SYNOPSYS®

DesignWare Cores Ethernet Quality-of-Service Databook

DWC-ETHERNET-QOS-SRC - Product Code: 6842-0

DWC-Ether-QOS-TSN1 Add-On - Product Code: C432-0

DWC-ETHERNET-QOS-TSN2 Add-On - Product Code: C433-0

DWC-Ethernet-QOS-ASP - Product Code: C118-0

Version 5.10a
December 2017

Copyright Notice and Proprietary Information

Synopsys, Inc.
690 E. Middlefield Road
Mountain View, CA 94043

www.synopsys.com

Contents

Revision History

The following table provides a summary of changes made to this Databook.

Date	Release	Description
December 2017	5.10a	■ Added a new chapter, Automotive Safety Features ■ Added a new appendix, DWC_ether_qos Automotive Safety ■ Added the following new sections ❑ Flexible Receive Parser ❑ Media Clock Generation and Recovery ❑ Comma Detection ■ Reorganized the chapter, Power Management and Energy Efficient Ethernet
September 2017	5.00a	Added the following new sections: ■ Time Sensitive Network (TSN) ■ Enhancements to Scheduled Traffic (EST) ■ Frame Preemption ■ Interrupt for MDIO Transaction Completion ■ VLAN Filter Fail Packets Queue ■ Broadcast/Multicast Packet Duplication ■ Fragmentation of IPv4 over UDP ■ Splitting Header on Receive ■ Recovering from DMA Channel Failure ■ Added Appendix C that lists the back-to-back registers ■ Updated CTXT and FD bit description in the Transmit and Receive Context descriptor ■ Updated CTXT and DE bit description in the Receive Normal (write-back) descriptor
June 2016	4.21a	Updates for ■ Programmable AV Slot Time ■ Posted writes for descriptors ■ DMA Start/Stop through sideband signals ■ Write 1 to clear interrupt status bits in CORE ■ Priority to writes in Tx SPRAM ■ VLAN Tagged IEEE 1588 PTP over Ethernet packets routing to Rx Queues ■ Multicast/Unicast DA fail packets routing to Rx Queues ■ Programmable control for providing Slave Error response when reserved CSR space is accessed ■ UDP layer Segmentation Offload ■ Support to avoid potential metastability due to software reset in EQOS-AHB/AXI configurations ■ Support to handle all 1s in the reserved fields of Tx and Rx DMA descriptors ■ Higher divisor for MDIO clock generation ■ 2.5 Gbps speed on GMII interface

Date	Release	Description
September 2015	4.20a	■ Added a new chapter, Queue/Channel Based VLAN Insertion on Tx ■ Added the following new sections: ❑ Extended Rx VLAN Filtering and Routing ❑ One-Step Time Stamping for PTP Over UDP ❑ SPRAM Interface ❑ Data Flow ❑ Application Clock Frequency Requirements ❑ SPRAM Timing ■ Updated AXI Low Power Interface ■ Updated the area and power numbers in Area and Power Appendix ■ Added the following details for the SPRAM support ❑ SPRAM Interface, data flow, and application clock frequency requirements ❑ SPRAM timing ■ Added the following new Registers: ❑ MAC_VLAN_Tag_Ctrl ❑ MAC_VLAN_Tag_Data ❑ MAC_VLAN_Tag_Filter ❑ MAC_HW_Feature3 ❑ AXI_LPI_Entry_Interval ■ Mac_Configuration ■ MAC_Ext_Configuration ■ MAC_Packet_Filter ■ MAC_VLAN_Tag ■ MAC_VLAN_Incl ■ MAC_Q[n]_Tx_Flow_Ctrl ■ MAC_Rx_Flow_Ctrl ■ MAC_TxQ_Prty_Map0 ■ MAC_TxQ_Prty_Map1 ■ MAC_RxQ_Ctrl0 ■ MAC_RxQ_Ctrl1 ■ MAC_RxQ_Ctrl2 ■ MAC_RxQ_Ctrl3 ■ MAC_Interrupt_Status ■ MAC_Interrupt_Enable ■ MAC_Rx_Tx_Status ■ MAC_PMT_Control_Status ■ MAC_LPI_Control ■ MAC_AN_Control ■ MAC_PHYIF_Control_Status ■ MAC_Version ■ MAC_HW_Feature1 ■ MAC_GPIO_Control Continued on next page

Date	Release	Description
September 2015	4.20a	■ Modified the Register fields for the following Registers: ❑ MAC_Address0_High ❑ MAC_Timestamp_Status ❑ MAC_Auxiliary_Control ❑ MTL_RxQ${i}_Operation_Mode ❑ MTL_RxQ${i}_Missed_Packet_Overflow_Cnt
October 2014	4.10a	■ Added the following configuration parameters: ❑ New drop-down EQOS-AXI4 configuration parameter for Application Interface Configuration ❑ New drop-down AXI4_Lite and APB4 interface configuration parameter for CSR Interface ❑ Enable IEEE 1588 Sub Nanoseconds Timestamp Support ❑ Enable PTP Timestamp Offload Feature ❑ Enable support for back to back register writes ■ The default value and dependencies of Enable One-Step Timestamp Feature configuration parameter is changed ■ Added the following new sections: ❑ APB4 Port Timing ❑ AXI4 Master Interface ❑ Transmit Queue Flush Operation ❑ Using PTP Timestamp Offload Function ❑ PTP Packet Generation ❑ PTP Message-Specific Fields ❑ Using the IPv4 ARP Offload Engine ■ Modified the following sections: ❑ Interrupts ❑ Address Assignments ❑ Transmit Path Functions ❑ Implementing Energy Efficient Ethernet ❑ Implementing Power Management through Magic Packet Detection ❑ System Considerations During Power Down ❑ Unified Power Format ■ Added illustrations to further explain the flows described in the following sections: ❑ Single-Packet Transmit Operation ❑ MAC Transmission ❑ MAC Reception ■ Modified the MAC Filter Status Word and Receive Status descriptions ■ Corrected the encoding value for Pdelay_Req used for ControlField in PTP Packet Generation ■ Modified signal description for phy_txd_o, ati_ctrl_type_i, tmi_wdata_o, tmi_rdata_i, trc_rd_data_i, and rrc_rd_data_i ■ Added the following signals: ati_txqueueflush_i, ati_txqueueflush_ack_o, sbd_perch_tx_intr_o[NTXQ-1:0], and sbd_perch_rx_intr_o[NRXCH-1:0] Continued on next page

Date	Release	Description
October 2014	4.10a	Added new signal tables: “AXI4 Master Interface Signals” on page 412 and “APB4 Interface Signals” on page 434 ■ Added the following new registers: ❑ DMA_CH0_TxDesc_List_HAddress ❑ DMA_CH0_RxDesc_List_HAddress ❑ DMA_CH0_Current_App_TxBuffer_H ❑ DMA_CH0_Current_App_RxBuffer_H ❑ MAC_1US_Tic_Counter ❑ MAC_PTO_Control ❑ MAC_Source_Port_Identity0 ❑ MAC_Source_Port_Identity1 ❑ MAC_Source_Port_Identity2 ❑ MAC_Log_Message_Interval ■ Modified the following register tables: ❑ MAC_Configuration ❑ MAC_RxQ_Ctrl1 ❑ MAC_PMT_Control_Status ❑ MAC_MDIO_Address ■ Modified the bit description for FCB, IPG, GPO, RI, TI, LPITCSE, TSIS, MMCTXIS, AV8021ASMEN, TXTSSTSM, ESTI, TSENMACADDR, TXTSSIS, OSTIAC register bits. ■ Added new bits: INTM, EIPG, EIPGEN, and SNSINC ■ The RFD and RFA bit descriptions and the entire register bit addressing is changed in MTL_RXQ0_Operation_Mode and MTL_RXQ1_Operation_Mode registers ■ Added new bit: and modified register bit addresses in MAC_Sub_Second_Increment ■ Modified the following sections inDescriptors chapter ❑ Transmit Normal Descriptor (Read Format) ❑ Transmit Context Descriptor ❑ Receive Descriptor ❑ RDES2 Normal Descriptor (Read Format) ❑ RDES3 Normal Descriptor (Read Format) ■ Updated the area and power numbers in Area and Power Appendix_ ■ Added new internal parameters appendix - Internal Parameter Descriptions.

Date	Release	Description
January 2014	4.00a-lp00	Minor document defects are fixed and a couple of minor doc enhancements are included in this release: ■ Corrected the number of Tx and Rx Receive paths mentioned in Audio Video Support ■ Corrected the description for the Enable Support for AV in Tx Queue 7 configuration parameter ■ Corrected the number of read and write accesses mentioned in MAC Control Interface ■ Updated the timing diagrams in “Transmit Path Timing”, “Receive Path Timing”, “MAC Transmit Timing” and “MAC Receive Interface (MRI)” ■ Corrected a cross-reference in “IP Header Checksum Engine” ■ Corrected the signal description for twc_wr_data_o signal ■ Corrected the bit descriptions for BLEN16, BLEN8, BLEN4, TXQEN, RQS, and SNPSVER ■ Corrected the description for IHE and MADRM descriptors
November 2013	4.00a	First version of the Databook

Preface

This document describes the Synopsys DesignWare Cores Ethernet Quality-of-Service (DWC_ether_qos) core, 5.10a. The DWC_ether_qos implements the Ethernet Quality-of-Service pertaining to the MAC layer.

The term DWC_ether_qos corresponds to DWC Ethernet QoS in the SolvNet database.

Databook Organization

The chapters of this databook are organized as follows:

- "Product Overview" describes the DWC_ether_qos features and provides an overview of the architecture.
- "Architecture" describes the DWC_ether_qos architecture and its main blocks.
- "VLAN and Double VLAN Insertion, Deletion, Replacement and Tagging" describes the VLAN insertion, replacement, and deletion; how to use the double VLAN tagging feature in which the MAC can process up to two VLAN tags.
- "Managing Buffers and Memories" describes the transmit and receive FIFOs and how to configure them.
- "Using PHY Interfaces" describes the Station Management module and the different PHY interfaces.
- "Packet Filtering" describes the different types of packet filtering.
- "IEEE 1588 Timestamp Support" describes the various IEEE 1588 features and their usage.
- "Multiple Channels and Queues Support" describes the multiple channels and queue support on the Transmit and Receive Paths.
- "TCP/IP Offloading Features" describes the Transmit and Receive offload engines, Segmentation, and Fragmentation features.
- "Power Management and Energy Efficient Ethernet" describes the low-power and energy management features including UPF.
- "MAC Management Counters" describes the MAC Management Counters, that store the statistics on the transmitted and received packets.
- "Flow Control" describes the flow control for Transmit and Receive paths.
- "Loopback Mode" describes the loopback support for the transmitted packets and the guidelines to use this mode.
- "Parameter Descriptions" describes the DWC_ether_qos configuration options and parameters.
- "Signal Descriptions" describes the top-level signals of the DWC_ether_qos.
- "Register Descriptions" maps and describes the function of the control and status registers of the DWC_ether_qos.
- "Internal Parameter Descriptions" lists the internal parameters of DWC_ether_qos that are mentioned elsewhere within the Databook
- "Descriptors" identifies and describes the DWC_ether_qos descriptors.
- "Programming" provides the programming sequences for initialization and other activities.
- Appendix A, "Area and Power" , lists the area and power numbers.
- Appendix B, "Endian Support" describes the endian support for 32-bit, 64-bit, and 128-bit data bus.
- Appendix C, "Back-to-Back Register Support" lists the supported back-to-back registers and the configurations.

DWC_ether_qos Reference Documentation

The following references contain useful information concerning the protocols addressed by this IP core:

- Carrier Sense Multiple Access With Collision Detection (CSMA/CD) Access Method and Physical Layer Specifications, Standard 802.3-2015, Institute of Electrical and Electronics Engineers (IEEE)
- Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems, Standard 1588-2008, IEEE
- Standards for Local and metropolitan area networks—Virtual Bridged Local Area Networks, IEEE 802.1Q-2014
- IEEE standards for Data Centre Bridging (DCB)
  - Enhanced Transmission Selection, Standard 802.1Qaz, IEEE
  - Priority-based Flow Control, Standard 802.1Qbb
  - MAC Control Frame for PFC, Standard 802.3bd, IEEE
- IEEE 802.1Qbb 2011 IEEE Standard for Local and Metropolitan Area Networks Virtual Bridged Local Area Networks - Amendment: Priority-based Flow Control (PFC)
- IEEE 802.1Qaz EEE Standard for Local and Metropolitan Area Networks-Virtual Bridged Local Area Networks - Amendment: Enhanced Transmission Selection (ETS)
- 802.1AS-Rev D5.0
  - Forwarding and Queuing Enhancements, Standard 802.1Qav, IEEE
  - Audio Video Bridging Systems, Standard 802.1BA-2009, IEEE
- IEEE Standards for Time Sensitive Networking (TSN)
  - Enhancements for Scheduled Traffic, Standard 802.1Qbv-2015, IEEE
  - Specification and Management Parameters for Interspersing Express Traffic, Standard 802.3br-2016, IEEE
  - Frame Preemption, Standard 802.1Qbu-2016, IEEE
- Standard for Energy Efficient Ethernet (EEE), IEEE Standard 802.3az-2010
- AMBA Specification, Revision 2.0, ARM Ltd. for AHB, APB interface
- AMBA4 AXI and ACE protocol specification, February 2013, ARM Ltd for AXI interface
- AMBA3 APB Protocol Specification, v1.0, ARM Ltd for APB3 interface
- MAC-PHY Interface Specifications
  - Reduced Gigabit Media Independent Interface (RGMII), Version 2.6, Broadcom Corp., Hewlett-Packard Co., Marvell Technology Group, Ltd.
  - Serial-GMII (SGMII) Specification, Revision 1.8, Cisco Systems
  - Serial-MII (SMII) Specification, Revision 2.1, Cisco Systems
  - RMII Specification, Revision 1.2, RMII Consortium
  - Reverse Media Independent Interface (RevMII) Block Architecture, Dmitriy Gusev
- RFC 768, User Datagram Protocol (UDP), DARPA Internet Program
- RFC 791, Protocol Specification (Internet Protocol, Version 4 (IPv4) Specification), DARPA Internet Program
- RFC 792, Internet Control Message Protocol Specification, DARPA Internet Program

- RFC 793, Transmission Control Protocol (TCP), DARPA Internet Program
- RFC 1071, Computing the Internet Checksum (memo), Network Working Group
- RFC 2460, Internet Protocol, Version 6 (IPv6) Specification, The Internet Society, Network Working Group
- RFC 2819, Remote Network Monitoring Management Information Base, The Internet Society, Network Working Group
- RFC 2965, HTTP State Management Mechanism, The Internet Society, Network Working Group
- RFC 4443, Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification, The Internet Society, Network Working Group
- 1588 Main Web Site, National Institute of Standards and Technology, http://ieee1588.nist.gov/
- Hardware-Assisted IEEE 1588* Implementation in the Intel® IXP46X Product Line (white paper), Intel, Inc., March 2005
- A Frequency Compensated Clock for Precision Synchronization using IEEE 1588 Protocol and its Application to Ethernet, Sivaram Balasubramanian, Kendal R. Harris, and Anatoly Moldovansky, Rockwell Automation, November, 2003
- UNH Ethernet Clause 4 MAC Test Suite - Annex D, University of New Hampshire InterOperability Laboratory
- Magic Packet Technology White Paper, Advanced Micro Devices, Inc.
- Device Class Power Management Reference Specification, version 2.0 by Microsoft Corporation and Advanced Micro Devices, Inc.
- Synopsys Low-Power Flow User Guide, Version E-2010.12, December 2010 (SolvNet ID required)
- Synopsys Low Power Verification Tools Suite User Guide, Version 2011.03, March 2011 (SolvNet ID required)
- Microsoft windows: Offloading the Segmentation of Large TCP Packets (NDIS 5.1)

Web Resources

The following websites are useful resources for accessing information regarding Synopsys DesignWare products and support:

- DesignWare IP product information: http://www.designware.com
- Your custom DesignWare IP page: http://www.designware.com
- Documentation through SolvNet: http://solvnet.synopsys.com (Synopsys password required)
- Synopsys Common Licensing (SCL): http://www.synopsys.com/keys

Customer Support

To obtain support for your product:

- First, prepare the following debug information, if applicable:
  - For environment setup problems or failures with configuration, simulation, or synthesis that occur within coreConsultant or coreAssembler, use the following menu entry:

    File > Build Debug Tar-file

    Check all the boxes in the dialog box that apply to your issue. This menu entry gathers all the Synopsys product data needed to begin debugging an issue and writes it to the file <core tool startup directory>/debug.tar.gz.
  - For simulation issues outside of coreConsultant or coreAssembler:
    - Create a waveforms file (such as VPD or VCD)
    - Identify the hierarchy path to the DesignWare instance
    - Identify the timestamp of any signals or locations in the waveforms that are not understood
- Then, contact Support Center, with a description of your question and supplying the above information, using one of the following methods:
  - For fastest response, use the SolvNet Web site. If you fill in your information as explained below, your issue is automatically routed to a support engineer who is experienced with your product. The Sub Product entry is critical for correct routing.

    Go to http://solvnet.synopsys.com/EnterACall and click on the link to enter a call. Provide the requested information, including:
    - Product: DesignWare Cores
    - Sub Product 1: Gigabit Ethernet
    - Sub Product 2: DWC_ethernet_qos
    - Tool Version: 5.00a
    - Problem Type:
    - Priority:
    - Title: Provide a short summary of the issue or list the error message you have encountered
    - Description: For simulation issues, include the timestamp of any signals or locations in waveforms that are not understood
    - After creating the case, attach any debug files you created in the previous step.
  - Or, send an e-mail message to support_center@synopsys.com (your e-mail will be queued and then, on a first-come, first-served basis, manually routed to the correct support engineer):
    - Include the Product name, Sub Product name, and Tool Version number in your e-mail (as identified above) so it can be routed correctly.
    - For simulation issues, include the timestamp of any signals or locations in waveforms that are not understood
    - Attach any debug files you created in the previous step.
  - Or, telephone your local support center:

- North America:
  Call 1-800-245-8005 from 7 AM to 5:30 PM Pacific time, Monday through Friday.
- All other countries:
  http://www.synopsys.com/Support/GlobalSupportCenters

1 Product Overview

This chapter provides an overview of the DesignWare Cores Ethernet Quality-of-Service (DWC_ether_qos).

The following components and features are discussed in this chapter:

1.1 General Product Description

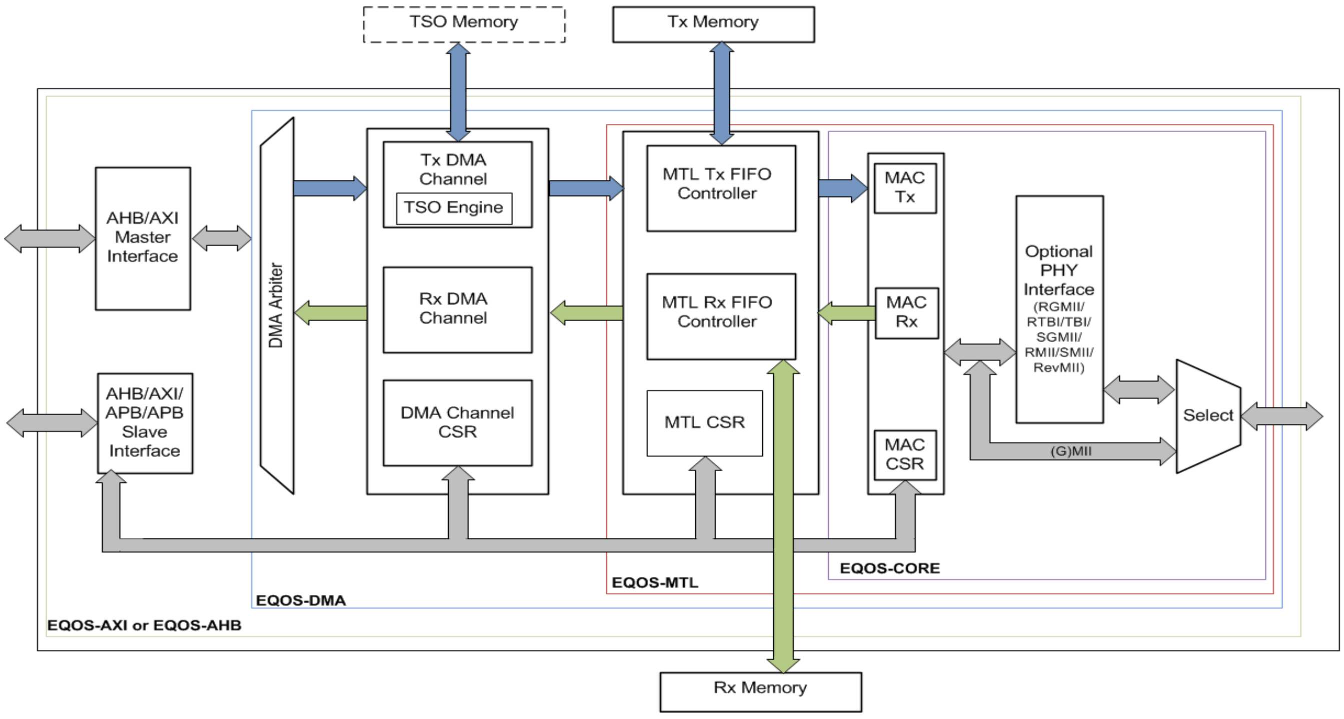
The DWC_ether_qos is compliant with the IEEE 802.3-2015 specification and can be used in applications such as AV bridges, AV nodes, switches, network interface cards, and data center bridges and nodes.

The DWC_ether_qos enables a host to transmit and receive data over Ethernet in compliance with the IEEE 802.3-2015. The DWC_ether_qos is a configurable product that can be optimized for gate count and latency to meet the application requirements.

1.1.1 System-Level Block Diagram

Figure 1-1 shows the system-level block diagram of DWC_ether_qos. A AHB or AXI Master interface is connected to all DMA channels. The DMA arbiter helps in arbitration of all the paths (Transmit and Receive) in all channels. Each channel has a separate set of Control and Status registers (CSR) for managing the Transmit and Receive functions, descriptor handling, and interrupt handling.

Figure 1-1 System-Level Block Diagram



Note When you configure the MAC for a single PHY interface, the MUX logic is removed and the corresponding PHY interface is directly connected to the ports.

1.1.2 Interfaces

The DWC_ether_qos controller supports the following interfaces:

- **Native Interface**

  The DWC_ether_qos subsystem can have one of the following interfaces:

  - Direct FIFO-type interface to the MAC through native CORE interface, which uses Ethernet PHY rate clocks at the interface
  - Direct FIFO-type interface to the MAC with MTL FIFO layer through native MTL interface, which uses an application-specific clock at the interface
  - Direct native DMA interface to the MAC, which uses an application-specific clock at the interface

  The native 32-bit Read or Write bus is provided for CSR access. You can select an APB, APB3, or APB4 port for CSR access instead of the default native 32-bit Read or Write MAC Control Interface.

- **AHB Interface**

  The AHB interface is designed to integrate with AMBA High-Performance Bus (AHB) on the application side. The AHB interface transfers the data to and from system memory through the AHB master interface. When the AHB interface is selected, the host CPU uses the default 32-bit AHB slave interface to access the Control and Status registers (CSRs) of the DWC_ether_qos subsystem. You can also select an APB, APB3, or APB43 port for CSR access instead of the AHB slave port.

- **AXI Interface**

  The AXI interface is designed to integrate with AMBA Advanced Extensible Interface Bus (AXI) on the application side. The AXI interface transfers the data to and from the system memory through AXI master interface. When the AXI interface is selected, the host CPU uses the AXI slave interface to access the Control and Status registers (CSRs) of the DWC_ether_qos subsystem. DWC_ether_qos can support either AXI3 or AXI4 compliant interface protocol on the master interface. Similarly, it can support AXI3 compliant or AXI4-Lite interface on the slave port. You can also select an APB, APB3, or APB4 port or AHB as slave port for CSR access instead of the AXI slave port.

- **PHY Interfaces**

  The DWC_ether_qos supports any one or a combination of the following PHY interfaces:

  - Gigabit Media Independent Interface (GMII)/Media Independent Interface (MII) [Default]
  - Reduced GMII (RGMII)
  - Serial GMII (SGMII)
  - Ten Bit Interface (TBI)
  - Reduced MII (RMII)
  - Serial MII (SMII)
  - Reduced TBI (RTBI)
  - Reverse MII (RevMII)

1.2 DWC_ether_qos Features

The DWC_ether_qos features are divided into the following categories:

1.2.1 Standard Compliance

In addition to the default interfaces defined in the IEEE 802.3 specifications, the DWC_ether_qos supports several industry standard interfaces to the PHY. The DWC_ether_qos is compliant with the following standards:

- IEEE 802.3-2015 for Ethernet MAC, Gigabit Media Independent Interface (GMII), Media Independent Interface (MII), Ten Bit Interface (TBI)
- IEEE 1588-2008 for precision networked clock synchronization
- IEEE 802.1AS-2011 and 802.1-Qav-2009 for Audio Video (AV) traffic
- IEEE 802.1AS-Rev/D4.0, Timing and Synchronization for Time-Sensitive Applications in Bridged Local Area Networks
- IEEE 802.3az-2010 for Energy Efficient Ethernet (EEE)
- IEEE 802.1Qbv-2015, 802.1Qbu-2016, and 802.1AS-Rev D5.0 for Time-Sensitive Networking (TSN) traffic
- IEEE 802.1Qaz-2011 and 802.1Qbb-2011 for Data Center Bridging (DCB) traffic
- AMBA 2.0 for AHB master, AHB slave, and APB slave ports
- AMBA 3.0 for AXI master and slave and APB3 slave ports
- AMBA4 AXI and ACE protocol specification, February 2013, ARM Ltd for AXI4 and APB4 interface
- RGMII/RTBI specification version 2.6 from HP/Marvell
- RMII specification version 1.2 from RMII consortium
- SGMII specification version 1.8 from Cisco/Marvell
- SMII specification version 2.1 from Cisco for SMII
- Reverse Media Independent Interface (RevMII) by Dmitriy Gusev

1.2.2 MAC Features

The DWC_ether_qos controller supports a number of Tx and Rx MAC features.

The MAC includes the following feature groups:

- "MAC Tx and Rx Features" on page 27
- "MAC Tx Features" on page 28
- "MAC Rx Features" on page 28

1.2.2.1 MAC Tx and Rx Features

- Separate transmission, reception, and control interfaces to the application
- Configurable big-endian and little-endian mode for Transmit and Receive paths
- 10, 100, and 1000 Mbps data transfer rates with the following PHY interfaces:
  - IEEE 802.3-compliant GMII/MII (default) interface to communicate with an external Gigabit or Fast Ethernet PHY
  - IEEE 802.3z-compliant TBI interface (optional), with auto-negotiation to communicate with an external PHY
  - RTBI interface (optional) to communicate with an external gigabit PHY
  - RGMII interface (optional) to communicate with an external gigabit PHY
  - SGMII interface (optional) with auto-negotiation to communicate with an external Gigabit PHY
  - SMII interface (optional) to communicate with an external Fast Ethernet PHY
  - RMII interface (optional) to communicate with an external Fast Ethernet PHY
  - RevMII interface (optional) to directly communicate with a remote MAC
- Half-duplex operation:
  - CSMA/CD Protocol support
  - Flow control using backpressure support (based on implementation-specific white papers and UNH Ethernet Clause 4 MAC Test Suite - Annex D)
  - Packet bursting and packet extension in 1000 Mbps half-duplex operation
- Standard IEEE 802.3az-2010 for Energy Efficient Ethernet in (G)MII and Reduced Gigabit Media Independent Interface (RGMII) PHYs.
- 32-bit, 64-bit, or 128-bit data transfer interface on the application side
- Full-duplex flow control operations (IEEE 802.3x Pause packets and Priority flow control)
- Optional network statistics with RMON or MIB Counters (RFC2819/RFC2665)
- Optional module to support Ethernet packet timestamping as described in IEEE 1588-2002 and IEEE 1588-2008 (64-bit timestamps given in the Tx or Rx status of PTP packet). Both one-step and two-step timestamping is supported in TX direction.
- Flexibility to control the Pulse-Per-Second (PPS) output signal (ptp_pps_o)
- Media clock generation and recovery
- Optional MDIO (Clause 22 and Clause 45) master interface for PHY device configuration and management

- Option to select up to 16 general purpose inputs and outputs
  The general purpose inputs are programmable to generate interrupts on rising or falling edges.

1.2.2.2 MAC Tx Features

- Preamble and start of packet data (SFD) insertion
- Separate 32-bit status for each packet transmitted from the application
- Automatic CRC and pad generation controllable on a per-packet basis
- Programmable packet length to support Standard or Jumbo Ethernet packets with up to 16 KB of size
- Programmable Inter Packet Gap (40–96 bit times in steps of 8)
- IEEE 802.3x Flow Control automatic transmission of zero-quanta Pause packet when flow control input transitions from assertion to de-assertion (in full-duplex mode)
- Source Address field insertion or replacement, and VLAN insertion, replacement, and deletion in transmitted packets with per-packet or static-global control
- Insertion, replacement, or deletion of up to two VLAN tags
- Option to transmit packets with reduced preamble size in full-duplex mode
- Insert, replace, or delete queue/channel-based VLAN tags
- Frame Preemption for MAC Tx

1.2.2.3 MAC Rx Features

- Automatic Pad and CRC Stripping options
- Option to disable Automatic CRC checking
- Preamble and SFD deletion
- Separate 112-bit or 128-bit status
- Programmable watchdog timeout limit
- Flexible address filtering modes:
  - Up to 31 additional 48-bit perfect (DA) address filters with masks for each byte
  - Up to 96 additional 48-bit perfect (DA) address filters that can be selected in blocks of 32 and 64 registers
  - Up to 31 48-bit SA address comparison check with masks for each byte
  - 32 bit, 64 bit, 128 bit, or 256 bit Hash filter (optional) for multicast and unicast (DA) addresses
  - Option to pass all multi-cast addressed packets
  - Promiscuous mode to pass all packets without any filtering for network monitoring
  - Pass all incoming packets (as per filter) with a status report
- Additional packet filtering:
  - VLAN tag-based: Perfect match and Hash-based (optional) filtering. Filtering based on either outer or inner VLAN tag is possible.
  - Layer 3 and Layer 4-based: TCP or UDP over IPv4 or IPv6
  - Extended VLAN tag based filtering 4, 8, 16, or 32 filter selection

- IEEE 802.1Q VLAN tag detection and option to delete the VLAN tags in received packets
- Optional module to detect remote wake-up packets and AMD magic packets
- Optional forwarding of received Pause packets to the application (in full-duplex mode)
- Optional Receive module for Layer 3/Layer 4 checksum offload for received packets
- Optional stripping of up to two VLAN Tags and providing the tags in the status.
- Frame Preemption for MAC Rx

1.2.3 Transaction Layer (MTL) Features

The DWC_ether_qos supports a number of Tx and Rx Transaction Layer (MTL) features.

The MTL includes the following feature groups:

- "MTL Tx and Rx Common Features" on page 29
- "MTL Tx Features" on page 29
- "MTL Rx Features" on page 30

1.2.3.1 MTL Tx and Rx Common Features

The DWC_ether_qos controller supports the following common features of the MTL Tx and Rx:

- 32-bit, 64-bit, or 128-bit Transaction Layer block (bridges the application and the MAC)
- Data transfers executed using simple FIFO protocol
- Synchronization for all clocks in the design (Transmit, Receive, and Application clocks)
- Optimization for packet-oriented transfers with packets delimiters
- Option to have dual-port RAM based asynchronous FIFO controllers or Single-port RAM based synchronous FIFO controllers
- RAM memory instantiation outside the top-level module to facilitate memory testing or instantiation
- Programmable burst length, up to half the size of the MTL Rx queue or Tx queue size, to support burst data transfer in the EQOS-MTL configuration
- Programmable threshold capability for each queue (default of 64 bytes)
- Optional Debug and slave mode operation on Queue 0 (default queue)

1.2.3.2 MTL Tx Features

The DWC_ether_qos controller supports the following MTL Tx features:

- Following FIFO sizes on transmission: 256 byte, 512 byte, 1 KB, 2 KB, 4 KB, 8 KB, 16 KB, 32 KB, 64 KB, or 128 KB
- Multiple queues (up to 8) on the Transmit path with a common memory for all Tx queues
- Store-and-Forward mechanism or threshold mode (cut-through) for transmission to the MAC
- Programmable queue size in configurations with multiple queues. Each queue size can be programmed in terms of 256 bytes
- Automatic retransmission of collision packets in half-duplex mode

- Discard packets on late collision, excessive collisions, excessive deferral, and under-run conditions with appropriate status
- Disabling of Data Memory RAM chip-select when inactive to reduce power consumption
- Optional module to calculate and insert IPv4 header checksum and TCP, UDP, or ICMP checksum
- Programmable interrupt options for different operational conditions
- Statistics by generating pulses for packets dropped (because of underflow) in the Tx FIFO
- Optional statistics related to bandwidth consumption by each queue of up to 16 blocks over a 125 µ?s period
- Optional packet-level control for
  - VLAN tag insertion or replacement
  - Ethernet source address insertion
  - Layer3/Layer4 Checksum insertion control
  - One-step timestamp
  - Timestamp control
  - CRC and pad control
- Following scheduling algorithms in configurations with multiple queues:
  - Weighted Round Robin (WRR)
  - (When Data Center Bridging is enabled) Deficit Weighted Round Robin (DWRR)
  - (When Data Center Bridging is enabled) Weighted Fair Queuing (WFQ)
  - Strict Priority (SP)
  - (When Audio-Video Bridging is enabled) Credit-based Shaper (CBS)
  - (When TSN is enabled), Enhancement to Scheduled Traffic (EST)
  - (When TSN is enabled), Time Based Scheduling (TBS)
- Option to support dropping of Tx Status to improve the Transmit throughput

1.2.3.3 MTL Rx Features

The DWC_ether_qos controller supports the following Rx MTL features:

- Following Rx queue sizes in the Receive path: 256 byte, 512 byte, 1 KB, 2 KB, 4 KB, 8 KB, 16 KB, 32 KB, 64 KB, 128 KB, or 256 KB
- Multiple queues (up to 8) on the Receive path with a common memory for all Rx queues
- Insertion of Rx Status vectors into the Rx queue after the EOP transfer (in Threshold mode) and before SOP (in Store-and-Forward mode) in EQOS-MTL configuration
- Programmable Rx queue threshold (default fixed at 64 bytes) in Threshold (or cut-through) mode
- Option to filter all error packets on reception and not forward them to the application in the store-and-forward mode
- Option to forward the undersized good packets
- Statistics by generating pulses for packets dropped (because of overflow) in the Rx FIFO
- Automatic generation of Pause packet control or backpressure signal to the MAC based on the Rx Queue fill level

- Arbitration among queues when multiple queues are present. The following arbitration schemes are supported:
  - Weighted Round Robin (WRR)
  - Weighted Strict priority (WSP)
  - Strict Priority (SP)
- Option to replicate received multicast packets for transfer by multiple Rx DMA channels
- Option to have a programmable lookup table based flexible Parser for filtering and steering the Rx packets

1.2.4 DMA Block Features

The DWC_ether_qos controller supports a variety of DMA block features.

The DMA block supports the following features:

- 32-bit, 64-bit, and 128-bit data transfers
- Multi-channel Transmit and Receive engines (up to 8 Transmit channels; up to 8 Receive channels)
- Separate DMA channel in the Transmit path for each queue in MTL
- Single or multiple DMA channels for any number of queues in MTL Receive path
- Fully synchronous design operating on a single application clock (except for CSR module, when a separate CSR clock is configured)
- Optimization for packet-oriented DMA transfers with packet delimiters
- Byte-aligned addressing for data buffer support
- Dual-buffer (ring) descriptor support
- Descriptor architecture to allow large blocks of data transfer with minimum CPU intervention (each descriptor can transfer up to 32 KB of data)
- Comprehensive status reporting for normal operation and transfers with errors
- Individual programmable burst length for Tx DMA and Rx DMA engines for optimal host bus utilization
- Programmable interrupt options for different operational conditions
- Per-packet Transmit or Receive Complete Interrupt control
- Round-robin or fixed-priority arbitration between the Receive and Transmit engines
- Start and Stop modes
- Separate ports for host CSR access and host data interface
- Optional support for TCP Segmentation Offload (TSO) and UDP Fragmentation Offload (UFO)
- Selectable number of Tx DMA channels with TSO/UFO feature enabled
- Routing of received packets to the DMA channels based on the DA or VLAN Priority in multi-channel DMA configurations
- Option to split the packet header (Layer 3 and Layer 4) and payload in a different buffers
- Time-sensitive conditional packet fetching from system memory by comparing the Slot Time or IEEE 1588 time information provided in the descriptor (useful for AV applications)
- Programmable control for Transmit Descriptor posted writes to improve the throughput
- Sideband signals to control starting and stopping of DMA channels

1.2.5 AHB Interface

The AMBA AHB interface features can be divided into the following categories:

- "AHB Master Interface Features" on page 32
- "AHB Slave Interface Features" on page 32

1.2.5.1 AHB Master Interface Features

The DWC_ether_qos controller supports the AHB Master interface as the application interface.

The AHB master interface supports the following features:

- Interfaces with the application through AHB
- Little-endian and big-endian modes
- 32-bit, 64-bit, or 128-bit data on the AHB master port
- Option to select address-aligned bursts from AHB master port
- Split, Retry, and Error AHB responses
- AHB 1K boundary burst splitting
- Software-selected type of AHB burst (fixed burst, indefinite burst, or mix of both)

The AHB master interface does not generate the following:

- Wrap burst
- Locked or Protected transfers

1.2.5.2 AHB Slave Interface Features

The DWC_ether_qos controller supports the AHB Slave interface as the CSR interface.

The AHB slave interface supports the following features:

- Interfaces with the application through AHB
- Little-endian and big-endian modes
- AHB slave interface (32-bit, 64-bit, or 128-bit) for CSR access in which only 32-bit or less (byte, half-word) accesses are possible
- All AHB burst types

The AHB slave interface does not generate the following responses:

- Split
- Retry

1.2.6 AXI Interface

The AMBA AXI interface features can be divided into the following categories:

- "AXI Master Interface Features" on page 33
- "AXI Slave Interface Features" on page 33

1.2.6.1 AXI Master Interface Features

The AXI master interface supports the following features:

- Interfaces with the application through AXI3 or AXI4 compatible interface
- 32-bit, 40-bit or 48-bit address width
- Little-endian and byte-invariant big-endian modes
- AXI low-power interface
- 32-bit, 64-bit, or 128-bit data
- OKAY, SLVERR, and DECERR responses
- Software-selected type of AXI burst (fixed and variable length burst) in AXI master interface; Option to select extended length fixed bursts of 32, 64, 128, and 256
- Option to select address-aligned bursts
- AXI 4 KB boundary burst splitting and support for limiting burst to 1 KB boundary
- Up to 32 outstanding Read and Write transactions
- Posted writes during multiple data transfers from the AXI master interface to maximize the bus utilization
- Handshaking on AXI Read and Write data channels

The AXI master interface does not support the following:

- Burst interleaving and reordering in AXI master write channel
- Atomic, Exclusive, Locked, Cache, and Protected accesses
- EXOKAY response from Slave

1.2.6.2 AXI Slave Interface Features

The AXI slave interface supports the following features:

- Interfaces with the application through AXI3 or AXI4-Lite compatible interface
- AXI slave interface (32-bit, 64-bit, or 128-bit) for CSR access
- 32-bit address width
- Little-endian and byte-invariant big-endian modes
- Narrow burst and FIXED or INCR burst

The AXI slave interface does not support the following accesses:

- Exclusive
- Atomic
- Locked
- Cache
- Protected

The AXI Slave interface does not generate the following response:

- EXOKAY

1.2.7 AMBA APB Slave Interface Features

The DWC_ether_qos includes a number of APB master and slave interfaces features.

- 32-bit access without any byte-enable
- APB3 interface with pready handshake signal
- APB4 interface with pstrb and pprot signal inputs

1.2.8 Audio and Video Features

DWC_ether_qos can be used in Audio Video (AV) mode, and the supported features are compliant to the industry standards for AV traffic.

DWC_ether_qos supports the following Audio Video (AV) features:

- Separate channels or queues for AV data transfer in 100 Mbps and 1000 Mbps modes
- Up to eight queues on the Receive paths for AV traffic and seven queues on the Transmit path for AV traffic
- IEEE 802.1-Qav specified credit-based shaper (CBS) algorithm for Transmit channels
- Single Tx FIFO and Rx FIFO (MTL) for all selected queues (system-side interface [AHB, AXI, or native] remains the same)
- Programmable Slot Interval with range from 1µs to 4096µs and granularity of 1µs.

1.2.9 Data Center Bridging Features

DWC_ether_qos can be used in Data Center Bridging (DCB) mode, and it uses different queuing and scheduling mechanisms that comply with industry standards.

DWC_ether_qos supports the following Data Center Bridging (DCB) features:

- Separate channels or queues for DCB data transfer in 100 Mbps and 1000 Mbps modes
- Up to eight queues on the Transmit and Receive paths for DCB traffic
- The following Tx queues scheduling mechanisms meet the compliance specification in IEEE 802.1-Qaz Enhanced Transmission Selection (ETS) algorithm:
  - Weighted Round Robin (WRR)
  - Deficit Weighted Round Robin (DWRR)
  - Weighted Fair Queuing (WFQ)
  - Strict Priority (SP)
- Common memory for all selected Tx or Rx queues [system-side interface (AHB, AXI, or native) remains the same]
- Priority-based Flow Control (PFC) on individual queues configured for DCB
- Programmable control to route received VLAN tagged packets to channels or queues

1.2.10 Time Sensitive Networking Features

DWC_ether_qos supports the following Time Sensitive Networking (TSN) features:

- IEEE 802.1Qbv-2015, Enhancements to Scheduling Traffic
- IEEE802.1Qbu/802.3br, Frame preemption and Interspersing Express Traffic

1.2.11 Generic Queuing Features

The DWC_ether_qos supports the following features for generic queuing.

- Programmable control for routing Receive packets with Multicast/Broadcast destination address to a programmable Receive Queue
- Support routing of Untagged Receive packets to a programmable Receive Queue
- Programmable control for routing VLAN tagged and untagged IEEE 1588 PTP over Ethernet Receive packets to a same or separate programmable Receive Queue
- Programmable control for routing Unicast/Multicast Receive packets that fail the destination address filter to a separate programmable Receive Queues

1.2.12 Automotive Safety Features

The automotive safety features are targeted to improve the reliability and reduce the time taken to detect faults in the controller:

Note These features are available only with DWC ASP Ethernet QOS (C118-0) product

DWC_ether_qos supports the following features for automotive safety:

- Error correction code (ECC) protection for memories
- On-chip data path parity protection
- FSM parity and timeout protection
- Application/CSR interface timeout protection

1.2.13 Monitoring, Testing, and Debugging Features

The monitoring, testing, and debugging mechanism in DWC_ether_qos help in effective analysis of the configured features.

DWC_ether_qos supports the following features for monitoring, testing, and debugging:

- Internal loopback from Tx to Rx on the GMII or MII for debugging
- DMA states (Tx and Rx) as status bits
- Debug status register that gives status of FSMs in Transmit and Receive data paths and FIFO fill-levels
- Application Abort status bits
- MMC (RMON) module
- Current Tx or Rx Buffer pointer as status registers
- Current Tx or Rx Descriptor pointer as status registers
- Statistical counters to calculate the bandwidth served by each Transmit channel when AV or DCB support is enabled
- Tx or Rx Queues memory accessible through Slave port for debug
- Optional protection of memory content with SECDED (Single bit Error Correction and Double bit Error Detection) ECC

- Ability to inject errors to check the safety features such as, ECC/Parity protection and FSM/Interface timeouts

1.3 DWC_ether_qos Configurations

DWC_ether_qos provides different configuration options using coreConsultant so that an optimized design can be achieved based on the required architecture, interfaces, and area and timing.

DWC_ether_qos can be configured to any of the following major architectures:

- EQOS-AHB: Subsystem with DMA and AHB interface (default)
- EQOS-AXI: Subsystem with DMA and AXI interface
- EQOS-DMA: Subsystem with DMA and native interface (without AHB and AXI)
- EQOS-MTL: Subsystem with transaction (FIFO) layer only (without DMA) and with native FIFO interface
- EQOS-CORE: The DWC_ether_qos with native FIFO interface (without DMA, FIFO layer, AHB and AXI)

In addition, the PHY can be configured to have any of the following interfaces (including an option to select them at reset), if multiple interfaces are configured:

- GMII or MII
- TBI
- RGMII
- RTBI
- SGMII
- RMII (only for 10 Mbps and 100 Mbps operations)
- SMII (only for 10 Mbps and 100 Mbps operations)
- RevMII

1.3.1 Configurable Features

You can also configure the following features in coreConsultant for an optimized design with respect to area and timing:

- 10/100 Mbps operation only
- 1000 Mbps operation only
- 10/100/1000 Mbps operation
- Full-duplex operation only
- Rx FIFO and Tx FIFO size in transaction layer
- Bus width of the AHB or AXI (or application) interface to 32, 64, or 128
- Address filtering option (Hash filter function)
- Packet filtering based on the TCP/IP headers
- Configurable number of MAC Address registers (up to 128) for filtering
- A fully programmable packet parser for flexible filtering
- RMON Counters (MMC) with individual registers and their width selection
- Power Management Module (PMT) with remote wake-up packet and magic packet processing options
- IP Checksum Offload support for TCP, UDP, or ICMP payloads encapsulated over IP

- TCP Segmentation and UDP Fragmentation Offload
- ARP Offload support
- IEEE 1588 Ethernet frame timestamp support
- AV, TSN, and DCB traffic queuing and forwarding support
- Endianness for all application-side data paths
- Station Management Agent (SMA) – MDIO module
- Choice of APB, APB3, APB4, AHB, AXI3, AXI4-Lite or native MCI interface for slave port
- Separate application clocks option for slave port
- SA and VLAN insertion or replacement
- Configurable number of general purpose inputs and outputs (up to 16 each)
- Packet transmission and reception through Slave interface instead of Master interface

1.4 DWC_ether_qos Deliverables

The deliverables included in the DWC_ether_qos software package help in the integration of the controller with RTL, verification testbenches, IP-XACT component, and so on.

The DWC_ether_qos is packaged as a dw_iip_DWC_ether_qos_5.10a.run file. The DWC_ether_qos requires the Synopsys coreConsultant tool to install, unpack, and configure the DWC_ether_qos. The license file required to install, unpack, and configure DWC_ether_qos is delivered separately.

The DWC_ether_qos image includes the following:

- Verilog RTL source code
- Synthesis script generation for Synopsys Design Compiler and Synplify Pro.
- Verification testbenches and test configurations derived from your parameter choices
  - Verilog-based testbench environment for EQOS-AHB and EQOS-AXI configuration using DW AMBA VIP. You can use this testbench, along with the provided test cases, as a reference to build a verification environment for an SoC in which EQOS-AHB or EQOS-AXI integration can be tested.

    For more information about the VTB used to test the EQOS-AHB/EQOS-AXI configurations using AMBA Verification IP (VIP), see *Design Ware Cores Ethernet Quality-of-Service, User Guide*.
  - Verilog-based testbench environment and test cases for EQOS-DMA, EQOS-MTL, and EQOS-CORE configurations.
  - Support for creating gate-level Device Under Test (DUT) models using a GTECH library.
  - Verification scripts for running with three simulators: Synopsys VCS, ModelSim, and NC-Verilog simulators.
- IP-XACT component representation of the IP (XML format). The coreConsultant supports the generation of configured core integration files that meet the IP-XACT specifications. The coreConsultant also supports the file generation for IP-XACT-compliant software viewing of the configured core.

Note The memory map is generated for all registers that are directly accessed by the host and are listed in Register Descriptions. However, the RevMII Registers are accessed indirectly through the MDIO interface. Therefore, these registers are not present in the Memory Map generated in the IP-XACT view.

Note Contact Synopsys Support Center for software driver

1.5 Related Synopsys Products

Synopsys offers demo drivers, demo boards, and PHYs that can be used with the DWC_ether_qos controller.

The following products are available for the DWC_ether_qos controller through separate part numbers and written agreements.

- Demo drivers
- Demo boards
- PHYs

2 Architecture

This chapter describes the DWC_ether_qos interfaces, protocols, functionality, and implementation. It contains the following sections:

2.1 CSR Slave Interface

You can access the CSR space by specifying a CSR slave interface in coreConsultant.

The CSR slave interface provides access to the CSR space.

You can select a slave interface by using the CSR Interface option in coreConsultant. The DWC_ether_qos can have one of the following slave interfaces:

- AHB Slave interface
- AXI3 or AXI4-Lite slave interface
- APB, APB3, or APB4 slave interface
- MCI slave interface

By default, all the slave interfaces of DWC_ether_qos except MCI slave interface (which do not have provision for providing error response) will provide OKAY response to all CSR accesses. If SEEN bit of MAC_CSR_SW_Ctrl register is programmed to 1, DWC_ether_qos will provide error response when reserved registers within the CSR space are accessed.

2.1.1 AHB Slave Interface

The AHB slave is supported in the EQOS-AHB and EQOS-AXI configurations. The 32-bit AHB slave interface provides access to the DMA, MTL, and MAC CSR space. The AHB slave interface supports the following features:

- Fully AMBA 2.0 Compliant AHB slave—no restrictions
- Single and all burst transfers
- Busy and early terminations
- 32-bit, 16-bit, and 8-bit write or read transfers to the CSR

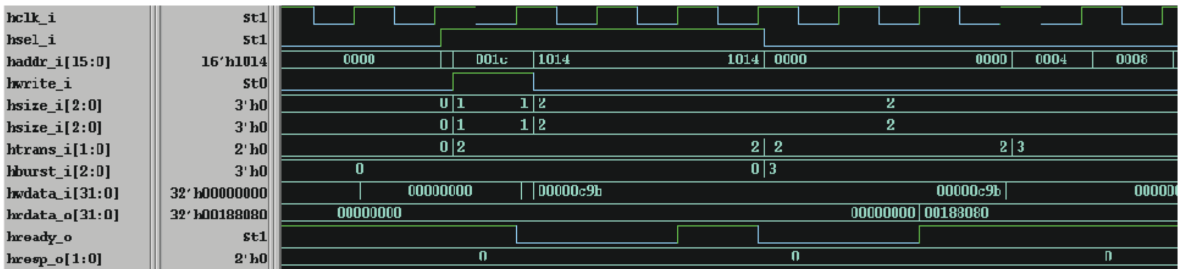
  The 32-bit CSR access is recommended to avoid any software synchronization problems.
- OKAY response

  The SPLIT or RETRY responses are not supported. When a reserved address is read in MAC register space, it gives all zeros data and generates an OKAY response (When SEEN bit is set, MAC generates ERROR response). Similarly, when a reserved address is written, the DWC_ether_qos ignores the write request and generates an OKAY response (When SEEN bit is set, MAC generates ERROR response).

2.1.1.1 AHB Slave Port Timing

Figure 2-1 shows a register write access (to address 0x001C) followed by a register read access (to address 0x1014) on the AHB slave port. The waveform is as per the AMBA 2.0 specification. The AHB slave port completes all read and write transfers in three clocks and therefore, hready_o is DE-asserted for two clocks for any register access. The AHB slave port supports burst operations, and the timing response remains the same (three clocks for completion of any read or write cycle).

Figure 2-1 Register Write Access Followed by Register Read Access on AHB Slave Port



2.1.1.2 Interfacing AHB Slave Port with a Wider AMBA Bus

To interface the 32-bit AHB slave port to a wider AMBA bus, extra logic described in Section 3-15 of AMBA 2.0 specification is required. You can include this logic by selecting the CSR Port Data Width option while configuring the controller. In such configurations, even though the AHB slave port indicates a 64-bit or 128-bit data bus, all slave port accesses must still be 32 bits or less. If a 64-bit or 128-bit access is performed, the AHB slave port internally completes a 32-bit access and gives an OKAY response. During the AHB slave write operation, the data on higher lanes is ignored. During the AHB slave Read transfer, the lower 32-bit lane is duplicated.

2.1.2 AXI Slave Interface

The AXI slave is supported in the EQOS-AXI configurations. The features supported for the AXI slave interface are listed in "AXI Slave Interface Features" on page 33.

The slave interface does not support the following features:

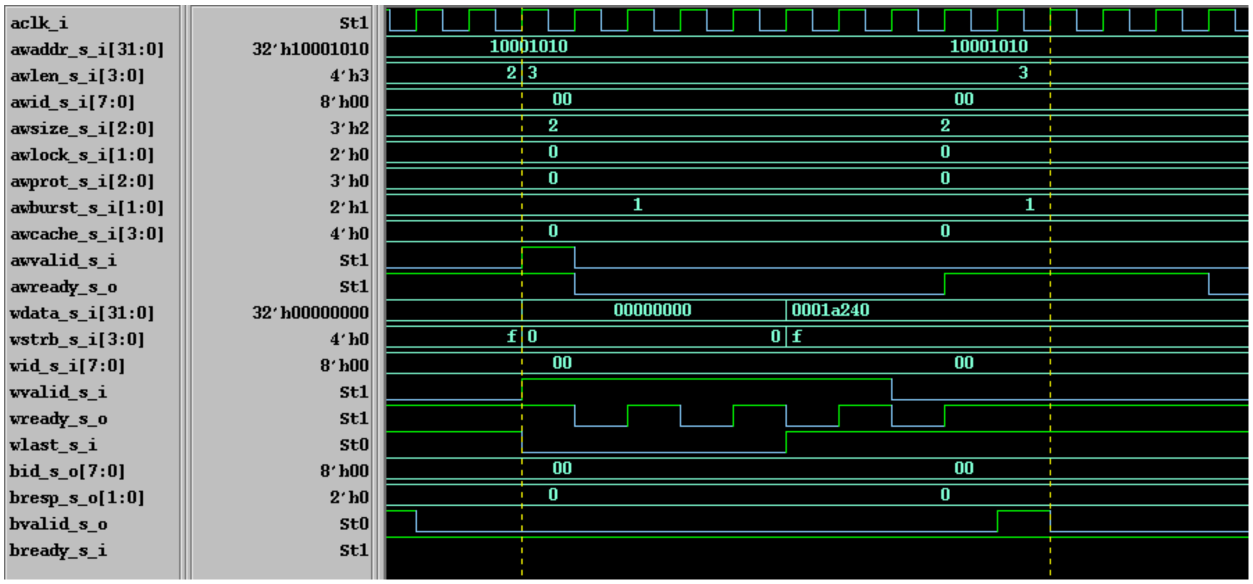
- Data reordering
- Data interleaving
- WRAP burst transfers
- Signals awcache/awprot or arcache/arprot
- Atomic accesses
- Exclusive Access

Note DWC_ether_qos provides only OKAY response on bresp_s_o and rresp_s_o. It does not provide the EXOKAY, SLVERR, and DECERR responses.

You can select the AXI4-Lite slave interface instead of the default AXI3 slave. In this configuration, only single beat transfers are supported and so many AXI3 control signals get removed as listed in the AMBA specification.

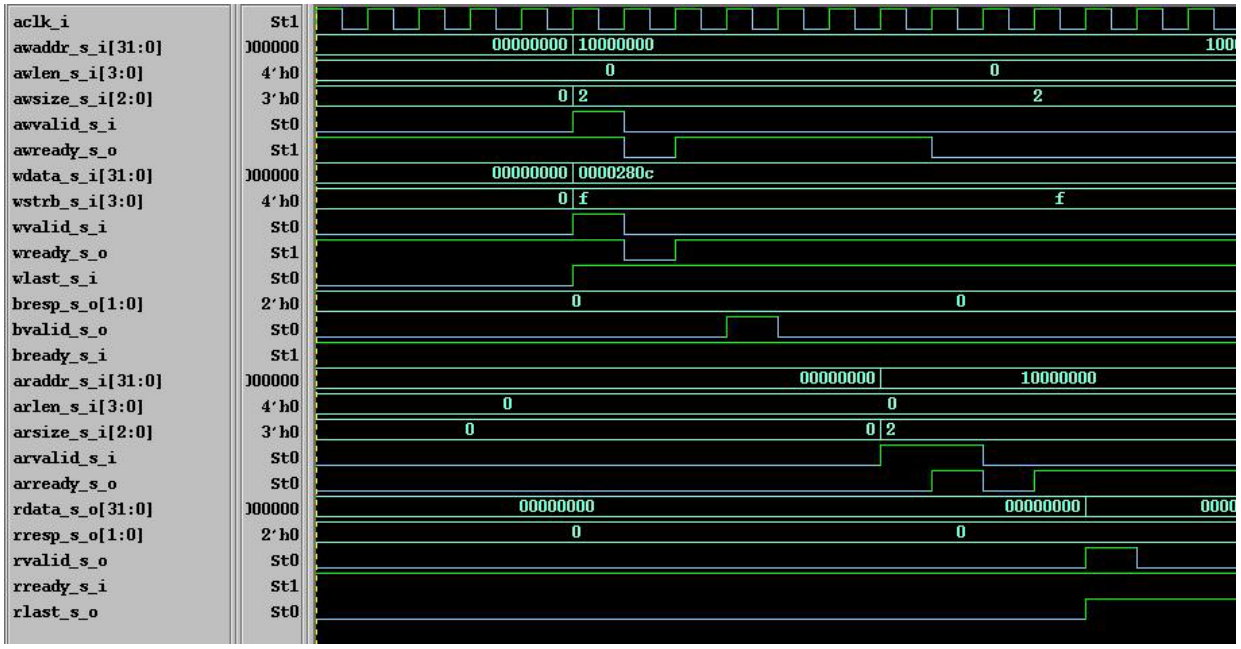
Figure 2-2 shows the timing on the AXI slave interface for a burst-write transfer.

Figure 2-2 Write Channel Timing of 32-Bit AXI Slave Interface



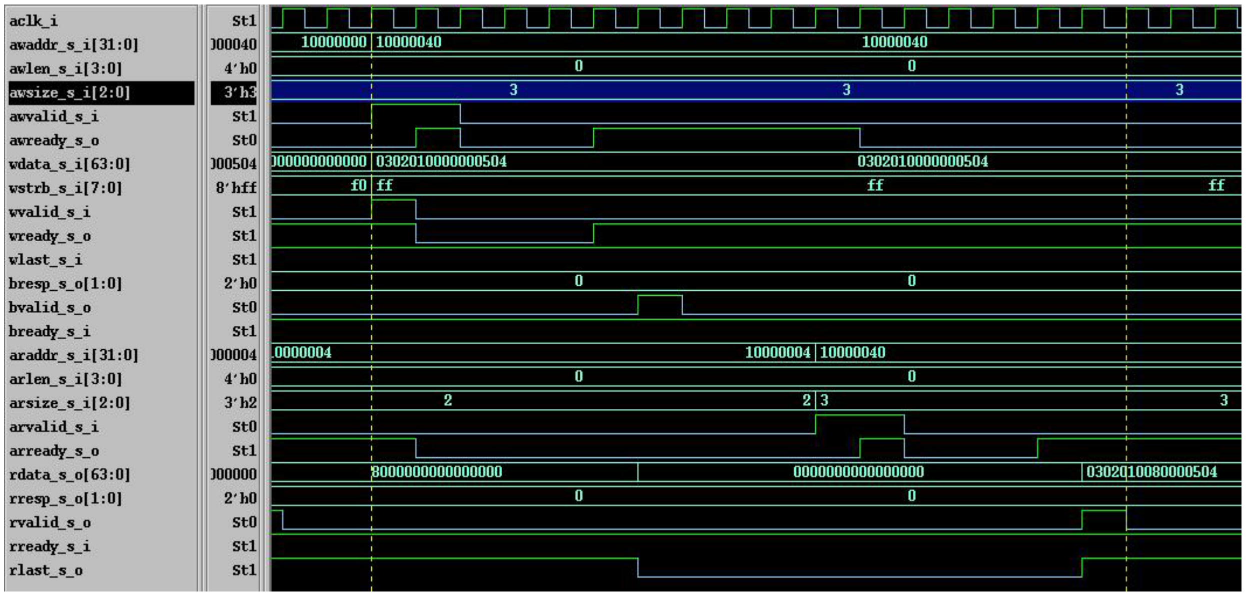
The AXI slave interface receives a write burst of length four, which gets accepted in seven clocks. The respective write response takes additional three clocks. Similarly, a single write transfer gets completed in four clocks, as shown in Figure 2-3.

Figure 2-3 Single Write and Read Transfer



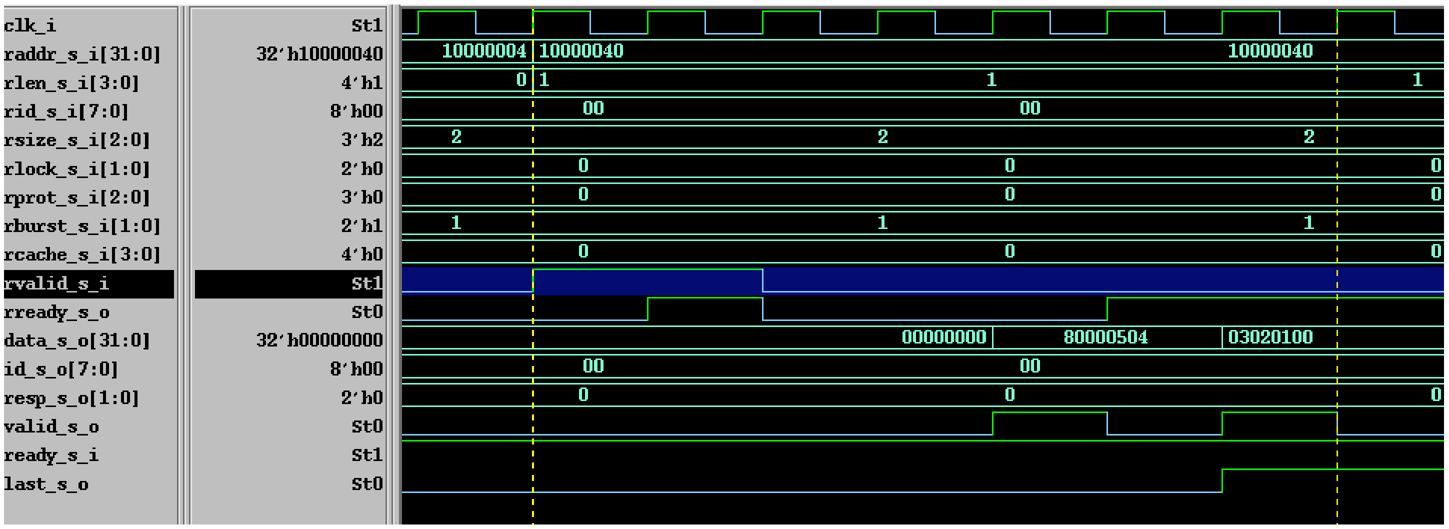
The AXI slave interface supports all types of byte, half-word, and word burst-size transfers. The latencies may vary for a 64-bit AXI slave interface, depending on the address offset and data bus width as shown in Figure 2-4

Figure 2-4 Latencies for a 64-bit AXI Slave Interface



In Figure 2-5, the AXI slave receives a read request of burst length 2. For the first data to be output on the AXI bus, the latency is four clocks. The remaining beats are output two clocks after the AXI master accepts the previous beat. In the AXI-read channel, the latencies depend on the burst length, data bus width, and the address offset.

Figure 2-5 AXI Slave Receives a Read Request of Burst Length 2



Note The EQOS-AXI configuration supports the little-endian and byte-invariant big-endian modes on the AXI slave ports. For more information about the supported endian modes, see "Endian Support" on page 1397.

2.1.3 APB, APB3, and APB4 Slave Interface

The APB, APB3, or APB4 slave interfaces are supported in all configurations. You can enable an APB port to access the DMA, MTL, and MAC CSR space.

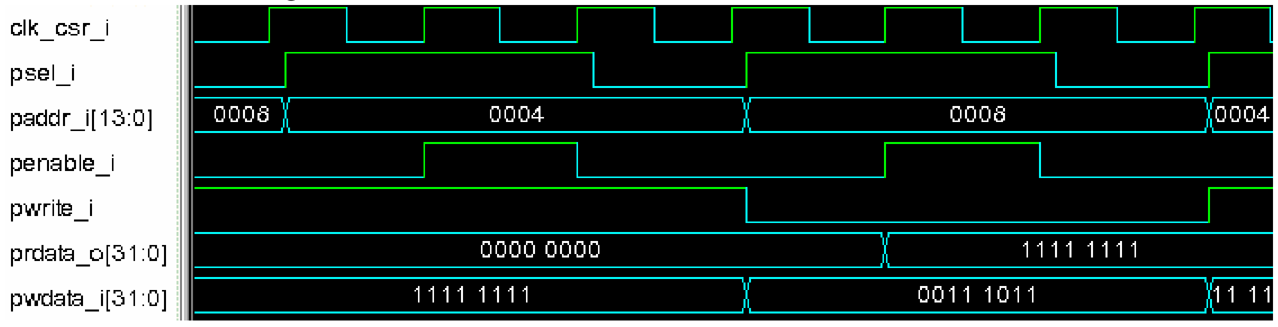
This topic discusses the following sections:

- "APB Port Timing" on page 46
- "APB3 Port Timing" on page 46
- "APB4 Port Timing" on page 47

2.1.3.1 APB Port Timing

Figure 2-6 shows a register write access followed by a register read access on the APB port. The APB port signals behave as per the AMBA 2.0 specification. All register accesses are completed in two clock cycles to support an APB port.

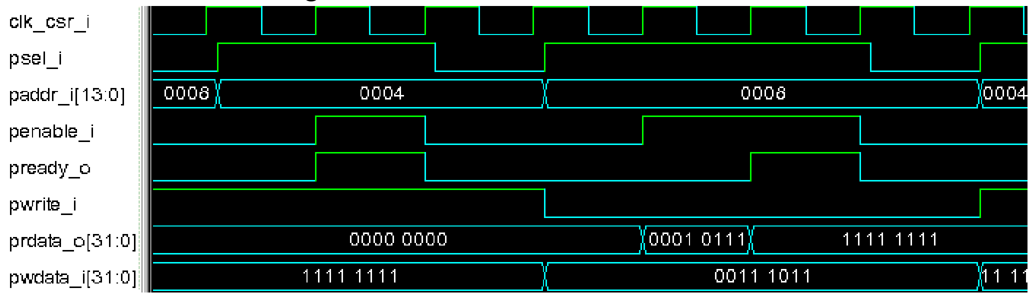
Figure 2-6 APB Port Timing



2.1.3.2 APB3 Port Timing

Figure 2-7 shows a register write access followed by a register read access on the APB3 port. The APB port signals behave as per the AMBA3 APB Protocol specification. Each write transfer completes in two clock cycles whereas each read transfer completes in three clock cycles.

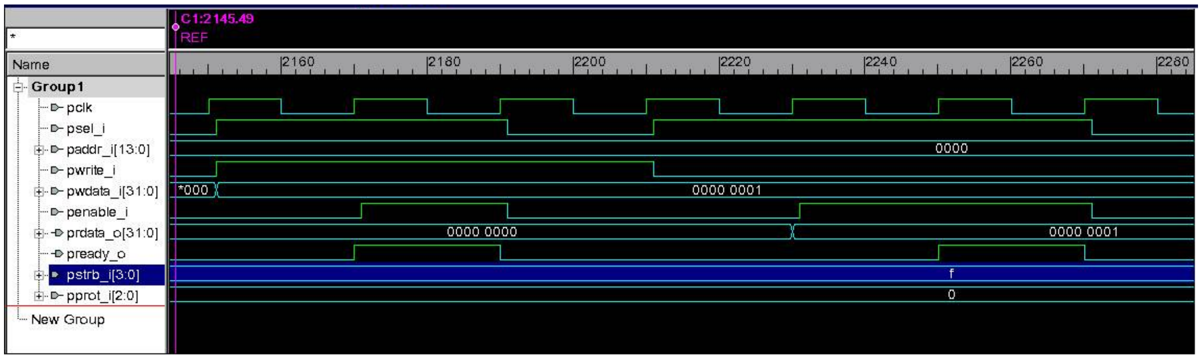
Figure 2-7 APB3 Port Timing



2.1.3.3 APB4 Port Timing

Figure 2-8 shows a register write access followed by a register read access on the APB4 port. The APB port signals behave as per the AMBA4 APB Protocol specification. Each write transfer completes in two clock cycles whereas each read transfer completes in three clock cycles. The pstrb_i signal enables the byte to be written.

Figure 2-8 APB4 Port Timing



2.1.4 MAC Control Interface

The MAC Control Interface (MCI) is supported in EQOS-CORE, EQOS-MTL, and EQOS-DMA configurations. This interface connects the application with the Control and Status register (CSR) module of the Media Access Controller to provide the control path for programming the MAC registers. The application initiates a register write or register read by asserting the command valid signal mci_val_i (with the valid data mci_wdata_i and valid byte enables mci_be_i), along with the register address on mci_addr_i. The mci_rdwn_i control signal indicates a Write (1'b0) or Read (1'b1) operation. The application assumes a successful data transfer when it receives an acknowledgment (mci_ack_o) asserted for one clock cycle. The CSR module drives the Read data on mci_rdata_o. The application can de-assert the command valid after receiving the acknowledgment.

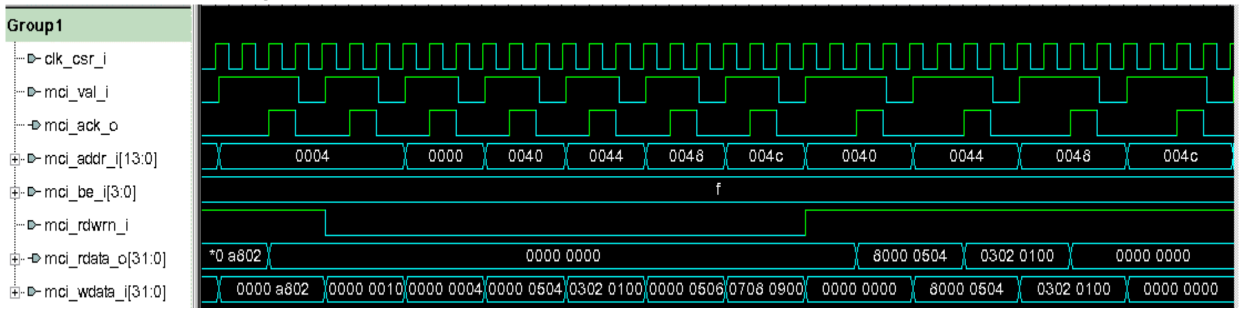
If the application continues to assert the command valid signal (on receiving an acknowledgment), the MAC considers it as the next access and responds to it by asserting the mci_ack_o after one clock cycle. The MAC completes the access to reserved registers with dummy read or write operations. This means that the write operations to the reserved addresses have no effect and the read operations are completed with an all-zero data output on the mci_rdata_o bus.

All write accesses to the CSR complete in two clock cycles and read accesses to the CSR complete in three clock cycles. The mac_ack_o signal is asserted after a delay of one clock cycle after the mci_val_i signal goes high. The mci_rdata_o signal is registered. Therefore, there is an additional delay of one clock in assertion of the mci_ack_o signal. Therefore, each read access is completed in three clock cycles for such configurations.

2.1.4.1 MCI Port Timing

Figure 2-9 shows read and write accesses on the MCI port.

Figure 2-9 MCI Port Timing



The flow shown in Figure 2-9 is as follows:

1. The first access to address 0x0004 is a single read cycle which completes in three clock cycles. The mci_val_i signal and the corresponding address and control signals should be kept asserted for the whole duration of the transaction.
2. The second and third accesses are single write operation cycles to addresses 0x0004 and 0x0000, respectively. The mci_ack_o signal is asserted one clock after the mci_val_i signal is asserted. The write operations are completed in two clock cycles.
3. The following accesses are burst write and burst read operations starting from address 0x0040.

   Each individual write or read transaction in the burst is identical to the single write or read transactions with respect to timing.

2.2 Application Master Interface

The DMA Controller interfaces with the application through one of the following interfaces:

- "AHB Master Interface" on page 49
- "AXI Master Interface" on page 54
- "AXI4 Master Interface" on page 60
- "AXI Low-Power Interface" on page 60

2.2.1 AHB Master Interface

In the EQOS-AHB configuration, the DMA controller interfaces with the application through the AMBA AHB interface. The AHB master interface controls the data transfers and the AHB slave interface accesses the CSR space.

The DMA can be used in the 32-bit, 64-bit, and 128-bit embedded applications where the DMA is required to optimize the data transfer between the MAC and the system memory. For information about the data transfers in little-endian and big-endian modes, see "Area and Power" on page 1387.

This section covers the following topics:

- "AHB Master Interface Features" on page 49
- "Burst Length Modes" on page 50
- "Aligning AHB Burst Transfers to An Address Value" on page 51
- "AHB Memory Read" on page 51
- "AHB Memory Write" on page 52
- "Early Burst Termination" on page 52
- "Error Response and Fatal Bus Error Interrupt" on page 52
- "Split or Retry Response" on page 53
- "Descriptor Write Back" on page 53

2.2.1.1 AHB Master Interface Features

The AHB master interface converts the internal DMA request cycles into AHB cycles. The AHB master interface has the following features:

- The AHB master interface is AMBA 2.0-compliant AHB master with no restrictions
- The AHB master interface supports the following burst length modes: fixed, unspecified, and mixed. These modes are explained in the "Burst Length Modes" on page 50.
- The AHB slaves interfacing with the AHB master support the SINGLE and INCR transfers.
- The AHB master interface can handle the AHB SPLIT, RETRY, and ERROR conditions. Any ERROR response halts all further transactions for that DMA and indicates the error as fatal through CSR and interrupt. The application can choose to give a hard or soft reset to the module to restart the operation.
- The AHB master interface can handle the AHB 1K boundary breaking.
- The AHB master interface can handle all 32-bit, 64-bit, or 128-bit data transfers (according to the data bus width configuration), except for Descriptor Status Write accesses (which may be 32-bit).

  In any burst data transfer, the address bus value is always aligned to the data bus width and need not be aligned to the beat size.

- The AHB master interface can align all AHB burst transfers to an address value. For more information, see "Aligning AHB Burst Transfers to An Address Value" on page 51.
- The DMA requests an AHB Burst Read transfer only when it can completely accept the received burst data. The data read from the AHB is always pushed into the DMA without any delay or Busy cycles. However, when multiple channels are selected, there may be some delay and Busy cycles may be inserted because of the shared-memory architecture for the channels.
- The DMA requests an AHB Burst Write transfer only when it has the sufficient data to completely transfer the burst. The AHB interface always assumes that it has the data available to push into the AHB bus. However, the DMA can prematurely indicate end-of-valid data (because of the transfer of EOP of an Ethernet packet) during the burst.

  In Fixed Burst Length mode, the AHB master interface continues the burst with dummy data until the specified length is completed. In INCR mode, it ends the burst transfer prematurely.

2.2.1.2 Burst Length Modes

The AHB master interface supports the following burst length modes:

- Fixed Burst Length

  In fixed burst length mode, the AHB master always initiates a burst with SINGLE, INCR4, INCR8, or INCR16 type. However, when such a burst is responded with SPLIT, RETRY, or Early Burst Termination (EBT), by default, the AHB master again initiates the pending transfers of the burst with INCR or SINGLE burst length type. When only one beat is remaining, the AHB master performs a SINGLE transfer. If more than one beat is remaining, the AHB master performs an INCR transfer. The AHB master terminates such INCR bursts when remaining beats of the original requested fixed burst are transferred.

  To ensure that the AHB master always performs the INCRx and SINGLE transfers even when a part of the burst is initiated again after a SPLIT, RETRY, or EBT response, use the RIB bit of the DMA_Mode register. However, when you enable this mode, the pending burst may further get split into multiple transfers as per the following rules before any new transfer is initiated:

  - If the DMA requests a burst transfer that is not equal to INCR4, INCR8, or INCR16, the AHB application interface splits the transfer into multiple burst transactions. For example, if the DMA requests a 15-beat burst transfer, the AHB interface splits it into multiple transfers of INCR8 and INCR4, and three SINGLE transactions.
  - If the burst length is more than 16, the AHB master interface splits it into multiple burst transfers. For example, if the burst length is 32 and the DMA starts a 32-beat transfer, the AHB master splits the burst into two INCR16.

- Unspecified Burst Length

  In unspecified burst mode, the AHB master always initiates a transfer with INCR and completes the DMA requested burst in one transfer.

- Mixed Burst Mode

  In mixed burst mode, the AHB master always initiates the bursts with fixed-size (INCRx) when the DMA requests transfers of size less than or equal to 16 beats. When the DMA requests bursts of length more than 16, the AHB master initiates such transfers with INCR and completes the request in one transfer.

  To ensure that the pending burst of a transfer is rebuilt only with INCRx transfers when an INCRx transfer is interrupted by RETRY, SPLIT, or EBT response, use the RB bit of the DMA_SysBus_Mode register. However, if an INCR burst initiated in this mode is interrupted, the AHB master rebuilds the burst only with INCR transfers.

You can choose the burst modes by programming the FB bit in the DMA_SysBus_Mode register.

Note The AHB master does not generate the WRAP bursts, and locked or protected transfers. Therefore, the DWC_ether_qos does not have the HLOCK or HPROT output signals.

2.2.1.3 Aligning AHB Burst Transfers to An Address Value

The AHB master interface can align all AHB burst transfers to an address value.

You can enable aligning all AHB burst transfers to an address value, by using the AAL bit in the DMA_SysBus_Mode register. If Bit 0 and Bit 12 of DMA_SysBus_Mode register are set to1, the AHB interface and the DMA ensure that all initiated beats are aligned to the address, completing the packet transfer in the minimum number of required beats.

For example, in 32-bit data bus mode, if the start address of a data buffer transfer is 0xF000_0008 and the DMA is configured for a maximum beat size of 16, the AHB transfers occur in the following sequence:

1. Two SINGLE transfers at addresses 0xF000_0008 and 0xF000_000C
2. One INCR 4 transfer at address 0xF000_0010
3. One INCR 8 transfer at address 0xF000_0020
4. All subsequent beats are INCR 16 transfers starting from address 0xF000_0040

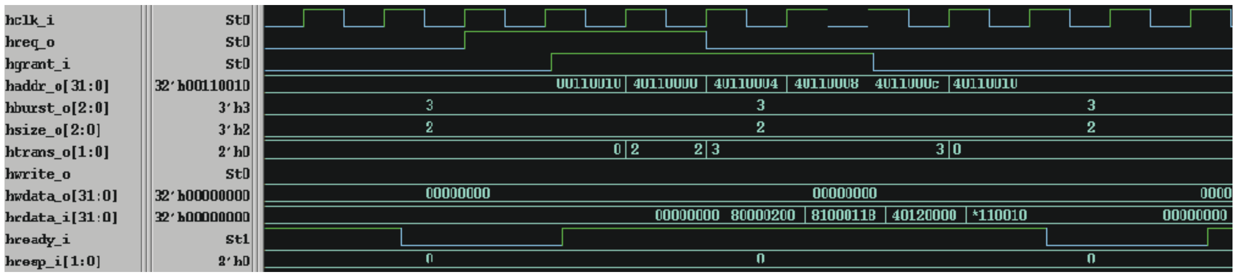
   For an address-aligned INCR 16 transfer, the six least-significant bits of the address must be 0.

Similarly, the AHB interface and the DMA split such unaligned address beats for a 64-bit (or 128-bit) data bus and initiate INCR16 beats only when the seven (or eight) least-significant bits of the address have a zero value.

2.2.1.4 AHB Memory Read

Figure 2-10 shows the AHB master performing a memory read starting from address 0x40110000. The AHB master port uses the INCR 4 burst in this scenario.

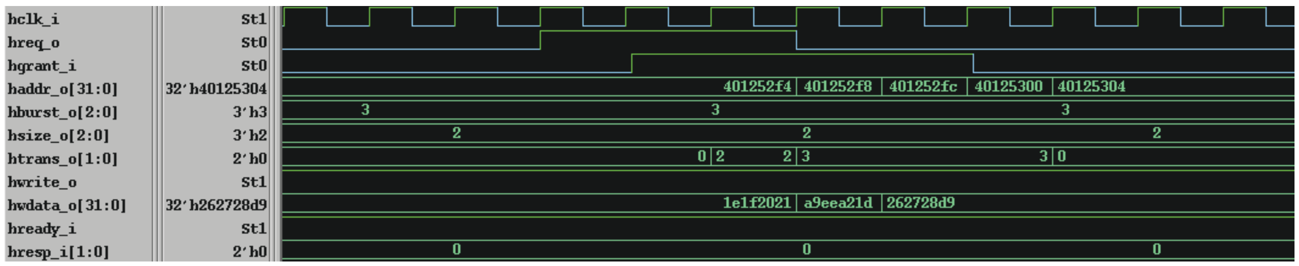
Figure 2-10 AHB Memory Read Timing



2.2.1.5 AHB Memory Write

Figure 2-11 shows the AHB master performing a burst memory write transaction, starting from address 0x401252F4. This data transfer corresponds to an Ethernet packet being written to the system memory by the Rx DMA. The AHB master uses the INCR4 burst. The data bus does not change values for last two beats in the INCR4 transaction. This is because the DMA to the AHB master port internally gave the EOP of the packet with the second beat transfer. Because the AHB master started a fixed INCR4 burst, it completes the burst with dummy data (same data) for last two beats.

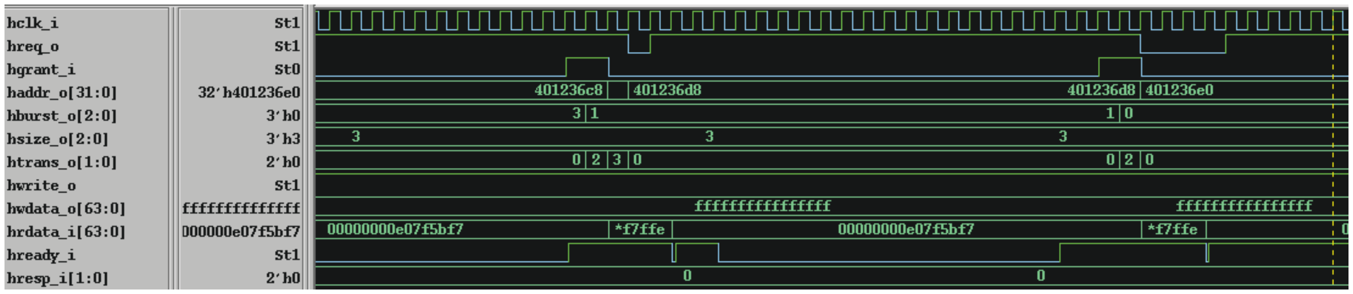
Figure 2-11 AHB Memory Write Timing



2.2.1.6 Early Burst Termination

Figure 2-12 shows an early burst termination occurring when the AHB master carries out an INCR (Write) transaction. This figure shows how the AHB master requests the bus grant again and completes the transfer with a SINGLE transaction.

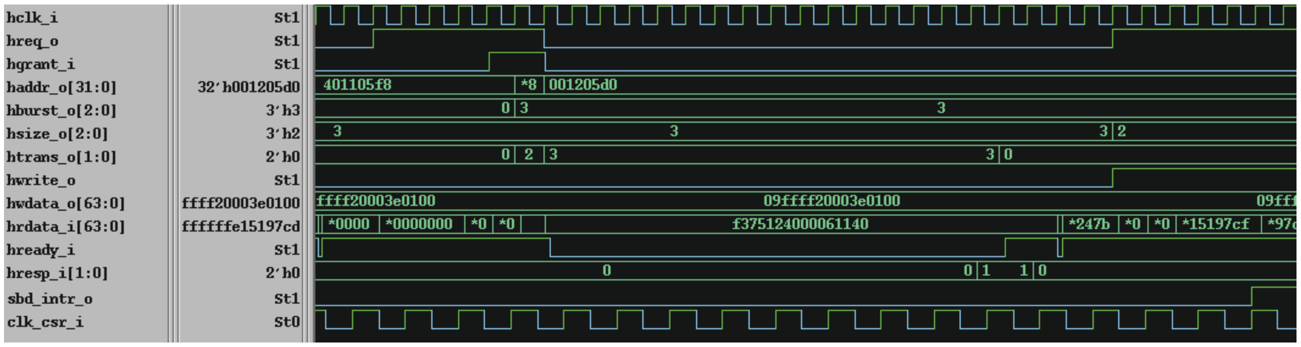
Figure 2-12 Early Burst Termination Timing



2.2.1.7 Error Response and Fatal Bus Error Interrupt

Figure 2-13 shows the behavior of the AHB master when an error response is received. The DUT tries to carry out the INCR 4 transaction. However, it receives a two-cycle error response during the data phase of the first beat. The sbd_intr_o interrupt is asserted as a result of this response. For more information about the interrupt functions, see "Interrupts" on page 145.

Figure 2-13 Error Response and Fatal Bus Error Interrupt Timing

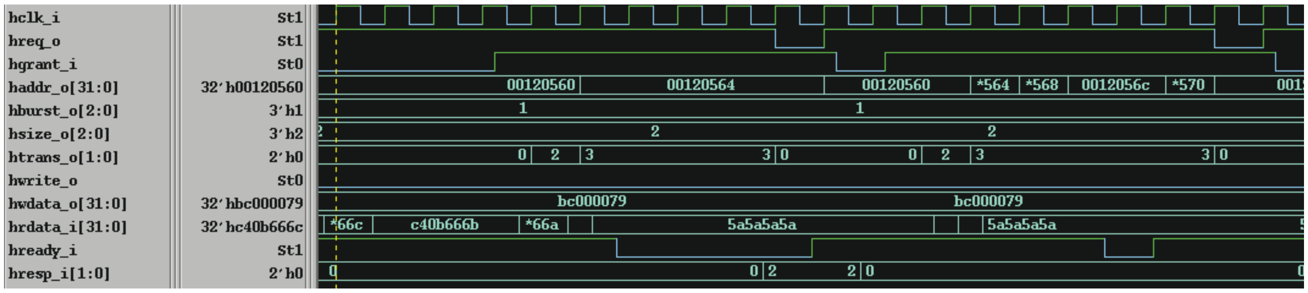


2.2.1.8 Split or Retry Response

Figure 2-14 shows how the AHB master port reacts to a split or retry response (to address 0x00120560) during a transaction. The AHB master receives a two-cycle retry response and immediately reconstructs the transaction and requests for the bus by again asserting the hreq_o signal. The hreq_o signal is de-asserted only for one cycle when the retry or split response is sampled in the first cycle of the two-cycle response.

The behavior remains same for Split response but the second request (hreq_o) from this master is masked in the AHB arbiter until the corresponding AHB slave is ready with the data.

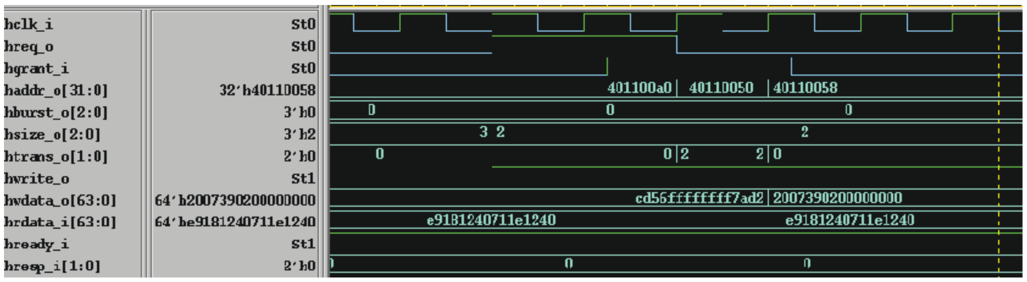
Figure 2-14 Split/Retry Response Timing



2.2.1.9 Descriptor Write Back

Figure 2-15 shows a descriptor being closed on a 64-bit AHB at address 0x40110050. For a big-endian configuration, only 32-bit status is written back to the memory and this appears on upper 32-bits of the hwdata_o bus. For a little-endian configuration, the 32-bit descriptor data appears on lower 32 bits of 64-bit AHB bus.

Figure 2-15 Descriptor Write-Back Timing



2.2.2 AXI Master Interface

In the EQOS-AXI configuration, the DMA controller interfaces with the application through the AMBA 3 AXI interface. The AMBA 3 AXI bus interface provides the characteristics to support highly-effective data traffic throughput. The system bus utilization is maximized by allowing simultaneous read and write transfers initiated from different DMA channels. For example, the Tx descriptor read and Rx descriptor write-back (status update) operations can happen simultaneously. However, the Tx descriptor read and Tx descriptor write-back operations cannot happen simultaneously because the Tx DMA (or Rx DMA) does not initiate the next transfer unless the previous one is complete.

The AXI master interface allows multiple numbers of burst requests to be queued for Read or Write operation. The system bus utilization can be further enhanced by supporting Requests reordering and Data interleaving. This gives the AXI master the flexibility to dynamically and independently select multiple slaves for handling Priority Requests and Slow peripheral slaves. The EQOS-AXI configuration also supports the low-power interface defined in the AXI specification.

Note You can select AXI4 compatible interface also. This interface has additional signals and capabilities as compared to the AXI3 signal interface.

2.2.2.1 Burst Splitting and Burst Selection

In DWC_ether_eqos, the AXI master can split the DMA requests into multiple bursts, in the AXI system bus.

The AXI master splits the DMA requests into multiple bursts on the AXI system bus. Splitting is based on the DMA count, the software-controllable burst enable bits (FB, BLEN256, BLEN128, BLEN64, BLEN32, BLEN16, BLEN8, BLEN4), and the software-controllable burst types (INCR and INCR_ALIGNED). For more information, see DMA_SysBus_Mode register.

The AXI master also handles splitting the transfers that cross 4 KB address boundary, with an option to limit it to 1-KB address boundary. The Burst Length Select Priority is in the following sequence: FB, BLEN256, BLEN128, BLEN64, BLEN32, BLEN16, BLEN8, and BLEN4.

Note The DWC_ether_qos supports the burst length of up to 256 beats in AXI3 and AXI4 modes although the maximum burst length specified in the AXI3 standard is 16 beats. You can configure the maximum burst length (16, 32, 64, 128, or 256) allowed on the AXI bus by using the AXI Maximum Burst Length option in coreConsultant. This feature is provided for the systems which require larger than 16-beats burst length for optimum utilization of the bus bandwidth.

2.2.2.2 INCR Burst Type

If FB is disabled, the AXI master always chooses the maximum allowed burst length based on the BLEN256, BLEN128, BLEN64, and BLEN32 bits (possible maximum burst lengths are 256, 128, 64, 32, or 16). The AXI master chooses a burst length of any value less than the maximum enabled burst length (all lesser burst length enables are redundant) in the following cases:

- The DMA requests are not multiples of the maximum allowed burst length
- The 4K/1K address boundary crossing condition is enabled

For example, when only BLEN64, BLEN16, and BLEN4 are enabled, and the DMA requests a burst transfer of 102 beats size, the AXI master splits it into two bursts of 64 and 38 beats size, respectively.

If FB is enabled, the burst length is based on the priority of the enabled bits in the following order: BLEN256, BLEN128, BLEN64, BLEN32, BLEN16, BLEN8, and BLEN4. When the DMA requests to transfer a burst, the AXI interface splits the requested bursts into multiple transfers by using only the enabled burst lengths. This splitting can occur in following cases:

- The requested burst is not a multiple of the maximum enabled burst
- The 4K/1K address boundary crossing condition is enabled

If the AXI interface cannot choose any of the enabled burst lengths, it selects the burst length as 1. For example, if only BLEN64, BLEN16, and BLEN4 are enabled and the DMA requests a burst transfer of size102 beats, the AXI interface splits it into multiple bursts of size 64, 16, 16, 4, 1, and 1 beats, respectively. The sequence is in decreasing burst lengths.

2.2.2.3 INCR_ALIGNED Burst Type

When the address-aligned burst type is enabled, then in addition to the burst splitting conditions explained in INCR Burst Type, the AXI interface splits the DMA requested bursts such that each burst length is aligned to the least significant bits of the start address. The AXI interface initially generates smaller bursts so that the remaining transfers can be transferred with the maximum possible (enabled) fixed burst lengths. For example, in the same setting as explained earlier for FB (BLEN64, BLEN16 and BLEN4 are enabled), the DMA requests a burst of size 102 beats at the start address of 0x0000_03A4 (32-bit bus). The AXI master starts the first transfer with size 23 such that the address of the next burst is aligned (0x0000_0400) for a burst of 64. Therefore, the sequence of bursts is 23, 64, and 15, respectively.

When FB is set, the sequence of burst transfers is 1, 1, 1, 4, 16, 64, 4, 4, 4, 1, 1, and 1 in same scenario, respectively. The sequence of smaller bursts at the beginning aligns the address to the next higher enabled burst lengths programmed in the register.

2.2.2.4 Outstanding Transactions

The AXI master supports configurable outstanding read/write requiests on the AXI bus. You can control the outstanding requests by programming the DMA_SysBus_Mode register.

The AXI master supports up to 4, 8, or 16 (configurable in coreConsultant) outstanding Read or Write requests on the AXI bus. You can also control these outstanding requests through software by programming Bits[24:21] and Bits[19:16] in the DMA_SysBus_Mode register. This is a useful workaround for any system-level issues with handling of multiple requests.

Note The maximum number of outstanding requests is divided equally between the requests generated by the two internal masters (Rx DMA and Tx DMA). This means that when you select eight outstanding AXI requests while configuring the core, each DMA is limited to a maximum of four outstanding requests in each Read or Write channel. When you reduce the outstanding request (for example, to four) by programming the respective control bits, all outstanding requests can be from the same DMA. The minimum number of outstanding requests per channel is two.

2.2.2.5 Priority of AXI Requests

The descriptor transfers have higher priority than the data transfers. Therefore, if there are two requests, Rx Descriptor Read and Tx Data Read, the Rx Descriptor Read is given higher priority so that the next Rx Data Write (subsequent to Rx Descriptor Read) need not wait for the Tx Data Read transfer to complete. If there are descriptor read requests from both DMAs, these requests are serviced based on the first-come first-serve basis. The Rx DMA has higher priority if descriptor read requests, generated from both DMAs, are in the same clock. Similarly, in the write channel, descriptor writes from any DMA have higher priority than the data-write transfers for the Rx DMA.

When you enable the AV or DCB feature, the DMA may contain multiple channels. If the requests are generated from different channels simultaneously in the same clock, the request priority among the channels is: Channel 7 (if present), Channel 6,..., Channel 1, and Channel 0.

Note The Maximum outstanding requests on the AXI is limited to parameter DWCXG_AXI_GM_MAX_RD_REQUESTS and DWCXG_AXI_GM_MAX_WR_REQUESTS corresponding to Read and Write Channels respectively. You can limit this by programming RD_OSR_LMT and WR_OSR_LMT of DMA_SysBus_Mode register.

2.2.2.6 Bursts Reordering and Data Interleaving

The AXI protocol allows reordering of data transfers with different AXI-IDs with respect to the sequence of requests. It also allows data interleaving between transfers of different AXI-IDs for Read channels.

In EQOS-AXI configurations, the requests from each internal master (Rx DMA and Tx DMA) are generated with different AXI-ID. Therefore, reordering and data interleaving can be performed as two DMAs operate independently and are allocated different address space on the slave memory.

The AXI master supports reordering and interleaving from the AXI slaves on the Read channel. Each DMA internally ensures that a Read request and Write request is never generated at the same time to the same address (it can occur for only descriptor accesses). The Tx DMA ensures that Tx Descriptor Read, Tx Data Read, and Tx Descriptor Write-Back requests are generated only after the previous requests are complete.

The Rx DMA generates only Descriptor reads on the Read channel with a different ID. Therefore, data reordering and interleaving on the Read channel does not cause any problems in the MAC.

Note The AXI master does not support data reordering and interleaving on the Write channel which is compliant with the latest AXI 4 specification.

2.2.2.7 AXI-ID Translation

The AXI master selects unique IDs for each DMA channel and also for each Tx/Rx DMA channel. The AXI master selects the IDs as follows:

- AXI Master ID[0] = 0

  Rx DMA access for Read and Write
- AXI Master ID[0] = 1

  Tx DMA access for Read and Write
- AXI Master ID[3:1] = 0, 1, 2, 3, 4, 5, 6, or 7

  DMA channel number when multiple DMA channels are present, that is, when you select the AV or DCB feature. Otherwise, it is hardwired to zeros.

Note The AXI system bus ID width is configurable. The default width for the AXI master is 4 bits. The unused bits are hardwired to zeros. The default width for the AXI slave interface is 8 bits.

2.2.2.8 Endianess Support

The AXI master supports the little-endian and byte-invariant big-endian modes on the AXI master ports. The AXI master does not support narrow transfers and always performs full data-width transfers. For more information about the supported endian modes, see “Endian Support”.

2.2.2.9 Posted Writes

In posted writes, the write channel of AXI master transfers an OKAY response to the DMA as soon as the last cycle or beat of data is accepted by the AXI interconnect. The AXI master sends this response without waiting for the response from the target AXI slave on the write response channel. In non-posted writes, the AXI master interface transfers the response received from the AXI slave channel to the DMA.

For transferring the Ethernet data to system memory, the Rx DMA always issues posted write requests to the AXI master interface. This enables pipelining of the data requests without delays. The AXI master does not support out-of-order write transfers. However, the Rx DMA ensures that the sequence is maintained for descriptor writes, descriptor reads, and data transfers.

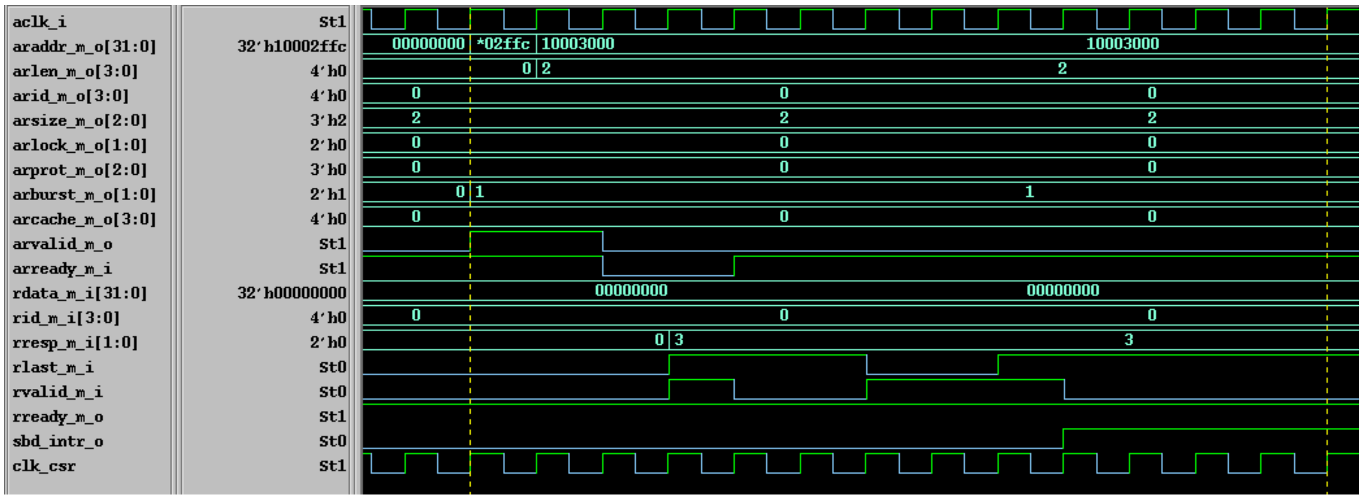
For writing descriptors (status or timestamp), the DMA always issues the non-posted write requests. This is needed because the Transfer Complete Interrupt is generated based on the descriptor writes for which the DMA needs completion response from the memory slave. This ensures no race condition between the hardware and the software because the interrupt is generated only after the data and descriptor are written to the slave memory.

Note For data transfer to system memory, the Rx DMA always issues posted writes. So, after the data is accepted by AXI interconnect, DMA moves out of Buffer data write state to descriptor write state without waiting for the response from the target AXI slave, on the write response channel. Therefore, after the ERROR response for the previous buffer write is received, DMA initiates a Descriptor Write and captures ERROR status (received during Descriptor Write) as bus error, into the CSR space.

2.2.2.10 Error Response Handling

Whenever there is an error response from the AXI slave (as shown in Figure 2-16 with rresp_m_i[1:0] = 3), the respective DMA channel which generated the request gets disabled. The AXI master asserts the interrupt (sbd_intr_o in Figure 2-16) when the corresponding "Fatal Bus Error" interrupt is enabled. Only the DMA whose specific transaction had the error response gets affected and goes to STOP state. The application can re-initialize that specific DMA and restart the operation. The traffic on the other channels does not get affected if an error is detected for a DMA Channel.

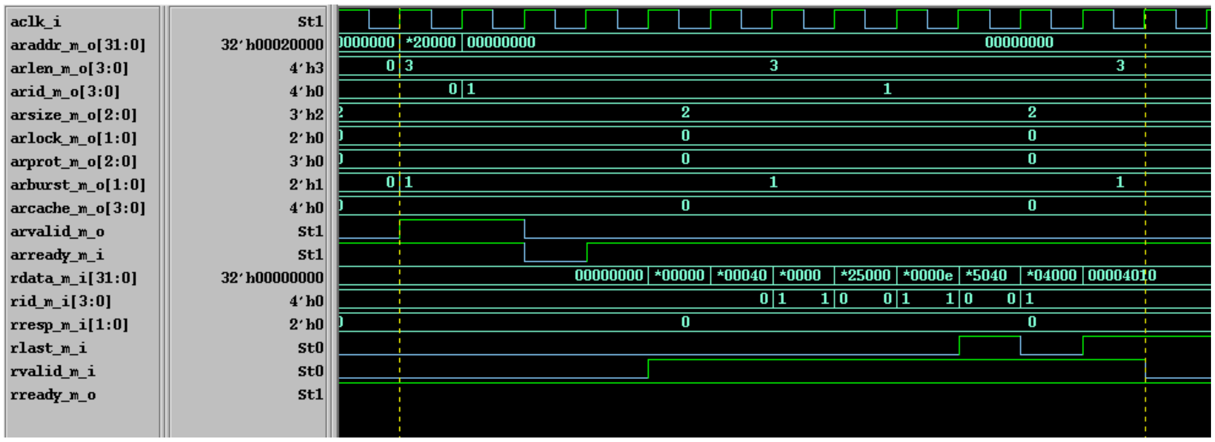
Figure 2-16 Error Response Timing



2.2.2.11 AXI Memory Read

Figure 2-17 shows the AXI master read timings. Two requests are placed for ID 0 and ID 1. The Read data is interleaved by the interconnect (by way of ID 0 and ID 1) and received by the AXI master. The Tx DMA in AXI master always ensures that it has enough space in its FIFO to accept the requested burst of data. Therefore, the r_ready_m_o response is always high.

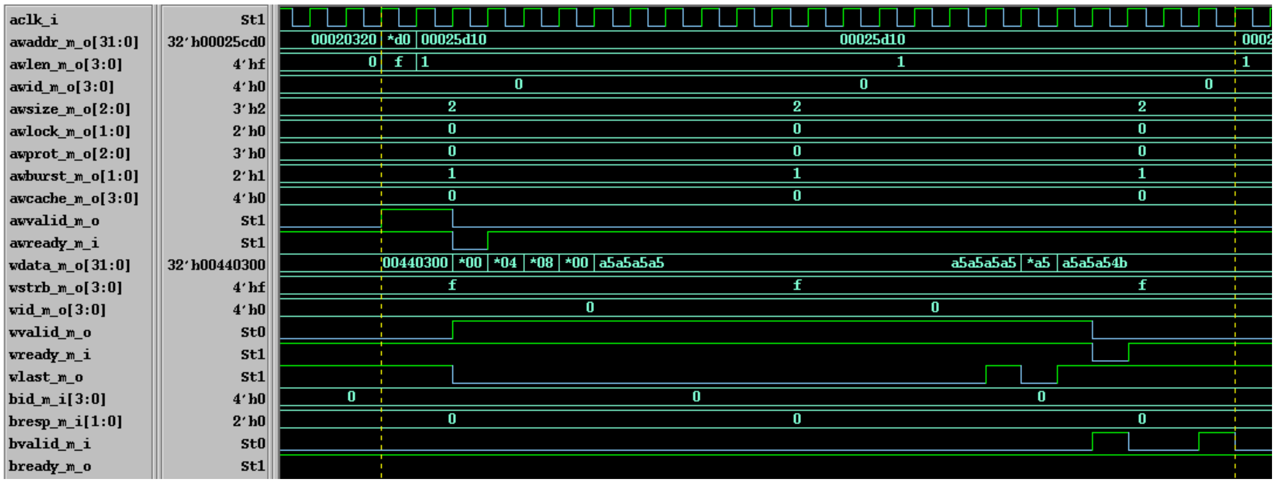
Figure 2-17 AXI Memory Read Timing



2.2.2.12 AXI Memory Write

Figure 2-18 shows the AXI write interface diagram. There are two requests for ID0. These requests are issued with burst lengths of 16 and 2. The latency between the first write request and the corresponding first data is two clocks. The Rx DMA ensures that it always has the requested burst of data available in its FIFO before the request is generated. Therefore, there are no further latencies or delays in the data transfer. The complete 18 beats of write data of the two requests are transferred in 18 clocks, provided there is no delay in the acceptance of that data by the AXI slave, as shown in Figure 2-18.

Figure 2-18 AXI Memory Write Timing



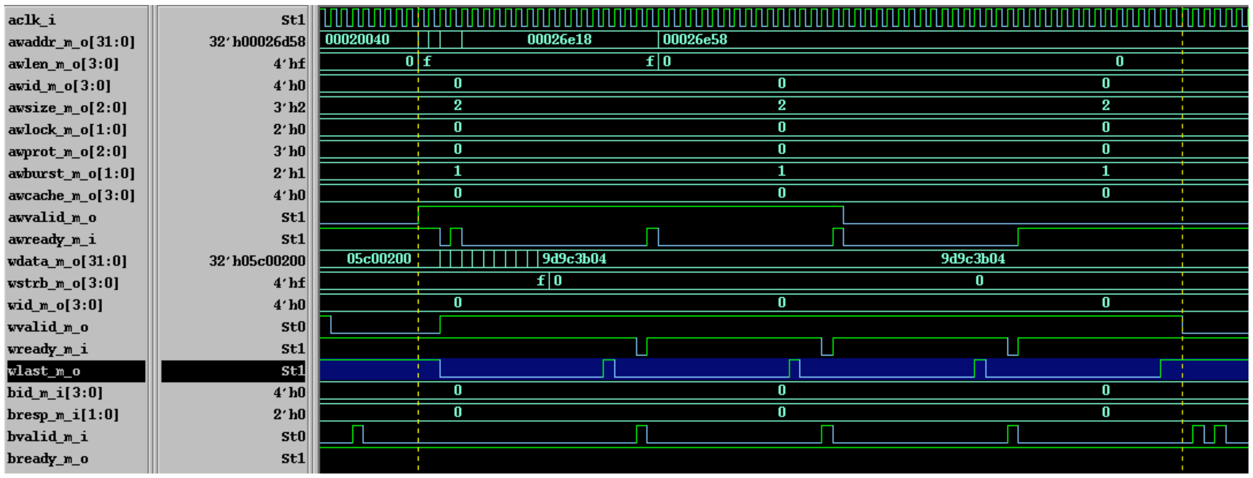
2.2.2.13 AXI Early Burst Termination

Figure 2-19 shows the early burst termination on Write channel. This can happen during data transfer when the EOP is read out of the Rx queue before the requested burst transfer is completed. Five requests are issued, but the EOP gets transferred in the middle of the first burst. The AXI master still continues the data phases of the outstanding requested bursts but does not allow further data to be written to the slave memory by de-asserting the write-strobes (as indicated by wstrb_m_o[3:0] = 0).

Two extra request commands (@address 0x00026e18 and 0x00026e58) are active when the write-strobes are de-asserted after EOP transfer. The first request has already been issued to the slave. Therefore, it is not changed. The second request is a dummy request with a single beat required to generate a last data transfer (with wlast high).

Note After transferring the EOP data to the system memory, a lot of AXI bandwidth may be wasted because of dummy writes. Therefore, it is recommended to have two or less than two outstanding requests on the AXI master write channel by appropriately programming DMA_SysBus_Mode.

Figure 2-19 AXI Early Burst Termination Timing



2.2.3 AXI4 Master Interface

In the EQOS-AXI4 configuration, the DMA controller interfaces with the application through the AMBA AXI4 interface. The EQOS-AXI4 configuration is same as EQOS-AXI configuration except for features listed in this section.

The EQOS-AXI4 configuration provides additional interface signals AxQOS (arqos_m_o and awqos_m_o) and AxDOMAIN (ardomain_m_o and awdomain_m_o).

Note

- The wid_m_o signal can be left unconnected in EQOS-AXI4 configurations.
- If the system does not support ACE-Lite, AxDOMAIN (ardomain_m_o and awdomain_m_o) signals can be left unconnected in EQOS-AXI4 configurations.

2.2.4 AXI Low-Power Interface

The EQOS-AXI configuration supports the low-power interface defined by the AXI specification. Bits[31:30, 10] of DMA_SysBus_Mode register control the behavior of the AXI master in the low-power mode. When Bit31 (EN_LPI) is set, the AXI Master enables the low-power feature. The setting of Bit[10] determines if the AXI Master can also initiate the entry to low-power state. Bit30 controls the low-power exit due to reception of packets from the line.

This means that the AXI master accepts the low-power request and goes into the low-power mode in which the application can remove the AXI clock when the following conditions are true for a duration equal to the value programmed in the AXI_LPI_Entry_Interval register.

- The MAC transmitter and receiver are in the Idle mode
- Both DMA engines are in Idle mode
- There is no data in the Transmit or Receive data path or queue
- Slave interface is idle

If the MAC is in the active state (not idle) at the time of low-power initiation request, the AXI master waits for a period of 16 AXI clock cycles. If the MAC is still in the active state at the end of this period, the low-power request is denied.

If the MAC goes to idle (inactive) within 16 clock cycles, the AXI master monitors the inactive state for a further period indicated by the AXI_LPI_Entry_Timer. If the MAC remains inactive for this period, the low-power request is accepted and acknowledged by going into the low-power state. Otherwise (if the MAC becomes active before the expiry of this interval), the request is denied.

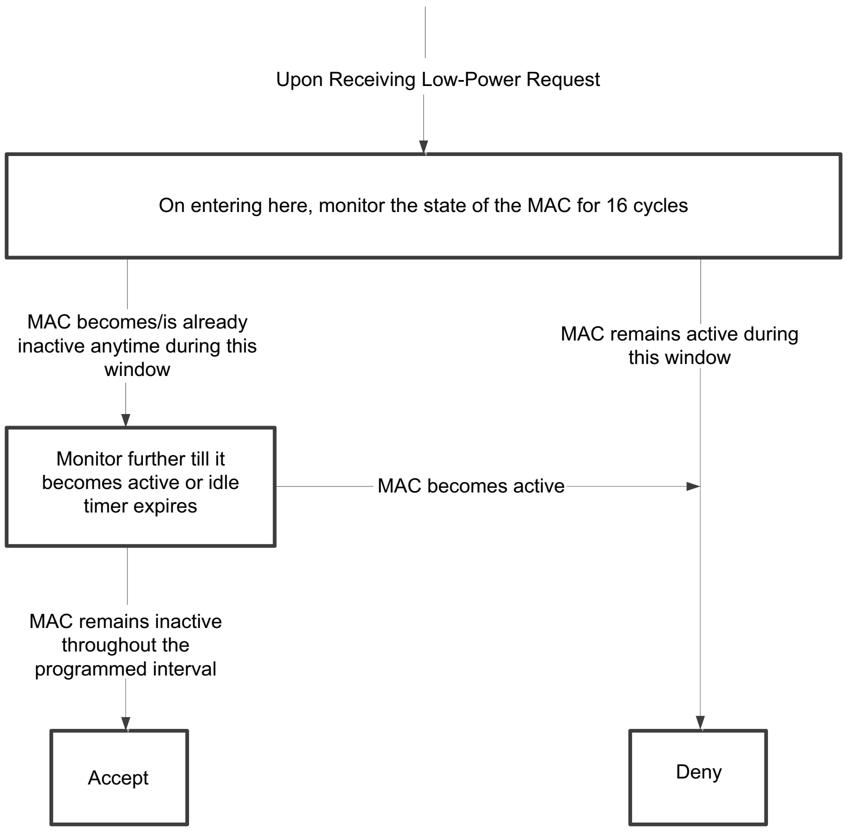
After the MAC goes into low-power state and acknowledges it (csysack_o is made low), the AXI system can gate-off or remove the AXI clock when cactive_o is made low,

Note The timing and behavior of the EQOS-AXI low-power interface signals is according to the Figure 12-2 of the AXI Specification.

2.2.4.1 AXI System Initiated Low-Power Entry

The AXI system initiates a low-power mode request to the AXI master by setting csysreq_i to 0. When EN_LPI is set, the AXI master checks the active state of the data paths within the controller before giving a response.

Figure 2-20 AXI Low-Power Entry Flow


Upon Receiving Low-Power Request
On entering here, monitor the state of the MAC for 16 cycles
MAC becomes/is already inactive anytime during this window
MAC remains active during this window
Monitor further till it becomes active or idle timer expires
MAC becomes active
MAC remains inactive throughout the programmed interval
Accept
Deny

2.2.4.2 Automatic AXI Low-Power Entry

In addition to the system initiated AXI low power entry (where DWC_ether_qos enters low power upon request from system controller), DWC_ether_qos also supports AXI low-power entry when there is no activity within the controller for a configured interval of time. This feature makes the DWC_ether_qos to automatically (that is, without any request from system controller) enter into AXI low-power state when the idle condition is reached.

This mode is enabled when AALE bit of DMA_SysBus_Mode Register (Bit 10) is set. The minimum idle period before the AXI low-power request is triggered by the MAC is programmed in the AXI_LPI_Entry_Interval register.

In this mode, when the MAC is in the inactive state (as mentioned in“AXI Low-Power Interface” on page 60) for the programmed duration, it automatically initiates the AXI low-power mode by making cactive_o low. The AXI system can gate-off (disable) the AXI clock in this state.

IDLE timer is the timer that gets loaded with the programmed interval when MAC is active. IDLE timer decrements when MAC is IDLE.

2.2.4.3 Coming Out of Low-Power Mode

AXI Master comes out of low-power mode when one of the following occurs:

- AXI system restores the AXI clock and initiates the exit by removing the low-power request (csysreq_i is made high); the AXI Master acknowledges it (by making csysack_o and cactive_o high) and exits the low-power state.
- The MAC itself initiates a low-power exit by making cactive_o high when it receives a packet from the line. If Bit[30] of DMA_SysBus_Mode register is set, it initiates the exit only if a valid magic packet or remote wake-up packet is received. In this scenario the request signal assertion (cactive_o) is synchronous to the clk_rx_i clock because the AXI clock is assumed to be absent.
- If the CSR slave port is operates with a different clock than AXI Master clock, the MAC initiates the low-power exit when any read/write access is made to any of the DMA or MTL registers. In this scenario the request signal assertion (cactive_o) is synchronous to the clk_csr_i clock because the AXI (aclk_i) clock is assumed to be absent.

Note

In case of AXI system initiated exit (due to csysreq_i assertion), the clock controller should enable the clock immediately along with csysreq_i. The cactive_o & csysack_o assertion is synchronous to this clock

For MAC initiated low-power exits, the clock controller must restore the AXI clock immediately after the cactive_o is made high. Otherwise, it can affect the CSR register operations due to misbehavior of CDC synchronization circuit especially for back-2-back register writes.

2.3 DMA Controller

The DMA has independent Transmit (Tx) and Receive (Rx) engines, and a CSR space. The Tx engine transfers data from the system memory to the device port (MTL), whereas the Rx engine transfers data from the device port to the system memory. The DMA engine uses descriptors to efficiently move data from source to destination with minimal application CPU intervention. The DMA is designed for packet-oriented data transfers such as packets in Ethernet. The DMA controller can be programmed to interrupt the application CPU for situations such as Packet Transmit and Receive Transfer completion, and other normal or error conditions.

The DMA and the application communicate through the following two data structures:

- Control and Status registers (CSR)
- Descriptor lists and data buffers

The DMA supports up to 8 Tx and 8 Rx Descriptor lists (or DMA channels). The base address of each list is written to the respective Tx Descriptor List Address register and Rx Descriptor List Address register. The descriptor list is forward linked and the next descriptor is always considered at a fixed offset to the current one. The offset is controlled by the DSL field of DMA_Ch[n]_Control register. The number of descriptors in the list is programmed in the respective Tx (or Rx) Descriptor Ring Length register. Once the DMA processes the last descriptor in the list, it automatically jumps back to the descriptor in the List Address register to create a descriptor ring.

The descriptor lists reside in the physical memory address space of the application. Each descriptor can point to a maximum of two buffers in the system memory. This enables two buffers to be used, physically addressed, rather than contiguous buffers in memory.

Note

- When DWC_ether_qos is configured with an address greater than 32 bits, the DMA considers that all the descriptors in a list are present in a memory page with maximum size of 4GB. In other words, address bits greater than 32 are taken to be constant and equal for all the descriptors in the ring. However, the upper bits of the various buffer addresses can be different and independent.
- When DWC_ether_qos is operated with an address greater than 32 bits, the Tx Descriptor can have only one buffer address. However, the Rx Descriptor can still have two buffer addresses.

A data buffer resides in the application physical memory space and consists of an entire packet or part of a packet but cannot exceed a single packet. Buffers contain only data. Buffer status is maintained in the descriptor. Data chaining refers to packets that span multiple data buffers. However, a single descriptor cannot span multiple packets. The DMA skips to the data buffer of next packet when EOP is detected.

The DWC_ether_qos supports the ring structure for the DMA descriptor. For more information on the descriptor, see "Descriptors" on page 1315 that describes the descriptor structure and how the DMA accesses the descriptors.

The following sections discuss in detail the various operations of a DMA controller:

- "DMA Application Bus Burst Access" on page 65
  - "DMA Application Data Buffer Alignment" on page 66
  - "DMA Buffer Size Calculations" on page 66
  - "DMA Arbiter (EQOS-DMA and EQOS-AHB Configurations)" on page 67
- "Transmission" on page 68
  - "Transmit DMA Operation: Default (Non-OSP) Mode" on page 68

  - ❑ "DMA Transmit Operation: OSP Mode" on page 71
  - ❑ "Transmit Packet Processing" on page 73
  - ❑ "Transmit Polling Suspended" on page 73
  - ❑ "DMA Transmit Channel Arbitration" on page 74
- "DMA Receive Operation" on page 75
  - ❑ "Receive Descriptor Acquisition" on page 78
  - ❑ "Receive Packet Processing" on page 78
- "Interrupts" on page 145
  - ❑ "Periodic Scheduling of Transmit and Receive Interrupt" on page 149
  - ❑ "Per Channel Transfer Complete Interrupt" on page 149
- "Error Response to DMA" on page 79
- "DMA Native Interface" on page 79

2.3.1 DMA Application Bus Burst Access

The DMA engines attempts to transfer data in a burst of maximum size as programmed in the PBL fields of Transmit Control and Receive Control registers of the respective DMA. The Rx and Tx descriptors are always accessed in the maximum possible (limited by PBL or 16 * 8/bus width) burst length for 16 bytes to be read. The burst transfers initiated by the DMA can be split into multiple burst transfers as per the Application Interface protocol (AHB or AXI) requirements and the settings of DMA_SysBus_Mode register.

The Tx DMA initiates a data transfer only when sufficient space is available in the MTL Tx Queue to accommodate either of the following:

- Bytes corresponding to the configured burst (PBL * bus_width/8)
- Remaining bytes in Tx Buffer without EOP
- Number of bytes till EOP

The Rx DMA initiates a data transfer in the following conditions:

- Sufficient data is available in MTL Rx Queue to accommodate the configured burst
- EOP (when it is less than the configured burst length) is detected in the Rx Queue

The DMA indicates the start address and the number of transfers required to the AHB or AXI master interface. When the AHB or AXI Interface is configured for fixed-length burst, it transfers the data by using the best combination of INCR4, INCR8, or INCR16 and SINGLE transactions. If EOP is reached before the fixed-burst ends on the AHB or AXI interface, dummy transfers are performed in-order to complete the fixed-burst. Otherwise [Bit 0 of DMA_SysBus_Mode register is reset], the DMA transfers the data using INCR (undefined length) and SINGLE transactions.

When the AHB or AXI interface is configured for address-aligned beats, both DMA engines ensure that the first burst transfer initiated by the AHB or AXI is less than or equal to the size of the configured PBL. Therefore, all subsequent beats start at an address that is aligned to the configured PBL. The DMA can only align the address for beats up to size 16 (for PBL > 16) for AHB interface because it does not support more than INCR16.

Similarly, the DMA can only align the address for beats up to 256 size (for PBL > 256) because the AXI interface does not support more than BLEN 256.

2.3.1.1 DMA Application Data Buffer Alignment

The Tx and Rx data buffers do not have any restrictions on start address alignment. For example, in systems with 32-bit memory, the start address for buffers can be aligned to any of the four bytes. However, the DMA always initiates write transfers with address aligned to the bus width and dummy data (old data) in the invalid byte lanes. This typically happens during the transfer of the beginning or end of an Ethernet packet. The software driver should discard the dummy bytes based on the start address of the buffer and size of the packet.

Table 2-1 Application Data Buffer Alignment Examples

Examples	
Buffer Read	If the Tx buffer address is 32'h00000FF2 (for 32-bit data bus), and 15 bytes need to be transferred, the DMA reads five full words from address 32'h00000FF0, but when transferring data to the MTL Tx queue, the extra bytes (the first two bytes) are dropped or ignored. Similarly, the last 3 bytes of the last transfer are also ignored. The DMA always ensures that it transfers a full 32-bit data to the MTL Tx queue, unless it is the end of packet.
Buffer Write	If the Rx buffer address is 32'h0000FF2 (for 64-bit data bus) and 16 bytes of a received packet need to be transferred, the DMA writes 3 full words from address 32'h00000FF0. However, the first 2 bytes of the first transfer and the last 6 bytes of the third transfer have dummy data. The DMA considers the offset address only if it is the first Rx buffer of the packet. The DMA ignores the offset address and performs full word writes for the middle and the last Rx buffer of the packet.

2.3.1.2 DMA Buffer Size Calculations

The DMA does not update the size fields in the Tx and Rx descriptors. The DMA updates only the status fields (RDES and TDES) of the descriptors. The driver has to perform the size calculations.

The Tx DMA transfers the exact number of bytes (indicated by buffer size field of TDES2) towards the MAC. If a descriptor is marked as first (FD bit of TDES3 is set), the DMA marks the first transfer from the buffer as SOP. If a descriptor is marked as last (LD bit of TDES3), the DMA marks the last transfer from that data buffer as EOP to the MTL.

The Rx DMA transfers data to a buffer until the buffer is full or the end of packet is received from the MTL. When the FD bit of a descriptor is set, the amount of valid data in a buffer is accurately indicated by the buffer size field (programmed in DMA Channel Receive Control register) minus the data buffer pointer offset. The offset is zero when the data buffer pointer is aligned to the data bus width. If a descriptor is marked as last, the buffer may not be full (as indicated by the buffer size in Bits[14:1] of Receive Control register). To compute the amount of valid data in this final buffer, the driver must read the packet length (PL bits of RDES3[14:0]) and subtract the sum of the buffer sizes of the preceding buffers in this packet. The Rx DMA always transfers the start of next packet with a new descriptor.

Note Even when the start address of a Rx buffer is not aligned to the data width of system bus, the system should allocate a Rx buffer of a size aligned to the system bus width. For example, if the system allocates a Rx buffer of 1,024 bytes (1 KB) starting from address 0x1000, the software can program the buffer start address in the Rx descriptor to have a 0x1002 offset.The Rx DMA writes the packet to this buffer with dummy data in the first two locations (0x1000 and 0x1001). The actual packet is written from location 0x1002. Therefore, the actual useful space in this buffer is 1,022 bytes, even though the buffer size is programmed as 1,024 bytes, because of the start address offset.

2.3.1.3 DMA Arbiter (EQOS-DMA and EQOS-AHB Configurations)

The arbiter inside the DMA module performs the arbitration between the Tx and Rx channel accesses to the AHB master interface. The following two types of arbitrations are supported:

- Round-Robin Arbitration

  When Bit 1 of the DMA_Mode register is reset and both Tx and Rx DMAs are simultaneously requesting for access, the arbiter allocates the data bus in ratio set by Bits[14:12] of DMA_Mode register.

- Fixed-Priority Arbitration

  When Bit 1 of the DMA_Mode register is set, the Rx DMA always gets priority over the Tx DMA for data access by default. When Bit 11 of DMA_Mode Register is also set, the Tx DMA gets priority over the Rx DMA as explained in Table 8-3 on page 286.

Note The Rx DMA places the next request only after the earlier request is complete. There is delay to find data availability in the MTL queue (ari_rdy and ari_pbl is enabled, if data equal to PBL is available; MTL asserts ari_rxwatermark which indicates required data availability) and place the request.

So, there is no overlap of Rx DMA request with Tx DMA request when operating in single DMA mode.

However, this is possible with multiple Rx DMA enabled.

2.3.1.4 DMA Start/Stop Control Through Sideband Signals

The DMA can be started/stopped by performing any or both of the following operations:

- Programming ST bit in DMA_CH[n]_Tx_Control and SR bit in DMA_CH[n]_Rx_Control register for Tx and Rx DMA respectively
- Appropriately driving the sbd_txdma_start_i[n] and sbd_rxdma_start_i[n] inputs for Tx and Rx DMA respectively

The DMA Start/Stop through Sideband Signals has higher precedence.

Table 2-2 shows the behavior when the above operations are performed simultaneously or distinctly:

Table 2-2 DMA Start/Stop Through Sideband Signals

Start(1)/Stop(0) through Sideband Signals	Start(1)/Stop(0) through software programming	Current State	New State	Comments
0	0	0	0	Default/Reset value
0 -> 1	0	0	1	DMA Started by Sideband
0	0 -> 1	0	1	DMA Started by SW
1	1	x	1	DMA Started by both Sideband Signals and software programming
1 -> 0	1, 0	1	0	DMA Stopped by Sideband Signals
1, 0	1 -> 0	1	0	DMA Stopped by SW programming
0 -> 1	1 -> 0	x	1	Sideband Signals Start DMA and SW programming Stop DMA in same clock cycle
1 -> 0	0 -> 1	x	0	Sideband Signals Stop DMA and SW programming Start DMA in same clock cycle

2.3.2 Transmission

2.3.2.1 Transmit DMA Operation: Default (Non-OSP) Mode

The Tx DMA engine in default mode proceeds as follows:

1. The application sets up the Transmit descriptor (TDES0–TDES3) and sets the Own bit (TDES3[31]) after setting up the corresponding data buffer(s) with Ethernet Packet data.
2. The application advances the Descriptor Tail pointer offset value of the Transmit Channel.
3. While in the Run state, the DMA runs an Arbitration cycle to select the next Tx DMA channel from which the packets requiring transmission should be processed.
4. The DMA fetches the descriptor from the application memory.
5. If the DMA detects one of the following conditions, the transmission from that channel is suspended and Bit 2 and Bit 16 of Status Register of corresponding DMA channel are set and the Tx Engine proceeds to step 11:
   - The descriptor is flagged as owned by the application (TDES3 [31] = 1'b0)
   - The Descriptor Tail pointer is equal to the Current Descriptor pointer in Ring Descriptor list Mode
   - An error condition occurs
6. If the acquired descriptor is flagged as owned by the DMA (TDES3[31] = 1’b1), the DMA decodes the Transmit Data Buffer address from the acquired descriptor.
7. The DMA fetches the Transmit data from the system memory and transfers the data to the MTL for transmission.

8. If an Ethernet packet is stored over data buffers in multiple descriptors, the DMA closes the intermediate descriptor and fetches the next descriptor. Steps 3 through 7 are repeated until the end of-Ethernet-packet data is transferred to the MTL.
9. When packet transmission is complete, if IEEE 1588 Timestamp feature was enabled for the packet (as indicated in the Tx status), the timestamp value obtained from MTL is written to the Tx descriptor (TDES0 and TDES1) that contains the EOP buffer. The status information is written to this Tx descriptor (TDES3). The application now owns this descriptor because the Own bit is cleared during this step.

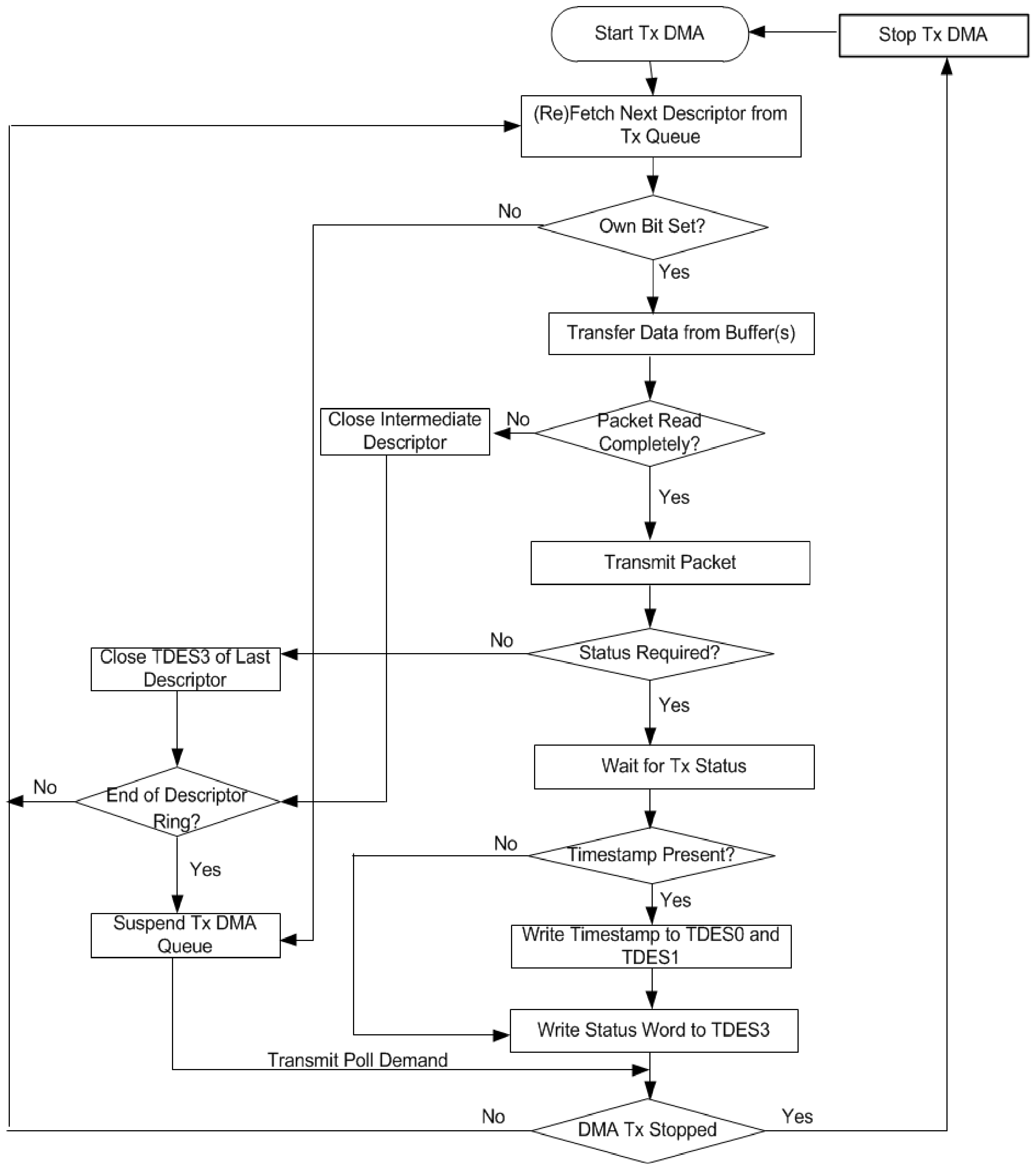
   If timestamp feature is not enabled for this packet, the DMA does not alter the contents of TDES0 and TDES1.

Note By default, the Tx DMA places subsequent request after the current descriptor write request is completed to the descriptor memory. If posted descriptor write is enabled by setting DSPW bit in DMA_Mode register, the Tx DMA will place subsequent request after the current descriptor write request is placed without waiting for its completion, thereby improving the throughput.

10. Bit 0 of Status Register of corresponding channel is set after completing transmission of a packet that has Interrupt on Completion (TDES2[31]) set in its Last Descriptor. The DMA engine returns to step 3.
11. In the Suspend state, the DMA tries to acquire the descriptor again (and thereby return to step 3). A poll demand command is triggered by writing any value to the DMA_CH0_TxDesc_Tail_Pointer when it receives a Transmit Poll demand and the Underflow Interrupt Status bit is cleared. If the application stopped the DMA by clearing Bit 0 of Transmit Control Register of corresponding DMA channel, the DMA enters the Stop state.

The Tx DMA transmission flow in default mode is shown in Figure 2-21.

Figure 2-21 Transmit DMA Operation in Default Mode



2.3.2.2 DMA Transmit Operation: OSP Mode

In the Run state, if Bit 4 is set in the Transmit Control Register of corresponding DMA channel, the Transmit process can simultaneously acquire two packets without closing the Status descriptor of the first packet. As the Transmit process finishes transferring the first packet, it immediately polls the Transmit Descriptor list for the second packet. If the second packet is valid, the Transmit process transfers this packet before writing the status information of the first packet.

In OSP mode, the Run state Tx DMA operates in the following sequence:

1. The DMA operates as described in step 1 to step 7 of the Tx DMA (default mode).
2. The DMA fetches the next descriptor without closing the last descriptor of previous packet.
3. If the DMA owns the acquired descriptor, the DMA decodes the transmit buffer address in this descriptor. If the DMA does not own the descriptor, the DMA goes into Suspend mode and skips to step 7.
4. The DMA fetches the Transmit packet from the system memory and transfers the packet to the MTL until the EOP data is transferred, closing the intermediate descriptors if this packet is split across multiple descriptors.
5. The DMA waits for the packet transmission status and timestamp of previous packet. When the status is available, the DMA writes the timestamp to TDES0 and TDES1 if such timestamp was captured (as indicated by a status bit). The DMA writes the status, with a cleared Own bit, to the corresponding TDES1, thus closing the descriptor.

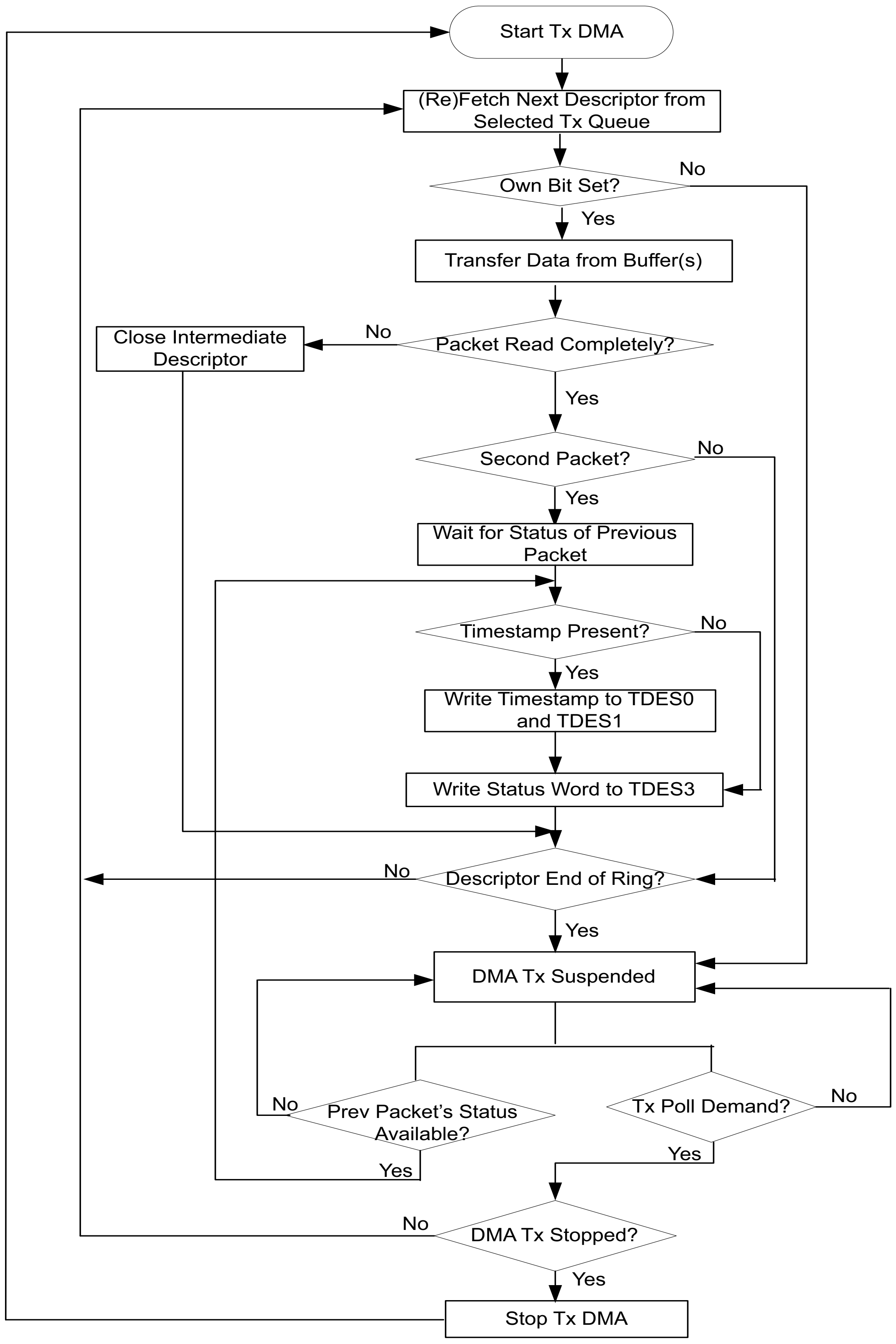
   If Timestamp feature is not enabled for the previous packet, the DMA does not alter the contents of TDES2 and TDES3.
6. The Transmit interrupt is set (if enabled). The DMA fetches the next descriptor and proceeds to step 3 (when Status is normal). If the previous transmission status shows an underflow error, the DMA goes into Suspend mode (step 7).
7. In Suspend mode, if a pending status and timestamp are received from the MTL, the DMA does the following:
   - Writes the timestamp (if enabled for the current packet) to TDES0 and TDES1
   - Writes the status to the corresponding TDES1
   - Sets relevant interrupts and returns to Suspend mode

   If no status is pending and the application stopped the DMA by clearing Bit 0 of Transmit Control Register of corresponding DMA channel, the DMA enters the Stop state.
8. The DMA can exit Suspend mode and enter the Run state (goes to step 1 or step 2 depending on pending status) only after receiving a Transmit Poll demand in Transmit Descriptor Tail Pointer register of corresponding channel.

Note The DMA fetches the next descriptor before closing the current descriptor. Therefore, the descriptor ring length must be more than two. Synopsys recommends having a minimum descriptor length of four.

The basic flow is described in Figure 2-22.

Figure 2-22 Tx DMA Operation in OSP Mode


Start Tx DMA
(Re)Fetch Next Descriptor from Selected Tx Queue
Own Bit Set?
No
Yes
Transfer Data from Buffer(s)
Close Intermediate Descriptor
No
Packet Read Completely?
Yes
Second Packet?
No
Yes
Wait for Status of Previous Packet
Timestamp Present?
No
Yes
Write Timestamp to TDES0 and TDES1
Write Status Word to TDES3
No
Descriptor End of Ring?
Yes
DMA Tx Suspended
No
Prev Packet's Status Available?
Yes
Tx Poll Demand?
No
Yes
No
DMA Tx Stopped?
Yes
Stop Tx DMA

2.3.2.3 Transmit Packet Processing

The Tx DMA expects that the data buffers contain complete Ethernet packets, excluding preamble, pad bytes, and FCS fields. The DA, SA, and Type/Length fields contain valid data. If the Tx Descriptor indicates that the MAC must disable CRC or PAD insertion, the buffer must have complete Ethernet packets (excluding preamble), including the CRC bytes.

Packets can be data-chained and can span several buffers. Packets must be delimited by the First Descriptor (TDES3[29]) and the Last Descriptor (TDES3[28]). As transmission starts, the First Descriptor must have TDES3[29] set. When this occurs, the packet data is transferred from the application buffer to the MTL Tx Queue. Concurrently, if the current packet has the Last Descriptor (TDES3[28]) clear, the Tx Process attempts to acquire the Next Descriptor. The Tx Process expects this descriptor to have TDES3[29] clear. If TDES3[28] is clear, it indicates an intermediary buffer. If TDES3[28] is set, it indicates the last buffer of the packet.

After the last buffer of the packet has been transmitted, the DMA writes back the final status information to the Transmit Descriptor 3 (TDES3) word of the descriptor that has the Last Descriptor Bit set in Transmit Descriptor 3 (TDES3[28]). At this time, if Interrupt on Completion (TDES2[31]) is set, Bit 0 of Status Register of corresponding DMA channel is set, the Next Descriptor is fetched, and the process repeats. The actual packet transmission begins after either of the following:

- The MTL Tx Queue has reached a programmable Transmit threshold (Bits[6:4] of Transmit Operation Mode register of corresponding MTL Transmit Queue)
- A full packet is contained in the FIFO

You can also use the store-and-forward mode (Bit 1 of MTL Transmit Operation Mode Register of a queue). In this mode, descriptors are released (Own bit TDES0[31] clears) when the DMA finishes transferring the packet.

Note To ensure proper transmission of a packet and the next packet, you must specify a non-zero buffer size for the Transmit descriptor that has the Last Descriptor (TDES3[28]) set.

2.3.2.4 Transmit Polling Suspended

Transmit polling can be suspended by any of the following conditions:

- The DMA detects a descriptor owned by the application (TDES3[31]=0).

  To resume, the driver must give descriptor ownership to the DMA and then issue a Poll Demand command by writing the Tail Pointer register. If the DMA goes into SUSPEND state because of this condition, Bit 15 and Bit 2 of Status Register of corresponding DMA channel are set.
- A packet transmission is aborted when a Transmit error is detected because of underflow.

  The appropriate Transmit Descriptor 3 (TDES3) bit is set. When this condition occurs, the following bits are set and the information is written to Transmit Descriptor 0, causing the suspension:
  - Bit 14 of Status Register of corresponding DMA channel
  - Transmit Underflow bit of corresponding queue in MTL_Interrupt_Status
- The DMA detects that the Tail Pointer is equal to the Current descriptor closed by the it.

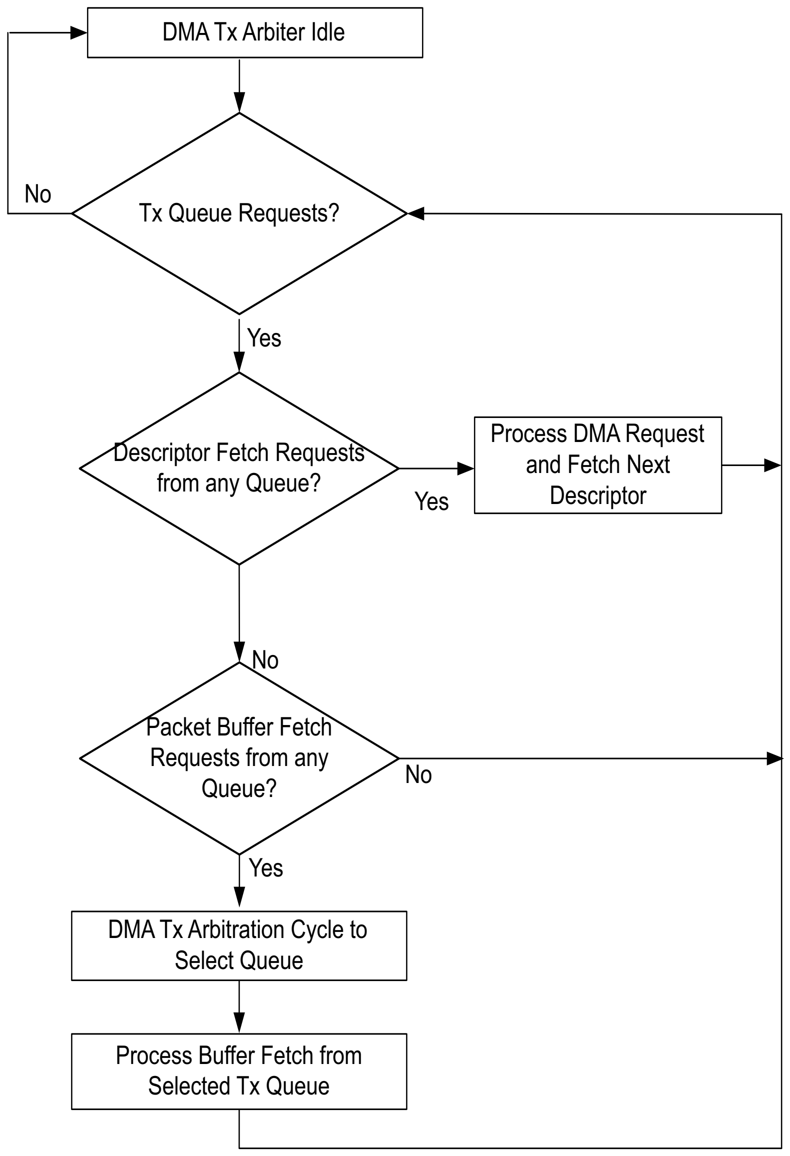
  To resume, the software driver must modify the Tail Pointer register.

In all conditions, the position in the Transmit List is retained. The retained position is that of the descriptor following the Last Descriptor closed by the DMA. The driver must explicitly issue a Transmit Poll Demand command after rectifying the suspension cause.

2.3.2.5 DMA Transmit Channel Arbitration

An arbiter provides access to multiple DMAs trying to access the Bus Interface Unit (BIU). Figure 2-23shows the arbitration process.

Figure 2-23 DMA Transmit Channel Arbitration Process


DMA Tx Arbiter Idle
Tx Queue Requests?
No
Yes
Descriptor Fetch Requests from any Queue?
Yes
Process DMA Request and Fetch Next Descriptor
No
Packet Buffer Fetch Requests from any Queue?
No
Yes
DMA Tx Arbitration Cycle to Select Queue
Process Buffer Fetch from Selected Tx Queue

When there is any request in the Tx queue, the DMA arbiter checks the type of the request: packet buffer fetch or descriptor fetch request. The descriptor fetch requests have higher priority than the buffer requests.Therefore, when there is a descriptor fetch request, the DMA arbiter acknowledges the DMA channel that is requesting for a descriptor fetch. If there is no descriptor fetch request, the arbiter looks for packet buffer fetch requests.

The DMA arbiter acknowledges the descriptor fetch request of one DMA channel at a time. Descriptor fetch requests are granted using a fixed priority with the higher channel having higher priority (Channel 1 having priority over Channel 0, Channel 2 having priority over Channel 1 and so on). For packet buffer fetches, the DMA arbiter uses the programmed channel weight and priority to decide which channel to acknowledge. The DMA arbiter performs a burst-by-burst arbitration based on one of the following algorithms:

- Weighted Strict Priority (WSP): In WSP arbitration mode, the arbiter first processes Channel 7 (or the last enabled channel) and then Channel 6, Channel 5, and so on. If any channel does not have a frame to transmit, the weight of that channel gets reassigned to Channel7 (or last enabled channel). If Channel 7 has no frames to transmit, the remaining weight is assigned to Channel 6 and so on.
- Weighted Round Robin (WRR): In WRR arbitration mode, the arbiter first selects the channel with the highest weight programmed, and then the channel with next highest weight, and so on. If any channel does not have a frame to transmit, the weight of that channel gets equally distributed to all channels that have frames to transmit.
- Fixed priority (FP): In Fixed priority mode, Channel 0 has the lowest priority and the last selected channel has the highest priority. The weight programmed in the Transmit Control register of a channel is ignored.

In WSP or WRR arbitration, the channel weight corresponds to the number of DMA burst transfers for which the DMA arbiter grants the bus to a channel. When a channel completes all the DMA burst transfers, the arbiter grants the bus to the next channel.

2.3.3 DMA Receive Operation

In the Receive path, the DMA reads a packet from the MTL receive queue and writes it to the packet data buffers of the corresponding DMA channel. The ARI data at the start of the frame indicates the channel number to which the current frame must be written. If only one DMA channel is selected, this information is not provided.

Figure 2-24 shows the reception sequence for Rx DMA engine. The following list describes this sequence:

1. The application sets up the Rx descriptors (RDES0-RDES3) and the Own bit (RDES3[31]).

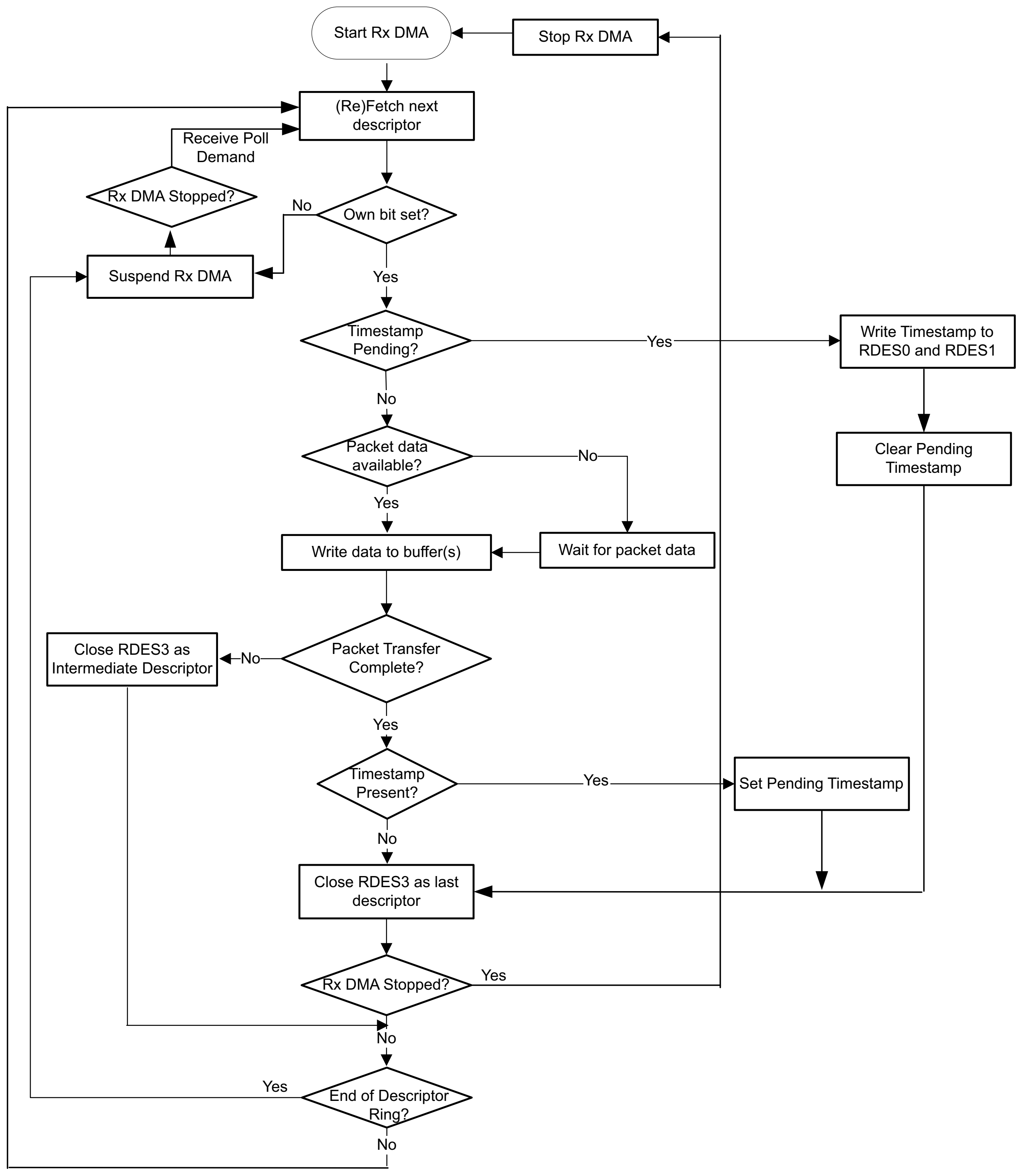
   The application must set the correct value in the Receive Descriptor Tail Pointer register of corresponding DMA channel.
2. When Bit 0 of Receive Control register of corresponding DMA channel is set, the DMA enters the Run state.

   The DMA looks for free descriptors based on the Rx Current Descriptor and Descriptor Tail Pointer register values. If there are no free descriptors, the DMA Channel enters the suspend state and goes to step 11.
3. The DMA fetches the next available descriptor in the ring and decodes the receive data buffer address from acquired descriptors.
4. If IEEE 1588 timestamping is enabled and the timestamp is available for the previous packet, the DMA writes the timestamp (if available) to the RDES0 and RDES1 of current descriptor and sets the CTXT field (RDES3[30]).
5. The DMA processes the incoming packets and places these in the data buffers of acquired descriptor.

6. If the current packet transfer is not complete, the DMA closes the current descriptor as intermediate and goes to step 10.
7. The DMA takes the status of the Receive frame from the MTL and writes the status word to current descriptor with the Own bit cleared and the Last Descriptor bit set.
8. The DMA writes the Frame Length to RDES3 and VLAN Tag to RDES0. The DMA also writes the MAC control frame opcode, OAM control frame code, and extended status information (if available) to RDES1 of the last descriptor.
9. If IEEE 1588 Timestamp feature is enabled, the DMA stores the timestamp (if available). The DMA writes the context descriptor after the last descriptor for the current packet (in the next available descriptor).
10. If more descriptors are available in the Rx DMA Descriptor Ring, go to step 3; otherwise, go to the Suspend state (step 11).
11. The Receive DMA exits the Suspend state when a Receive Poll demand is given and the application advances the Receive Tail Pointer register of a channel.

    The engine proceeds to step 2 and re-fetches the next descriptor.

Figure 2-24 Receive DMA Operation



2.3.3.1 Receive Descriptor Acquisition

The Receive Engine always attempts to acquire an extra descriptor in anticipation of an incoming packet. Descriptor acquisition is attempted if any of the following conditions is satisfied:

- Bit 0 of Receive Control Register of corresponding DMA channel is set immediately after being placed in the Run state.
- The Descriptor Tail pointer register value is ahead of the Current Descriptor acquired by the Rx DMA.
- The controller has completed packet reception, but the current Receive Descriptor is not yet closed.
- A Receive poll demand is issued (update of the Tail Pointer register).

2.3.3.2 Receive Packet Processing

The sequence for processing a Receive packet is as follows:

1. The MAC transfers the received packets to the MTL memory only if the packet passes the address filter. If the packet fails the address filtering, it is dropped in the MAC block (unless Bit 31 of MAC_Packet_Filter register is set).
2. If packet size is greater than or equal to configurable threshold bytes set for Rx Queue of MTL, or when the complete packet is written to the queue in the store-and-forward mode, the MTL block requests the DMA block to begin transferring the packet data to the Receive Buffer pointed by the current descriptor.

   Packets smaller than 64 bytes, because of collision or premature termination, are removed from the MTL Receive Queue.
3. When the DMA application Interface (AHB/AXI or MDC) becomes ready, it transfers the data and sets the following:
   - If the packet fits in a single descriptor, the DMA sets both Last Descriptor (RDES3[28]) and First Descriptor (RDES3[29]).
   - If the packets fits into more than one descriptor, the DMA sets the First Descriptor (RDES3[29]) to delimit the packet.
4. The DMA releases the descriptors by resetting the Own (RDES3[31]) bit to 1'b0, either because the Receive buffer filled up or the last segment of the packet is transferred to the Receive buffer. The received packets status is updated in the last descriptor.
5. If Interrupt Enabled on Completion (RDES3[30]) bit is set in any of the Descriptors between the First and Last Descriptor of the Packet and Bit 6 of Interrupt Enable Register of corresponding DMA channel is set, the DMA sets Bit 6 of Status register of corresponding DMA channel.

   The same process repeats unless the DMA encounters a descriptor flagged as being owned by the application or when there are no more descriptors in the ring. When the DMA finds a descriptor owned by the application and if Bit 7 of Interrupt Enable Register of corresponding DMA channel is set, the Receive Process sets Bit 7 of Status register of corresponding DMA channel and then enters the Suspend state. The position in the receive list is retained.

2.3.4 Error Response to DMA

For any data transfer initiated by a DMA channel, if the slave replies with an error response, the DMA stops all operations and updates the error bits and the Fatal Bus Error bit in the Status Register of corresponding DMA channel. The application can either perform a reset to DWC_ether_qos or re-initialize the DMA descriptor list and start again. The rest of the DMA channels are not affected by such errors. This is applicable for the following configurations:

- EQOS-DMA: In this configuration, the DMA receives the error response through mdc_error_i signal (shown in Figure 2-31 on page 82).
- EQOS-AHB: In this configuration, the DMA receives the error response through hresp_i signal (shown in Figure 2-13 on page 53).
- EQOS-AXI: In this configuration, the DMA receives the error response through rresp_m_i (shown in Figure 2-16 on page 58) or bresp_m_i signal.

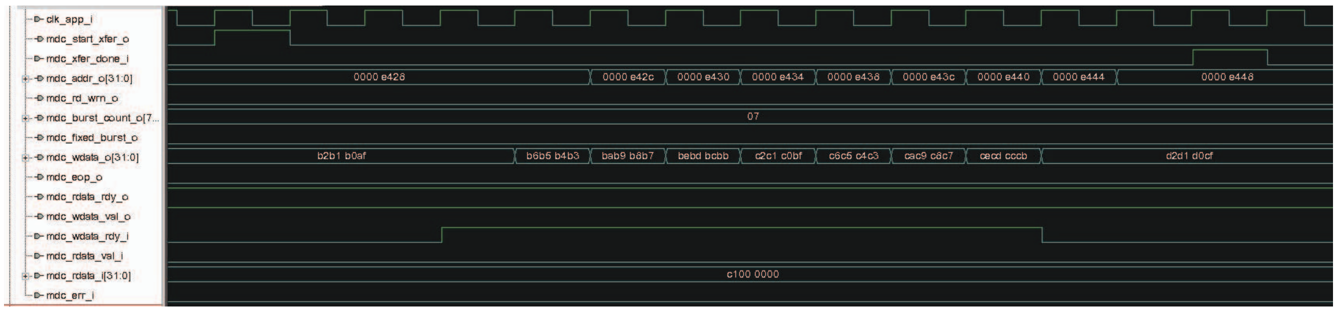
2.3.5 DMA Native Interface

The subsystem can be configured (EQOS-DMA) to have DMA with a native FIFO-like interface on the application side and an MCI (Optionally APB) interface for CSR port. All functions and features of the DMA remain the same as described in earlier sections.

On the native interface, the DMA uses a simple hardware protocol to initiate the data or descriptor transfers. The DMA always starts a transaction with the proper values on the address bus, transfer size, burst length, and type of transfer (whether read or write). For read transfers, the application can provide valid data by asserting the mdc_rdata_val_i signal. For write transfers, the DMA can provide valid data by asserting the mdc_wdata_val_i signal. The application can pause the transfers by de-asserting the respective mdc_rdata_val_i or mdc_wdata_rdy_i control signals.

Note The values on the address bus, burst length, and transfer size are not updated immediately on receiving the mdc_rdata_val_i or mdc_wdata_rdy_i signals. The address, burst length, and transfer size are valid only at the start of the transfer. In certain types of access, the burst length may not even change during the transfers as shown in Figure 2-25. Before asserting the mdc_xfer_done_i signal for proper transfer of data, the application must assert the mdc_rdata_val_i or mdc_wdata_rdy_i signal for clocks equal to the burst length requested at the start of transfer (unless a premature write transfer is indicated by DMA as explained in the next section).

Figure 2-25 MDC Interface Burst Transfers

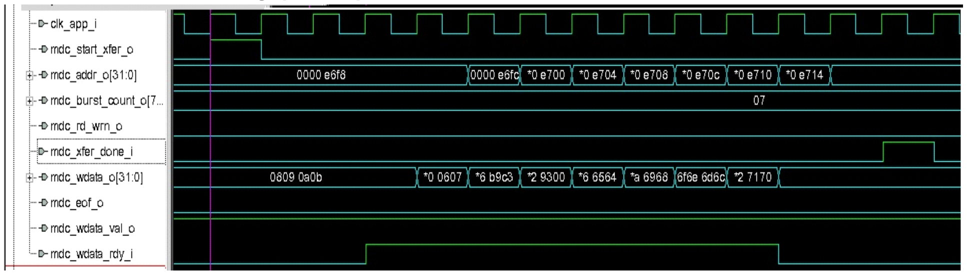


After transferring all of the requested data, the application must terminate the DMA transaction with the mdc_xfer_done_i signal. The application can indicate an error by activating the mdc_error_i signal and then the respective DMA (Rx or Tx) goes to the Fatal Error state and all operations stop. The application must provide a soft reset to restart the DMA operation.

2.3.5.1 Write Data Transfer

Figure 2-26 shows the transfer sequence of a data buffer being written to system memory. The sequence is as follows:

Figure 2-26 Write Data Transfer Timing (1 of 3)



1. The subsystem initiates the transaction by asserting the mdc_start_xfer_o (pulse) signal along with the following:
   - System memory address (mdc_addr_o)
   - Read/Write control signal (mdc_rd_wrn_o = 0)
   - Number of beats (mdc_burst_count_o)
   - Transfer size (mdc_xfer_size_o)

   It also drives the first data beat of the burst on the mdc_wdata_o bus along with mdc_wdata_val_o asserted.
2. The application asserts the mdc_wdata_rdy_i signal to accept the data.
3. If the application is not ready to handle the data, it can de-assert mdc_wdata_rdy_i and delay the data transfer. The DMA holds the values on the data bus when the "rdy" control is de-asserted.
4. After all of the data has been transferred as indicated by mdc_burst_count_o, the application must end the DMA transfer by asserting the mdc_xfer_done_i (pulse) signal.
5. The DMA can prematurely end the burst transfer if an EOP is being transferred to the application. The DMA marks the EOP data on mdc_wdata_o with the assertion of mdc_eop_o. The application must then complete the transaction by asserting the mdc_xfer_done_i signal after accepting the EOP data even if the initial burst length given at start of transaction is not completed.

Figure 2-27 shows the transfer sequence when a data buffer is written to system memory with the burst transfer ending prematurely because of the assertion of mdc_eop_o. The DMA terminates the burst by asserting mdc_eop_o after completing 3 cycles of the requested 4-cycle burst. The application must respond by asserting the mdc_xfer_done_i signal without asserting any more mdc_wdata_rdy_i.

The application should sample mdc_eop_o along with the corresponding data being accepted, that is, mdc_eop_o should be sampled when mdc_wdata_rdy_i is high. The mdc_eop_o signal can go high when the data on mdc_wdata_o is updated with the EOP data even before the start of next transfer indicated bymdc_start_xfer_o. It need not go low immediately after the application accepts the EOP data because the MDC may not have new data to update. It can go low anytime before the start of next write transfer. The mdc_eop_o signal may also go high with the last beat of the requested burst transfer if it coincides with the EOP data being presented to the application.

When the mdc_eop_o signal goes high, it may remain high even if the DMA inserts a single descriptor write transfer (which has higher priority) before completing the EOP data transfer. If a descriptor write occurs with two or more beats, the mdc_eop_o signal is removed.

Figure 2-27 shows the transfer sequence when EOP of a received packet is written to memory.

Figure 2-27 Write Data Transfer Timing (2 of 3)

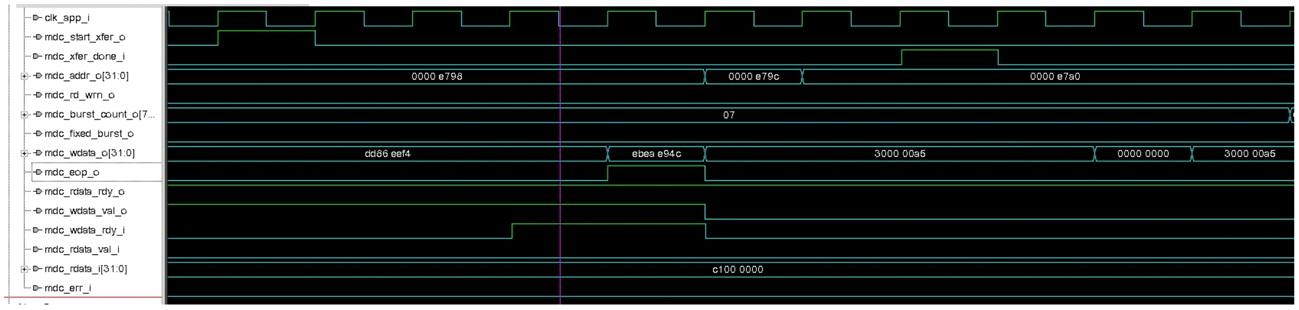
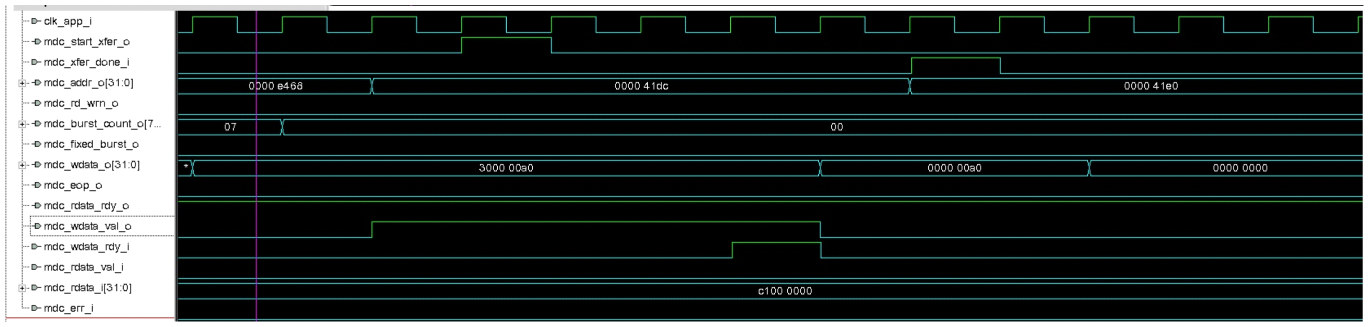


Figure 2-28 shows the transfer sequence when a descriptor is written to system memory.

Figure 2-28 Write Data Transfer Timing (3 of 3)



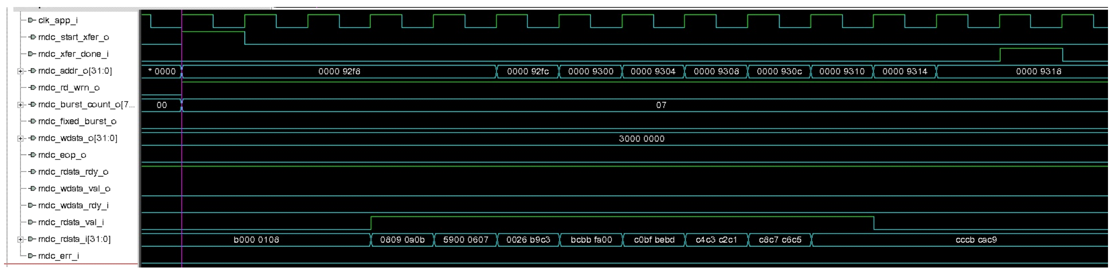
2.3.5.2 Read Data Transfer

Figure 2-29 shows the transfer sequence during data buffer being read from system memory. The sequence is as follows:

1. The subsystem initiates the transaction by asserting the mdc_start_xfer_o (pulse) signal along with the following:
   - System memory address (mdc_addr_o)
   - Read/Write control signal (mdc_rd_wrn_o = 1)
   - Number of beats (mdc_burst_count_o)
   - Transfer size (mdc_xfer_size_o)
2. The DMA accepts the data on the mdc_rdata_i bus by asserting the mdc_rdata_rdy_o signal whenever the application asserts the mdc_rdata_val_i signal.

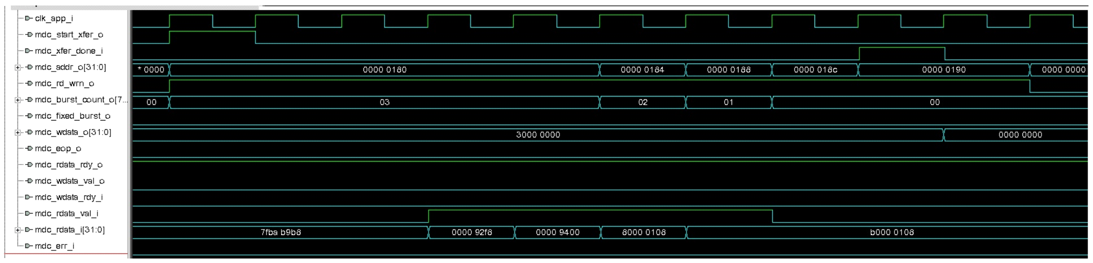
3. If the application is not ready with the data, it can de-assert mdc_rdata_val_i and delay the data transfer.
4. After all of the data has been transferred as indicated by mdc_burst_count_o, the application must end the DMA transfer by asserting the mdc_xfer_done_i (pulse) signal.

Figure 2-29 Read Data Transfer Timing (1 of 2)



[Figure 2-30](#) shows the transfer sequence during a descriptor being read from system memory.

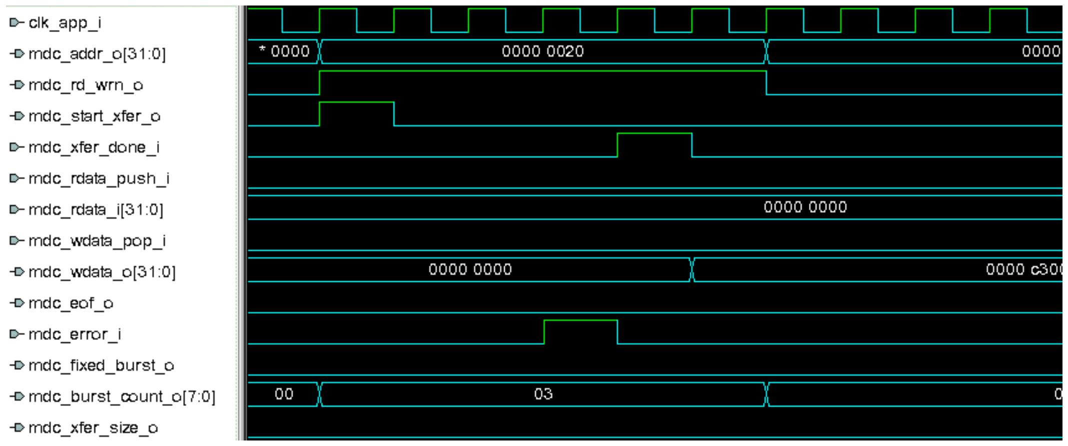
Figure 2-30 Read Data Transfer Timing (2 of 2)



2.3.5.3 Data Transfer With Error

For any write or read transfer started by the DMA, the application can assert mdc_error_i if it cannot complete the transaction as shown in Figure 2-31. The application should also complete the transfer by asserting mdc_xfer_done_i in a different clock cycle. If the application asserts both signals together, the DMA may restart the transfer to the same address and the behavior of the DMA is not predictable in such scenario.

Figure 2-31 Data Transfer With Error Timing



2.4 MAC Transaction Layer

The MAC Transaction Layer (MTL) provides the FIFO memory Interface to buffer and regulate the packets between the application system memory and the MAC. It also enables the data to be transferred between the application clock and MAC clock domains. The MTL layer has two data paths: Transmit path and Receive Path. The data path for both directions can be 32-bit, 64-bit, or 128-bit wide.

The MTL communicates with the application through Application Transmit Interface (ATI) on the Transmit path and Application Receive Interface (ARI) on the Receive path. The MTL also provides the MAC Control Interface (MCI) as a control path. The default and optional I/O signals for the ATI, ARI, and MCI interfaces are described in "Signal Descriptions" on page 471.

2.4.1 Transmit Path

In EQOS-MTL configurations, the application module drives all transactions related to the Transmit path through ATI. In EQOS-AHB, EQOS-AXI, and EQOS-DMA configurations, the internal DMA handles all transactions for the Transmit path through ATI.

The application or internal DMA pushes the Ethernet packets read from the application or system memory into the corresponding queue. The packet is then popped out and transferred to the MAC when the queue threshold is reached (threshold mode) or complete packet is in the queue (store-and-forward mode). When EOP is transferred, the status of the transmission is taken from the MAC and transferred back to the application or internal DMA.

The Tx queue has a default size of 2K bytes. The fill level of the queue is indicated to the application or internal DMA (using PBL and watermark) so that it can initiate a data fetch in required bursts from the application or system memory. The application or internal DMA indicates the SOP and EOP as packet delimiters through ATI interface. In configurations with multiple queues, the application or internal DMA should also indicate the queue number for which the transaction is being addressed.

2.4.1.1 Transmit Control Word

The following control information related to packet transmission is provided as a part of the Control Word through ATI interface:

- Packet Length (if DCB is enabled with WFQ scheduling algorithm)
- CRC Pad Control
- Source Address Insertion Control (if ***Enable SA and VLAN Insertion on Tx*** option is selected)
- VLAN Insertion and Replacement Control and VLAN Tag for Outer and Inner VLAN Tags (if only ***Enable SA and VLAN Insertion on Tx*** option is enabled. Inner VLAN Tag is supported if both ***Enable SA and VLAN Insertion on Tx*** and ***Enable Double VLAN Processing*** options are enabled)
- TCP/IP Checksum Insertion Control (if ***Enable Transmit TCP/IP Checksum Offload*** option is selected)
- One-step Timestamping Control Correction (if ***Enable One-Step Timestamp Feature*** option is selected)
- Transmit Timestamp Enable (if ***Enable IEEE 1588 Timestamp Support*** option is selected)

2.4.1.1.1 Transmit Control Word Format

The control word format for 32-bit datawidth configuration is shown in Figure 2-31. Control Word 0 is always present, when per packet control is needed. Control Words 1, 2, and 3 (in 32-bit configuration) are present based on the packet control field of Control Word 0.

Control Word 1 is present in the following configurations:

- When only Enable SA and VLAN Insertion on Tx option is selected and VTV bit is set.
- When both Enable SA and VLAN Insertion on Tx and Enable Double VLAN Processing options are enabled and VTV or IVTV bits are set
- When Enable One-Step Timestamp Feature is selected, and OSTC bit is set
- When Enable One-Step Timestamp for PTP over UDP/IP Feature is selected, and both OSTC and OST_AVAIL bits are set

Table 2-3 lists and describes the contents of Control Word 1 based on different configurations and register bit settings.

Table 2-3 Content of Control Word 1 Depending on Configuration and Register Bit Settings

Contents of Control Word 1	Enable SA and VLAN Insertion on TX	Enable Double VLAN Processing	Enable One-Step Timestamp Feature	VTV	IVTV	OSTC and OST_AVAIL
VLAN Tags are present	Yes	No	Dont care	Yes	No	Dont care
Inner VLANTag	Yes	Yes	Dont care	No	Yes	Dont care
Outer VLANTag	Yes	Yes	Dont care	Yes	No	Dont care
Inner and Outer VLANTags are present	Yes	Yes	Dont care	Yes	Yes	Dont care
Timestamp Low (lower 32-bit value)	Yes	Yes	Yes	No	No	Yes
Timestamp High (higher32-bit value)	No	Not configurable	Yes	Not configurable	Not configurable	Yes

Control Word 2 is present only when Enable One-Step Timestamp Feature is selected and OSTC bit is set or when Enable One-Step Timestamp for PTP over UDP/IP Feature is selected and both OSTC and OST_AVAIL bits are set.

Table 2-4 lists and describes the contents of Control Word 2 depending on the configuration and register bit settings.

Table 2-4 Content of Control Word 2 Depending on Configuration and Register Bit Settings

Contents of Control Word 2	Enable One-Step Timestamp Feature	Enable SA and VLAN Insertion on TX	OSTC and OST_AVAIL	VTV	ITV
Timestamp Low (lower 32-bit value)	Yes	Yes	Yes	Yes	Yes
Timestamp High (higher 32-bit value)	Yes	No	Yes	No	No

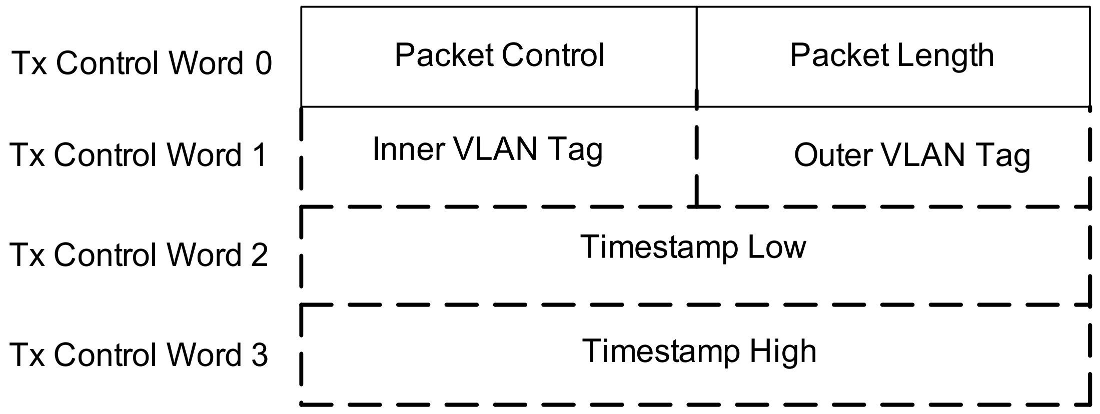
If Enable SA and VLAN Insertion on Tx is not selected, it contains the higher 32-bit value. If Enable SA and VLAN Insertion on Tx is selected and VTV bit is set, it contains the lower 32-bit value.

Control Word 3 is present only when Enable SA and VLAN Insertion on Tx, Enable Double VLAN Processing, and Enable One-Step Timestamp Feature is selected and OSTC, (Enable One-Step Timestamp for PTP over UDP/IP Feature is selected and both OSTC and OST_AVAIL bits are set), VTV, and IVTV bits are set. Its value is Timestamp High (higher 32-bit value).

Control Word 1 is present when bits VTV (when Enable SA and VLAN Insertion on Tx option is selected) and/or IVTV (when both Enable SA and VLAN Insertion on Tx and Enable Double VLAN Processing options are enabled) is/are set, and contains the Inner and/or Outer VLAN Tags, based on the VTV/SVTV bits respectively.

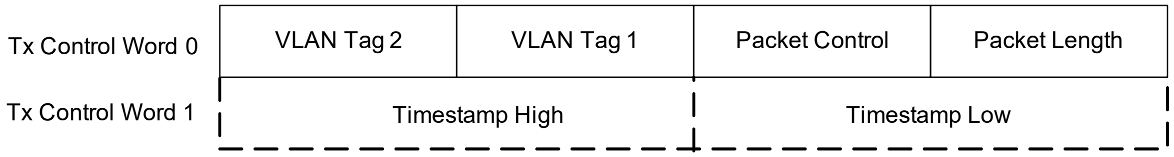
Control Word 1 is also present when OSTC (when Enable One-Step Timestamp Feature is selected) bit is set and both VTV and SVTV bits are not set. In this case, Control Word 1 contains the lower 32-bit value and Control Word 2 contains the higher 32-bit value of the 64 -bit timestamp. When both OSTC and VTV and/or SVTV bits are set, Control Word 1 contains the Inner and/or Outer VLAN Tags, Control Word 2 contains the lower 32-bit value, and Control Word 3 contains the higher 32-bit value of the 64 bit timestamp. The Control Word number is indicated through the ati_ctrl_type_i signal of the ATI interface.

Figure 2-32 Tx Control Word Structure for 32-Bit Datawidth Configurations



The control word format for 64-bit datawidth configuration is shown in Figure 2-33. Control Word 1 is present only when the Enable One-Step Timestamp Feature option is selected in your configuration. The Control Word number is indicated through the ati_ctrl_type_i signal of the ATI interface.

Figure 2-33 Tx Control Word Structure for 64-Bit Datawidth Configurations



The control word format for 128-bit datawidth configuration is shown in Figure 2-34.

Figure 2-34 Tx Control Word Structure for 128-Bit Datawidth Configuration

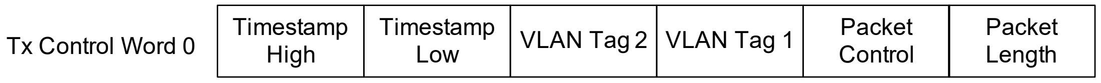


Table 2-5 describes the Packet Length and Packet Control Fields.

Table 2-5 Packet Length and Control Fields

Field	Field Name	Description
31	RSVD	Reserved
30	IVTV	Inner VLAN Tag Valid: When set, this bit indicates that the VLAN Tag to be inserted or replaced is available in TX Control Word 1 [31:16]. This bit is valid only when both Enable SA and VLAN Insertion on Tx and Enable Double VLAN Processing options are selected.
29	VTV	VLAN Tag Valid When set, this bit indicates that the VLAN Tag1 to be inserted or replaced is available in the TX Control Word1[15:0] or VLT1[15:0] field of TX control word1. This bit is valid only when Enable SA and VLAN Insertion on Tx option is selected.
28:27	VTIR	VLAN Tag Insert or Replace When set, these bits request the MAC to perform VLAN tagging or untagging before transmitting the packets. The application must set the appropriate CRC Pad Control value when this field is enabled.The following list describes the values of these bits: ■ 2'b00: Do not add a VLAN tag. ■ 2'b01: Remove the VLAN tag from the packets before transmission. This option should be used only with the VLAN packets. ■ 2'b10: Insert a VLAN tag. ■ 2'b11: Replace the VLAN tag. This option must be used only with VLAN packets. The tag value for insertion or replacement can be taken from the Outer VLAN field of the Control Word or VLT field of MAC_VLAN_Incl register, based on the VLTI bit in that register. These bits are valid when the Enable SA and VLAN Insertion on Tx option is selected. This field is valid only for the first descriptor.

Field	Field Name	Description
26:24	SAIC	SA Insertion Control These bits request the MAC to insert or replace the Source Address field in the Ethernet packet with the value given in the MAC Address 0 register. If the Source Address field is modified in a packet, the MAC automatically recalculates and replaces the CRC bytes. Bit 26 specifies the MAC Address Register (1 or 0) value that is used for Source Address insertion or replacement. The following list describes the values of Bits[25:24] ■ 2'b00: Do not include the source address ■ 2'b01: Include or insert the source address. For reliable transmission, the application must provide frames without source addresses. ■ 2'b10: Replace the source address. For reliable transmission, the application must provide frames with source addresses. ■ 2'b11: Reserved SA Insertion Control can also be controlled by programming the SARC bits of MAC_Configuration register. SARC bits determine whether the SA insertion control is through the control word or register. These bits are valid when the Enable SA and VLAN Insertion on Tx option is selected.
23:22	IVTIR	Inner VLAN Tag Insert or Replace These bits control the inner VLAN tagging or untagging before transmitting the packets. The outer VLAN tag must be present for this operation. The application must set appropriate CRC Pad Control value when this control is enabled. The following list describes the values of these bits. ■ 2'b00: Do not add inner VLAN tag. ■ 2'b01: Remove the inner VLAN tag from the packets before transmission. This option should be used only with the stacked VLAN packet ■ 2'b01: Remove the inner VLAN tag from the packets before transmission. This option should be used only with the stacked VLAN packets ■ 2'b10: Insert inner VLAN tag with the tag value programmed in MAC Register 25 (MAC_Inner_VLAN_Incl register). ■ 2'b11: Replace the inner VLAN tag in packets with the Tag value programmed in MAC Register 25 (MAC_Inner_VLAN_Incl register). This option must be used only with stacked VLAN packets. The tag value for insertion or replacement can be taken from the Inner VLAN field of the Control Word or VLT field of MAC_Inner_VLAN_Incl register, based on the VLTI bit in that register. These bits are valid when the Double VLAN and SA-VLAN Insertion options are selected during core configuration.
21:20	CIC	Checksum Insertion Control. These bits control the checksum calculation and insertion. The following list describes the bit encoding: ■ 2'b00: Checksum Insertion Disabled. ■ 2'b01: Only IP header checksum calculation and insertion are enabled ■ 2'b10: IP header checksum and payload checksum calculation and insertion are enabled but pseudo-header checksum is not calculated in hardware ■ 2'b11: IP Header checksum and payload checksum calculation and insertion are enabled and pseudo-header checksum is calculated in hardware. This field is valid when the Enable Transmit TCP/IP Checksum Offload option is selected.

Field	Field Name	Description
19	OSTC	One-Step Timestamp Correction When this bit is set, the MAC performs the one-step timestamp correction with reference to the timestamp values provided in Tx Control Word 1/2 and Tx Control Word 2/3.This bit is valid when the Enable One-Step Timestamp Feature option is selected.
18	TTSE	Transmit Timestamp Enable When set, this bit enables the IEEE1588 hardware time stamping for this Transmit frame.This bit is valid when the Enable IEEE 1588 Timestamp Support option is selected.
17:16	CPC	CRC Pad Control This field controls the CRC and Pad insertion for Tx frame.The following list describes the bit encoding: ■ 2'b00: CRC and Pad Insertion The MAC appends the cyclic redundancy check (CRC) at the end of the transmitted packet of length greater than or equal to 60 bytes. The MAC automatically appends padding and CRC to a packet with length less than 60 bytes. ■ 2'b01: CRC Insertion (Disable Pad Insertion) The MAC appends the CRC at the end of the transmitted packet but it does not append padding. The application should ensure that the padding bytes are present in the packet being transferred from the Transmit Buffer, that is, the packet being transferred from the Transmit Buffer is of length greater than or equal to 60 bytes. ■ 2'b10: Disable CRC Insertion The MAC does not append the CRC at the end of the transmitted packet. The application should ensure that the padding and CRC bytes are present in the packet being transferred from the Transmit Buffer. ■ 2'b11: CRC Replacement The MAC replaces the last four bytes of the transmitted packet with recalculated CRC bytes. The application should ensure that the padding and CRC bytes are present in the packet being transferred from the Transmit Buffer. When checksum insertion (CIC), SA-VLAN insertion (SAIC), or one-step timestamp (OSTC) features are enabled, the application must provide appropriate CRC Pad Control.
15	OST_AVAIL	One-step Timestamp Available When OSTC bit and this bit are set, it indicates that the reference Timestamp is available in the TTSH and TTSL fields of TX control word.The Tx control word corresponding to TTSH and TTSL fields must not be provided if this bit is reset.This bit is valid when the Enable One-Step Timestamp for PTP over UDP/IP Feature option is selected.
14:0	PL	Transmit Packet Length This field is equal to the length of the packet to be transmitted in bytes. This field is needed only when DCB is enabled in your configuration and WFQ scheduling algorithm is selected. **Note:** If CRC and padding is done by the MAC, the application must provide the appropriate length, which is (approximately) equal to the length of the transmitted packet by the MAC. This information is used by the Transmit scheduler.

Table 2-6 provides description of Outer VLAN Tag and Inner VLAN Tag fields of ATI Control Word.

Table 2-6 Outer VLAN Tag and Inner VLAN Tag Fields of ATI Control Word

Field	Field Name	Description
31:16	VLT2	Inner VLAN Tag: This field contains the inner VLAN Tag to be inserted or replaced in the transmitted packet.This field is valid when the following are true: ■ The Enable SA and VLAN Insertion on Tx and Enable Double VLAN Processing options are selected. ■ The VLTI bit is set in MAC_Inner_VLAN_Incl register ■ The IVTIR field in Tx Control Word 0 indicates inner VLAN Tag insertion or replacement.
15:0	VLT1	Outer VLAN Tag: This field contains the outer VLAN Tag to be inserted or replaced in the transmitted packet. This field is valid when the following are true: ■ The Enable SA and VLAN Insertion on Tx option is selected. ■ The VLTI bit is set in the MAC_VLAN_Incl register. ■ The VTIR field in Tx Control Word 0 indicates VLAN Tag insertion or replacement.

Table 2-7 provides description of Timestamp Low field of ATI Control Word.

Table 2-7 Timestamp Low Field of ATI Control Word

Field	Field Name	Description
31:0	TSSL	Transmit Packet Timestamp Low This field contains the lower 32-bits of timestamp when one-step timestamp correction is enabled. This field is validated by OSTC bit in Tx Control Word 0. This field is valid when the Enable One-Step Timestamp Feature option is selected.

Table 2-8 provides description of Timestamp High field of ATI Control Word.

Table 2-8 Timestamp High Field of ATI Control Word

Field	Field Name	Description
31:0	TSSH	Transmit Packet Timestamp High This field contains the higher 32-bits of timestamp when one-step timestamp correction is enabled. This field is validated by the OSTC bit in Tx Control Word 0. This field is valid when the Enable One-Step Timestamp Feature option is selected.

2.4.1.2 Transmit Operation

The following two modes of operation trigger reading of the data towards the MAC:

- **Threshold mode:**

  In Threshold (or cut-through) mode, as soon as the number of bytes in the Queue cross the configured threshold level (or when the end of packet is written before the threshold is crossed), the data is ready to be popped out and forwarded to the MAC. The threshold level is configured by using the TTC bits of MTL_TxQ0_Operation_Mode register corresponding to an MTL queue.

- **Store-and-forward mode:**

  In store-and-forward mode, the MTL pops out the packet towards the MAC only when one or more of the following conditions are true:

  - A complete packet is stored in the Queue
  - The Tx FIFO becomes almost full
  - The ATI watermark becomes low

  The watermark becomes low when the requested Queue does not have space to accommodate the requested burst length on the ATI. Therefore, the MTL when operating in the store-and-forward mode allows packet transmission even if the packet length is bigger than the Tx Queue size.

The application can flush complete content of the Tx Queue by setting Bit 0 (FTQ) of Transmit Operation Mode register corresponding to an MTL queue. This bit is self-clearing and initializes the Queue pointers to the default state. If the FTQ bit is set during a packet transfer from the MTL to the MAC, the MTL stops further transfer because the queue is considered to be empty. Therefore, an underflow event occurs at the MAC transmitter.

For information about the initialization and transmit operations for the MTL Layer, see Section 2.4.1.3 through Sections 2.4.1.5. The timing diagrams are provided in "Transmit Path Timing" on page 93.

2.4.1.3 Initialization Flow

Upon reset, the MTL is ready to manage the flow of data to and from the application or DMA, and MAC. For configurations with a single Transmit Queue, there are no initialization requirements for enabling the MTL. For configurations with multiple Transmit Queues, you need to initialize the Queue size for each of the queues by programming the TQS bits of MTL_TxQ0_Operation_Mode register corresponding to a Transmit queue. You also need to initialize the MAC block. For EQOS-AHB, EQOS-AXI, and EQOS-DMA configurations, the internal DMA controllers must be individually enabled through their respective CSRs.

2.4.1.4 Data Flow Between DMA and MTL

This section describes the interaction process between the MTL and the DMA in the EQOS-AHB, EQOSAXI, and EQOS-DMA configurations.

2.4.1.4.1 Single-Packet Transmit Operation

The following flow is valid when the core is operating in non-OSP mode, that is, when bit 4 (OSP) of the DMA_CH(#i)_TX_Control (for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1) register of the DMA channel is set to 0.

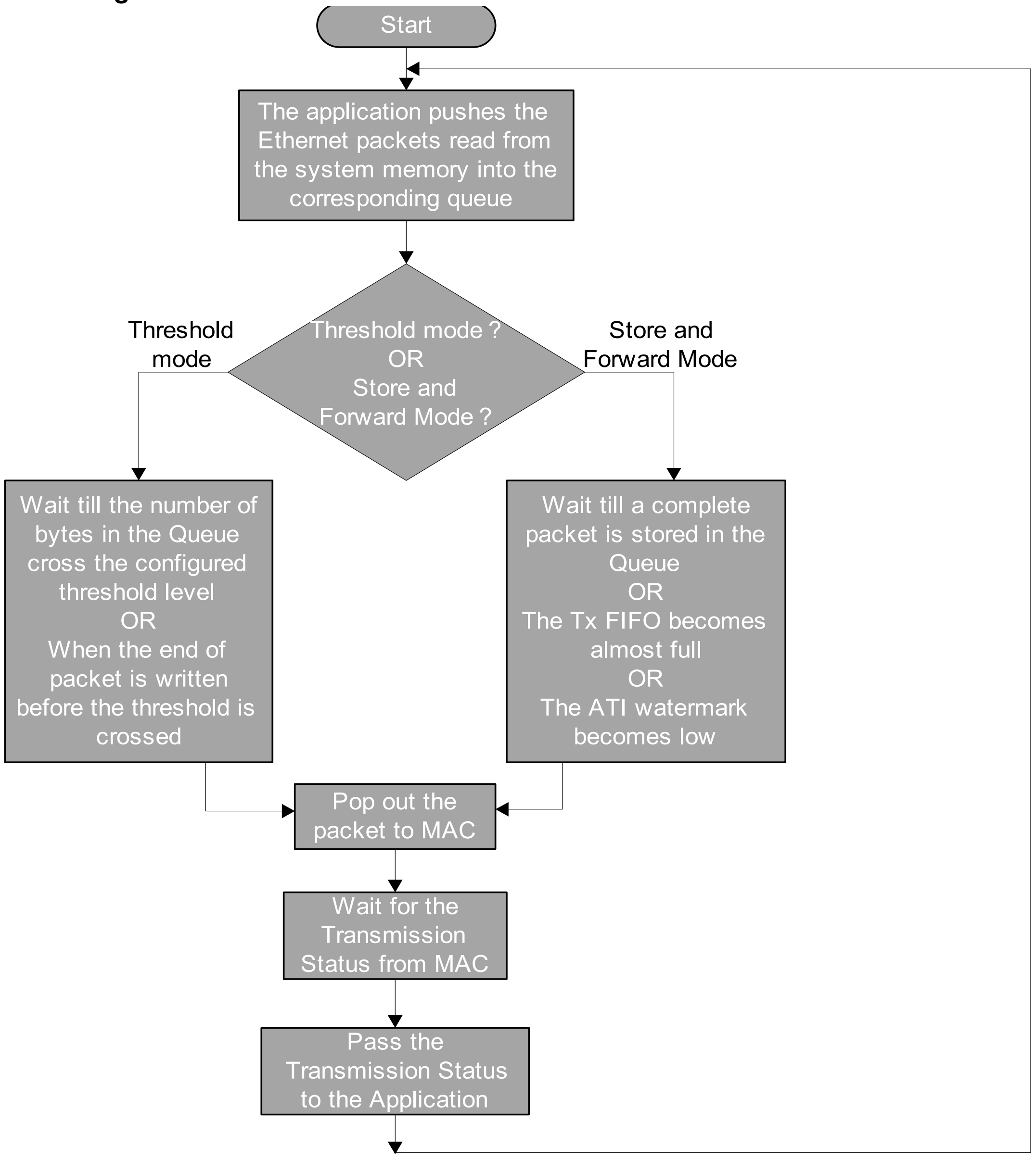
During a Transmit operation, the MTL block is a slave for the DMA controller. The general sequence of events for a Transmit operation is as follows:

1. If the system has data to be transferred, the DMA controller (if enabled) fetches the data from the application through the AHB, AXI, or Native DMA master interface and starts forwarding it to the MTL. The DMA descriptor memory contains the information related to packet control which is used to drive the ATI Control Word. The DMA data buffer memory contains the packet data.
2. The MTL pushes the data received from the DMA into the corresponding queue. This process continues until the EOP is transferred.

3. When the threshold level is crossed or a full packet of data is received into the queue, the MTL reads the packet data and drives it to the MAC.
4. The queue controller continues to transfer data from the queue until a complete packet is transferred to the MAC.
5. When the packet transfer is complete, the MTL receives the status from the MAC and notifies the DMA controller.

Figure 2-35 shows the MTL single-packet transmit operation.

Figure 2-35 MTL Single Packet Transmit Flow


Start
The application pushes the Ethernet packets read from the system memory into the corresponding queue
Threshold mode
Threshold mode ? OR Store and Forward Mode ?
Store and Forward Mode
Wait till the number of bytes in the Queue cross the configured threshold level OR When the end of packet is written before the threshold is crossed
Wait till a complete packet is stored in the Queue OR The Tx FIFO becomes almost full OR The ATI watermark becomes low
Pop out the packet to MAC
Wait for the Transmission Status from MAC
Pass the Transmission Status to the Application

Note When the Disable Transmit Status in MTL option is selected or Bit 1 (DTXSTS) of MTL_Operation_Mode register is set, the MTL does not provide any status to the DMA.

2.4.1.4.2 Transmit Operation - Two Packets in the Buffer

The following flow is valid when the core is operating in OSP mode, that is, when bit 4 (OSP) of the DMA_CH(#i)_TX_Control (for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1) register of the DMA channel is set to 1.

The flow is similar to the flow described in "Single-Packet Transmit Operation" on page 90". However after fetching a packet DMA instead of waiting for the status, it continues to fetch another packet if available in the system memory. So, MTL can receive the second packet while it is processing the first packet. This flow improves the performance because the DMA can process two packets back to back before waiting for the status of the first packet.

2.4.1.4.3 Relationship Between the Number of Tx DMA Channels and Tx Queues

The number of Tx DMA channels is always equal to the Tx queues selected in your configuration. This is because the DMA is designed to arbitrate among multiple channels (to fetch the descriptor and packet data from the system memory) in terms of PBLs (Programmable Bursts Lengths). However, the Tx queues are packet-based storage memory. Therefore, one to one mapping is required between the DMA channels and Tx queues to maintain the packet-level coherency.

2.4.1.4.4 Transmit Operation - Multiple Packets in Buffer

In the EQOS-MTL configuration, the Tx FIFO can be configured to accept more than two packets at a time. This option limits the number of status words that can be stored in the MTL before it is transferred to the DMA or application. By default, this number is limited to two but it can be configured for 4 or 8. When the MTL queue accepts the number of packets equal to the depth of the Status FIFO, it stops accepting further packets unless the Tx Status is given and accepted by the application.

2.4.1.4.5 Retransmission During Collision

While a packet is being transferred from the MTL to the MAC, a collision event can occur on the MAC line interface in half-duplex mode. The MAC indicates a Retry attempt to the MTL by giving the status even before the EOP is transferred from the MTL. The MTL then enables the retransmission by popping out the packet again from the queue.

After more than 96 bytes (or 548 bytes in 1000-Mbps mode) are read out towards the MAC, the Queue Controller frees up that space and makes it available to the application or the DMA to push in more data. This means that the retransmission is not possible after this threshold is crossed or when the MAC indicates a late-collision event.

When a packet transmission is aborted because of underflow and a collision event immediately follows (initiating a retry), retry has higher priority than abort, as described in Table 2-15.

2.4.1.5 Transmit Status Word

At the end of transfer of the Ethernet packet to the MAC and after the MAC completes the transmission of the packet, the MTL provides the Transmit status validity by asserting ati_txstatus_val_o[] on respective queues.For configurations with a single queue, ati_txstatus_o indicates the status of the queue when ati_txstatus_val_o is asserted. The application can read the status by asserting the ati_txstatus_ack_i signal.

For configurations with multiple Tx queues, the ati_txstatus_val_o signal indicates the availability of status on individual queues and application should read the status on ati_txstatus_o[17:0] by driving the Tx queue number on ati_txsqnum_i and by asserting ati_txstatus_ack_i.

The detailed description of the Transmit Status is the same as for Bits[17:0] of TDES3 normal descriptor in write-back format, given in Table 21-10. If IEEE 1588 Timestamp feature is enabled, the MTL returns 64-bit timestamp of the packet, along with the Transmit status of ATI on ati_txstatus_o[81:18]. A valid timestamp is indicated by an active high on ati_txstatus_o[17].

If the Disable Transmit Status in MTL option is selected while configuring the core, the Transmit Status is not provided to the application or DMA. When this option is selected, the application (or DMA) can push as many packets as possible into the Tx queues because the application or DMA is not required to read the status words from the Status FIFO. If this option is not selected, you can program Bit 1 (DTXSTS) of MTL_Operation_Mode register to disable the dependency of the application or DMA to read the status words from the status FIFO.

Note If the Enable IEEE 1588 Timestamp Support option is not selected, the width of ati_txstatus_o[] is 18. If this options is selected, the width of the ati_txstatus_o is 82. The upper 64-bits indicate the timestamp captured for that packet if the IEEE 1588 timing stamping is enabled for the packet through Packet Control Word (Bit 19).

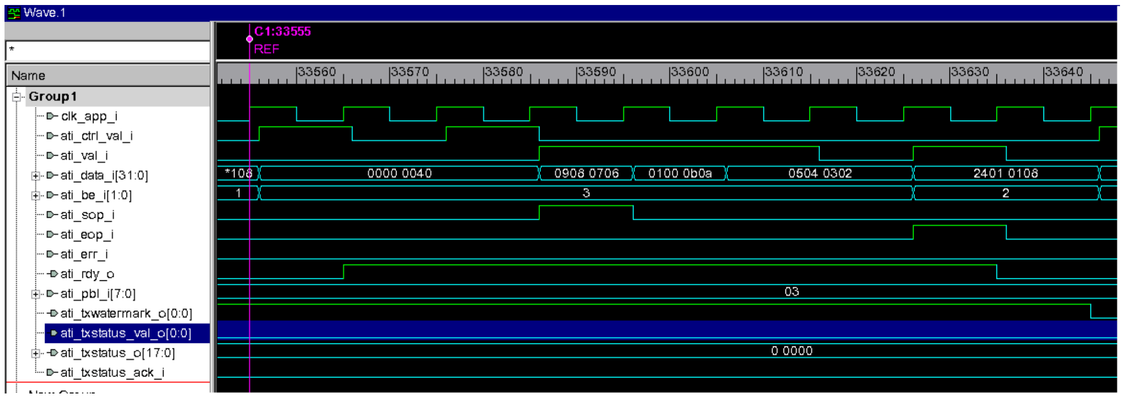
2.4.1.6 Transmit Path Timing

This section describes the timing specifications for the MAC Transaction layer (MTL) interface signals in the Transmit path.

2.4.1.6.1 Application Transmit Interface

Figure 2-36 illustrates the timing specifications for the Application Transmit Interface (ATI) signals. The handshake mechanism involved in packet transmission is as follows:

Figure 2-36 Application Transmit Interface (ATI) Timing (1 of 2)



1. Before starting a burst transfer of data to ATI, the application should check whether the MTL Tx queue has space to accommodate the burst. The length of the proposed burst transfer is put on ati_q#_pbl_i on the corresponding queue.
2. In response, the MTL asserts or de-asserts ati_txwatermark_o[ ] on the corresponding queue one clock later, indicating whether it can accommodate the burst data transfer or not.

3. At any point during the transfer, the MTL can alert the application to stop the transfer by de-asserting the ati_rdy_o signal. The ati_rdy_o signal is de-asserted only when either of the following happens:
   - The queue is almost full
   - The Tx queue is being flushed or checksum is being written if the Enable Transmit TCP/IP Checksum Offload option is selected

     The Tx queue-full event does not occur if the application ensures that enough space is available before the start of data transfer, as described in step 1 and step 2.
4. The application starts a packet transfer by asserting the ati_ctrl_val_i and ati_ctrl_type_i signals (for writing the control word) followed by the ati_val_i and ati_sop_i signals. The MTL accepts the control word or data whenever ati_rdy_o is asserted along with ati_ctrl_val_i or ati_val_i.

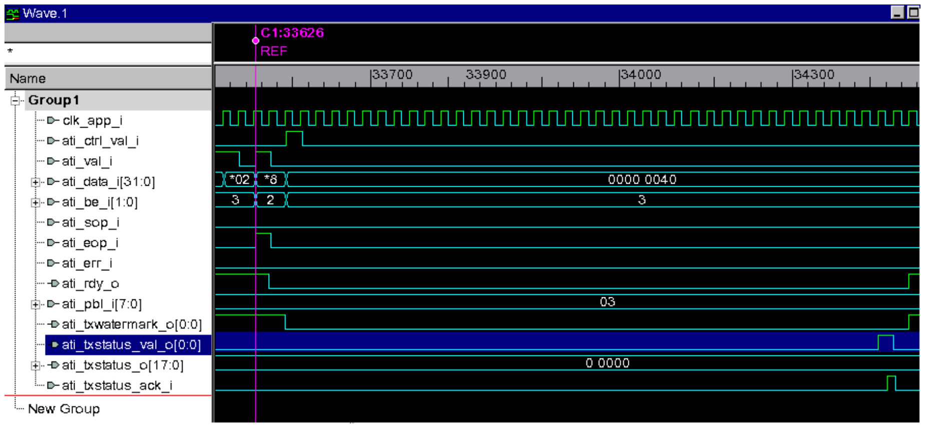
   In configurations with multiple Tx queues, the ati_qnum_i[] signal should be associated with the ati_c-trl_val_i and ati_val_i signals to indicate the Tx queue to which the control word or data is addressed.
5. When the application wants to transfer the last bytes of a packet, ati_eop_i should be asserted along with the byte enables (ati_be_i), indicating the valid number of data bytes on the bus. All data transfers except the EOP transfer must have the byte-enables as all-ones.
6. After transferring the packet to the Ethernet PHY, the MAC gives the transmission status to the MTL (not shown in the diagram).
7. In configurations with a single queue, the MTL gives the Transmit status of the packet to the application through ati_txstatus_o by asserting ati_txstatus_val_o. The application accepts the status by asserting ati_ack_i.

   In configurations with multiple queues, the MTL asserts the ati_txstatus_val_o[] to indicate the status availability on the corresponding queue. The application should read the status on ati_txstatus_o[] by driving Tx queue number on ati_txsqnum_i[] and asserting ati_txstatus_ack_i.
8. The application need not wait for step 6 and step 7 to start the transfer of next packet on the ATI. It can go to step 2 by driving the new PBL on ati_q#_pbl_i to check for the watermark.

   Figure 2-36 shows the application driving the control word followed by the packet data.

   Figure 2-37 shows the status (driven by MTL) of the packet after transmission is complete.

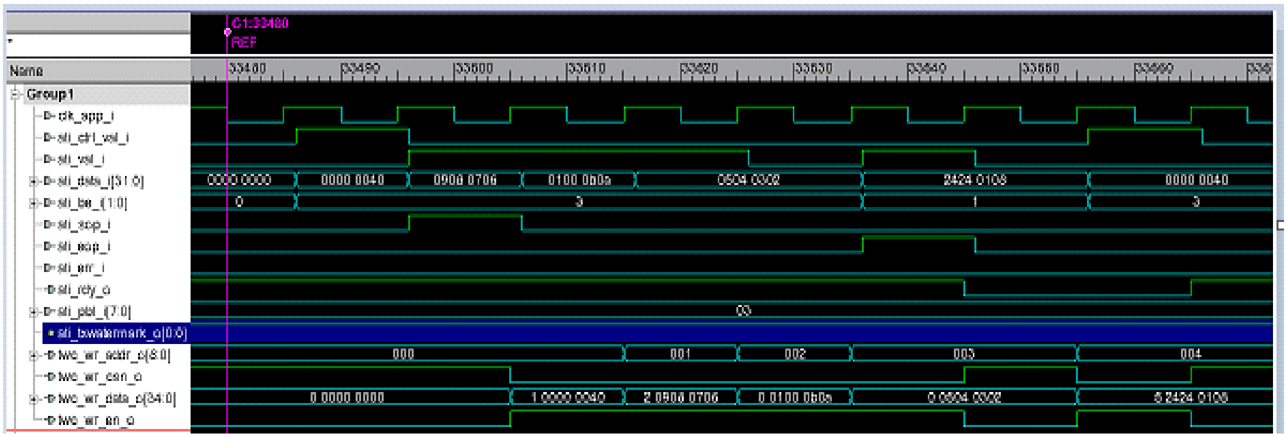
Figure 2-37 Application Transmit Interface (ATI) Timing (2 of 2)



2.4.1.6.2 Transmit Queue Write Interface

The MTL Tx Queue controller transfers the data accepted through ATI to the external memory. Timing specifications for the memory interface signals (twc_wr_* signals) are shown in Figure 2-38. This figure also shows the timing relationship of the memory interface signals with respect to the ATI signals.

Figure 2-38 Transmit Queue Write Interface Timing



The external memory used for storing packet must be synchronous two-port RAM as shown in Figure 2-53 on page 112.

2.4.1.6.3 Transmit Queue Read Interface

The Tx Queue Read controller reads the data from the Tx queue and transfers it to the MAC on the MAC Transmit Interface (MTI). The timing relationship of the MTI signals with the Transmit Queue Read Interface signals is shown in Figure 2-39. The external memory of Tx Queue is configured as synchronous RAM in the figure.

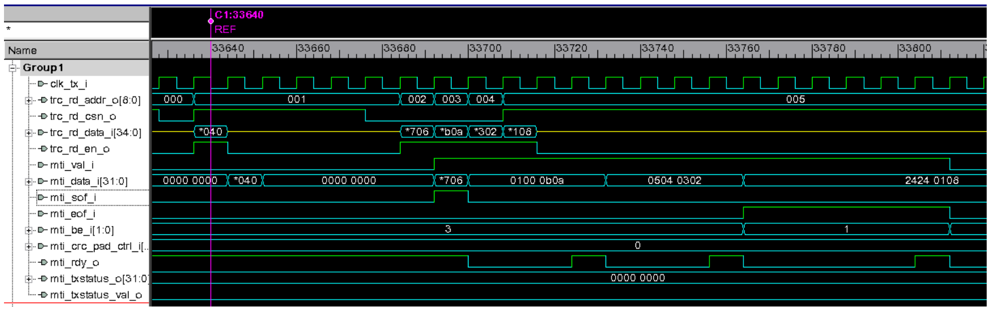
The sequence of events during the transfer of a packet from Tx Queue to the MTI is as follows:

1. When the Tx queue has data (not shown in the diagram), the Read Controller starts the read transfers by asserting the trc_rd_addr_o and trc_rd_csn_o signals. The assertion of trc_rd_en_o is delayed by 1 clock cycle as per synchronous RAM timing requirements. The first location of the packet has Packet Control and Packet Length. These are required to assert some of the sideband signals (such as mti_crc_pad_ctrl_i) of the MAC.

Note In configurations with multiple Tx queues, the arbitration scheme selects the nonempty Tx queue for transmission. Before Tx queue arbitration, the first location from each Tx queue is pre-fetched (containing the packet control and packet length). The packet length information is required for Tx queue arbitration (WFQ and DWRR arbitration schemes).

2. The data is read on the trc_rd_data_i bus in the next clock cycle. Because the data is directly transferred to the mti_data bus to the MAC (MTI), mti_val_i is also asserted. The mti_sop_i signal is also asserted along with mti_val_i in Figure 2-39 because the data corresponds to start of packet.
3. The Tx Queue Read Controller can pre-fetch data from three locations (two in Asynchronous RAM port) so that it can sustain continuous transfers to the MTI interface without any delay.
4. If the MAC accepts the data (active mti_rdy_o), more data is read from the queue. If the MAC deasserts mti_rdy_o, the read operations are suspended in the next clock cycle and trc_rd_csn_o is deasserted.

Figure 2-39 Transmit Queue Read Interface Timing



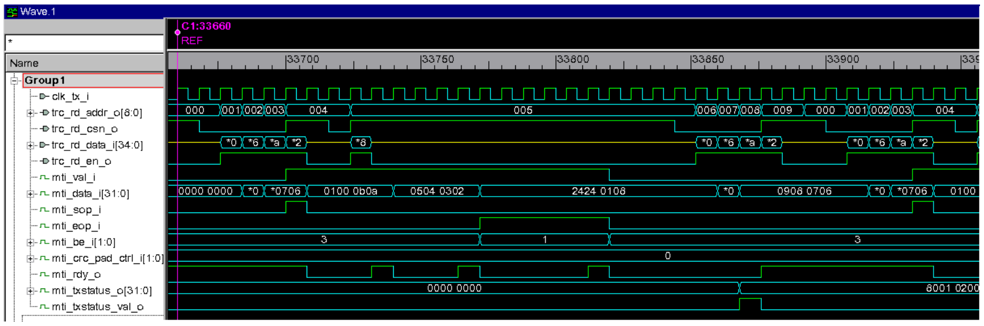
2.4.1.6.4 Transmit Packet Retransmission

The MAC transmitter may abort the transmission of a packet because of collision, Tx Queue underflow, loss of carrier, jabber timeout, no carrier, excessive deferral, and late collision. When packet transmission is aborted because of collision, the MAC requests retransmission of the packet. The sequence of events shown in Figure 2-40 is as follows:

1. The MTL starts the transfer of a packet to the MAC by asserting mti_val_i and mti_sop_i.

   The subsequent data is sent when the MAC asserts the mti_rdy_o signal.
2. When the MAC experiences a collision on the Ethernet bus (not shown in diagram), packet transmission is suspended first by keeping mti_rdy_o de-asserted.
3. The MAC transfers the transmission status on mti_txstatus, and requests retransmission by asserting Bit 31 of mti_txstatus.
4. The MTL retransmits the packet. The Tx Queue read address (trc_rd_addr) rewinds back to the start of packet address 0x000 for retransmission.

Figure 2-40 Transmit Packet Retransmission Timing



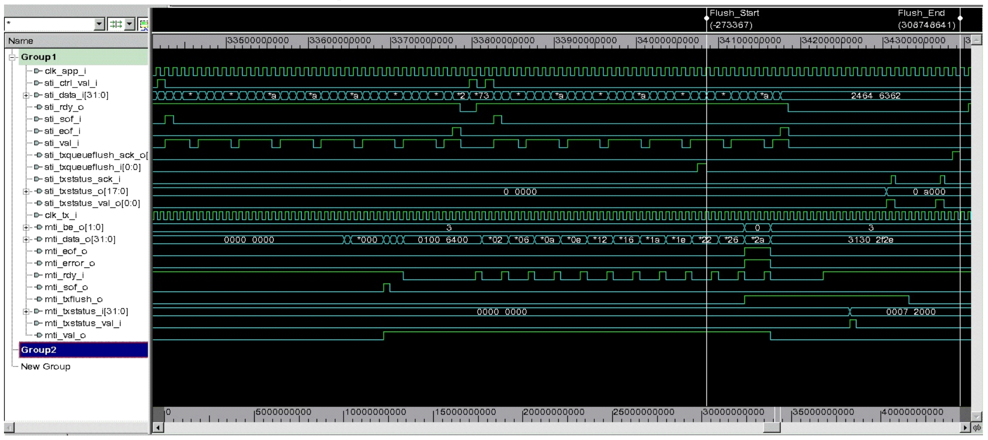
2.4.1.6.5 Transmit Queue Flush Operation

The MTL allows a Tx Queue to be flushed at any moment through Bit 0 of MTL_TxQ[n]_Operation_Mode register.

1. The application pushes two packets into the Tx queue on the ATI interface.
2. The application performs a write to the Transmit Operation mode register of corresponding MTL queue and sets the Flush Transmit Queue bit.
3. The application reads back the same register to check the status of the Flush operation. The bit[0] is asserted high during CSR read access, indicating the progress of the Flush operation.

4. If the MTL has already transferred a (partial) packet to the MAC, then it terminates that immediately and clears the Tx Queue. The MTL waits for the status of the partial/full packet transfers to the MAC.
5. The MTL indicates the completion of the Flush operation by asserting ati_txfifoflush_clr (internal), which clears Bit 0 in Transmit Operation Mode register of corresponding MTL queue.

Figure 2-41 Transmit Queue Flush Operation



2.4.2 Receive Path

The MTL Rx module receives the packets from the MAC and pushes them into the Rx Queue. The status (fill level) of the queue is indicated to the application or DMA when it crosses the configured Receive threshold (RTC bits[1:0] of MTL_RxQ0_Operation_Mode register of corresponding MTL queue), or the complete packet is received. The MTL also indicates the fill level of the queue so that the DMA can initiate preconfigured burst transfers towards the AHB or AXI interface.

Sections Receive Operation through Receive Status Word Format describe receive operations for the MTL Layer. Timing diagrams are provided in "Receive Path Timing Diagrams" on page 108.

2.4.2.1 Receive Operation

During a Receive operation, the MTL is a slave for the MAC. The general sequence of events is as follows:

1. When the MAC receives a packet, it indicates the availability of receive data.
2. The MAC indicates the SOP and EOP delimiters.
3. The MTL accepts the data and pushes it into corresponding Rx queue.
4. After the EOP is transferred, the MAC drives the status word which is also pushed into the corresponding Rx queue by the MTL.

Note In Threshold (cut-through) mode, the status words are stored after the packet EOP. In store-and-forward mode, the location for the maximum status words are reserved before writing the SOP and the status is written to reserved locations after writing the EOP.

5. If IEEE 1588 timestamp feature is enabled and the 64-bit timestamp is available along with the packet status, it is pushed into the Rx queue as a part of the status word. Therefore, in 32-bit data bus mode, two additional locations are taken per packet to store the timestamp in the Rx queue. In 64-bit, 128-bit, or 256-bit mode, one additional location is taken.
6. The MTL takes the data out of the queue and sends it to the DMA depending on the mode:

2.4.2.2 Threshold Mode

In the (default) Threshold mode, the MTL reads the data and indicates its availability to the application or DMA when one of the following occurs:

- data bytes equal to the threshold amount are written to the Rx queue (RTC bits[1:0] of MTL_RxQ0_Operation_Mode register of corresponding MTL queue)
- a full packet of data is received into the queue

Note When the split header feature is enabled or when multiple Rx DMA is present in a configuration, the MTL operates in dynamic threshold mode even though it is programmed for threshold mode. In dynamic threshold mode, MTL starts forwarding to the application only after it receives threshold number of bytes, split-header length, and DMA number from MAC.

Single Queue Configuration:

1. (Optional) The application asserts ari_q#_pbl_i and waits for ari_rxwatermark_o.
2. The MTL indicates data availability by asserting ari_val_o along with the data on ari_data_o, and byte enables on ari_be_o. During first transfer MTL also asserts ari_sop_o to indicate the start of the packet.
3. In turn, the application or DMA can accept the data by asserting ari_ack_i. Steps step b and step c are repeated till the end of the packet delimiter is received (as indicated by ari_eop_o).
4. After the entire packet data is received the MTL asserts ari_rxstatus_val_o and the first status word on ari_data_o. The first status word also indicates the number of status words that follow.
5. The application accepts the status by asserting ari_ack_i. Steps step d and step e are repeated till all the status words are sent out.

Note When Rx timestamp status is available then the transfer takes place as described in steps 4 and 5 where the MTL indicates the timestamp availability by asserting ari_timestamp_val_o.

Multiple Queue Configuration

1. (Optional) The application can assert ari_q#_pbl_i signal and wait for ari_rxwatermark_o on the corresponding queue. The assertion of ari_rxwatermark_o assures the application that the MTL has ari_q#_pbl_i amount of data ready in the corresponding Queue.
2. The application asserts the ari_ready_i (per Queue) signal, which indicates that the application is ready to accept the ari_q#_pbl_i amount of data from the MTL on the corresponding Rx Queue.
3. The MTL then internally arbitrates among Queues on which ari_ready_i is asserted and selects one of the Queues and indicates the Queue number on ari_qnum_o.

4. The MTLindicates the data availability on the selected Queue by asserting ari_val_o along with the data on ari_data_o, and byte enables on ari_be_o. During the first transfer, MTL also asserts ari_sop_o to indicate the start of the packet.

2.4.2.3 Store-and-Forward Mode

In the store-and-forward mode (when Bit 5 of MTL_RxQ0_Operation_Mode register of a queue is set to 1), the initial Rx queue locations are reserved for the status words before writing the SOP. A packet is read out only after it is completely written into the Rx queue. In this mode, all error packets are dropped (if configured through Bit 4 of MTL_RxQ0_Operation_Mode register of a queue) such that only valid packets are read and forwarded to the application.

Single Queue Configuration

1. (Optional) The application asserts ari_q#_pbl_i and waits for ari_rxwatermark_o.
2. The MTL indicates the status availability by asserting ari_rxstatus_val_o along with the status on ari_-data_o. The first status word also indicate the number of status words that follow.
3. The application accepts the status word by asserting ari_ack_i.
4. Steps step b and step c are repeated till the end of the packet delimiter is received (as indicated by ari_eop_o).

Note When Rx timestamp status is available the transfer happens as described in steps 2 and 3 where the MTL indicates the timestamp availability by asserting ari_timestamp_val_o.

5. After transferring all the status words, the MTL indicates the data availability by asserting ari_val_o along with the data on ari_data_o, and byte enables on ari_be_o. During the first transfer, MTL also asserts ari_sop_o to indicate the start of the packet.
6. The application or DMA accepts the data by asserting ari_ack_i.
7. step e and step f are repeated till all the status words are sent out.

Multiple Queue Configuration

1. (Optional) The application asserts ari_q#_pbl_i and waits for ari_rxwatermark_o on the corresponding queue. The assertion of ari_rxwatermark_o assures the application that the MTL has ari_q#_pbl_i amount of data ready in the corresponding Queue.
2. The application asserts the ari_ready_i (per Queue) signal, which indicates that the application is ready to accept ari_q#_pbl_i amount of data from the MTL on the corresponding Rx Queue.
3. The MTL then internally arbitrates among Queues on which ari_ready_i is asserted and selects one of the Queues and indicates the Queue number on ari_qnum_o.

Note step d and step e are applicable only when the currently selected queue has a new packet to transfer. The status word transfer in step d and step e are not a part of the current PBL (as indicated by ari_q#_pbl_i of the currently selected Queue) requests. Instead it is additional information provided before the PBL amount of data transfer that begins in step f.

4. The MTL indicates the status availability by asserting ari_rxstatus_val_o, along with the status on ari_-data_o for the selected queue as indicated by ari_qnum_o. The first status word also indicates the number of status words that follow.
5. The application accepts the status word by asserting ari_ack_i.

Note When Rx timestamp status is available then the transfer happens as described in step d and step e where the MTL indicates the timestamp availability by asserting ari_timestamp_val_o.

6. The MTL indicates the data availability on the selected Queue by asserting ari_val_o along with the data on ari_data_o, and byte enables on ari_be_o. During first transfer, MTL also asserts ari_sop_o to indicate the start of the packet.
7. The application or DMA accepts the data by asserting ari_ack_i.
8. step f and step g are repeated till the requested PBL amount of the data is transferred (as indicated by ari_q#_pbl_i for the currently selected Queue) or end of the packet delimiter is accepted (as indicated by ari_eop_o).

   All steps are repeated till all the status words are sent out.

2.4.2.4 Multi-Packet Receive Operation

In Threshold mode, the packet status is available immediately after the packet data. In store-and-forward mode, the packet data is available after the packet status. The MTL is capable of storing any number of packets into the queue as long as it is not full.

If the MAC receives a packet when the corresponding Rx queue is full, the MTL ignores that packet and the ari_fifo_ovf_o[] output pulse is generated on the corresponding Rx queue. In addition, the MTL increments the overflow counter in the MTL_RxQ0_Missed_Packet_Overflow_Cnt of corresponding queue.

2.4.2.5 Error Handling in Receive Operation

If the MTL Rx queue is full before it receives the EOP data from the MAC, the following happens:

1. An overflow is declared
2. The whole packet (including the status word) is dropped
3. The overflow counter in the DMA (Overflow Counter register of corresponding MTL queue) is incremented.

This is true even if Bit 4 (FEP) of MTL_RxQ0_Operation_Mode register of corresponding MTL queue is set.

If the start address of such a packet has already been transferred to the Read Controller, the rest of the packet is dropped and a dummy EOP is written to the queue along with the status word with overflow status. The status indicates a partial packet because of overflow. In such packets, the Packet Length field is invalid. If the MTL Receive Queue is configured to operate in the store-and-forward mode and the length of the received packet is more than the queue size, overflow occurs and all such packets are dropped.

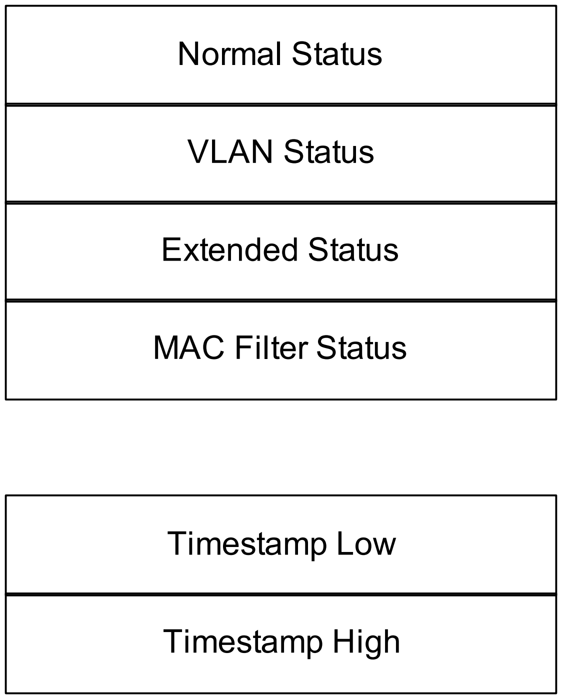
The MTL Rx Control logic can filter error and undersized packets, if enabled by using the FEP and FUP bits of MTL_RxQ0_Operation_Mode register of corresponding MTL queue. If the start address of such a packet has already been transferred to the Rx Queue Read Controller, that packet is not filtered. The start address of the packet is transferred to the Read Controller after the packet crosses the receive threshold set by Bits[1:0] of MTL_RxQ0_Operation_Mode register of corresponding MTL queue.

If the MTL Receive Queue is configured to operate in the store-and-forward mode, all error packets can be filtered and dropped. For the application or DMA to flush the error packet being read from the queue, it must assert the ari_pkt_flush_i signal. The MTL then stops transferring data to the application (DMA). It internally reads the rest of the packet and drops it. The MTL then starts the transfer of next packet (if available).

2.4.2.6 Receive Status Word Format

The Rx packet status words are sent to the application after the packet data in Threshold mode and before the packet data in the Store and Forward Mode as explained in "Multi-Packet Receive Operation" on page 100. Figure 2-42 illustrates the format of the Receive Status Word order for 32-bit data width configuration.

Figure 2-42 Format of the Receive Status Word Order for 32-bit Datawidth Configuration



The Normal status Word is always present. The contents of the Normal Status Word indicates the number of status words that follow.

The VLAN Status Word is present only when Bit [25] or Bit [26] of the Normal status field is set to 1. Bit[25] or Bit[26] is set only when corresponding VLAN Status Word is non-zero, this is applicable even when received VLAN Tag value is 0.

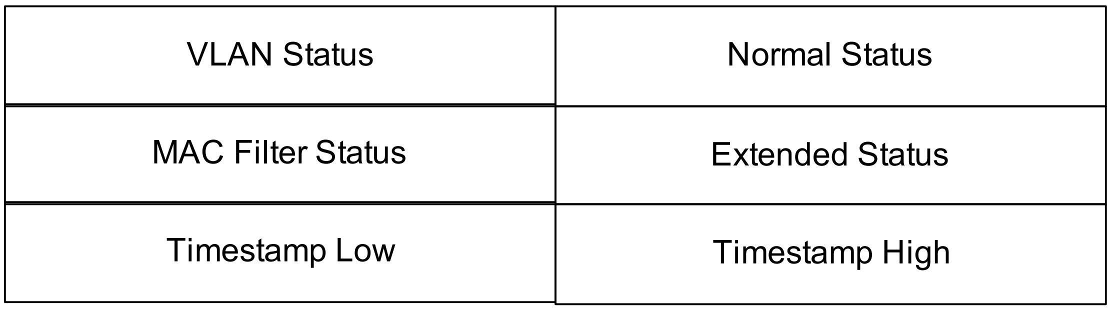
The Extended Status Word is present only when the Bit[27] of the Normal status field is set to 1. Bit[27] is set only when Extended Status Word is non-zero.

The MAC Filter Status word is present only when the Bit[29] of the Normal status field is set to 1. Bit[29] is set only when the corresponding MAC Filter Status Word is non-zero, this is applicable even when the received DA matches MAC Address 0 register with Bit[18] reset.

The Timestamp Status Low and Timestamp Status High status words are present only when bit [30] of the Normal Status Word is set to 1 and bit [15] of the Extended status Word is set to 0.

Figure 2-43 shows the format of the Receive Status Word order for 64-bit datawidth configuration.

Figure 2-43 Format of the Receive Status Word Order for 64-bit Datawidth Configuration



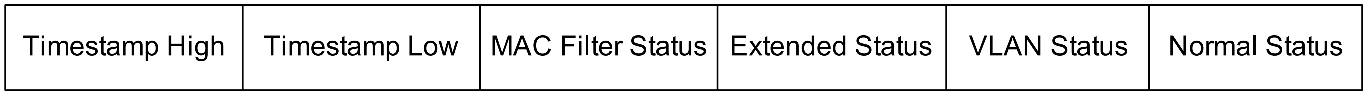
The Normal Status Word or VLAN Status Field is always present. Its contents indicate the number of status words that follow.

The Extended Status Word or Mac Filter Status word is present only when bit [27] or bit [29] of Normal status Word is set to 1.

The Timestamp Status Low and Timestamp Status High status words are present only when bit [30] of Normal Status Field is set to 1 and bit [15] of Extended status Field is set to 0.

Figure 2-44 shows the format of the Receive Status Word order for 128-bit datawidth configuration.

Figure 2-44 Format of the Receive Status Word Order for 128-bit Datawidth Configuration



Format of Normal Status Word

Table 2-9 describes the format of the Normal Status Word.

Table 2-9 Normal Status Word Format

Field	Description
31	This field reserved
30	Timestamp Status Received The Timestamp status is received from the MAC. **Note:** The Timestamp status can be received from the MAC but dropped inside the MTL Queue because of the unavailability of the space for the Timestamp Status. The availability of the timestamp status is indicated through the bit[10] of the Extended Status Word.
29	MAC filter Status Available When set, it indicates that the MAC Filter Status is present.
28	This field is reserved.
27	Extended Status Available When set, it indicates that the Extended status is present.
26	Inner VLAN Tag Status Available When set, it indicates that Inner VLAN Tag stripped from the received packet is on VLAN Status Word[31:16]. When this bit is set then the VLAN Status Word will also be available in the Rx status Word.

Field	Description
25	Outer VLAN Tag Status Available When set, it indicates that Outer VLAN Tag stripped from the received packet is present on VLAN Status Word [15:0]. When this bit is set then the VLAN Status Word is also present in the Rx Status Word.
24	CRC Error When set, it indicates that a cyclic redundancy check (CRC) error occurred on the received packet.
23	Giant Frame When set, it indicates that the packet length exceeds the maximum Ethernet specified size of 1518, 1522, or 2000 bytes (9018 or 9022 bytes if jumbo packet enable is set). Giant frame is only a packet length indication, and it does not cause any packet truncation.
22	Watchdog Timeout When set, it indicates that the RPE module has received a packet with byte count greater than 2,048 (10,240 if Jumbo packet is enabled) bytes (DA + SA + LT + DATA + PAD + FCS). This bit is not set if the watchdog timer is disabled in the MAC Register 0 (MAC Configuration Register). However, even if the watchdog timer is disabled, this bit is set when the frame is greater than 32 KB in size.
21	Overflow Error When set, this bit indicates that the received frame is damaged because of buffer overflow in Rx Queue.Note: This bit is set only when the MTL transfers a partial frame to the application. This happens only when the Rx FIFO is operating in the threshold mode. In the store-and-forward mode, all received partial frames which encounters overflow are dropped completely in Rx Queue.
20	GMII Error When set, it indicates that the gmii_rxer_i was asserted during the reception of this packet. This error also includes carrier extension error in GMII and half-duplex mode. Error can be of less/no extension, or error (rxd = 1f) during extension.
19	Dribbling Bit When set, it indicates that the packet contained a non-integer multiple of eight bits (valid only in MII mode). This bit is not valid if a collision is seen or if runt packet bits are set. If this bit is set and the CRC error is reset, then the packet is valid.
18:16	Length/Type Field This field indicates if the packet received was a length packet or a type packet. The encoding of the 3 bits is as follows: ■ 3'b000: The packet is a length packet ■ 3'b001: The packet is a type packet ■ 3'b100: The packet is a type packet with VLAN Tag ■ 3'b101: The packet is a type packet with Double VLAN Tag ■ 3'b110: The packet is a MAC Control packet type ■ 3'b111: The packet is a OAM packet type ■ 3'b010-3'b011: Reserved **Note:** The packet type indication in this field is purely based on the LT field of received packet. For example, VLAN Tag or Double VLAN Tag indication does not ensure that corresponding VID match or VLAN filter pass.

Field	Description
15	Error Summary Indicates the logical OR of the following error bits: ■ CRC Error ■ Dribble Error ■ Receive Error ■ Watchdog Timeout ■ Overflow Error ■ Giant Packet
14:0	Packet Length Indicates the length, in bytes, of the received packet (including pad if applicable), and/or the FCS field when the Auto Pad/CRC Strip bit is de-asserted. Otherwise, it indicates the length without pad and/or FCS field. It also includes the 2-byte IP Checksum appended after the FCS when the IP Checksum module is present and enabled by setting the IPC bit of MAC Register 0 (MAC Configuration Register),and if the received frame is not a MAC control frame. It indicates the length without FCS field for Type packets (Length/Type >= 0x600), when Type CRCStrip bit is set. It indicates the length without VLAN Tag for VLAN tagged packets, when VLAN Tag stripping is enabled

Format of VLAN Status Word

Table 2-10 describes the format of the VLAN Status Word.

Table 2-10 VLAN Status Word Format

Field	Description
31:16	Inner VLAN Tag This field indicated the inner VLAN Tag Value This is valid only when the when bit[26] (outer VLAN tag Available) of the Normal Status Field is set to 1
15:0	Outer VLAN Tag This field indicates the outer VLAN Tag value.This is valid only when the when bit[25] (outer VLAN tag Available) of the Normal Status Field is set to 1

Format of Extended Status Word

This field is identical to the RDES1 field of the DMA in Write back Format shown in Table 19-23 on page 1341

Format of MAC Filter Status Word

Table 2-11 describes the format of the MAC Filter Status Word.

Table 2-11 MAC Filter Status Word Format

Field	Description
31:29	Layer 3 and Layer 4 Filter Number Matched These bits indicate the number of the Layer 3 and Layer 4 Filters that matched the received frame. ■ 000: Filter 0 ■ 001: Filter 1 ■ 010: Filter 2 ■ 011: Filter 3 ■ 100: Filter 4 ■ 101: Filter 5 ■ 110: Filter 6 ■ 111: Filter 7 This field is valid only when bits 28 or 27 are set high. When more than one filter matches, these bits give only the lowest filter number.
28	Layer 4 Filter Match When set, this bit indicates that the received frame matches one of the enabled Layer 4 Port Number fields. This status is given only when one of the following conditions is true: ■ Layer 3 fields are not enabled and all enabled Layer 4 fields match. ■ All enabled Layer 3 and Layer 4 filter fields match. When more than one filter matches, this bit gives the layer 4 filter status of the filter indicated by bits [31:29].
27	Layer 3 Filter Match When set, this bit indicates that the received frame matches one of the enabled Layer 3 IP Address fields. This status is given only when one of the following conditions is true: ■ All enabled Layer 3 fields match and all enabled Layer 4 fields are bypassed ■ All enabled filter fields match When more than one filter matches, this bit gives the layer 3 filter status of filter indicated by bits [31:29].
26:19	MAC Address Match/Hash Value When bit 18, Hash Filter Status is reset, this field contains the number of the MAC address register which matches the Destination address of the received packet. When bit 18, Hash Filter Status is set, this field contains the hash value that is computed by the MAC.The bit corresponding to the hash value in the hash filter register is set, and so the frame passes the hash filter.
18	Hash Filter Status When set, this bit indicates that the packet passed the MAC address hash filter. The bits [26:19] indicate the hash value.
17	DA Filter Fail When this bit is set, it indicates that the Destination Address filter failed. This means that this packet did not clear the DA filter.
16	SA Filter Fail When this bit is set, it indicates that the Source Address filter failed. This means that this packet did not clear the SA filter.

Field	Description
15	VLAN Filter status When set, this bit indicates that the VLAN Tag of the received packet passed the VLAN filter. This bit is valid only when DWC_EQOS_ERVFE is not enabled. If DWC_EQOS_ERVFE is enabled, the bit is redefined as Outer VLAN Tag Filter Status (OTS). For more details about OTS bit, see bit[15] “RDES2 Normal Descriptor (Write-Back Format)” on page 1343
14	Inner VLAN Tag Filter Status This bit is valid only when DWC_EQOS_ERVFE is enabled. For more details about ITS bit, see bit[14] “RDES2 Normal Descriptor (Write-Back Format)” on page 1343
13:6	Reserved
5	ND Advertisement Not Generated When set, it indicates that the MAC has not generated ND Advertisement for the received ND Solicitation packet. This bit is valid when bit 4 (ND Solicitation Packet Received) is set. This bit is set when the MAC receives ND Solicitation Packet with Authentication extension header or non-zero extension headers. This bit is reserved when Enable ND Offload feature is not selected.
4	ND Solicitation Packet Received When set, it indicates that ND Solicitation packet is received. This bit is reserved when Enable ND Offload feature is not selected.
3	Reserved
2	ARP Reply Not Generated When set, it indicates that the MAC has not generated ARP Reply for the received ARP Request packet. This bit is set when the MAC is busy transmitting ARP reply to an earlier ARP request (only one ARP request processed at a time). This bit is reserved when Enable ARP Offload feature is not selected.
1	Inner VLAN Tag Type When set, it indicates that Inner VLAN type is S-VLAN in the received packet.When reset, it indicates that Inner VLAN type is C-VLAN in the received packet. This field is reserved when Enable Double VLAN feature is not selected.
0	Outer VLAN Tag Type When set, it indicates that Outer VLAN type is S-VLAN in the received packet.When reset, it indicates that Outer VLAN type is C-VLAN in the received packet.

Format of Timestamp Status Low Field

This field is identical to the RDES0 field of the Context Descriptor of the DMA shown in Table 19-26 on page 1348.

Format of Timestamp Status High Field

This field is identical to the RDES1 field of the Context Descriptor of the DMA shown in Table 19-27 on page 1348.

2.4.2.7 Receive Path Timing Diagrams

Timing specifications for the MAC Transaction layer (MTL) interface signals in the receive path are illustrated in detail in the following sections. The timing diagrams are organized to show the sequence of transactions that occur when a received packet is transferred from the MAC to the application.

2.4.2.7.1 Receive FIFO Write Interface

A packet received by the MAC is transferred to the MTL over the MRI interface. The transferred data is accepted and written to the Rx Queue by the write control logic. (See Figure 2-45.)

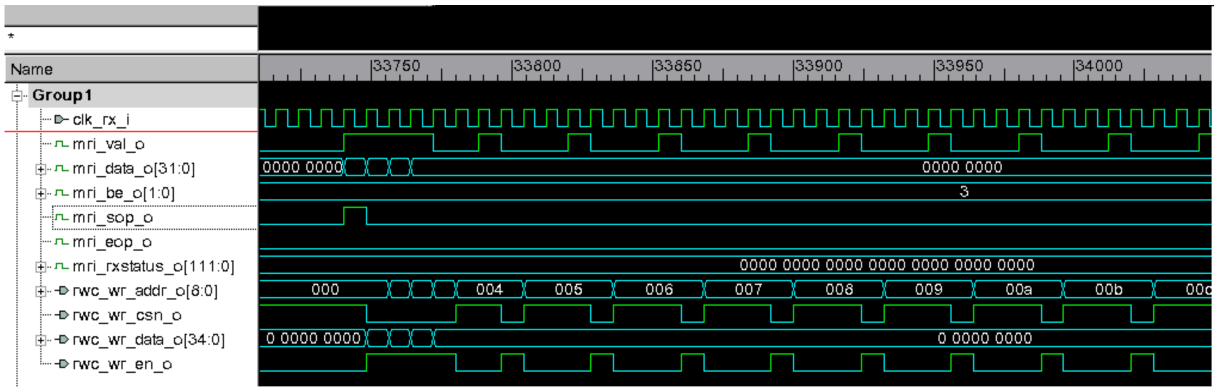
1. The MAC starts transferring the received packet to the MTL by asserting mri_val_o and mri_sop_o along with the corresponding 128-bit data on mri_data_o.

   This is directly transferred to the Rx queue in the next clock cycle. This can be seen by the assertion of rwc_csn_o and rwc_wr_en_o along with the corresponding address and data. The MSB 5 bits of rwc_wr_data correspond to the value on mri_eop_o and mri_be_o signals accepted by the MTL.

   In store-and-forward mode, the MTL reserves locations for the status word before writing the first packet data. In Threshold mode, the MTL does not reserve any locations for the status because the status is stored after the packet data.
2. The MAC completes the packet transfer by the asserting the mri_eop_o signal along with the valid data. The Receive packet status on mri_rxstatus_val_o is also valid with mri_eop_o.
3. After writing the EOP data, the Rx Queue Write logic transfers the received status to the external RAM in the next clock cycle.

   In Threshold mode, the status words are written after the packet data. In store-and-forward mode, the status is written to the locations that were reserved earlier (before the packet data).

Figure 2-45 Receive Queue Write Interface Timing



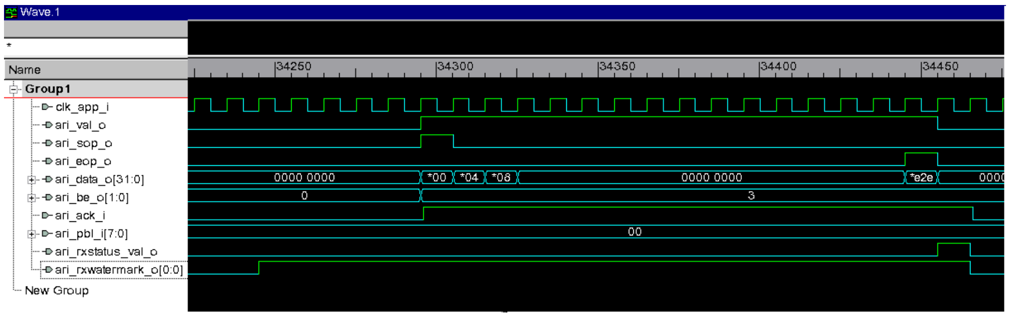
2.4.2.7.2 Application Receive Interface (ARI)

The data received from the MAC and stored in the Rx Queue is transferred to the Application Receive Interface with the following sequence of events, as shown in Figure 2-46

1. The application can check whether the Rx Queue has the amount of data (or the complete packet) to support the required burst length on ari_qnum_pbl_i.
2. The MTL updates ari_watermark_o on the corresponding queue in the next clock cycle which indicates that the MTL has enough data to sustain the burst transfer of length given on ari_pbl_i (in step 1).
3. In case of the Multiple Queue configuration, the application can drive ari_ready_i[] to indicate the queues the application is ready to service.

4. In configurations with multiple Rx queues, the MTL Read controller performs the arbitration among all non-empty queues, and all Queues for which ari_ready_i is asserted, and then selects one queue. The selected queue number is indicated through ari_qnum_o[] signal. The MTL indicates the availability of data by asserting the ari_val_o (in Threshold mode) or ari_rxstatus_val_o (in store-and-forward mode) signal along with the data. The ARI transfers the data or status in every clock cycle as long as data or status is available in the Rx queue.
5. The application accepts the data from the MTL by asserting ari_ack_i. The MTL updates the data in the next clock cycle if it is continuing the burst transfer. The application can pause the transfer by deasserting ari_ack_i, as shown in Figure 2-46.
6. The MTL transfers the remaining data by asserting the ari_val_o, ari_rxstatus_val_o, or ari_timestamp_val_o signal and the corresponding data or status. When the last few bytes of packet (less bytes than the data bus width) are being transferred, ari_eop_o is also asserted along with ari_be_o, indicating a valid number of bytes in that data phase.
7. In Threshold mode, the Receive packet status (indicated with the assertion of ari_rxstatus_val_o or ari_-timestamp_val_o) is transferred after the EOP data. Packet transfer is complete only after the ari_ack_i acknowledgment for status.

Figure 2-46 Application Receive Interface (ARI) Timing



2.4.2.7.3 ARI With Timestamp Feature in Threshold Mode

Figure 2-47 shows the ARI timing when IEEE 1588 Timestamp feature is enabled. After the ARI outputs the EOP data, as indicated by ari_eop_o high, it gives the timestamp value on ari_data_o. Asserting ari_timestamp_val_o validates the timestamp. In this figure, the 64-bit timestamp is output in two cycles, because the data bus is only 32-bits wide. After the application accepts the timestamp (by asserting ari_ack_i for two clocks), the Rx status is given on the data bus and validated with the assertion of ari_rxstatus_val_o.

Figure 2-47 ARI Timing With IEEE 1588 Timestamping Enabled

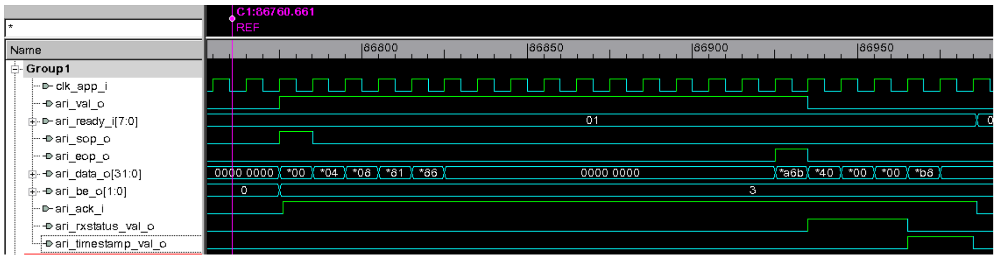
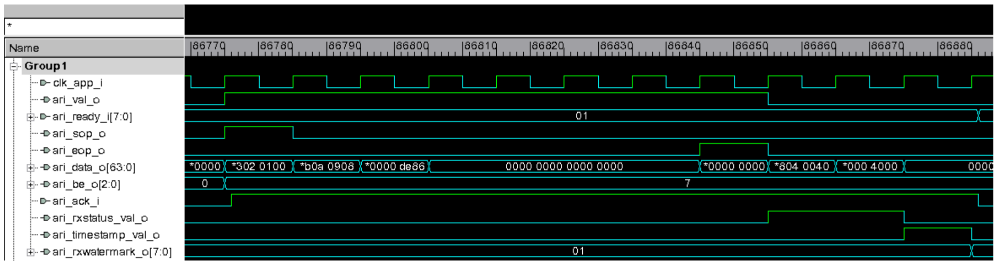


Figure 2-48 shows the timing of the timestamp transfer when the MAC is configured for a 64-bit data bus. The 64-bit timestamp is given in one cycle between the EOP data transfer and the Receive Status transfer.

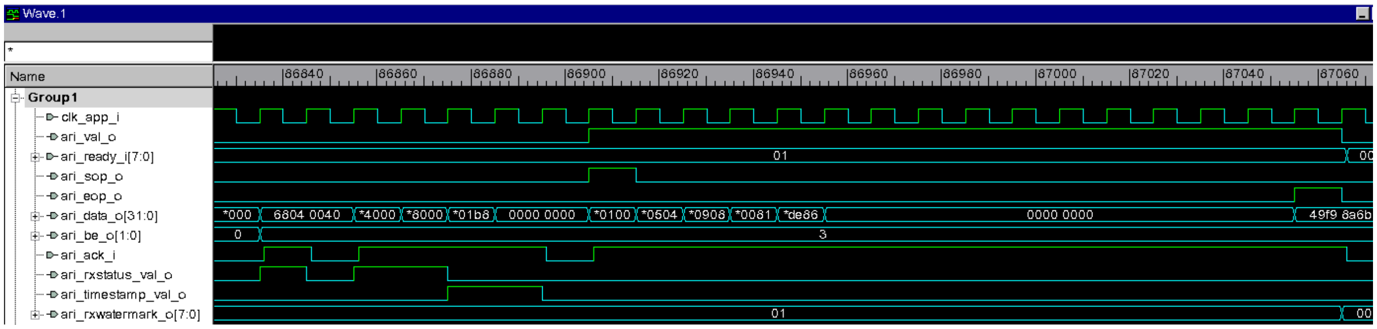
Figure 2-48 ARI Timing for 64-Bit Bus, IEEE 1588 Timestamp Feature Enabled



2.4.2.7.4 ARI With Timestamp Feature in Store-and-Forward Mode

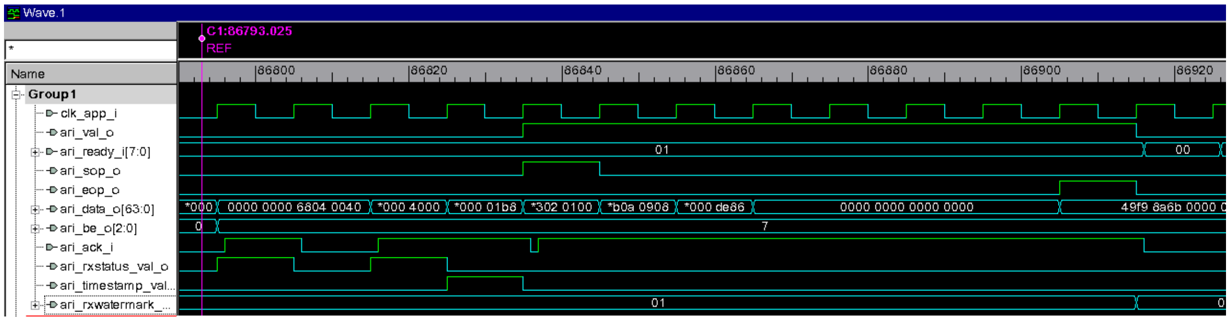
In store-and-forward mode, the MTL starts the packet transfer with the status by asserting the ari_rxstatus_val_o and ari_data_o signals containing the status. After the application accepts the status words by asserting ari_ack_i, the MTL asserts ari_timestamp_val_o to indicate the availability of the timestamp on ari_data_o.

Figure 2-49 ARI Timing with IEEE 1588 Timestamp Feature in Store-and-Forward Mode



In Figure 2-50, the 64-bit timestamp is output in two cycles because the data bus is only 32-bits wide. After the application accepts the timestamp (by asserting ari_ack_i for two clocks), the MTL starts driving the packet data by asserting the ari_val_o, ari_sop_o and ari_data_o.

Figure 2-50 ARI Timing for 64-Bit Bus, IEEE 1588 Timestamp Feature in Store-and-Forward Mode

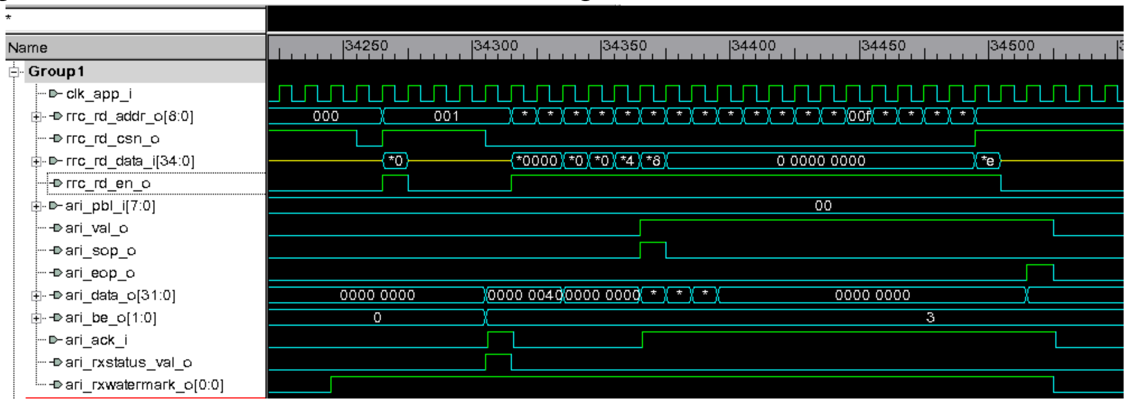


2.4.2.7.5 Receive Queue Read Interface

The queue read timing diagrams for the synchronous two-port RAM along with the timing relationship with signals on the ARI is shown in this section.

The MTL Rx Queue Read Controller pre-fetches the data (rrc_rd_* signals) from the Rx queue when it finds that the Rx queue is not empty and the application is ready to service the Queue (in case of multiple Queue configuration). The pre-fetching corresponds to maximum of three locations for three locations for synchronous RAM interface. This pre-fetching is done to sustain the burst data transfers without any delay on the ARI interface. The (pre-)fetched data is registered and given out on the ARI with respective ari_* signals. When the application acknowledges the data with ari_ack_i, further data is read from the queue. Figure 2-51 shows the Queue read interface timing for synchronous SRAM. The timing relationship of the rrc_rd_* signals is similar to that of trc_rd_* signals.

Figure 2-51 Receive FIFO Read Interface Timing



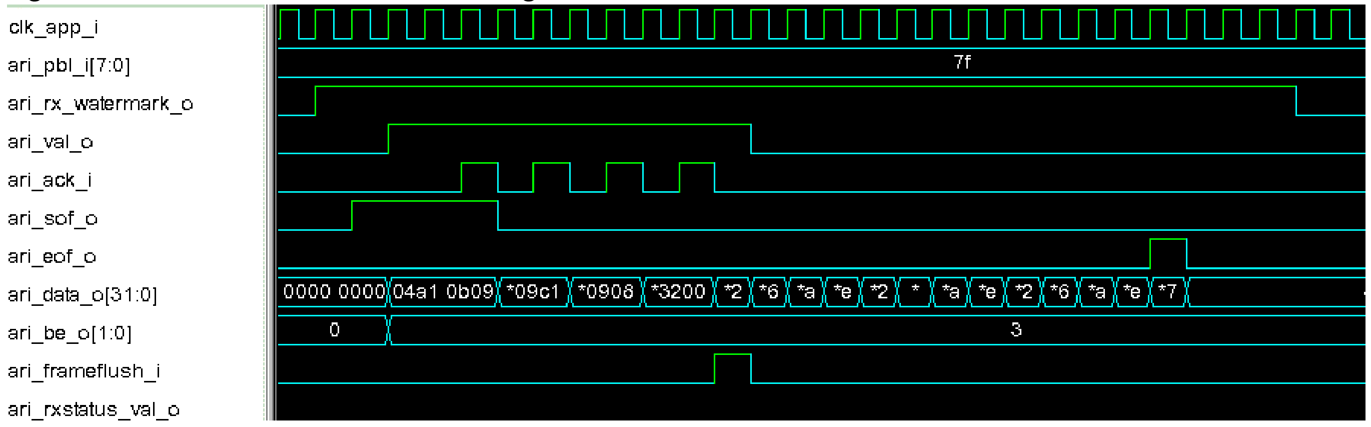
2.4.2.7.6 Receive Packet Flush

When the application decides to drop the packet (at the top of the Rx Queue) during the start, middle, or end of a packet transfer (ari_val_o, ari_rxstatus_val_o, or ari_timstamp_val_o is asserted), it can assert ari_packetflush_i for one clock on the corresponding Rx queue. The MTL response is shown in Figure 2-52

The MTL de-asserts ari_val_o, ari_rxstatus_val_o, ari_timestamp_val_o, and ari_rxwatermark_o when it finds an active pulse on ari_packetflush_i[] for the currently active queue number ari_qnum_o[]. It continues the Read transfer from the Rx queue but does not transfer it to ARI. It flushes the entire packet, including the status. The MTL is now ready to transfer the next packet.

If the ari_packetflush_i[] is asserted for currently inactive queue (there is no activity on the ARI interface for the queue to which ari_packetflush_i[] is issued), the MTL internally flushes the topmost packet in that queue.

Figure 2-52 Receive Packet Flush Timing



2.4.3 Interfacing With External Two-Port RAMs

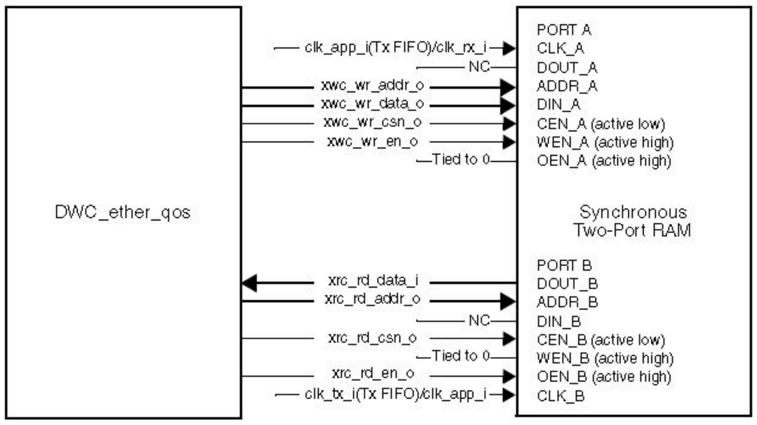
The DWC_ether_qos provides a generic memory interface for easy interconnection to any vendor-specific, two-port RAM for implementing the Tx queue and Rx queue. This section describes how the DWC_ether_qos can be interconnected with different types of two-port RAMs.

The two memory interfaces provided in DWC_ether_qos for connectivity with Transmit and Receive two-port RAMs are identical. Therefore, the connections with the Tx queue and Rx queue RAMs are illustrated in the following sections in a single diagram for each type of RAM. The “x” variable in the signal names can be replaced with a “t” for the Tx queue interface and an “r” for the Rx queue interface.

2.4.3.1 Low-Power, Synchronous Two-Port SRAM Connection

The synchronous, two-port SRAM shown in Figure 2-53 consists of two ports: Port A and Port B. These ports can be used for both read and write operations. The DWC_ether_qos (configured for synchronous RAM interface) uses this SRAM as an asynchronous FIFO for transferring data from one clock domain to another clock domain. Therefore, a port (Port A, for example) is used to write the packet data and the other port is used to read the packet data, with the writes and reads happening in different clock domains. (Consequently, the Read data (DOUT_A) from Port A and write data (DIN_B) to Port B are left unconnected in Figure 2-53)

Figure 2-53 Low-Power, Synchronous Two-Port SRAM Connection



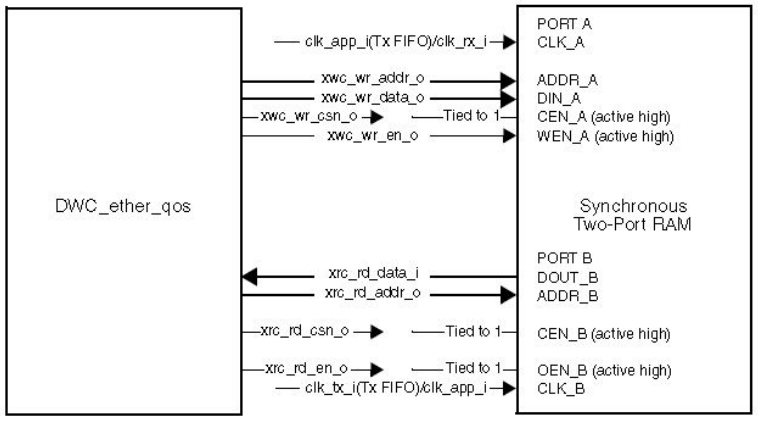
The SRAM chip enable (CEN) and write enable (WEN) are connected to chip select (*_csn_o) and write enable (*_wr_en_o) signals from the MAC. Based on the active states of CEN and WEN, the DWC_ether_qos memory interface signals are directly connected or have inverted polarity. Connecting the chip select signal ensures that the SRAM I/O drivers are turned off when read or write operation is not scheduled and thereby reduces power consumption.

If the memory module (such as Xilinx FPGA Block Select RAM) has a single enable for the read port (No Output Enable), xrc_rd_csn_o must be connected to the enable, with polarity taken into consideration.

2.4.3.2 High-Speed, Synchronous Two-Port SRAM Connection

For applications where power consumption is not a factor and memory access times are too critical, the chip enable and read enable signals can always be tied to active high inputs. In such applications, the generic memory interface signals can be connected to synchronous SRAM as shown in Figure 2-54. The unused outputs of the generic memory interface are also shown as floating outputs.

Figure 2-54 High-Speed, Synchronous Two-Port SRAM Connection



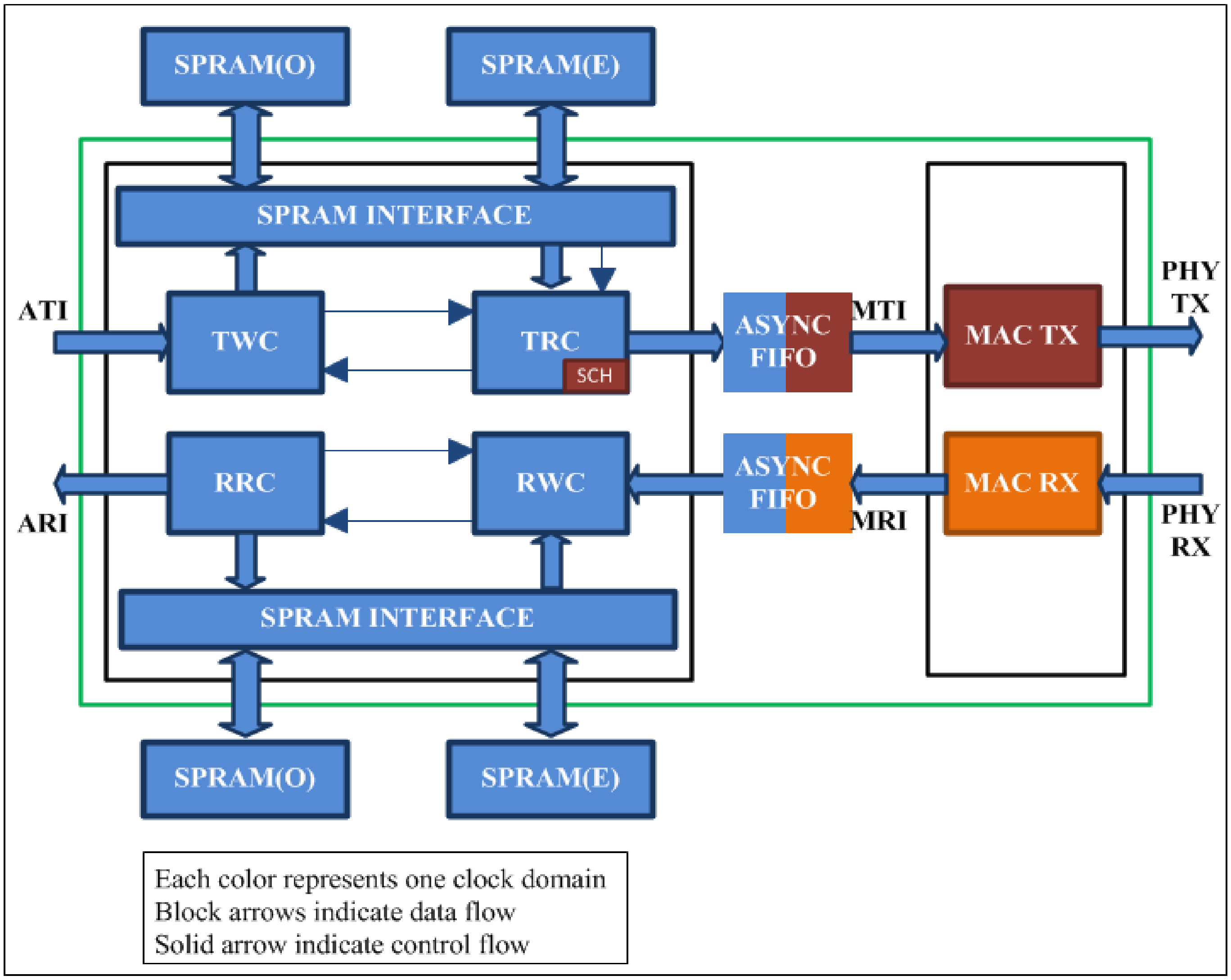
2.4.4 SPRAM Interface

The MAC supports Single Port RAM (SPRAM) option to implement the MTL FIFO buffers. In this configuration, each transmit and Receive FIFO memory is split into two SPRAM blocks.

Figure 2-55 shows the high level architectural approach for supporting single port RAM interface. It describes the data flow, clock domains and memory interfaces in EQOS-MTL configuration with GMII/MII PHY interface.

As compared to a single DPRAM instance (with application clock on 1 port and GMII/MII clock on the other port), each Transmit and Receive FIFO memory is split into two SPRAM blocks (Odd and Even) and operates with the application clock. Each SPRAM block will have the same width as earlier (DPRAM) but will have half the depth of the total FIFO memory selected in the configuration. Thus the total memory space will be retained as configured.

Figure 2-55 SPRAM Architecture Block Diagram



2.4.4.1 Data Flow

In Transmit direction, the data is read from the SPRAM memory and then transferred from application clock domain to GMII TX clock domain using a small asynchronous register FIFO at the MTI interface between the MTL module and MAC blocks. Similarly, the data received from the MAC Receiver in GMII/MII RX clock domain is transferred to application clock domain using a register FIFO at the MRI interface.

In configurations with multiple TX Queues, there is a scheduler on the Read-side of the TXFIFO which selects the TxQueue from which the next packet is to be transmitted. This scheduler logic continues to operate in GMII/MII clock domain so that the AVB related CBS algorithm is not affected by this change.However, there can be an impact on the minimum IPG (Inter-Packet Gap) that can be supported with SPRAM configuration due to the presence of ASYNC FIFO in TX path, scheduler in TX GMII/MII clock domain while the packet data is read from SPRAM in application clock domain. Additional CDC synchronizers will get added as compared to the DPRAM architecture which adds latency delays between packet transmissions.

The area increase with the SPRAM configuration option is mainly due to this asynchronous register FIFOs.

The data to the FIFO are stored alternatively in the Odd and Even SPRAM blocks based on the odd and even addresses of the FIFO pointer respectively. As the memory is used as a FIFO structure, data is normally written and read in a sequential manner (incrementing address). Hence with this organization, writes and reads will alternate between the two Odd and Even SPRAMs. This architecture enables simultaneous reads and writes to the FIFO (in 1 clock cycle) so that the data transfer throughput as achieved by a DPRAM memory architecture is maintained. This also allows continuous burst transfers to the host bus like AHB, AXI in configurations with DMA and thus utilize the bus bandwidth effectively.

However, as the read and write operations are triggered by independent engines, there can be a contention between them for accessing a SPRAM block in the same clock. There is an arbiter block between the Read and Write Engines that handles the contention as given as follows:

For TX FIFO, write operation is given priority over the read operation.

The read operation is delayed by a clock in such cases assuming that the write operation switches to the other memory in the next clock (due to sequential write).

The MAC transmitter do not need continuous data (For example, in 32-bit configurations and operating in 1Gbps mode, the MAC transmitter needs 32-bit word every 4 Tx clock cycles). The depth of small asynchronous FIFO towards the MAC is selected to avoid under run, by pre-fetching data whenever there is opportunity to read from the memory.

For RX FIFO, read operation is given priority over write operation.

The write operation is delayed by a clock in such cases assuming that the read operation switches to the other memory in the next clock (due to sequential reads).

The MAC receiver does not provide continuous data (For example, in 32-bit configurations and operating in 100Mbps mode, the MAC receiver provides 32-bit word every 8 Rx clock cycles) except at the beginning and towards end of the Rx packet. The depth of small asynchronous FIFO towards the MTL is selected to avoid the overrun.

2.4.4.2 Application Clock Frequency Requirements for Memory Transfers

Due to the requirement of simultaneous read and write transfers from the FIFO memories, the application clock frequency should be such that the data transfer bandwidth of memory must be at least twice the bandwidth as that of the line.

In addition to the previous requirement, additional bandwidth is required to allow delays due to contention between read and writes to a single memory block (Odd or Even). The probability of contention increases with multiple Queues (in Tx Side) especially in EQOS-AXI and EQOS-MTL configurations. These are mainly due to

- Possibility of receiving and writing interleaved data of different queues, all of which are to the same (odd or even) memory clocks in consecutive clock cycles.
- Non-sequential reads & writes of control words for each packet by scheduler in TX side
- Non-sequential writes of status words for each packet in RX side.

Considering the above, the minimum application clock frequency estimate in the various modes/speeds/configurations to avoid underflow in the "Async FIFO" in TX path is given in Table 2-12.

Table 2-12 Application Clock Frequency Estimate in Various Modes/Speeds

Configuration	No of Tx Queues	DW = 32	DW = 64	DW = 128
EQOS-AXI, EQOS-MTL	2 or less	75 MHz	38 MHz	19 MHz
	4 or less	125 MHz	62.5 MHz	31.25 MHz
	6 or less	200 MHz	100 MHz	50 MHz
	8 or less	250 MHz	125 MHz	62.5 MHz
EQOS-AHB, EQOS-DMA	Any	62.5 MHz	31.25 MHz	16 MHz

When the speed is 100Mbps (MII mode), the minimum application clock frequency is reduced by factor of 10.

In the RX side, contention can happen if the slave memory does not accept data continuously and in the worst case there can be a contention for every memory access if the slave memory introduces one busy cycle after every transfer. However, status word writes steal cycles during the IPG (Inter-packet Gap) of received packets. The number of status words to be written depends on the configuration (for example, IEEE 1588 requires 64-bit timestamps, VLAN stripping requires 32-bit VLAN tags, etc). The depth (or area) of the register Asynchronous FIFO has to be larger to avoid overflows.

However, the minimum application clock frequency will be the same in all configurations in 1 Gbps speed as shown in Table 2-13.

Table 2-13 Minimum Application Clock Frequency in Various Configurations

Configuration	No of Rx Queues	DW = 32	DW = 64	DW = 128
Any	Any	69 MHz	36 MHz	21 MHz

For 100Mbps operating mode, the application clock frequency can be reduced by a factor of 10.

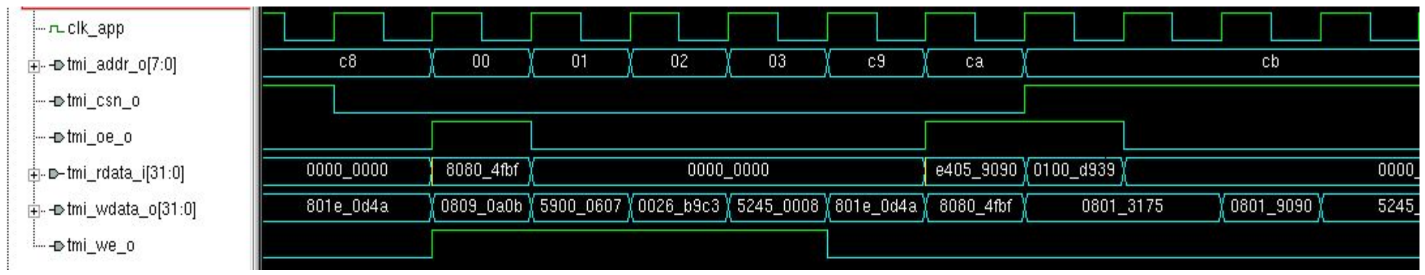
Note

- The previous tables are pessimistic and estimated with a worst case scenario assumptions. These will be validated in simulation and updated later.
- To handle valid frames of size < 64 bytes received in the line, increase the application clock frequency further, as the overhead due to status is considerably large on Rx side.
- The previous computation assumes 12 bytes of IPG and 8 bytes of preamble+SFD. To handle reduced IPG and preamble, increase the application clock frequency further, to transfer the status in the available gaps between packets on the application interface (MRI)

2.4.4.3 SPRAM Timing

Timing of the SPRAM memory is same as the TSO memory, as shown in Figure 2-56.

Figure 2-56 SPRAM Interface Timing Diagram



2.5 MAC

The MAC supports many interfaces towards the PHY chip. The PHY interface can be selected only once after reset. The MAC communicates with the application side with the MAC Transmit Interface (MTI), MAC Receive Interface (MRI), and MAC Control Interface (MCI).

2.5.1 MAC Transmission

The MAC supports many interfaces towards the PHY chip. The PHY interface can be selected only once after reset. The MAC communicates with the application side with the MAC Transmit Interface (MTI), MAC Receive Interface (MRI), and MAC Control Interface (MCI).

The MAC transmission process is as follows:

1. Transmission is initiated when the MTL application pushes in data with the SOP (mti_sop_i) signal asserted.
2. When the SOP signal is detected, the MAC accepts the data and begins transmitting to the GMII or MII.

   The time required to transmit the packet data to the GMII or MII after the application initiates the transmission depends on delay factors such as IPG delay, time to transmit preamble or SFD, and any back-off delays for half-duplex mode. While the packets data is being transmitted, the MAC can stop accepting the data received from the MTL by de-asserting the mti_rdy_o signal.
3. After the EOP (mti_eop_i) is transferred to the MAC, the MAC does one of the following:
   - The MAC completes the normal transmission and gives the transmission status to the MTL.
   - If a normal collision (in half-duplex mode) occurs during transmission, the MAC gives the Transmit Status with retry bit set to the MTL. The MAC gives the Retry request till one of the following is true:

     Packet is successfully transmitted

     Maximum retry requests expire

     When maximum retry requests expire, the MAC aborts the packet transmission with Excessive Collision Transmit Status. The MAC accepts and drops all further data until the next SOP is received. The MTL block should retransmit the same packet from SOP on observing a Retry request (in the Status) from the MAC.
   - If any one of the following happens, the MAC aborts the packet transmission:

     No carrier (half-duplex mode)

     Loss of carrier (half-duplex mode)

     Excessive deferral (half-duplex mode)

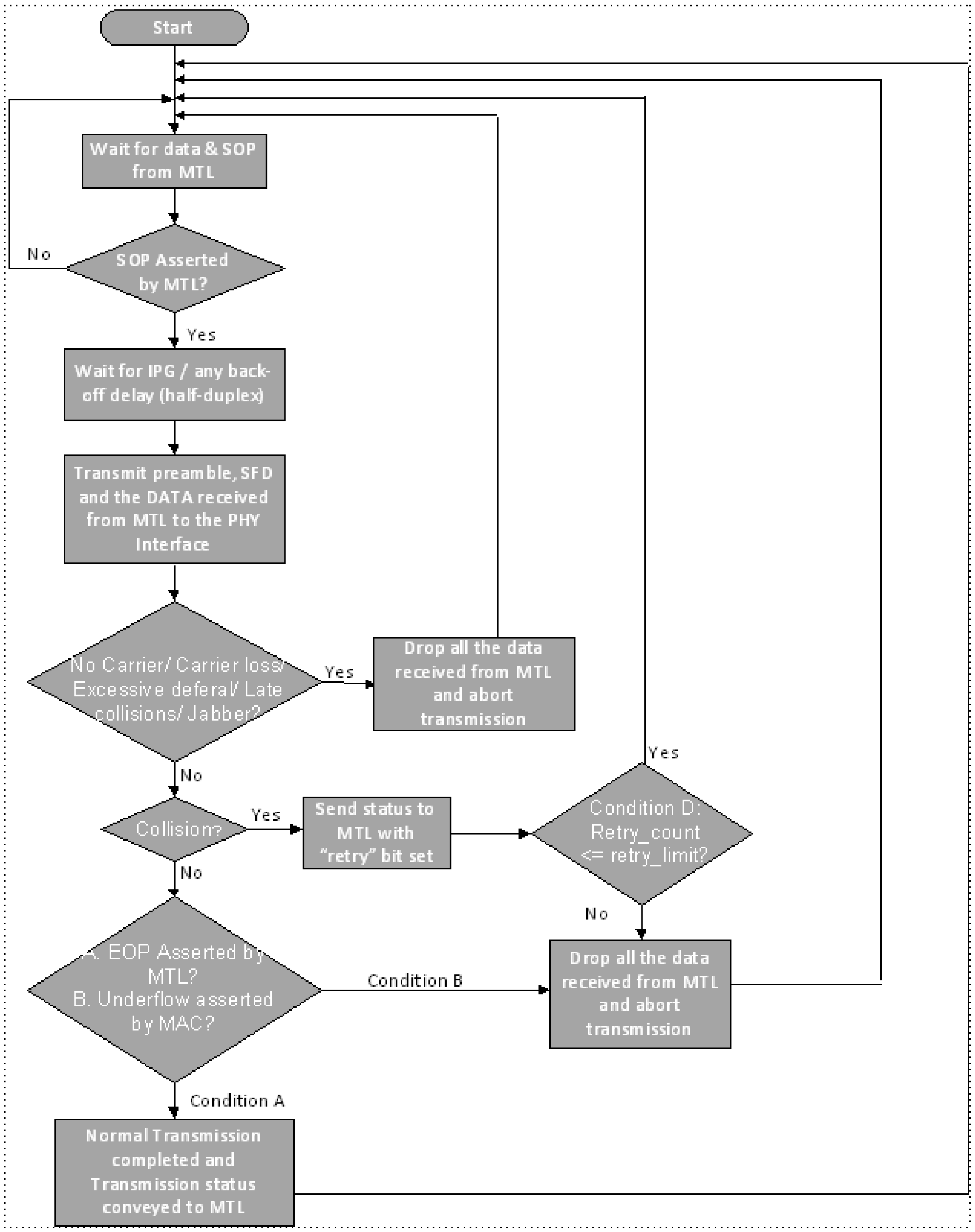
     Late collisions (half-duplex mode)

     Jabber

     The MAC accepts and drops all further data until the next SOP is received.
4. The MAC issues an underflow status if the MTL is not able to provide the data continuously during the transmission. The MAC accepts and drops all further data until the next SOP is received.
5. During the normal transfer of a packet from MTL, if the MAC receives a SOP without getting an EOP for the previous packet, it ignores the SOP and considers the new packet as continuation of the previous packet.

Figure 2-57 illustrates the MAC transmission process flow.

Figure 2-57 Overview of MAC Transmission Process Flow


Start
Wait for data & SOP from MTL
SOP Asserted by MTL?
No
Yes
Wait for IPG / any back-off delay (half-duplex)
Transmit preamble, SFD and the DATA received from MTL to the PHY Interface
No Carrier/ Carrier loss/ Excessive deferal/ Late collisions/ Jabber?
Yes
Drop all the data received from MTL and abort transmission
No
Collision?
Yes
Send status to MTL with "retry" bit set
Condition D: Retry_count <= retry_limit?
Yes
No
No
A. EOP Asserted by MTL? B. Underflow asserted by MAC?
Condition B
Drop all the data received from MTL and abort transmission
Condition A
Normal Transmission completed and Transmission status conveyed to MTL

The following six modules constitute the transmission function of the MAC:

- "Transmit Bus Interface Module" on page 120
- "Transmit Packet Controller Module" on page 120
- "Transmit Protocol Engine Module" on page 121
- "Transmit Module" on page 122
- "Transmit CRC Generator Module" on page 123
- "MAC Transmit Flow Control Module" on page 123

2.5.1.1 Transmit Bus Interface Module

The MAC Transmit Bus Interface (TBU) accepts data in 32, 64, or 128-bit wide bus and runs on the clk_tx_i clock of GMII interface.

The Transmit Bus Interface (TBU) module connects the Transmit path of the MAC with an external packet through a FIFO interface.

The TBU accepts data in 32, 64, or 128-bit wide bus and runs on the clk_tx_i clock of GMII interface. The TBU module performs the following functions:

- Outputs the (32-bit) Transmit status to the application at the end of normal transmission or collision
- Outputs the Transmit snapshot register value to the mti_timestamp_o signal and asserts the mti_status_valid signal
- Performs the Endian conversion of data bus by swapping the byte lanes and corresponding byte enables
- Converts the input data into an 8-bit bus towards the Transmit Packet Controller

2.5.1.2 Transmit Packet Controller Module

The Transmit Packet Controller (TPC) module consists of eight registers to hold the data and the last data control received from the TBU.

The register provides a buffer between the application and the Transmit Protocol Engine (TPE) to regulate data flow.

When the number of bytes received from the application are less than 60 (DA+SA+LT+DATA), the state machine that interfaces with TBU automatically appends zeros to the packet being transmitted. This is done to make the data length exactly 46 bytes (provided mti_crc_pad_ctrl_i is 2'b00) to meet the minimum data field requirement of IEEE 802.3. To program the MAC to not append any padding, you can use the mti_crc_pad_ctrl_i sideband signal from the MTI.

The cyclic redundancy check (CRC) for the Frame Check Sequence (FCS) field is calculated before transmission to the TPE module. This value is computed by the CTX module. The TPC module receives the computed CRC and appends it to the data being transmitted to the TPE module. When the MAC is programmed (through mti_crc_pad_ctrl_i sideband signal from the application interface) to not append the CRC value to the end of Ethernet packets, the TPC module ignores the computed CRC and transmits only the data received from the TBU module to the TPE module. An exception to this rule is that when the MAC is programmed to append pads for packets (DA+SA+LT+DATA) less than 60 bytes sent by the TBU module, the TPC module appends the CRC at the end of padded packet irrespective of the value on the mti_crc_pad_ctrl_i signal.

2.5.1.3 Transmit Protocol Engine Module

The Transmit Protocol Engine (TPE) module consists of a Transmit State Machine that controls the operation of Ethernet packet transmission.

The Transmit State Machine of this module contains the following features to meet the IEEE 802.3/802.3z specification:

- Generates preamble and SFD
- Generates jam pattern in the half-duplex mode after normal collision
- Generates carrier extension in the half-duplex (only in the GMII) mode when packet is smaller than 512 bytes
- Supports packet bursting in the half-duplex (only in the GMII) mode
- Supports jabber timeout
- Supports flow control for the half-duplex mode (backpressure)
- Generates Transmit packet status
- Contains timestamp snapshot logic for IEEE 1588 support

When the TPC module requests the TPE module for a new packet transmission, the Transmit State Machine sends out the preamble and SFD, followed by the data received. The preamble is defined as 7 bytes of 8’b10101010 pattern and the SFD is defined as 1 byte of 8b’10101011 pattern.

The collision window is defined as 1 slot time (512-bit times for 10/100 Mbps Ethernet and 4,096 bit times for 1,000 Mbps Ethernet). The jam pattern generation is applicable only to half-duplex mode, not to full duplex mode. In full-duplex mode, the Transmit State Machine ignores the phy_col_i signal from the PHY.

In MII mode, if a collision occurs any time from the beginning of the packet to the end of the CRC field, the Transmit State Machine sends a 32-bit jam pattern of 32’h55555555 on MII to inform all other stations that a collision has occurred. If the collision is seen during the preamble transmission phase, the Transmit state machine completes the transmission of preamble and SFD, and then sends the jam pattern.

In GMII mode, if a collision occurs any time between the beginning of the packet and the end of the extension field, the Transmit State Machine sends a 32-bit jam pattern of 32’h55555555 on GMII to inform all other stations of the collision. If the collision is seen during the preamble transmission phase, the Transmit State Machine completes the transmission of preamble and SFD, and then sends the jam pattern. If a collision occurs during the extension field, the Transmit State Machine sends a 32-bit jam pattern of 32'h1F1F1F1F.

If the collision occurs after the collision window and before the end of the FCS field (or the end of Burst if the Packet Burst mode is enabled), the Transmit State Machine sends a 32-bit jam pattern and sets the late collision bit in the Transmit packet status.

Note At the GMII or MII interface, the collision signal (phy_col_i) being asynchronous is checked by the transmitter after it is double-synchronized to clk_tx domain. This additional latency delays the recognition of collision or late-collision event. When the output of phy_col_i synchronizer is asserted after the complete packet is transmitted, it is not recognized as collision even if the COL signal is high at the GMII interface before the end of transmission. Similarly, an assertion of COL signal at the GMII input at the last byte of normal collision window might be identified as late collision because of the synchronizer delay.

In GMII half-duplex mode (1000 Mbps), the Transmit State Machine ensures that all valid carrier events exceed a slot time of 4,096 bit times. To accomplish this, any Transmit packet shorter than 512 bytes from the TFC module is extended using a carrier extension. On GMII, this is signaled to the PHY by asserting phy_txer_o, de-asserting phy_txen_o, and setting phy_txd_o[7:0] to 8h'0F.

When the Packet Burst mode is enabled, only the first packet of the burst is carrier extended if it is shorter than 512 bytes. The carrier extension is not applicable for MII half-duplex and GMII or MII full-duplex modes. When the Packet Burst mode is enabled, the Transmit State Machine transmits a burst of packets (as long as packets are available from the TFC module) without releasing the carrier of the PHY. To accomplish this, the state machine inserts the carrier extension for a minimum IPG period (96 bit times) between the packets. The Transmit State Machine continues to burst packets as long as additional packets are available from the TPC module and a burst limit of 8,192 byte times has not been exceeded. If an additional packet is not available at the end of the IPG period in the middle of the burst, the Transmit State Machine releases the carrier by de-asserting the phy_txer_o and phy_txen_o signals on GMII.

Packet bursting is applicable only for the GMII half-duplex mode. It is not applicable in the MII and GMII full-duplex modes. In the GMII half-duplex mode, the size of the first packet in a packet burst should at least be equal to the slot time. If the first packet (including carrier extension) is less than the slot time, the MAC considers all data bytes, received for the first packet and subsequent packets, which end immediately after the slot time is reached as first packet in the burst. This may result in CRC error. However, if the packet burst ends (both RXDV and RXER go low) before the slot time, the subsequent packet is considered as a new packet. This behavior is according the IEEE 802.3.

The TPE module maintains a jabber timer to stop the transmission of Ethernet packets if the TFC module transfers more than 2,048 (default) bytes. The timeout is changed to 10,240 bytes when the Jumbo packet is enabled.

The Transmit State Machine uses the deferral mechanism for flow control (backpressure) in the half-duplex mode. When the application requests to stop receiving packets, the Transmit State Machine sends a JAM pattern of (8'h55) 32 bytes whenever it senses a reception of a packet, provided the Transmit flow control is enabled. This results in a collision and the remote station backs off. The application requests the flow control through a sideband signal mti_flowctrl_i (or by setting BPA bit of the Flow Control Register of corresponding MTL queue). If the application requests a packet to be transmitted, it is scheduled and transmitted even when the backpressure is activated. If the backpressure is kept activated for a long time (and more than 16 consecutive collision events occur), the remote stations abort their transmissions because of excessive collisions.

If IEEE 1588 timestamp is enabled for the Transmit packet, this block takes a snapshot of the system time when the SFD is put onto the Transmit GMII or MII bus. The system time source is either an external input or it is internally generated according to the configuration selected.

Note The Collision input is an asynchronous signal and should be asserted for at least 32-bit times (4 clocks for 1Gbps, and 8 clocks for slower speeds).

2.5.1.4 Transmit Module

The Transmit (STX) module is responsible for scheduling the packet transmission on GMII or MII. It provides an enable signal to the TPE module after satisfying the IPG and back-off delays. The STX module performs the following functions:

- Maintains the inter-packet gap between two transmitted packets

  The STX module maintains an idle period of the configured inter-packet gap (IPG bits of MAC_Configuration register) between any two transmitted packets. If packets from the TPC arrive at the TPE module sooner than the configured IPG time, the TPE module waits for the enable signal from the STX module before starting the transmission on GMII or MII. The STX module starts its IPG counter as soon as the carrier signal of GMII or MII goes inactive. At the end of programmed IPG value, the module issues an enable signal to the TPE module in the full-duplex mode.

  In the half-duplex mode and when IPG is configured for 96-bit times, the STX module follows the rule of deference specified in the IEEE 802.3, Section 4.2.3.2.1. The module resets its IPG counter if a carrier is detected during the first two-thirds (64-bit times for all IPG values) of the IPG interval. If the carrier is detected during the final one-third of the IPG interval, the STX module continues the IPG count and enables the transmitter after the IPG interval.

- Implements the Truncated Binary Exponential Back-off algorithm

  The STX module implements the Truncated Binary Exponential Back-off algorithm when it operates in the half-duplex mode.

2.5.1.5 Transmit CRC Generator Module

The MAC Transmit CRC Generator (CTX) module interfaces with the TFC module to generate CRC for the FCS field of the Ethernet packet.

The Transmit CRC Generator (CTX) module interfaces with the TFC module to generate CRC for the FCS field of the Ethernet packet.

The TPC module sends the packet data and any necessary padding to the CTX module through an 8-bit interface.

The CTX module calculates the 32-bit CRC for the FCS field of the Ethernet packet. The encoding is defined by the following generating polynomial:

G (x) = x32 + x26 + x23 + x22 + x16 + x12 + x11 + x10 + x8 + x7 + x5 + x4 + x2 + x + 1

The CTX module gets the byte data of Ethernet packet from the TPC module (DA + SA + LT + DATA + PAD) qualified with a Data Valid signal. The TPC also indicates to the CTX when to reset the previously calculated CRC and to start the new CRC calculation for the coming packet. The TPC module issues the start command before sending the new packet data for calculation. The calculated CRC is valid on the next clock after the data is received.

In GMII mode, the Data Valid signal is valid for every clock, from the first data byte through the last data byte. In MII mode, this signal is valid every alternate clock.

2.5.1.6 MAC Transmit Flow Control Module

The Transmit Flow Control (FTX) module generates and transmits the Pause packets to the TFC module based on the flow control triggers in full-duplex mode. The TFC module receives the Pause packet from the FTX module, appends the calculated CRC, and sends the packet to the TPE module.For more information about flow control, see “Flow Control” on page 389.

2.5.2 MAC Transmit Interface Protocol

The MAC Transmit Interface (MTI) connects the application (MTL module in DWC_ether_qos) with the MAC to provide the Ethernet data for transmission.

The MAC Transmit Interface (MTI) connects the application (MTL module in DWC_ether_qos) with the MAC to provide the Ethernet data for transmission.

The application initiates the Ethernet packet transmission by writing the first data (mti_sop_i) of the packet to the MAC, provided the MAC is ready to accept data (indicated by mti_rdy_o). Each valid data transfer is indicated by an active mti_valid_i signal. The application can push-in data as long as the MAC is ready to accept it. Therefore, the application can assume a successful data transfer if the mti_rdy_o signal is asserted.

The application indicates the last data of the packet by asserting the end-of-packet mti_eop_i signal, along with the last data and byte enables (mti_be_i). The number of valid byte lanes for last transfer is decoded with the mti_be_i signal as shown in Table 2-14. At the end of normal transmission of the Ethernet packet, the MAC outputs the Transmit status to the application. This is indicated by an active mti_txstatus_val_o signal.

Table 2-14 mti_be_i Function

mti_be_i	Valid byte lanes (Little-Endian)	Valid byte lanes (Big Endian)
128-Bit Databus		
0000	mti_data_i[7:0] – Byte lane 0	mti_data_i[127:120] – Byte lane 15
0001	mti_data_i[15:0] – Byte lane 0-1	mti_data_i[127:112] – Byte lane 15-14
0010	mti_data_i[23:0] – Byte lane 0-2	mti_data_i[127:104] – Byte lane 15-13
...		
1110	mti_data_i[119:0] – Byte lane 0-14	mti_data_i[127:8] – Byte lane 15-1
1111	mti_data_i[127:0] – Byte lane 0-15	mti_data_i[127:0] – Byte lane 15-0
64-Bit Databus		
000	mti_data_i[7:0] – Byte lane 0	mti_data_i[63:56] – Byte lane 7
001	mti_data_i[15:0] – Byte lane 0-1	mti_data_i[63:48] – Byte lane 7-6
010	mti_data_i[23:0] – Byte lane 0-2	mti_data_i[63:40] – Byte lane 7-5
...		
110	mti_data_i[55:0] – Byte lane 0-6	mti_data_i[63:8] – Byte lane 7-1
111	mti_data_i[63:0] – Byte lane 0-7	mti_data_i[63:0] – Byte lane 7-0
32-Bit Databus		
00	mti_data_i[7:0] – Byte lane 0	mti_data_i[31:24] – Byte lane 3
01	mti_data_i[15:0] – Byte lane 0-1	mti_data_i[31:16] – Byte lane 3-2

mti_be_i	Valid byte lanes (Little-Endian)	Valid byte lanes (Big Endian)
10	mti_data_i[23:0] – Byte lane 0-2	mti_data_i[31:8] – Byte lane 3-1
11	mti_data_i[31:0] – Byte lane 0-3	mti_data_i[31:0] – Byte lane 3-0

If the packet transmission is not successful (because of underflow, collision, jabber timeout, or excessive deferral events), the MAC asserts the Transmit status even before the EOP is received. The application takes the appropriate action as per the status. The MAC drops all further data input to it until the next SOP.

Table 2-15 Transmit Status at the MAC Interface

Bit	Description
31	Packet Retry Requested When set, this bit indicates that the MAC transmitter requested retransmission of the entire packet from the application. This bit has a higher priority than the Bit 0 (Packet Aborted). The MAC expects a retransmission of the packet when this bit is set irrespective of the value of Packet Aborted bit.
30	Timestamp Status When set, this bit indicates that the MAC transmitter has captured the IEEE1588 timestamp of transmitted packet. The captured timestamp is available along with the Transmit status on the mti_timestamp_o[63:0]output. This bit is enabled only when the IEEE1588 Timestamp feature is enabled. Otherwise, it is reserved.
29	VLAN Packet When set, this bit indicates to the application that the transmitted packet is a VLAN tagged packet, that is, the Type field is equal to one of the following: 16'h88a8 when Bit 18 of the MAC_VLAN_Tag register is set to 16'fh8100.
28:27	Address Type This field indicates the type of destination address transmitted: 2'fb00: Unicast 2'fb01: Multicast 2'fb10: Reserved 2'fb11: Broadcast
26:13	Transmit Byte Count This 14-bit counter indicates the number of bytes transmitted.
12:9	Collision Count This 4-bit counter indicates the number of collisions that occurred before the packet was transmitted. The count is not valid when the Excessive Collisions bit is set.
8	Deferred When set, this bit indicates that the MAC defers before transmission because of the presence of carrier.
7	Underflow When set, this bit indicates that the MAC aborted the transmission because of data underflow. The Underflow error indicates that no more data is available for transmission in the middle of transmission.
6	Excessive Collisions When set, this bit indicates that the transmission is aborted after 16 successive collisions while attempting to transmit the current packet. If Bit 8 in the MAC_Configuration register is set, this bit is set after the first collision. The transmission of the packet is aborted.

Bit	Description
5	Late Collision When set, this bit indicates that the packet transmission was aborted because a collision occurred after the collision window (64 bytes including Preamble in MII mode and 512 bytes including Preamble and Carrier Extension in GMII mode). This bit is not valid if the Underflow error is set.
4	Excessive Deferral This bit is set when Bit 4 is set high in the MAC_Configuration register. When set, this bit indicates that the transmission has ended because of excessive deferral of over 24,288 bit times (155,680 in 1000 Mbps mode or if Jumbo packet is enabled).
3	Loss of Carrier When set, this bit indicates that the loss of carrier occurred while transmitting the packet, that is, the phy_crs_i signal was inactive for one or more transmission clock periods during packet transmission. This bit is valid only for packets transmitted without collision in the half-duplex mode.
2	No Carrier When set, this bit indicates that the carrier signal from the PHY was not present at the end of preamble transmission. This bit is valid only when the MAC operates in the half-duplex mode.
1	Jabber Timeout When set, this bit indicates the MAC transmitter has experienced a jabber timeout. This bit is set only when Bit 17 of the MAC_Configuration register is not disabled (de-asserted).
0	Packet Aborted When set, this bit indicates that the transmission of the current packet is aborted because of one or more of the following conditions: ■ Jabber Timeout ■ No Carrier ■ Loss of Carrier ■ Excessive Deferral ■ Late Collision ■ Retry Count exceeds the attempt limit ■ Data underrun When reset, this bit indicates that the current packet was successfully transmitted on the GMII or MII interface (provided Bit 31 is zero).

2.5.3 MAC Transmit Timing

The following sections describe the timing specifications for MAC (EQOS-CORE) interface signals in the Transmit path:

- "MAC Transmit Interface (MTI)" on page 127
- "MTI With Timestamp Feature" on page 128
- "Collision During Transmission" on page 129
- "Underflow During Transmission" on page 130

2.5.3.1 MAC Transmit Interface (MTI)

Figure 2-58 shows the timing specifications for the MTI signals. The packet transmission handshake mechanism is as follows:

1. The application can initiate a packet transfer by asserting mti_val_i along with valid data on mti_data_i. The mti_sop_i signal should be asserted when the start of packet is being transferred.
2. The MAC accepts the data as long as mti_rdy_o is active. The application should update the data (or de-assert mti_val_i) on observing an active mti_rdy_o signal to transfer data in bursts.
3. The MAC starts transmitting the packet on the GMII as soon as it gets hold of the channel.

   At any point during the transfer, the MAC can alert the application to stop the transfer by de-asserting mti_rdy_o. This occurs when the internal FIFO of the MAC is full.
4. When the application wants to transfer the last bytes of a packet, mti_eop_i should be asserted along with the byte enables (mti_be_i), indicating the valid number of data bytes on the bus. All data transfers except the EOP transfer must have the byte enables as all ones.
5. After transferring the EOP, the application should wait until the MAC gives the transmission status of that packet. The MAC also de-asserts mti_rdy_o after receiving the EOP so that the application cannot transfer the next packet.
6. The MTI gives the transmit status of the packet back to the application through mti_txstatus_o by asserting mti_txstatus_val_o (see Figure 2-59). The application should accept it immediately
7. The application can start the transmission of the next packet.

Figure 2-58 MAC Transmit Interface (MTI) Timing (1 or 2)

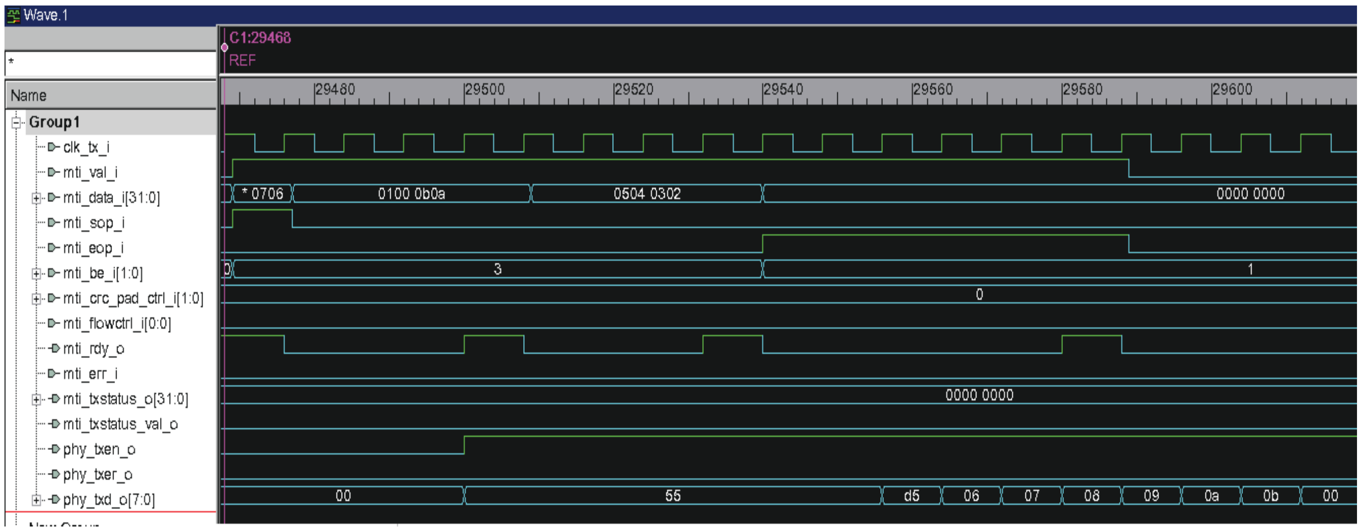
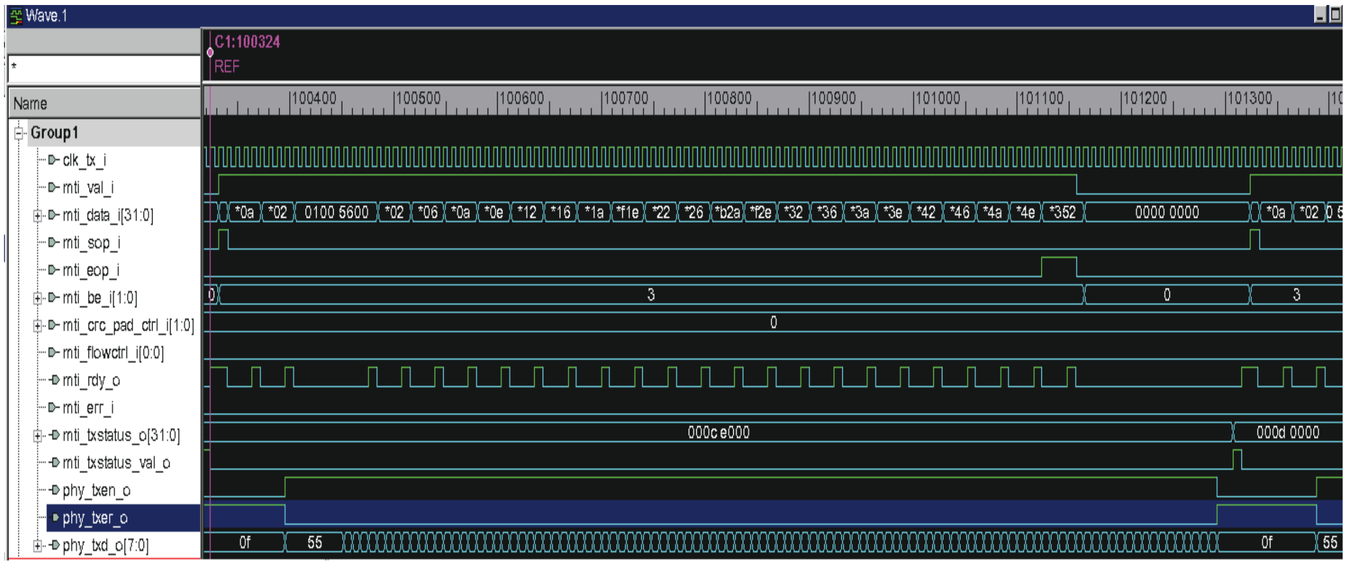


Figure 2-59 shows the timing of the GMII signals during transmission of a packet. It shows how the MAC meets the IPG timing on GMII during packet-burst transmission.

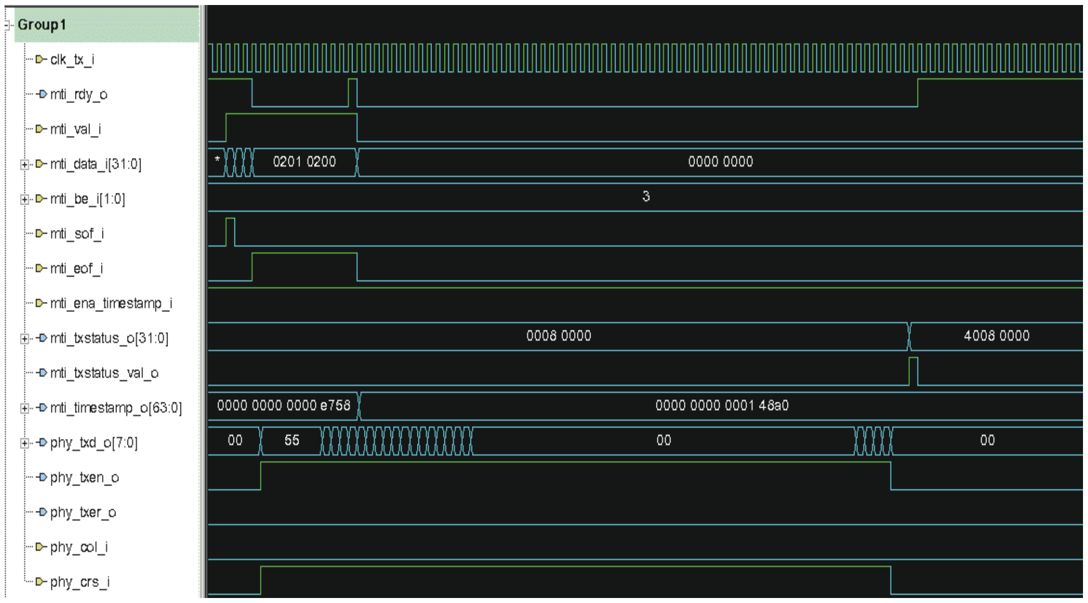
Figure 2-59 MAC Transmit Interface (MTI) Timing (2 of 2)



2.5.3.2 MTI With Timestamp Feature

Figure 2-60 shows the MTI timing when the IEEE 1588 timestamp feature is enabled. The mti_ena_timestamp_i signal is sampled along with the mti_sop_i signal for an input packet. When the packet status (0x40080000) is given on mti_txstatus_o, the timestamp captured for the packet is also output on mti_timestamp_o. The validity of the timestamp is indicated by mti_txstatus_o[30] being asserted when mti_txstatus_val_o is high.

Figure 2-60 MTI Timing With IEEE 1588 Timestamp Feature Enabled



2.5.3.3 Collision During Transmission

Figure 2-61 and Figure 2-62 show how the MAC behaves when collisions occur on the GMII interface. The process is as follows:

1. The MAC accepts the data from the application on MTI and starts transmission in the half-duplex mode.
2. When a collision is recognized, the MAC de-asserts mti_rdy_o to stop taking more data from the application on MTI. In Figure 2-61 this signal is already low because the internal FIFOs are full.
3. After JAM pattern (4 bytes of 0x55) transmission, the MAC gives the transmission status (mti_txstatus_o) to the application by asserting mti_txstatus_val_o. It requests the application to retry the packet transfer by setting Bit 31 of mti_txstatus_o.
4. In the next clock cycle, the MAC asserts mti_rdy_o, indicating that it is ready to accept the next packet for transmission. The MAC (MTI) ignores or drops all data that is given after the assertion of mti_txstatus_val_o until the application indicates a start of packet transfer by asserting mti_sop_i.

Figure 2-61 Collision During Transmission (1 of 2)

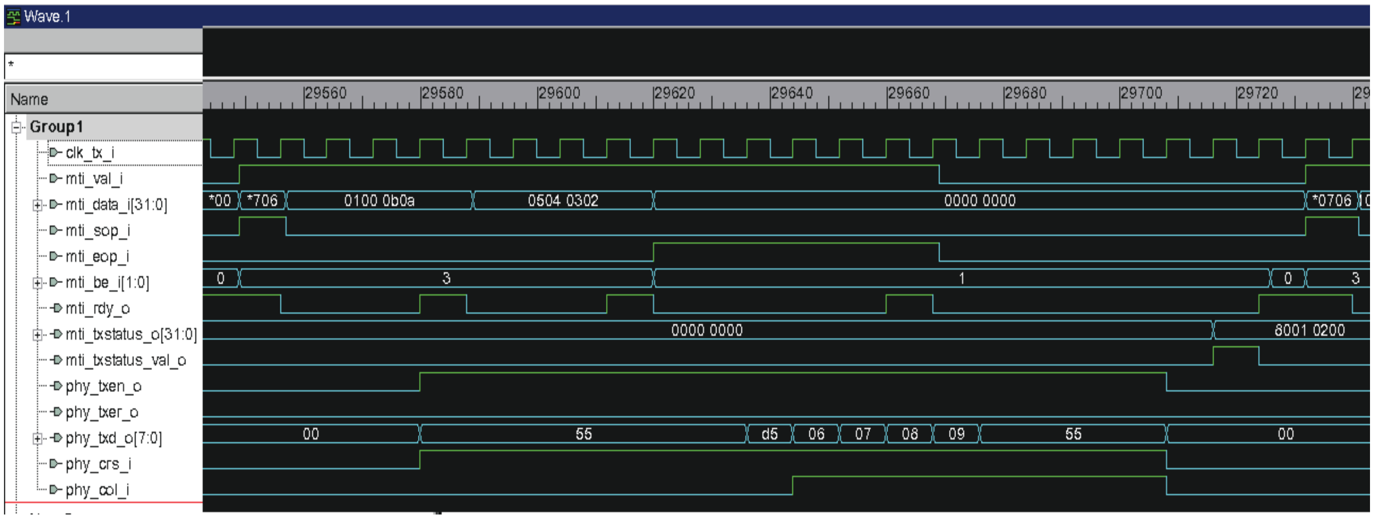
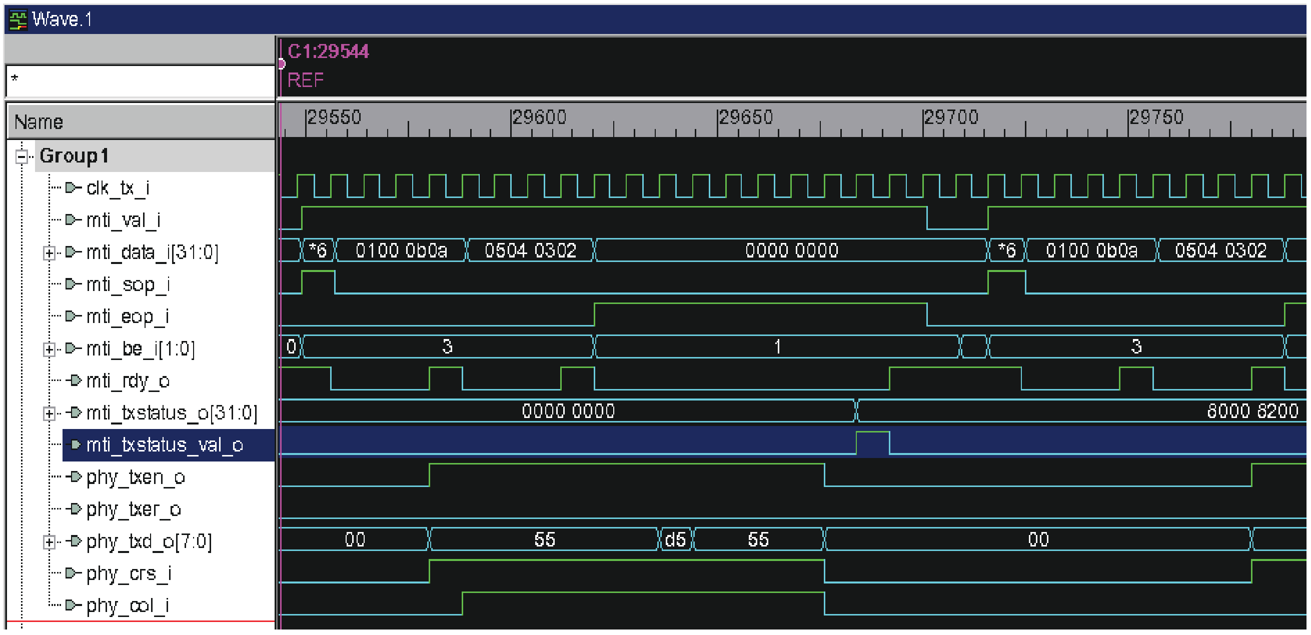


Figure 2-62 shows how the MAC transmitter starts a JAM pattern only after the transmission of the SFD even though the collision was recognized well in advance during the preamble transmission.

Figure 2-62 Collision During Transmission (2 of 2)

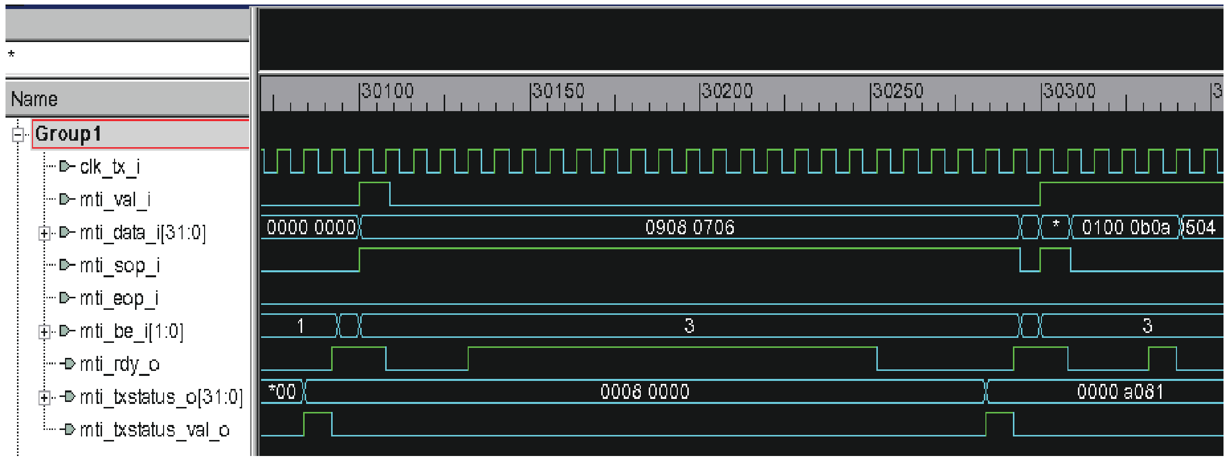


2.5.3.4 Underflow During Transmission

Figure 2-63 shows how a packet is aborted because of insufficient data being input to the MAC. The process is as follows:

1. The MAC schedules and starts transmission of a packet as soon it receives the start of packet on MTI.
2. The MAC finds that it does not have the EOP data because the application did not input data on MTI after the start of packet, and it de-asserts mti_rdy_o.
3. The MAC aborts transmission after 5 bytes (the fifth byte is a dummy byte).
4. The MAC gives the Transmit Abort status (0x0000a081) to the application.
5. The MAC indicates that it is ready to accept new data in the next clock cycle by asserting mti_rdy_o.

Figure 2-63 Underflow During Transmission Timing



2.5.4 MAC Reception

A receive operation is initiated when the MAC detects an SFD on GMII or MII. The MAC strips the preamble and SFD before proceeding to process the packet. The header fields are checked for filtering and the FCS field used to verify the CRC for the packet. The received packet is stored in a shallow buffer until the address filtering is performed. The packet is dropped in the MAC if it fails the address filter.

The following are the functional blocks in the Receive path of the MAC.

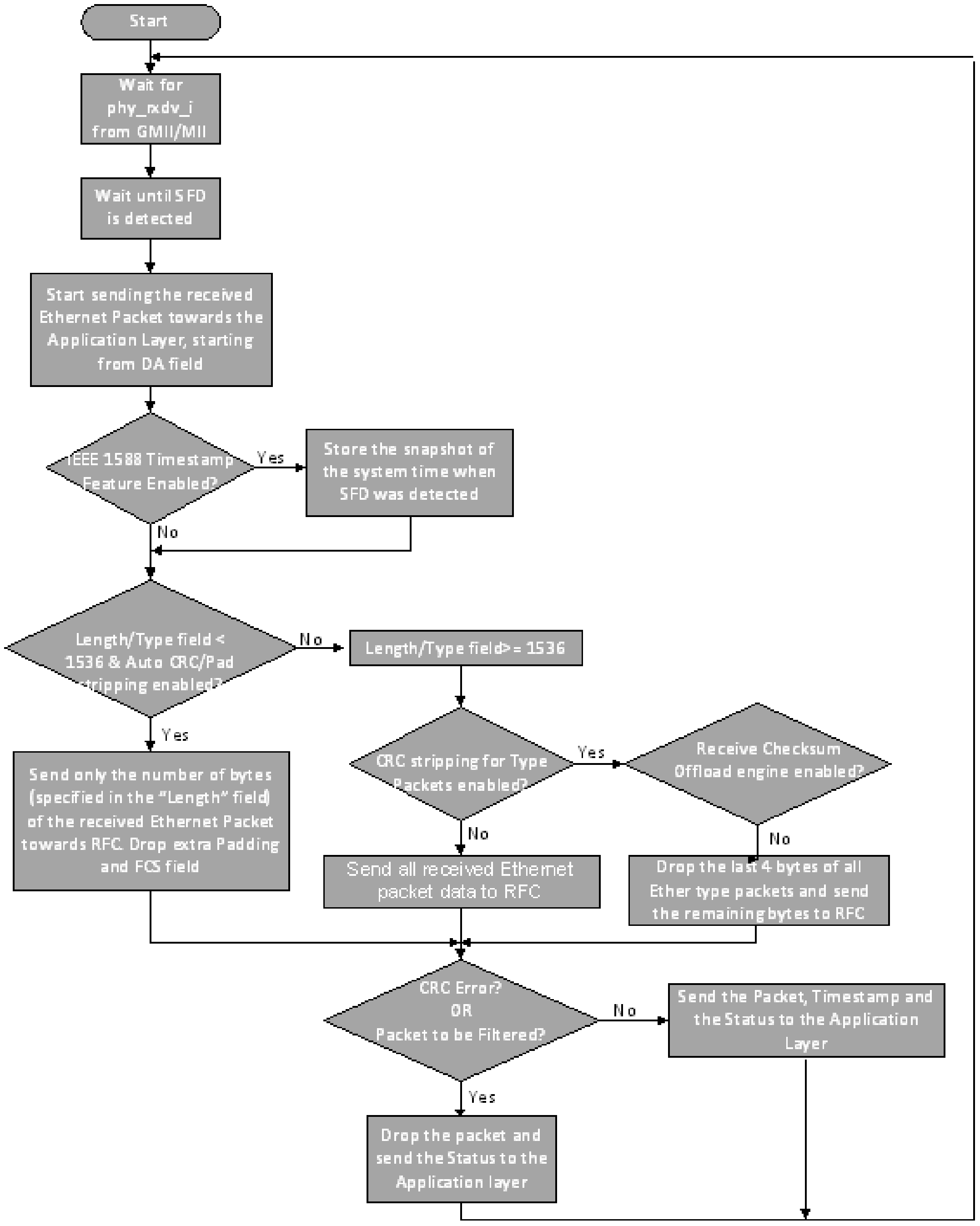
- "Receive Protocol Engine Module" on page 131
- "Receive CRC Module" on page 133
- "Receive Packet Controller Module" on page 134
- "Receive Flow Control Module" on page 135
- "Receive Bus Interface Unit Module" on page 135
- "Address Filtering Module" on page 135

2.5.4.1 Receive Protocol Engine Module

The Receive Protocol Engine (RPE) consists of the Receive State Machine which strips the preamble, SFD, and carrier extension of the received Ethernet packet (in half-duplex 1000-Mbps mode).

Figure 2-64 illustrates the Receive transmission flow in the RPE.

Figure 2-64 MAC Receive Flow Transmission


Start
Wait for phy_rxdv_i from GMII/MII
Wait until SFD is detected
Start sending the received Ethernet Packet towards the Application Layer, starting from DA field
IEEE 1588 Timestamp Feature Enabled?
Yes
Store the snapshot of the system time when SFD was detected
No
Length/Type field < 1536 & Auto CRC/Pad stripping enabled?
No
Length/Type field>= 1536
Yes
Send only the number of bytes (specified in the "Length" field) of the received Ethernet Packet towards RFC. Drop extra Padding and FCS field
CRC stripping for Type Packets enabled?
Yes
Receive Checksum Offload engine enabled?
No
Send all received Ethernet packet data to RFC
No
Drop the last 4 bytes of all Ether type packets and send the remaining bytes to RFC
CRC Error? OR Packet to be Filtered?
No
Send the Packet, Timestamp and the Status to the Application Layer
Yes
Drop the packet and send the Status to the Application layer

The sequence is as follows:

1. When the phy_rxdv_i signal of GMII or MII becomes active, the Receive State Machine of RPE starts looking for the SFD field (byte 0x5D in GMII mode; 0xD nibble in MII mode).

   The state machine drops received packets until it detects SFD.

2. When SFD is detected, the state machine begins sending the data of Ethernet packet to the RPC module, beginning with the first byte following the SFD (destination address).

3. If IEEE 1588 Timestamp feature is enabled, the RPE takes a snapshot of the system time at which SFD of any packet is detected on GMII or MII. If this packet is not dropped during MAC filtering, the timestamp is passed to the application. In MII mode, the RPE converts the received nibble data into bytes and forwards the valid packet data to the RFC module.
4. The Receive State Machine of the RPE module decodes the Length/Type field of the receiving Ethernet packet.

   If the Length/Type field is less than 1,536 and if the MAC is programmed for the Auto CRC/Pad Stripping (Bit 20 of the MAC_Configuration register), the state machine sends the data of the packet up to the count specified in the Length/Type field and starts dropping bytes (including the FCS field). The state machine of the RPE module decodes the Length/Type field and checks for the Length interpretation.

Note In Audio Video (AV) or Data Center Bridging (DCB) mode, when you select additional Rx queues, the packets that are less than or equal to 16 bytes in length after pad stripping, always get dropped inside the MAC receiver. This happens even if the packets have passed the address filter and have no CRC error.

   If the Length/Type field is greater than or equal to 1,536, the RPE module sends all received Ethernet packet data to the RFC module if you have not enabled the CRC stripping for Type packet in Bit 21 of the MAC_Configuration register. However, if you have enabled the CRC stripping for Type packets and not enabled the Receive Checksum Offload Engine, the MAC strips and drops the last 4 bytes of all packets of Ether type before forwarding the packets to the application.
5. By default, the MAC is programmed for watchdog timer to be enabled, that is, packets above 2,048 (10,240 if Jumbo Packet is enabled) bytes (DA + SA + LT + DATA + PAD + FCS) are cut off at the RPE module. In addition, you can use a programmable watchdog timer (Bit 16 of MAC_Watchdog_Timeout register) to override the fixed timeout of 2,048 or 10,240 bytes. You can disable the watchdog timer by programming Bit 19 of MAC_Configuration register. However, even if the watchdog timer is disabled, a packet greater than 32 KB size is cut off and a watchdog timeout status is given.

At the end of every received packet, the RPE module generates received packet status and sends it to the RPC module. Control, missed packet, and filter fail status are added to the receive status in the RPC module.

Note
- In half-duplex mode, the first packet in a burst should be at least slot time in length. If the first packet is smaller than the slot time, the MAC considers the bytes received up to the slot time as first packet, which results in CRC error for the packet.
- Packet bursting is applicable only for the GMII half-duplex mode. It is not applicable in the MII and GMII full-duplex modes. In the GMII half-duplex mode, the size of the first packet in a packet burst should at least be equal to the slot time. If the first packet (including carrier extension) is less than the slot time, the MAC considers all data bytes, received for the first packet and subsequent packets, which end immediately after the slot time is reached as first packet in the burst. This may result in CRC error. However, if the packet burst ends (both RXDV and RXER go low) before the slot time, the subsequent packet is considered as a new packet. This behavior is according the IEEE 802.3.

2.5.4.2 Receive CRC Module

The MAC Receive CRC (CRX) interfaces with the RPE module to check any CRC error in the packet being received.

The Receive CRC (CRX) interfaces with the RPE module to check any CRC error in the packet being received. This module calculates the 32-bit CRC for received packet that includes the Destination address field through the FCS field.

The encoding is defined by the following generating polynomial:

G (x) = x32 + x26 + x23 + x22 + x16 + x12 + x11 + x10 + x8 + x7 + x5 + x4 + x2 + x + 1

The module gets the data from the RPE module (DA+SA+LT+DATA+PAD+FCS). The RPE module also sends a control signal that indicates the validity of the data. Irrespective of the Auto Pad or CRC strip, the CRX module receives the entire packet to compute the CRC check for received packet.

2.5.4.3 Receive Packet Controller Module

The MAC Receive Packet Controller (RPC) receives the Ethernet packet data and status from the RPE module.

The Receive Packet Controller (RPC) receives the Ethernet packet data and status from the RPE module.

The RPC module consists of a FIFO of parameterized depth (default set to 4 deep and 33-bits wide) and two state machines for writing and reading the FIFO. The FIFO holds the received Ethernet packet data and byte enables, along with a control bit to indicate the last data. The state machines manage the FIFO and provide a packet buffering for the Ethernet packet being received from the RPE module. The following are main functions of the RPC module:

- Converting Data path — converts 8-bit data to 32-bit data to the RBU module
- Packet filtering
- Attaching the calculated IP Checksum input from IPC
- Updating the Receive Status and forwarding it to RBU

If RA bit of the MAC_Packet_Filter register is set, the RPC module initiates the data transfer to the RBU module when 4 bytes of Ethernet data are received from the RPE module. At the end of the data transfer, the RPC module sends out the received packet status that includes the packet filter bits (SA Filterfail and DA Filterfail) and status from the RPC module. These bits are generated based on the filter-fail signals from the AFM module. This status bit indicates to the application whether the received packet has passed the filter controls (both address filter and Packet Filter controls from CSR). The RPC module does not drop any packet on its own in this mode.

If the RA bit is reset, the RPC module performs packet filtering based on the destination or source address. If the application does not want to receive any bad packets such as runt, CRC error packets, the application still needs to perform another level of filtering. The RPC module waits to receive the first 14 bytes of received data (type field) from the RPE module. Until then, the module does not initiate any transfers to the RBU module. After receiving the destination or source address bytes, the RPC checks the filter-fail signal from the AFM module for an address match. On detecting filter-fail from AFB, the packet is dropped at the RPC module and not transferred to the application.

On a delayed filter response from the AFM (this can only occur if you change the AFM logic), the RPC module waits until the FIFO is full, and then proceeds with the packet transfer to the RBU module. However, the RPC module still takes the delayed response from the AFM module and if it is a (DA or SA) filter failure, it drops the rest of the packet and sends the Rx Status Word (with zero packet-length, CRC Error and Runt Error bits set) immediately indicating the filter fail. If there is no response from the AFM until EOP is transmitted, the filter fail status in the Rx Status Word is updated accordingly.

When the optional PMT module is present and configured for power-down mode, this block drops all received packets and does not forward the packets to the application.

2.5.4.4 Receive Flow Control Module

The Receive Flow Controller (FRX) detects the Pause packet being received and pauses the packet transmission for the delay specified within the received Pause packet. The FRX module is enabled only in the full-duplex mode.

2.5.4.5 Receive Bus Interface Unit Module

The Receive Bus Interface Unit (RBU) converts the 32-bit data received from the RPC module into a 32-bit, 64-bit, or 128-bit FIFO protocol on the Application side. The RBU module interfaces with the application through the MAC receive interface (MRI). This block also performs the endian conversion if the EQOSCORE is configured for big-endian mode.

If IEEE 1588 Timestamp feature is enabled, the RBU module also outputs the timestamp captured from the received packet, along with the status on the mri_timestamp_o bus in EQOS-CORE configuration. The value on this bus is valid when the mri_eop_o signal is asserted.

2.5.4.6 Address Filtering Module

The Address Filtering (AFM) module performs the destination and source address checking function on all received packets and reports the address filtering status to the RPC module.

The address checking is done based on different parameters (Packet Filter register) chosen by the application. These parameters are inputs to the AFM module as control signals, and the AFM module reports the status of the address filtering based on the combination of these inputs. The AFM module does not filter the receive packets but reports the status of the address filtering (whether to drop the packet or not) to the RFC module. The AFM module also reports address filter status and whether the received packet is a multicast packet or a broadcast packet.

The AFM module probes the 8-bit receive data path between the RPE module and the RFC module and checks the destination and source address field of each incoming packet. In GMII mode, the module takes 8/14 clocks (from the start of packet) to compare the destination/source address of the packet being received. Similarly, in MII mode, the module takes 14/26 clocks (from the start of packet) to compare the destination or source address of the receiving packet. The AFM module gets the physical (MAC) address of the station and the Multicast Hash table from CSR module for address checking. The CSR module provides the Packet Filter register parameters to AFM.

2.5.5 MAC Receive Interface Protocol

The MRI interface connects the application to the receive data path of the MAC with a simple FIFO-protocol interface. The RBU initiates an Ethernet packet transfer by asserting the mri_sop_o along with the data (mri_data_o). All valid data transfers are indicated by an active high on the mri_val_o signal. The RBU module transfers the last data of the packet by driving the signal mri_eop_o along with the data. The number of byte lanes having valid data in the last transfer (EOP) is indicated by the mri_be_o[3:0] signal. The encoded values of mri_be_o is the same as for mti_be_i as given in Table 2-16. The 112-bit Receive status (mri_rxstatus_o, described in Table 2-16) is also valid along with the EOP data transfer.

The MAC always assumes that the application accepts the data output by it in one clock cycle. Therefore, it does not have any acknowledgment from the application before performing the next data transfer.

Table 2-16 Receive Status at the MAC Interface

Bit	Description
127:112	Inner VLAN Tag These bits contain the Inner VLAN Tag stripped from the received packet and validated by Bit 26. This field is not available when the Enable Double VLAN Processing option is not selected.
111:96	Outer VLAN Tag These bits contain the Outer VLAN Tag stripped from the received packet and validated by Bit 25.
95:80	OAM Sub-Type Code, or MAC Control Packet opcode ■ OAM Sub-Type Code: If Bits[18:16] are set to 3'b111, this field contains the OAM sub-type and code fields. ■ MAC Control Packet opcode: If Bits[18:16] are set to 3'b110, this field contains the MAC Control packet opcode field.
79	Reserved
78	Timestamp Available If timestamp is present, when set, this bit indicates that the timestamp comes after the Rx status. The timestamp on Rx status is capture and provided in Rx status only for the valid Ethernet packets, that do not have any of the following errors: ■ Gaint packet error ■ Length error ■ CRC error ■ Invalid code error ■ watchdog timeout error
77	PTP Version When set, this bit indicates that the received PTP message has the IEEE 1588 version 2 format in the version field. When reset, it indicates the version 1 format in the version field. This bit is valid only if the Message Type is non-zero.
76	PTP Packet Type When set, this bit indicates that the PTP message is sent directly over Ethernet.

Bit	Description
75:72	PTP Message Type These bits are encoded to give the type of the message received: ■ 0000: No PTP message received ■ 0001: SYNC (all clock types) ■ 0010: Follow_Up (all clock types) ■ 0011: Delay_Req (all clock types) ■ 0100: Delay_Resp (all clock types) ■ 0101: Pdelay_Req (in peer-to-peer transparent clock) ■ 0110: Pdelay_Resp (in peer-to-peer transparent clock) ■ 0111: Pdelay_Resp_Follow_Up (in peer-to-peer transparent clock) ■ 1000: Announce ■ 1001: Management ■ 1010: Signaling ■ 1011-1110: Reserved ■ 1111: PTP packet with Reserved message type
71	IP Payload Error When this bit is set, it indicates either of the following: ■ The 16-bit IP payload checksum (that is, the TCP, UDP, or ICMP checksum) calculated by the MAC does not match the corresponding checksum field in the received segment. ■ The TCP, UDP, or ICMP segment length does not match the payload length value in the IP Header field. ■ The TCP, UDP, or ICMP segment length is less than minimum allowed segment length for TCP, UDP, or ICMP. Bit 15 (ES) of RDES3 is not set when this bit is set.
70	IP Checksum Bypassed When set, this bit indicates that the checksum offload engine is bypassed.
69	IPv6 Packet Received When set, this bit indicates that the received packet is an IPv6 packet.
68	IPv4 Packet Received When set, this bit indicates that the received packet is an IPv4 packet.
67	IP Header Error When this bit is set, it indicates either of the following: ■ The 16-bit IPv4 header checksum calculated by the MAC does not match the received checksum bytes. ■ The IP datagram version is not consistent with the Ethernet Type value. ■ Ethernet packet does not have the expected number of IP header bytes. This bit is valid when either Bit 5 or Bit 4 is set. This bit is available when you select the Enable Receive TCP/IP Checksum Check feature.

Bit	Description
66:64	Payload Type These bits indicate the type of payload, encapsulated in the IP datagram, processed by the Receive Checksum Offload Engine (COE). These bits also indicate received AV frame type. ■ 3'b000: Unknown Type or COE did not process the IP or AV payload ■ 3'b001: UDP ■ 3'b010: TCP ■ 3'b011: ICMP ■ 3'b110: AV Tagged data packet ■ 3'b111: AV Tagged control packet ■ 3'b101: AV Untagged control packet ■ 3'b100: IGMP if IPv4 Packet Received status bit is set else, DCB (LLDP) Control Packet If the COE does not process the payload of IP datagram because of an IP header error or fragmented IP, it sets these bits to 3?fb000.
63:61	Layer 3 and Layer 4 Filter Number Matched These bits indicate the number of the Layer 3 and Layer 4 Filters that matched the received packet: ■ 000: Filter 0 ■ 000: Filter 1 ■ 010: Filter 2 ■ 011: Filter 3 ■ 100: Filter 4 ■ 101: Filter 5 ■ 110: Filter 6 ■ 111: Filter 7 This field is valid only when Bit 60 or Bit 59 is set high. When more than one filter matches, these bits give only the lowest filter number.
60	Layer 4 Filter Match When set, this bit indicates that the received packet matches one of the enabled Layer 4 Port Number fields.This status is given only when one of the following conditions is true: ■ Layer 3 fields are not enabled and all enabled Layer 4 fields match ■ All enabled Layer 3 and Layer 4 filter fields match When more than one filter matches, this bit gives the layer 4 filter status of filter indicated by Bits[63:61].
59	Layer 3 Filter Match When set, this bit indicates that the received packet matches one of the enabled Layer 3 IP Address fields.This status is given only when one of the following conditions is true: ■ All enabled Layer 3 fields match and all enabled Layer 4 fields are bypassed ■ All enabled filter fields match When more than one filter matches, this bit gives the layer 3 filter status of filter indicated by Bits[63:61].
58:51	MAC Address Match or Hash Value When Bit 50 is reset, this field contains the number of the MAC address register that matched the Destination address of the received packet. When Bit 50 is set, this field contains the hash value computed by the MAC. The bit corresponding to the hash value is set in the hash filter register indicating that the frame passed the hash filter.

Bit	Description
50	Hash Filter Status When set, this bit indicates that the packet passed the MAC address hash filter. Bits[58:51] indicate the hash value.
49	DA Filter Fail When set, this bit indicates that the Destination Address filter failed, that is, the packet did not pass the DA filter.
48	SA Filter Fail When set, this bit indicates that the Source Address filter failed, that is, the packet did not pass the SA filter.
47	VLAN Filter Status When set, this bit indicates that the VLAN Tag has matched one of the VLAN Tag Filters. If this bit is reset, it indicates that the VLAN Tag of the received packet passed the filter. This bit is valid only when DWC_EQOS_ERVFE is not enabled. If DWC_EQOS_ERVFE is enabled, the bit is redefined as Outer VLAN Tag Filter Status (OTS). For more details about OTS bit, see bit[15] "RDES2 Normal Descriptor (Write-Back Format)" on page 1343
46	Inner VLAN Tag Filter Status This bit is valid only when DWC_EQOS_ERVFE is enabled. For more details about ITIS bit, see bit[14] "RDES2 Normal Descriptor (Write-Back Format)" on page 1343
45:35	Reserved
34	ARP Reply Not Generated When set, it indicates that the MAC did not generate the ARP reply for received ARP request packet. This bit is set when the MAC is busy transmitting ARP reply to earlier ARP request (only one ARP request processed at a time). This bit is reserved when the Enable IPv4 ARP Offload option is not selected.
33	Inner VLAN Tag Type When set, it indicates that the Inner VLAN type is S-VLAN in the received packet. When reset, it indicates that the Inner VLAN type is C-VLAN in the received packet. This field is reserved when the Enable Double VLAN Processing option is not selected.
32	Outer VLAN Tag Type When set, it indicates that the Outer VLAN type is S-VLAN in the received packet. When reset, it indicates that the Outer VLAN type is C-VLAN in the received packet.
31:27	Reserved
26	Inner VLAN Tag Available When set, this bit indicates that the Inner VLAN Tag stripped from the received packet is available in Bits[127:112]. This field is reserved when the Enable Double VLAN Processing option is not selected.
25	Outer VLAN Tag Available When set, this bit indicates that Outer VLAN Tag stripped from the received packet is available in Bits[111:96].

Bit	Description
24	CRC Error When set, this bit indicates that a cyclic redundancy check (CRC) error occurred on the received packet.
23	Giant Packet When set, this bit indicates that the packet length exceeds the maximum Ethernet specified size of 1,518, 1,522, or 2,000 bytes (9,018 or 9,022 bytes if jumbo packet enable is set). Giant packet indicates the length of a packet, and it does not cause any packet truncation.
22	Watchdog Timeout When set, this bit indicates that the RPE module received a packet with byte count greater than 2,048(10,240 if Jumbo packet is enabled) bytes (DA + SA + LT + DATA + PAD + FCS). This bit is not set if the watchdog timer is disabled in MAC_Configuration register. However, even if the watchdog timer is disabled, this bit is set when the packet is greater than 16 KB in size.
21	Runt Packet When this bit is set, it indicates that the MAC received packets of less than 64 bytes or a collision was observed before receiving the 64th byte.
20	GMII Error When set, this bit indicates that gmii_rxer_i was asserted during the reception of this packet. This error also includes carrier extension error in the GMII and half-duplex mode. Error can be less/no extension error (rxd = 1f) during extension
19	Dribbling Bit When set, this bit indicates that the packet contained a non-integer multiple of eight bits (valid only in MII mode). This bit is not valid if a collision is seen or runt packet bits are set. If this bit is set and the CRC error is reset, the packet is valid.
18:16	Length/Type Field This field indicates the type of the received packet: Length packet or Type packet. The encoding is as follows: ■ 3'b000: The packet is a length packet ■ 3'b001: The packet is a type packet ■ 3'b100: The packet is a type packet with VLAN Tag ■ 3'b101: The packet is a type packet with Double VLAN Tag ■ 3'b110: The packet is a MAC Control packet type ■ 3'b111: The packet is a OAM packet type ■ 3'b010 - 3'b011: Reserved The packet type indication in this field is based on the LT field of received packet. For example, VLAN Tag or Double VLAN Tag indication does not ensure that corresponding VID matched or VLAN filter passed.
15	Reserved

Bit	Description
14:0	Packet Length When set, this field indicates the length (in bytes) of one or both of the following: ■ Received packet (including pad if applicable) ■ FCS field when the Auto Pad/CRC Strip bit is de-asserted When reset, it indicates the length without pad and/or FCS field. When the IP Checksum module is present and enabled by setting Bit 10 of MAC_Configuration register and the received packet is not a MAC control packet, this field also includes the 2-byte IP Checksum appended after FCS.When Type CRC Strip bit is set, this field indicates the length without FCS field for Type packets (Length/Type >= 0x600). When VLAN Tag stripping is enabled, it indicates the length without VLAN Tag for VLAN tagged packets. For more details, see Table 2-18 on page 142

Figure 2-17 shows how the settings of S2KP and JE bits of the MAC_Configuration register impact the giant packet status.

Table 2-17 Giant Packet Status Based on S2KP and JE Bits

Length/Type Field	Received Packet Length	S2KP	JE	Giant Packet Status
Untagged Packet	>1518	0	0	1
	>2000	1	0	1
	>9018	x	1	1
VLAN Tagged Packet	>1522	0	0	1
	>2000	1	0	1
	9022	x	1	1
NOTE: For all other combinations, the Giant Packet status is 0				

Table 2-18 shows how the settings of the CST and ACS bits of the MAC Configuration register impact the inclusion of CRC length in the packet length.

Note If Type/Length < 1536, the CST field has no effect. Similarly, if Type/Length field > 1536, the ACS field has no effect.

Table 2-18 Packet Length Based on CST and ACS Bits

Receive Checksum Offload Engine	Received Packet Length	CST	ACS	FCS Stripping Done
IPCKSUM_EN = 0 and IPC_FULL_OFFLOAD = 0 or IPCHKSUM_EN = 1 and IPC_FULL_OFFLOAD = 1	<1536	x	0	No
		x	1	Yes (for Ethernet packets)
	>=1536	0	x	No
		1	x	Yes (for Type packets)
IPCKSUM_EN = 1and IPC_FULL_OFFLOAD = 0	<1536	x	0	No
		x	1	Yes (for Ethernet packets)
	>= 1536	x	x	No

2.5.6 MAC Receive Timing

This section provides timing specifications for the MAC (EQOS-CORE) interface signals in the receive path.

2.5.6.1 MAC Receive Interface (MRI)

The timing diagrams show the timing specifications for MRI signals and GMII receive interface. The sequence of events is as follows:

1. The MAC receives packets on the (G)MII interface.
2. The MAC stores the packet until it makes the decision to filter the packet (if configured). When the packet is passed by the filter, the MAC puts the data on mri_data_o and indicates the validity by asserting mri_val_o. If the data corresponds to the start of a packet, the MAC also asserts mri_sop_o.
3. The application must accept the data as soon as the MAC presents it because the MAC may output the next data (if available) in the next clock cycle.
4. The MAC indicates transfer of the end-of-packet data with the assertion of mri_eop_o. The mri_be_o signal may also have a non all-one value during the EOP, showing how many lanes on the data bus have valid data.
5. The MAC updates the receive status bits on mri_rxstatus_o during the transfer of the EOP.

Figure 2-65 MAC Receive Interface Timing (1 of 2)

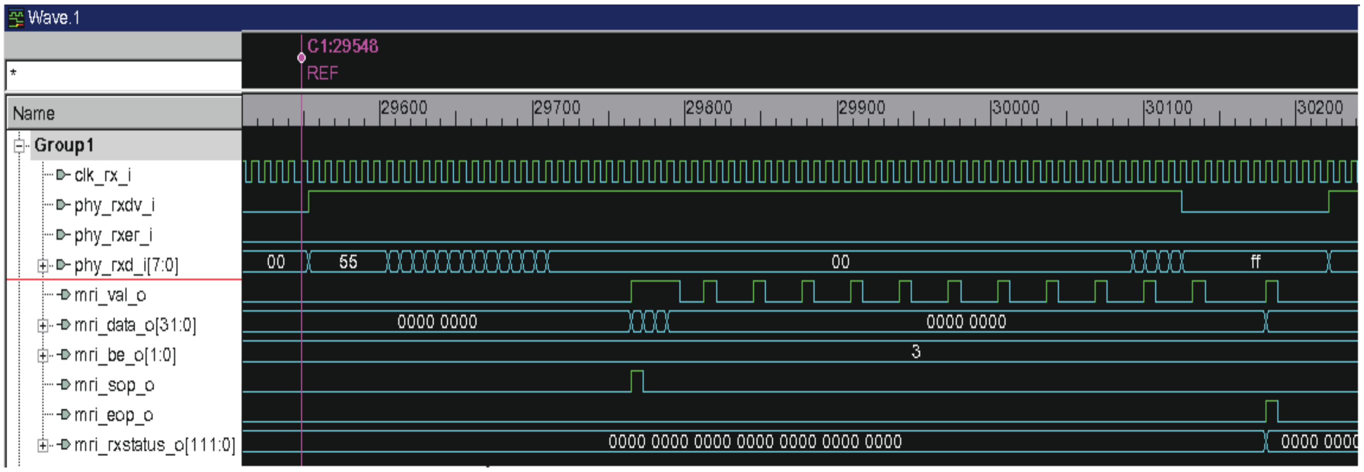
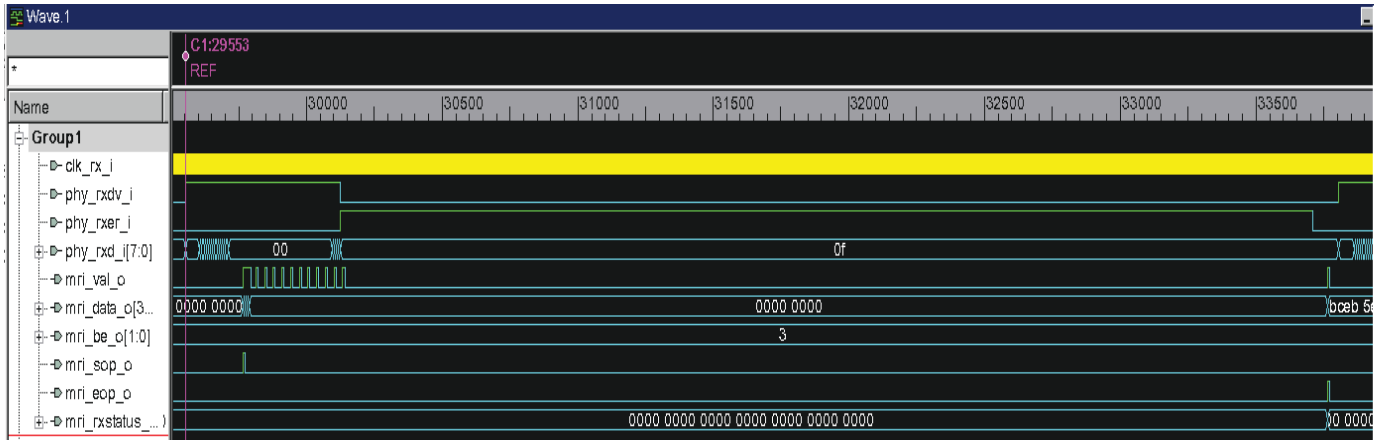


Figure 2-66 shows the reception and transfer of a packet in half-duplex mode or gigabit mode. The EOP is not transferred until the packet-extension is complete so that valid receive status can be output together.

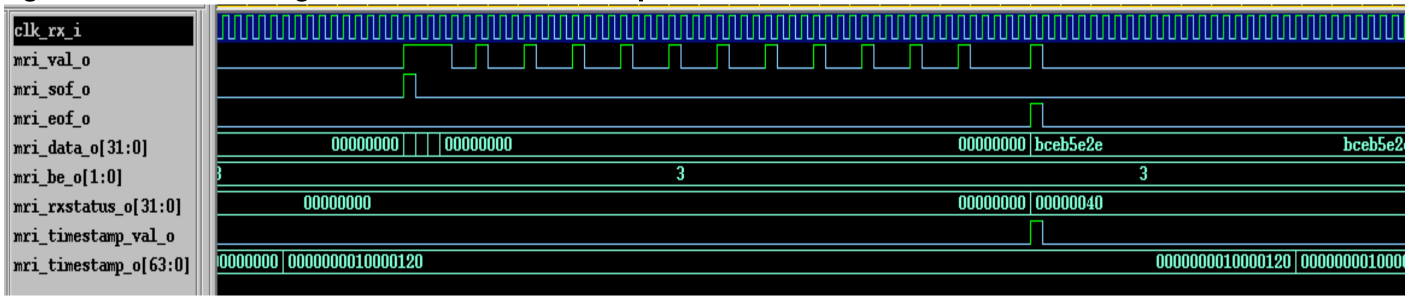
Figure 2-66 MAC Receive Interface Timing (2 of 2)



2.5.6.2 Packet Reception With Timestamp Feature Enabled

Figure 2-67 shows the MRI timing when IEEE 1588 Timestamp feature is enabled. When the MRI outputs the EOP data, as marked by mri_eop_o high, the timestamp value is also given on mri_timestamp_o. The timestamp is validated with the assertion of mri_timestamp_val_o.

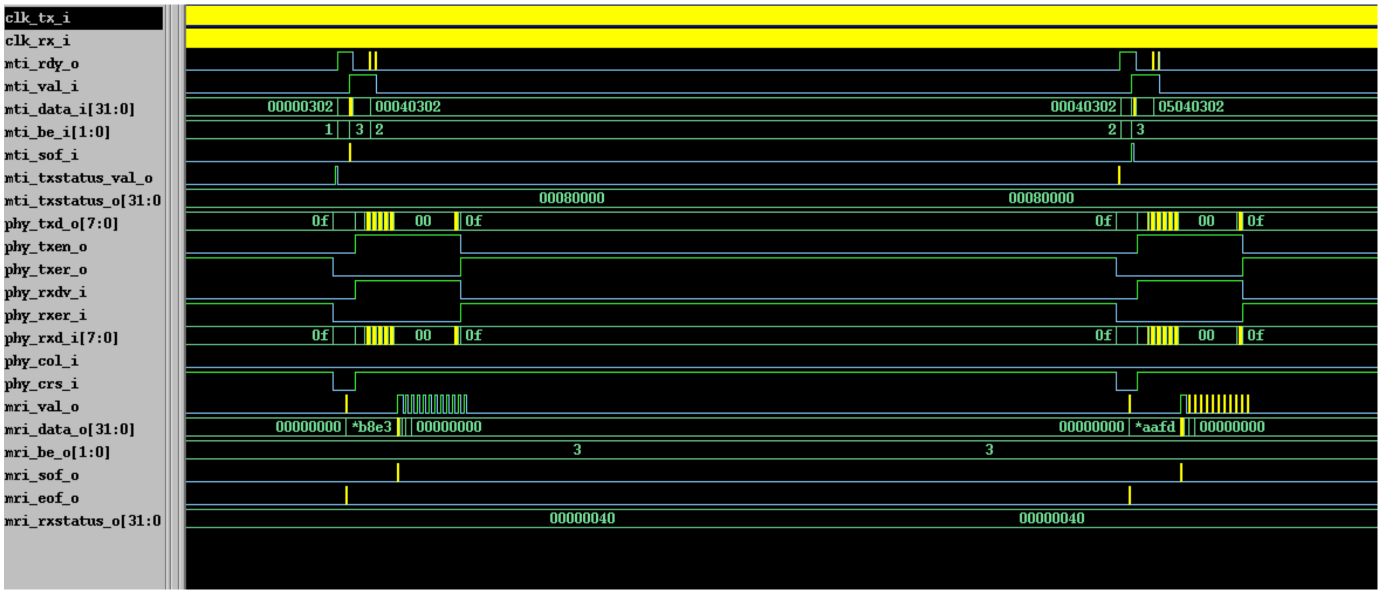
Figure 2-67 MRI Timing with IEEE 1588 Timestamp Feature Enabled



2.5.6.3 Packet Transmission and Reception

Figure 2-68 shows the timing for MTI and MRI interface signals along with the GMII signal for a packet transmission looped back to the GMII receiver in the testbench environment.

Figure 2-68 Packet Transmission and Reception



2.6 Interrupts

Interrupts can be generated as a result of various events in the DWC_ether_qos controller. These events are captured in status registers, and interrupt enables are provided for each source of an interrupt such that the interrupt signal (sbd_intr_o) is asserted for an event only when the corresponding interrupt enable is set.

In EQOS-AXI configuration, the interrupt status and corresponding enable registers are organized in an hierarchical manner so that it is easier for software to traverse and identify the source of an interrupt event quickly. When sbd_intr_o is asserted, the DMA_Interrupt_Status register is the first level that indicates the major blocks for the interrupt event source. This register is read-only, and it contains bits corresponding to each DMA channel (TX and RX pair), the MTL, and the MAC. The software application must then read one (or more) of the following registers corresponding to the bits that are set:

- MAC_Interrupt_Status register
- MTL_Interrupt_Status register
- DMA_CH(#i)_Status (for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1) register

2.6.1 Interrupts from the MAC

Interrupts can be generated from the MAC as a result of various events in the MAC Receiver, Transmitter, or the optional modules/functions such as RMON counters, EEE and so on.

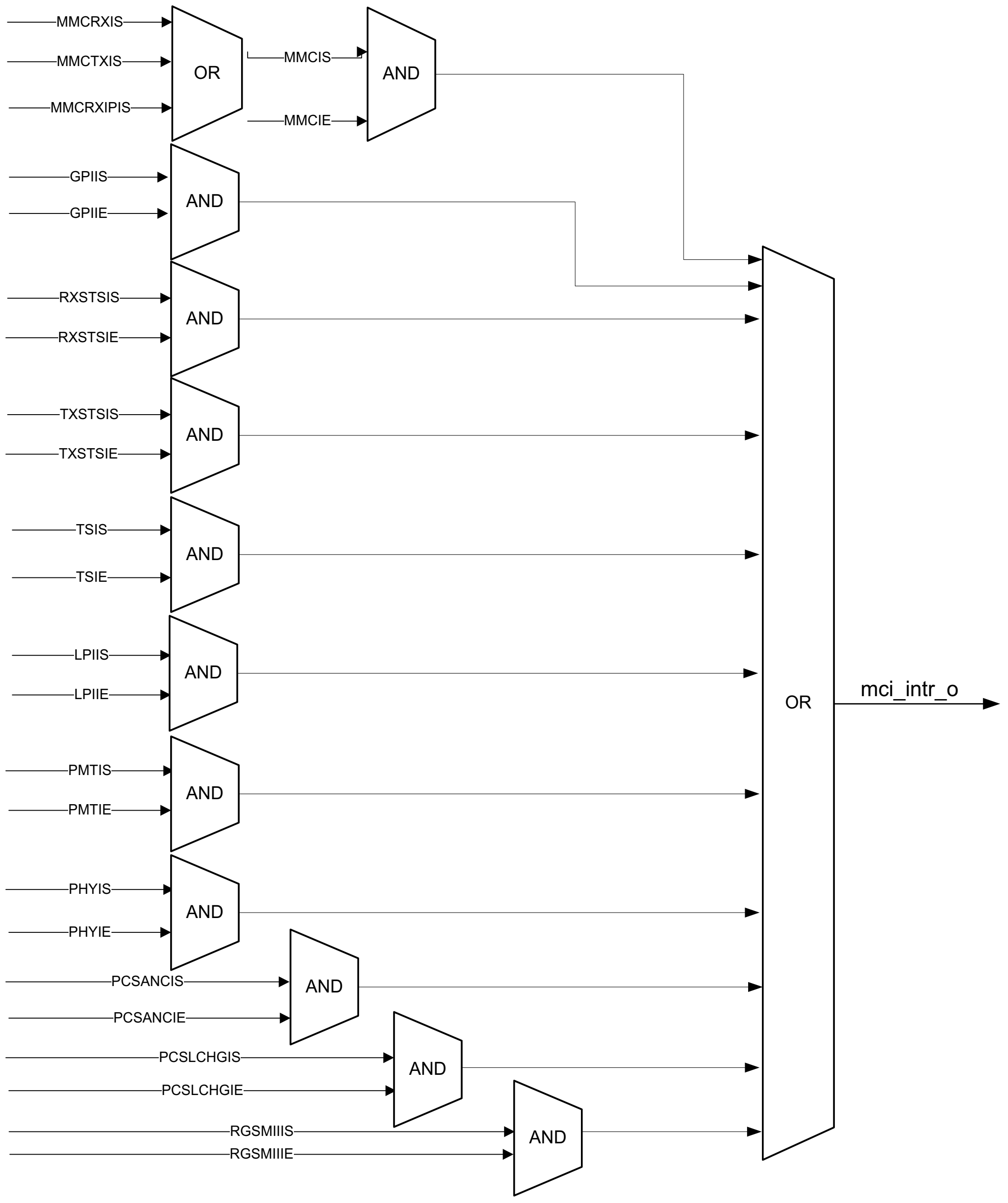
In EQOS-CORE and EQOS-MTL configurations, mci_intr_o is the interrupt signal. In EQOS-DMA and EQOS-AHB configurations, these interrupt events are combined with the events in the DMA on the sbd_intr_o signal. The MAC interrupts are of level type, that is, the interrupt remains asserted (high) until it is cleared by the application or software.

The MAC_Interrupt_Status register describes the events that can cause an interrupt from the MAC. The MAC interrupts are disabled by default. Each event can assert the interrupt on the mci_intr_o or sbd_intr_o signals when the corresponding bit is set in the MAC_Interrupt_Enable register.

The interrupt register bits only indicate the block from which the event is reported. You must read the corresponding status registers and other registers to clear the interrupt. For example, when set high, Bit 0 of the MAC_Interrupt_Status register indicates that the link status on the RGMII, SGMII, or SMII interface has changed. You must read the MAC_PHYIF_Control_Status register to clear this interrupt event.

The interrupts from the RGMII, SGMII, SMII, and PCS blocks and the optional General Purpose Inputs are combined (OR'ed) and given as the GLI bit in the DMA_Interrupt_Status register.

Figure 2-69 MAC Interrupt Enabling Scheme



Note By default, the MAC interrupt status bits are cleared when the register that contains the source of the interrupt is read. If RCWE bit in MAC_CSR_SW_Ctrl register is programmed to 1, the MAC interrupt status bits are cleared when the bit that contains the source of the interrupt is explicitly written to 1.

2.6.2 Interrupts from MTL

DWC_ether_qos can generate interrupts as a result of events in the MAC layer or in the MTL modules. In the EQOS-MTL configuration, the mci_intr_o output is the interrupt signal, which is a level signal (asserted until the interrupt source is read and cleared). In the MTL, the interrupts are mainly related to exception events in the TxQ or RxQ in the Transmit and Receive paths respectively. The interrupt status are captured and organized in a hierarchical manner in order to identify the root cause quickly.

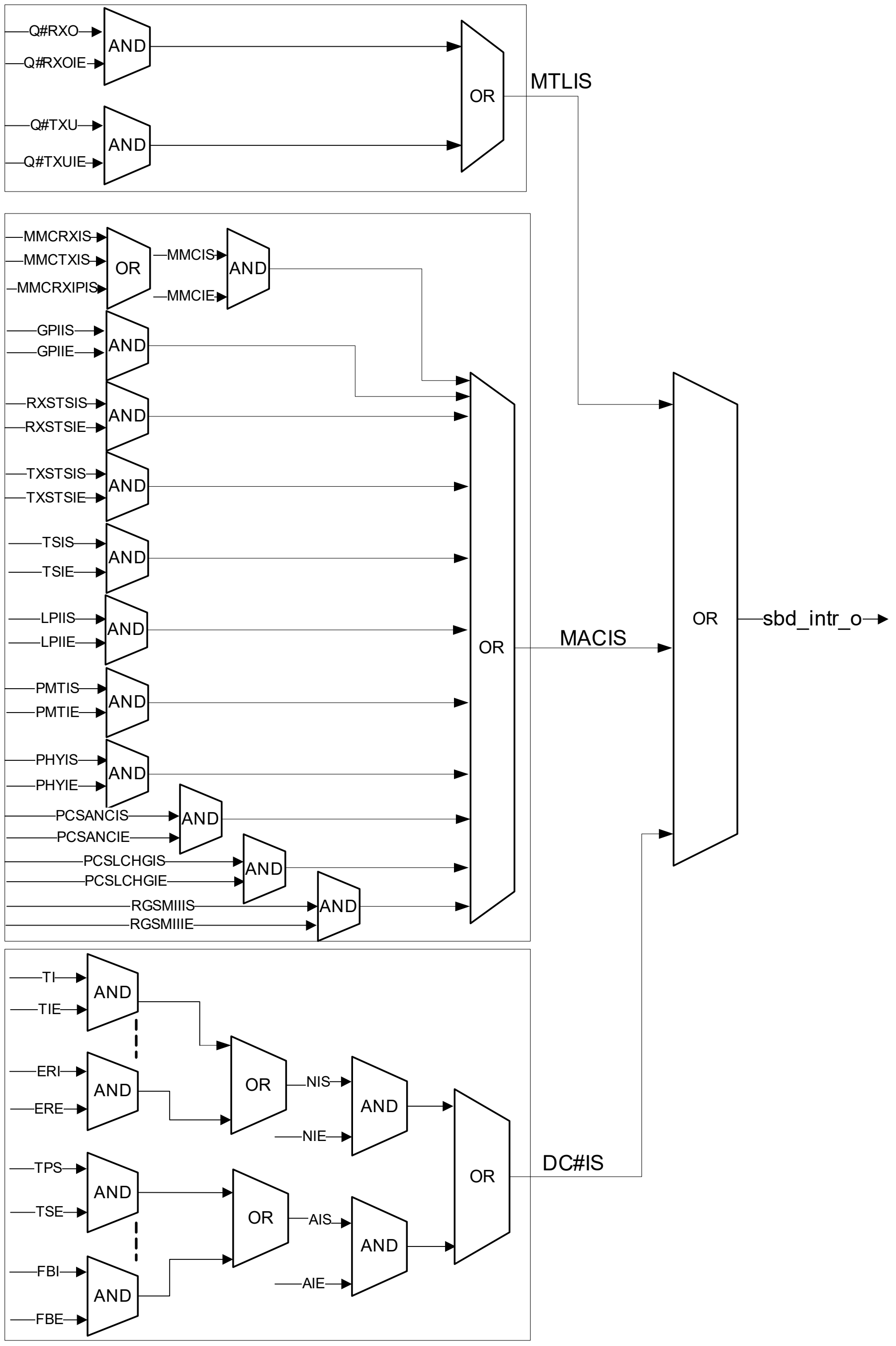
The MTL_Interrupt_Status register identifies the top level modules that can cause the interrupt to be asserted.

- Bits[7:0] identify 8 Queues (Tx or Rx). They are read-only bits and the application should read the corresponding MTL_Q[n]_Interrupt_Status register to identify the exact cause and set the corresponding bits to 1′b1 to clear the interrupt event. The assertion of mci_intr_o due to these events is enabled by the corresponding enable bits in MTL_Q[n]_Interrupt_Enable register.
- Bit[16] is made high in case the event is present in the MAC core. The application should in turn read the MTL_Interrupt_Status register to identify the cause. For more information, see "Interrupts from the MAC" on page 145.
- Bit[17] is related to indirect access completion events to the TxFIFO and RxFIFO memory in Debug Mode. For more information, see "Accessing Memory In Slave/Debug Mode" on page 161.

2.6.3 Interrupts from DMA

As shown in Figure 2-70, sbd_intr_o interrupt is a level signal and gets de-asserted only when all the enabled interrupt events are cleared in their respective status registers and correspondingly all the bits in the DMA_Interrupt_Status register are cleared.

Figure 2-70 Sbd_Intr_O Generation



The DMA_CH[n]_Status register captures all the interrupt events of that Tx DMA and Rx DMA channel pair. The DMA_CH[n]_Interrupt_Enable register contains the corresponding enable bits for each of the interrupt event. There are two groups of interrupts in the DMA channel namely Normal and Abnormal interrupts. They are indicated by Bits[15:14] of DMA_CH[n]_Status register respectively. The normal group is for events that happen during the normal transfer of packets (TI, RI, TBU) while the abnormal interrupt events are for error events. Interrupt events are cleared by writing 1’b1 to the corresponding bit position. When all the enabled interrupt events are cleared (including the NIS and AIS), the interrupt source from the DMA Channel is cleared and the corresponding bit in DMA_Interrupt_Status register is also cleared.

Interrupts are not queued. If the same interrupt event occurs again before the driver responds to the previous one, no additional interrupts are generated. For example, Receive Interrupt Bit[6] of DMA_CH[n]_Status register indicates that one or more packets were transferred to the application buffer. The driver must scan all descriptors, from the last recorded position to the first one, owned by the DMA to determine how many packets are received.

An interrupt is generated only once for multiple events. The driver must scan the DMA_Interrupt_Status register for the cause of the interrupt and clear the source in the respective Status register. The sbd_intr_o is cleared only when all the bits of DMA_Interrupt_Status register are cleared.

2.6.3.1 Periodic Scheduling of Transmit and Receive Interrupt

To improve the throughput and performance, DWC_ether_qos supports interrupt timer and transmit descriptor to generate interrupts periodically, instead of every DMA transfer.

It is not preferable to generate interrupts for every packet transferred by DMA (RI and TI) for system throughput performance reasons. The DWC_ether_qos gives the flexibility to schedule the interrupt at regular intervals using two methods:

- Set Interrupt on Completion bit in Transmit descriptor (TDES2[31]) once for every “required” number of packets to be transmitted.
- Similarly, set the IOC (RDES3[30] in Table 19-21) bit only at some specific intervals of Receive descriptors. This way, whenever a received packet transfer to system memory is complete and any of the descriptors used for that packet transfer has the IOC bit set, only then the RI event is generated.

In addition to above, an interrupt timer (DMA_CH[n]_Rx_Interrupt_Watchdog_Timer) is given for flexible control and periodic scheduling of Receive Interrupt. When this interrupt timer is programmed with a nonzero value, it gets activated as soon as the Rx DMA completes a transfer of a received packet to system memory without asserting the Receive Interrupt because the corresponding interrupt of completion IOC bit (RDES3[30] in Table 19-21) is not set. When this timer runs out as per the programmed value, RI bit is set and the interrupt is asserted if the corresponding RIE is enabled in DMA_CH[n]_Interrupt_Enable register.The timer is stopped and cleared before it expires, if the RI is set for a packet transfer whose descriptor’s IOC was set. The timer is reactivated automatically after the next packet transfer is complete without the RI event being generated.

2.6.3.2 Per Channel Transfer Complete Interrupt

The Transmit Transfer complete interrupt (TI) and Receive Transfer complete interrupt (RI) is reflected in DMA_CH[n]_Status register. The TI bit is set whenever the Tx DMA channel closes the transmit descriptor in which the IOC (Interrupt On Completion - TDES2[31]) bit is set. Similarly, the RI bit is set whenever the Rx DMA channel closes the receive descriptor with LD bit set and in any of the descriptors used for transferring that packet, IOC (Interrupt Enable on completion - RDES3[30]) bit is set.

The common sbd_intr_o output signal is asserted for the Transfer complete interrupts only when the corresponding interrupts are enabled in DMA_CH[n]_Interrupt_Enable register.

EQOS-AXI also supports the following per Channel Transfer Complete interrupt signals.

- sbd_perch_tx_intr_o[] (Transmit Per Channel Interrupts)
- sbd_perch_rx_intr_o[] (Receive Per Channel Interrupts)

The behavior of the RI/TI/ sbd_perch_tx_intr_o[]/sbd_perch_rx_intr_o[] changes depending on the settings of INTM field bit[17:16] in DMA_Mode. Table 2-19 explains the Transfer Complete Interrupt behavior.

Table 2-19 DWC_ether_qos Transfer Complete Interrupt Behavior

Sl No	Interrupt Mode	Behavior of the sbd_perch_tx_intr_o[] and sbd_perch_rx_intr_o[]	Behavior of TI/RI and sbd_intr_o
1.	INTRM=0	A pulse is asserted on these output signals when corresponding TX/RX transfer complete event (for which IOC descriptor bits are enabled) is detected, irrespective of the corresponding interrupt status.	The TI/RI status signals are set whenever the "Transfer complete" event is detected. The bits get cleared whenever the software driver writes'1' to these bits. The sbd_intr_o is asserted when ever the corresponding interrupts are also enabled inDMA_CH[n]_Interrupt_Enable register.
2.	INTM=1	These signals reflect the value of corresponding TI/RI bits in DMA_CH[n]_Status register when the corresponding interrupt enable is set. Hence they are level signals and are cleared by the application by writing 1'b1 to the RI/TI status bits. This signal will not be asserted when the corresponding interrupt enable bit is not set.	The TI/RI is set as explained above.However, the sbd_intr_o signal and NIS status bit are not asserted for any RI/TI events.
3.	INTM=2	In this mode, RI/TI interrupts are queued. These signals reflect the value of corresponding TI/RI bits inDMA_CH[n]_Status register when the corresponding interrupt enable is set. They are level signals and cleared by software by writing 1'b1 to the RI/TI status bits. However, it is set again if another TI/RI event(s) is detected before the TI/RI bits are cleared forthe previous event.	The RI/TI status bits are set whenever the Transfer Complete event is detected and gets reset whenever software driver clears those bits by writing 1. However, if another Transfer Complete event is detected before it is cleared (serviced) by the software, then DWC_ether_qos automatically set these status bits again.However, the sbd_intr_o signal is not generated based on TI/RI

3 VLAN and Double VLAN Insertion, Deletion, Replacement and Tagging

This chapter describes the double VLAN tagging feature. It contains the following sections:

3.1 Double VLAN Processing

DWC_ether_qos supports two VLAN tags, namely inner and outer, for processing double VLANs.

3.1.1 Description of Double VLAN Processing

This feature is referred as the double VLAN tagging feature in which the MAC can process two VLAN tags. With this feature, the DWC_ether_qos supports the following:

- Insertion, replacement, or deletion of up to two VLAN tags in the Transmit path.
- Packet filtering and stripping based on any one of the two VLAN Tags in the Receive path. Stripping and providing up to two VLAN Tags in the Receive path as a part of the Receive status.

3.1.1.1 Transmit Path

Table 3-1 describes the features supported by the MAC on the Transmit side.

Table 3-1 Double VLAN Processing Features in Transmit Path

Feature	Description
Support for C-VLAN and S-VLAN Tag types	The inner or outer VLAN tag can be of C-VLAN and S-VLAN type. The VLAN type is specified through the CSVL bit of MAC_VLAN_Incl and MAC_Inner_VLAN_Incl registers. The DWC_ether_qos supports processing of any sequence of outer and inner VLAN tags. **Note:** The DWC_ether_qos does not support the C-VLAN S-VLAN sequence. The MAC does not check whether the packet provided by the application has a valid sequence of the VLAN Tag types or the insertion or replacement operation results in invalid sequence of VLAN Tag type. Therefore, the application must provide correct sequence of VLAN Tag types and program the MAC in such a way that it results in correct sequence of VLAN Tag types in the transmitted packet. The application must ensure the following: ■ The inner tag should not be S-VLAN when outer C-VLAN Tag insertion is enabled. ■ The outer tag should not be C-VLAN when inner S-VLAN Tag insertion is enabled. ■ The inner tag should not be S-VLAN when outer tag should be replaced with C-VLAN. ■ The outer tag should not be C-VLAN when inner tag should be replaced with S-VLAN.
VLAN Tag deletion	You can enable the VLAN tag deletion for outer or inner tag through VLC field in the MAC_VLAN_Incl or MAC_Inner_VLAN_Incl register, respectively. When VLAN deletion is enabled, the MAC deletes the tag present at the corresponding position. When a packet has only one tag, it is considered as the outer tag. If inner tag deletion is enabled and the packet has only one tag, the MAC does not delete the tag.
VLAN Tag Insertion or Replacement	You can enable the VLAN tag insertion or replacement for outer or inner tag through VLC field in the MAC_VLAN_Incl or MAC_Inner_VLAN_Incl register, respectively. When VLAN tag insertion or replacement is enabled, the VLTI bit in the MAC_VLAN_Incl or MAC_Inner_VLAN_Incl register is used to determine whether the VLAN tag should be taken from the register or the Control Word.

3.1.1.2 Receive Path

Table 3-2 describes the features supported by the MAC on the Receive side and the corresponding bits in the MAC_VLAN_Tag register.

Table 3-2 Double VLAN Processing in Receive Path

Feature	Description
Outer or inner VLAN tag-based filtering	The MAC can filter packets based on the outer or inner VLAN tag through the ERIVLT bit.
C-VLAN or S-VLAN tag-based filtering	The MAC can filter packets based on the C-VLAN or S-VLAN type based on the ERSVLM bit.
Outer and Inner VLAN Tag stripping	The MAC can strip the outer and inner VLAN Tags from received frame based on the EVLS and EIVLS bits.
16-bit outer and inner VLAN Tag and Type in Rx status	The MAC can provide the 16-bit outer and inner VLAN Tag and Type in the Rx status based on the EVLRXS and EIVLRXS bits, respectively.
Disabling or skipping checking of outer VLAN Tag type	The MAC can disable or skip checking of outer VLAN Tag type to match C-VLAN or S-VLAN based on the DOVLTC bit.

3.1.2 Double VLAN-Related Registers

Following are the Double VLAN-related registers:

- MAC_VLAN_Incl
- MAC_Inner_VLAN_Incl
- MAC_VLAN_Tag

3.1.3 Enabling Double VLAN Processing

To enable this feature, select the **Enable Double VLAN Processing** option under the General Features section during the Specify Configuration activity in coreConsultant.

1. Select the **Enable Double VLAN Processing** option under the General Features section during the **Specify Configuration** activity in coreConsultant.

   For details about this option, see General Features Parameters in the "Parameter Descriptions" on page 419.

Note See the "Enabled" field of the parameter descriptions to understand the dependencies

3.2 Source Address and VLAN Insertion, Replacement, or Deletion

DWC_ether_qos supports SA and VLAN insertion, replacement and deletion.

3.2.1 Description of SA and VLAN Insertion, Replacement, or Deletion

The source address (SA) and VLAN fields are Tx packet-related control information that are provided as part of the control word through ATI interfaces. DWC_ether_qos supports the feature to insert or replace the source address based on the information in the MAC Address Registers, and also the feature to insert, replace or delete the VLAN fields (VLAN Type and VLAN Tag) based on the setting of the VLTI bit in the MAC_VLAN_Incl register. You can enable the SA insertion or replacement feature for all Transmit packets or selective packets. Similarly, you can enable the VLAN insertion, replacement, or deletion feature for all Tx packets or selective packets.

3.2.2 Enabling SA and VLAN Insertion, Replacement, or Deletion

To enable this feature while configuring the controller in the coreConsultant GUI:

1. Select the **Enable SA and VLAN Insertion on Tx** option under the General Features section during the **Specify Configuration** activity in coreConsultant.

   For details about this option, see General Features Parameters in "Parameter Descriptions" on page 419.

Note See the "Enabled" field of the parameter descriptions to understand the dependencies

3.2.3 Programming Source Address Insertion or Replacement

The software can use the SA insertion or replacement feature to instruct the MAC to do the following for Tx packets:

- Insert the content of the MAC Address Registers in the SA field
- Replace the content of the SA field with the content of the MAC Address Registers

When SA insertion is enabled, the application must ensure that the packets sent to the MAC do not have the SA field. The MAC does not check whether the SA field is present in the Transmit packet and it inserts the content of MAC Address Registers in the SA field. Similarly, when SA replacement is enabled, the application must ensure that the SA field is present in the packets sent to the MAC. The MAC replaces the six bytes following the Destination Address field in the Transmit packet with the content of the MAC Address Registers.

You can enable the SA insertion or replacement feature for all Transmit packets or selective packets:

- Enabling SA insertion or replacement for all packets

  To enable this feature for all packets, program the SARC field of the MAC_Configuration register.
- Enabling SA insertion or replacement for selective packets

  To enable this feature for selective packets, use the following method depending on the configuration:
  - **EQOS-MTL or EQOS-CORE**:

    Use the SA Insertion Control from ATI Control Word bit[26:24] and mti_sa_ctrl_i [2:0] input signal, respectively. When Bit 26 of ATI Control Word or mti_sa_ctrl_i signal is high, it indicates insertion or replacement by the MAC Address1 registers. When Bit 26 of ATI Control Word or mti_sa_ctrl_i signal is low, it indicates insertion or replacement by the MAC Address0 registers.

- **EQOS-AHB, EQOS-AXI, or EQOS-DMA**:

  Program the SA Insertion Control field (Bits[25:23] of TDES3) in the first Transmit descriptor of the packet. When Bit 25 of TDES3 is set, the SA Insertion Control field indicates insertion or replacement by MAC Address1 registers. When Bit 25 of TDES3 is reset, it indicates insertion or replacement by MAC Address 0 registers.

If MAC Address1 Registers are not enabled, the MAC Address0 registers are used for insertion or replacement irrespective of the value of the most-significant bit of the SA Insertion Control field.

3.2.4 Programming VLAN Insertion, Replacement, or Deletion

The software can use the VLAN insertion, replacement, or deletion feature to instruct the MAC to do the following for Tx packets:

- Delete the VLAN Type and VLAN Tag fields
- Insert or replace the VLAN Type and VLAN Tag fields

  Insertion or replacement is done based on the setting of VLTI bit in the MAC_VLAN_Incl register as described Table 3-3

Table 3-3 VLAN Insertion or Replacement Based on VLTI Bit

Condition	Description
VLTI bit is set	The MAC inserts or replaces the following: ■ VLAN Type field (C-VLAN or S-VLAN as indicated by the CSVL bit of MAC_VLAN_Incl register) ■ VLAN Tag field with one of the following depending upon the configuration: ❑ **EQOS-MTL**: The Outer VLAN Tag or Inner VLAN Tag field of ATI control word. ❑ **EQOS-CORE**: Content of the mti_vlan_tag_i signal ❑ **EQOS-AHB, EQOS-AXI, or EQOS-DMA**: Content of the VT field of Transmit context descriptor of the packet
VLTI bit is reset	The MAC inserts or replaces the following: ■ VLAN Type field (C-VLAN or S-VLAN as indicated by the CSVL bit of MAC_VLAN_Incl register) ■ VLAN Tag field with the VLT field of MAC_VLAN_Incl register

When VLAN replacement or deletion is enabled, the MAC checks if the VLAN Type field (0x8100 or 0x88a8) is present after the DA and SA fields in the Transmit packet. The replace or delete operation does not occur if the VLAN Type field is not detected in two bytes following the DA and SA fields. However, when VLAN insertion is enabled, the MAC does not check the presence of VLAN Type field in the Transmit packet and just inserts the VLAN Type and VLAN Tag fields.

You can enable the VLAN insertion, replacement, or deletion feature for all Tx packets or selective packets:

- Enabling VLAN insertion, replacement, or deletion for all packets

  To enable this feature for all packets, program the VLC and VLP fields of MAC_VLAN_Incl register.
- Enabling VLAN insertion, replacement, or deletion for selective packets

  To enable this feature for selective packets, use the following method depending upon the configuration:
  - **EQOS-MTL**: Program the VTIR field of ATI control word.
  - **EQOS-CORE**: Use the mti_vlan_ctrl_i input.
  - **EQOS-AHB, EQOS-AXI, or EQOS-DMA**: Program the VTIR field of TDES2 Normal Descriptor (see Figure 19-5 on page 1324).

  In addition, the VLP (VLAN Priority control) bit must be reset in Register 24 (for outer VLAN) and Register 25 (in inner VLAN) for the MAC to take the control inputs from the host, depending on the configuration.

3.3 Queue/Channel Based VLAN Tag Insertion on Tx

DWC_ether_qos supports channel/queue based VLAN tag insertion on all transmitted packets.

3.3.1 Accessing Queue/Channel Specific VLAN Tag Registers

Queue/Channel specific VLAN tag registers are accessed using indirect addressing via the MAC_VLAN_Incl register. VLAN type and tag value can be independently programmed for each queue/channel.

3.3.2 Enabling Queue/Channel Based VLAN Tag Insertion on Tx

To enable this feature while configuring the controller in the coreConsultant GUI:

1. Select the **Enable Queue/Channel based VLAN tag insertion on Tx** option under the General Features section during the **Specify Configuration** activity in coreConsultant.

For details about this option, see General Features Parameters in the "Parameter Descriptions" on page 419.

Note See the "Enabled" field of the parameter descriptions to understand the dependencies

When you enable this feature and the CBTI field is set in the MAC_VLAN_Incl register, the VLAN tag is inserted on every packet that is transmitted from a queue/channel, with the programmed tag value taken from queue/channel specific VLAN tag register

3.3.3 Programming Guidelines for Channel Based VLAN Tag Insertion

See "Programming Sequence for Queue/Channel Based VLAN Inclusion Register" on page 1379

4

Managing Buffers and Memories

This chapter describes buffers and memories of DWC_ether_qos. It contains the following sections:

4.1 Introduction to Transmit and Receive FIFOs

The Transmit FIFO (Tx FIFO) buffers the data transferred from the application to the DWC_ether_qos. Similarly, the Receive FIFO (Rx FIFO) stores the packets received from the line until they can be transferred to the application. These are asynchronous FIFOs because they also transfer the data between the application clock and the MAC line clocks. Tx memory and Rx memory are required to be two-ported RAM of 35-bit or 68-bit or 133-bit width for 32, 64 or 128 data bus widths, respectively. The extra bits are used for storing the byte-enable information.

DWC_ether_qos also provides an option to utilize SPRAM for the Tx memory and Rx memory instead of the two-ported RAM. In this configuration, the transfer of data across the clock domains is done using small register based FIFOs inside the controller, while the FIFO memory is utilized only for buffering the data.

When multiple queues are selected, all Tx queue share the Tx FIFO memory and all Rx queue share the Rx FIFO memory. The application can program the size of the FIFO memory allocated to each Tx or Rx queue.

4.2 Description of Transmit and Receive FIFOs

For ease of integration and testing, both memories are instantiated outside the core. You must add technology-specific memory modules and related BIST logic outside the core, and connect the data bus, address, and control signals to DWC_ether_qos. DWC_ether_qos provides a direct read or write access to the FIFO memories in Debug mode.

4.3 Transmit and Receive FIFO-Related Registers

Following are the Transmit and Receive FIFO-related registers:

- MTL_TxQ0_Operation_Mode
- MTL_RxQ0_Operation_Mode

4.4 Configuring Transmit and Receive FIFO Details

1. Specify the FIFO size, address bus width, and the data width under the **Buffer Management** section during the **Specify Configuration** activity in coreConsultant.

For details about this option, see Buffer Management Parameters in the "Parameter Descriptions" on page 419.

Note See the "Enabled" field of the parameter descriptions to understand the dependencies

4.5 TCP/IP Header Buffer

The TCP/IP Header Buffer exists when a separate memory is enabled for the TCP/IP headers. This memory is external to DWC_ether_qos and requires a single-port SRAM.

This buffer is present only when **Enable Separate Memory for TCP/IP Headers** option is selected in the coreConsultant GUI. A single-port SRAM is required for this memory and is external to DWC_ether_qos. DWC_ether_qos supports TCP/IP segmentation (TSO) offloading, and the TCP/IP Header memory is used to store the TCP/IP headers for segmented packets. The memory size can be configured from 64 bytes to 1KB in powers of 2 for each TSO channel. The total size gets multiplied by the number of DMA channels requiring TSO feature. The memory width is the same as the data-width configured (32, 64 or 128). For more details see, "External TSO Memory Interface" on page 356

4.6 Accessing Memory In Slave/Debug Mode

The DWC_ether_qos supports the debug access to the FIFO memory. When this feature is enabled, the MAC inserts the pad (if required) and the computed CRC in the Transmit packet. The FIFOs operate in the store-and-forward mode.

When the FIFO Debug Access is enabled, ARI/ATI interface should not be active. For MTL configurations, the ATI interface should not be driven with any transactions, and ARI_READY should not be asserted on the ARI interface. For DMA configurations, the corresponding Tx/Rx DMA should be disabled.

This section describes accessing memory in debug mode. It contains the following sub-sections:

- "Enabling Memory Access in Slave/Debug Mode" on page 161
- "Sending or Receiving a Packet through a Slave Interface" on page 162
  - "Transmitting Packets through a Slave Interface" on page 162
  - "Receiving Packets through a Slave Interface" on page 164
- Operating in Debug Mode
  - "Writing to FIFO Memory in Debug Mode" on page 165
  - "Reading from FIFO Memory in Debug Mode" on page 165

Note The debug access mode is not designed for dynamic switching from normal operation mode to debug access mode. The software/Application needs to ensure that the Transmit and Receive paths are idle (All the Debug status registers in CSR space are read as all-zeros). The Software/Application should wait for idle state or use mechanism such as Tx Queue flush or soft reset to reach the idle state faster, prior to switching to Debug access mode.

4.6.1 Enabling Memory Access in Slave/Debug Mode

To enable this feature while configuring the controller in the coreConsultant GUI:

1. Select the **Enable Debug Memory Access** option under the Buffer Management section during the **Specify Configuration** activity in coreConsultant
2. Program the entire selected Tx memory to Tx Queue 0 and Rx memory to Rx Queue 0.

Do not enable offload features such as Checksum Offload Engine and IEEE timestamp with this feature.

For details about Enable Debug Memory Access option, see Buffer Management Parameters in the "Parameter Descriptions" on page 419.

Note See the “Enabled” field of the parameter descriptions to understand the dependencies

4.6.2 Sending or Receiving a Packet through a Slave Interface

You can use this mode to send or receive a packet through a slave interface. The following sections describe how to transmit or receive a packet through a slave interface.

In the slave/debug mode access flow, any two successive access should have a minimum time-gap of (3 cycles of Tx clock + 3 Cycles of CSR clock) while accessing the Tx and Rx Memory, and (3 cycles of App clk+ 3 Cycles of CSR clock) while accessing the TSO memory.

4.6.2.1 Transmitting Packets through a Slave Interface

To write a packet to the Transmit memory through a slave interface, complete the steps given in Table 4-1.

Table 4-1 Steps to Write a Packet to Transmit Memory Through a Slave Interface

Register Name	Step Number	Description
MTL_DBG_CTL	1	Select the FIFO for slave access by using the DBGMOD and FDBGEN fields.
MTL_FIFO_Debug_Data	2	Write the Packet control word. For more information, see “Transmit Control Word Format” on page 84.
MTL_DBG_CTL	3	Complete the following: 1. Set the PKTSTATE field to Control field. 2. Set the FIFOSEL field to Tx FIFO. 3. Set the FIFOWREN bit. 4. Reset the FIFORDEN bit.
MTL_DBG_STS	4	Read the FIFOBUSY bit and wait till this bit is low. When this bit is low, read the LOCR field to know the number of free locations.
MTL_FIFO_Debug_Data	5	Write the SOP, the first word of the packet data.
MTL_DBG_CTL	6	Complete the following steps: 1. Set the PKTSTATE field to SOP. 2. Set the FIFOSEL field to Tx FIFO. 3. Set the FIFOWREN bit.
MTL_DBG_STS	7	Read the FIFOBUSY bit and wait till this bit is low. When this bit is low, read the LOCR field to know the number of free locations.
MTL_FIFO_Debug_Data	8	Write subsequent data words.

Register Name	Step Number	Description
MTL_DBG_CTL	9	Complete the following steps: 1. Set the PKTSTATE field to Packet Data. 2. Set the FIFOSEL field to Tx FIFO. 3. Set the FIFOWREN bit.
MTL_DBG_STS	10	Read the FIFOBUSY bit and wait till this bit is low. When this bit is low, read the LOCR field (if required).
MTL_DBG_CTL and MTL_DBG_STS	11	Repeat step 8 and step 10 till the penultimate word of the packet data.
MTL_FIFO_Debug _Data	12	Write the last data word in the FDBGDATA field.
MTL_DBG_CTL	13	Complete the following steps: 1. Set the PKTSTATE field to EOP. 2. Set the FIFOSEL field to Tx FIFO. 3. Set the FIFOWREN bit. 4. Set the number of valid bytes in the field.
MTL_DBG_STS	14	Read the FIFOBUSY bit and wait till this bit is low. When this bit is low, read the LOCR field to know the number of free locations.

After Step 4 in Table 4-1, the core automatically initiates the packet transmission. The status of the packet transmission can be obtained as explained in Table 4-2. To get the status of a Tx packet, complete the steps given in Table 4-2.

Table 4-2 Steps to Get the Status of a Transmit Packet

Register Name	Step Number	Description
MTL_DBG_STS	1	Wait for the STSI interrupt bit to set.
MTL_DBG_CTL	2	Complete the following steps: 1. Set the FIFOSEL field to Tx Status FIFO. 2. Reset the FIFWREN bit. 3. Set the FIFORDEN bit.
MTL_DBG_STS	3	Read the FIFOBUSY bit and wait till this bit is low.
MTL_FIFO_Debug _Data	4	Read the FDBGDATA field.

4.6.2.2 Receiving Packets through a Slave Interface

To read a packet from the Rx memory through a slave interface, complete the steps given in Table 4-3.

Table 4-3 Steps to Read a Packet from Rx Memory Through a Slave Interface

Register Name	Step Number	Description
MTL_DBG_CTL	1	Select the FIFO for slave access by using the DBGMOD and FDBGEN fields.
MTL_DBG_STS	2	Wait for the PKTI interrupt.
MTL_DBG_CTL	3	Complete the following: 1. Set the FIFOSEL field to Rx FIFO. 2. Reset the FIFOWREN bit. 3. Set the FIFORDEN bit.
MTL_DBG_STS	4	Read the FIFOBUSY bit and wait till this bit is low.
MTL_DBG_CTL	5	Read the PKTSTATE and fields.
MTL_FIFO_Debug_Data	6	Read the data in the FDBGDATA field.
MTL_DBG_STS	7	Read the LOCR field to know the number of available locations.
MTL_DBG_STS, MTL_DBG_CTL, MTL_FIFO_Debug_Data	8	Repeat Step 4 through Step 7 till the complete packet, along with the Rx status, is received. The format of the received packet for MTL configuration is as follows: ■ First Status Word (One or multiple words) ■ Last Status Word (One Word. Acts as delimiter between the status and data) ■ Frame Data Word (multiple) ■ Last Frame Data Word (One Word. Acts as a delimited among subsequent packets on the received Queue) Format for the received packet for QOS-DMA, EQOS-AXI, EQOS-AHB configurations is as follows: ■ DMA Header Status (present only in case of multiple Rx DMA channel configuration) The bit[`DWC_EQOS_NRXCH] indicates the DMA Channel number format. ❑ When set to 1 it indicates that the DMA channel number is indicated as per bit format in bit[`DWC_EQOS_NRXCH-1:0] of the Header Status. ❑ When set to 0, it implies that the DMA Channel number is indicated by channel number in bit[`DWC_EQOS_NRXCH-1:0] of the Header Status. ■ Frame Data Word (multiple) ■ Last Frame Data Word (one word. Acts as a delimiter among subsequent packets on the received Queue The packet status format in the debug memory access mode is same as that of non-debug mode. For more details of receive packet status format, see "Receive Status Word Format" on page 101

4.6.3 Operating in Debug Mode

Debug Mode enables you to do both the following operations:

- Write and Read back the Tx Queue
- Read and Write to the Tx Queue

This mode is useful for Memory BIST applications.

4.6.3.1 Writing to FIFO Memory in Debug Mode

To write to FIFO memory in the debug mode, complete the steps given in Table 4-4.

Table 4-4 Steps to Write to FIFO Memory in Debug Mode

Register Name	Step Number	Description
MTL_DBG_CTL	1	Select the FIFO for debug access by using the DBGMOD and FDBGEN fields.
MTL_FIFO_Debug_Data	2	Write the data in the FDBGDATA field.
MTL_DBG_CTL	3	Complete the following: 1. Set the FIFOSEL field to Tx FIFO, Rx FIFO, or TSO FIFO. 2. Reset the FIFORDEN bit. 3. Set the FIFOWREN bit.
MTL_DBG_STS	4	Read the FIFOBUSY bit and wait till this bit is low. When this bit is low, read the LOCR field (if required).
MTL_DBG_CTL, MTL_DBG_STS	5	Repeat Step 3 and Step 4 twice to write one complete data word in 64-bit mode. Repeat these steps four times in 128-bit mode.

4.6.3.2 Reading from FIFO Memory in Debug Mode

To read from FIFO memory in the debug mode, complete the steps given in Table 4-5.

Table 4-5 Steps to Read from FIFO Memory in Debug Mode

Register Name	Step Number	Description
MTL_DBG_CTL	1	Select the FIFO for debug access by using the DBGMOD and FDBGEN fields.
MTL_DBG_CTL	2	Complete the following: 1. Set the FIFOSEL field to Tx FIFO, Rx FIFO, or TSO FIFO. 2. Reset the FIFOWEN bit. 3. Set the FIFORDEN bit.
MTL_DBG_STS	3	Read the FIFOBUSY bit and wait till this bit is low. When this bit is low, read the LOCR field (if required).
MTL_FIFO_Debug_Data	4	Read the data in the FDBGDATA field.

Register Name	Step Number	Description
MTL_DBG_STS	5	Read the LOCR field (if required).
MTL_DBG_CTL, MTL_DBG_STS, MTL_FIFO_Debug_Data	6	Repeat Step 2 and Step 5 twice to read one complete data word in 64-bit mode. Repeat these steps four times in 128-bit mode.

5 Using PHY Interfaces

This chapter describes the Station Management module and different PHY interfaces. It contains the following sections:

5.1 Station Management Agent

5.1.1 Introduction to Station Management Agent

The application can access the PHY registers through the Station Management Agent (SMA) module. SMA is a two-wire Station Management interface (MIM).

5.1.2 Functional Description of Station Management Agent

For MIM accesses, the maximum operating frequency of the MDC (gmii_mdc_o) is 2.5 MHz, as specified in the IEEE 802.3. In the DWC_ether_qos core, the gmii_mdc_o clock is derived from the application clock or clk_csr_i, using a divider-counter. The divide factor depends on the clock range setting (CR field) in the MAC_MDIO_Address register

Select the clock divide factor as mentioned in the description of CR field of MAC_MDIO_Address register, to meet IEEE specifications. However, if your system supports higher clock frequencies on the MIM interface, there is a provision to select a different divider.

The MDIO frame structure is as follows:

Table 5-1 MDIO Clause 45 Frame Structure

Field	Description
IDLE	The mdio line is in tri-state; there is no clock on gmii_mdc_o signal.
PREAMBLE	32 continuous bits of value 1
START	Start of packet is 2’b00
OPCODE	■ 2’b00 ■ 2’b01 ■ 2’b10 ■ 2’b11
PHY ADDR	5-bit address select for one of 32 PHYs
DEV ADDR	5-bit address select for one of 32 devices
TA	Turnaround ■ 2’bZ0: Read and post-read increment address ■ 2’b10: Write and address MDIO accesses Where Z is the tri-state level
DATA/ADDRESS	16-bit value: For an address cycle (OPCODE = 2'b00), this frame contains the address of the register to be accessed on the next cycle. For the data cycle of a write frame, this field contains the data to be written to the register. For read or post-read increment address frames, this field contains the contents of the register read from the PHY. ■ In address and data write cycles, the DWC_ether_qos drives the MDIO line during the transfer of these 16 bits. ■ In read and post-read increment address cycles, the PHY drives the MDIO line during the transfer of these 16 bits.

The frame structure for Clause 22 frames is also supported. The C45E bit in the MAC_MDIO_Address register can be programmed to enable Clause 22 or Clause 45 mode of operation. Table 5-2 shows the Clause 22 frame format.

Table 5-2 MDIO Clause 22 Frame Structure

Field	Description
IDLE	The mdio line is in tri-state; there is no clock on gmii_mdc_o.
PREAMBLE	32 continuous bits of value 1
START	Start of packet is 2’b01
OPCODE	■ 2’b01 for Write ■ 2’b10 for Read
PHY ADDR	5-bit address select for one of 32 PHYs
DEV ADDR	5-bit address to select the register within each MMD
TA	Turnaround ■ 2’bZ0: Read and post-read increment address ■ 2’b10: Write and address MDIO accesses Where Z is the tri-state level
DATA/ADDRESS	Any 16-bit value: ■ In a write operation, the DWC_ether_qos drives MDIO. ■ In read operation, the PHY drives MDIO.

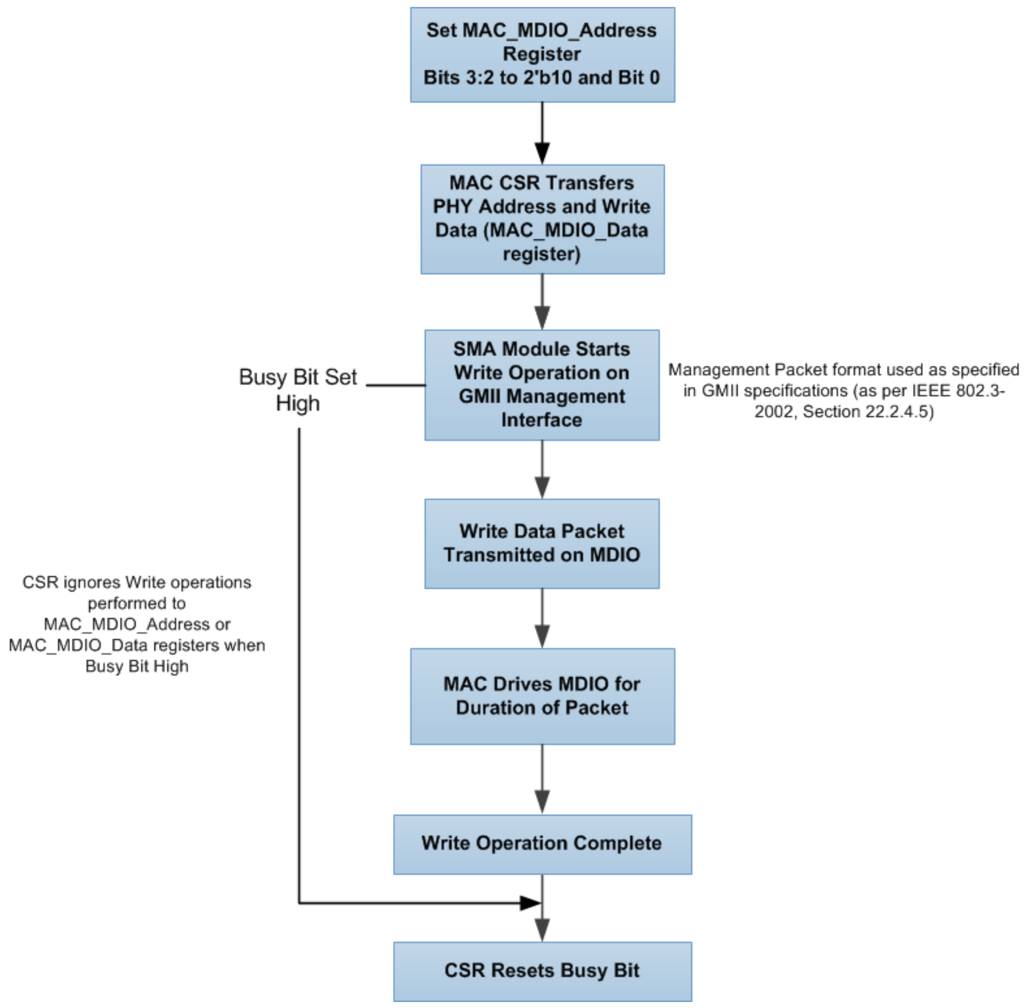
In addition to normal read and write operations, the SMA also supports post-read increment address while operating in Clause 45 mode.

5.1.2.1 GMII/MII Management Write Operation

After the Station Management Agent receives the PHY address and the write data from the MAC CSR module, the SMA starts a Write operation to the PHY registers.

Figure 5-1 illustrates the flow for a write operation from the SMA module to the PHY registers.

Figure 5-1 SMA Write Operation Flow


Set MAC_MDIO_Address Register Bits 3:2 to 2'b10 and Bit 0
MAC CSR Transfers PHY Address and Write Data (MAC_MDIO_Data register)
SMA Module Starts Write Operation on GMII Management Interface
Busy Bit Set High
Management Packet format used as specified in GMII specifications (as per IEEE 802.3-2002, Section 22.2.4.5)
Write Data Packet Transmitted on MDIO
CSR ignores Write operations performed to MAC_MDIO_Address or MAC_MDIO_Data registers when Busy Bit High
MAC Drives MDIO for Duration of Packet
Write Operation Complete
CSR Resets Busy Bit

When you set Bit[3:2] to 2’b01 and Bit 0 in the MAC_MDIO_Address register, the MAC CSR module transfers the PHY address, the register address in PHY, and the write data (MAC_MDIO_Data register) to the SMA to initiate a Write operation into the PHY registers. At this point, the SMA module starts a Write operation on the GMII Management Interface using the Management Packet Format specified in the GMII specifications (as per IEEE 802.3-2002, Section 22.2.4.5).

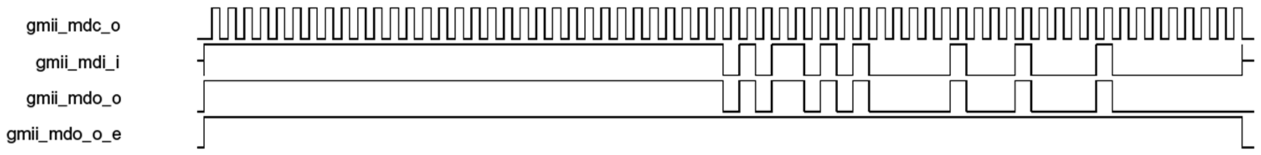
When the SMA module starts a Write operation, the write data packet is transmitted on the MDIO line. The MAC drives the MDIO line for complete duration of the packet. The Busy bit is set high until the write operation is complete. The CSR ignores the Write operations performed to the MAC_MDIO_Address register or the MAC_MDIO_Data register during this period (the Busy bit is high). When the Write operation is complete, the SMA module indicates this to the CSR, and the CSR resets the Busy bit.

The packet format for the Write operation is shown in the following table:

IDLE	PREAMBLE	START	OPCODE	PHY ADDR	REG ADDR	TA	DATA	IDLE
Z	1111...11	01	01	AAAAA	RRRRR	10	DDD....DDD	Z

Figure 5-2 is a reference for the Write operation.

Figure 5-2 Management Write Operation



5.1.2.2 GMII/MII Management Read Operation

After the Station Management Agent receives the PHY address and the register address in the PHY from the MAC CSR module, the SMA initiates a Read operation to the PHY registers.

The flow of a Management read operation is as follows:

1. Set Bit[3:2] to 2'b11 and Bit 0 in the MAC_MDIO_Address register
2. The MAC CSR module transfers the PHY address and the register address in PHY to the SMA to initiate a Read operation in the PHY registers
3. The SMA module starts a Read operation on the GMII Management Interface using the Management Packet Format specified in the GMII specifications (as per IEEE 802.3-2002, Section 22.2.4.5).
4. When the SMA module starts a Read operation on the MDIO, the CSR ignores the Write operations to the MAC_MDIO_Address or MAC_MDIO_Data register during this period (the Busy bit is high) and the transaction is completed without any error on the MCI interface.
5. When the Read operation is complete, the SMA indicates this to the CSR.
6. The CSR resets the Busy bit and updates the MAC_MDIO_Data register with the data read from the PHY.

For more information about the communication from the application to the PHYs, see the Reconciliation Sublayer and Media Independent Interface Specifications sections of the IEEE 802.3z, 1000BASE Ethernet.

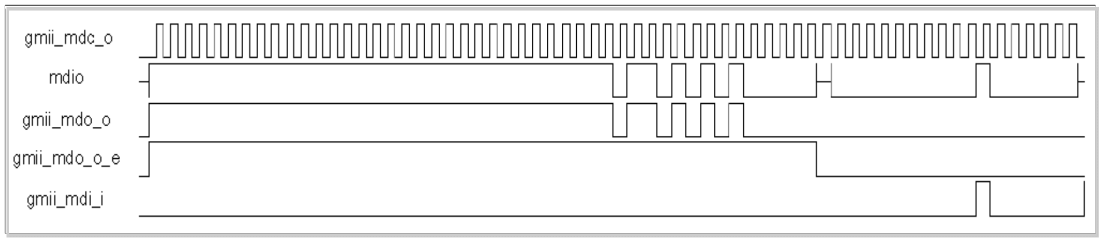
The packet format for the Read operation is as follows:

IDLE	PREAMBLE	START	OPCODE	PHY ADDR	REG ADDR	TA	DATA	IDLE
Z	1111...11	01	10	AAAAA	RRRRR	Z0	DDD....DDD	Z

Note The MAC drives the MDO on negedge of MDC so that it can be sampled on posedge with sufficient setup or hold time (half cycle of MDC). Similarly, the PHY can drive MDI on negedge so that the MAC samples it on posedge, thus providing half cycle setup or hold time.

Figure 5-3 is a reference for the read operation.

Figure 5-3 Management Read Operation



5.1.2.3 Preamble Suppression

The IEEE standard specifies 32-bit preamble (all-ones) for the MDIO frames. The DWC_ether_qos provides controls to support preamble suppression, it will transmit MDIO frames with only 1 preamble bit. The preamble suppression can be enabled by setting PSE bit of MAC_MDIO_Address register.

5.1.2.4 Trailing Clocks and Back to Back transactions

The DWC_ether_qos drives MDC clock on the gmii_mdc_o port for duration of the MDIO frame. There is no clock driven during the idle period. The trailing clocks feature can be used if the PHY needs the MDC clock to be active for some cycles after the MDIO frame. The NTC field in MAC_MDIO_Address register allows programming of trailing clocks from 0 to 7.

The DWC_ether_qos supports Back to Back transactions which allows start of next MDIO frame even before the trailing clocks are completed for previous MDIO frame. This feature can be enabled by setting BTB bit in MAC_MDIO_Address register when trailing clocks feature is also enabled. When Back to Back transactions is enabled, the GMII Busy will be cleared immediately after MDIO frame completion allowing SW to issue next command, which will be executed by DWC_ether_qos while trailing clocks are still on for previous MDIO frame. When Back to Back transactions is not enabled, the GMII Busy will be cleared after the trailing clocks are completed for MDIO frame.

5.1.3 Interrupt for MDIO Transaction Completion

This enhancement is to generate an interrupt on completion of MDIO read or write transactions. As a result of this enhancement, the application need not poll the GMII Busy bit of MAC_MDIO_Address register to know the completion of MDIO commands.

5.1.4 Enabling Station Management Agent

To enable this feature while configuring the controller in the coreConsultant GUI:

1. Select the **Enable Station Management (MDIO Interface)** option under the PHY Interface section during the **Specify Configuration** activity in coreConsultant.

   For details about this option, see PHY Interface Parameters in "Parameter Descriptions" on page 419.

Note See the "Enabled" field of the parameter descriptions to understand the dependencies.

5.2 Physical Coding Sub-Layer (PCS)

The DWC_ether_qos provides an IEEE 802.3z compliant 10-bit Physical Coding Sublayer (PCS) interface that you can use when the MAC is configured for the TBI, RTBI, or SGMII PHY interface. The PCS interface transmits 10-bit code groups with a 125-MHz Transmit clock. In addition, the interface can receive 10-bit code groups with two out-of-phase 62.5-MHz clocks in TBI interface mode and a single 125 MHz clock in the SGMII or RTBI interface mode.

In PCS interface, the auto-negotiation process is implemented as described in IEEE 802.3z, Figure 37-6. The auto-negotiation provides an automatic process to share configuration information between the DWC_ether_qos core (MAC) and the link partner. The MAC and the link partner use this information to operate at the best of their common abilities. You can also use manual configuration by using the MAC_AN_Control and MAC_AN_Advertisement register and disabling the auto-negotiation in the MAC_AN_Control register.

5.2.1 PCS Functions

The PCS interface performs the following tasks:

- Encapsulation in the Transmit path
- Decapsulation in the Receive path
- Synchronization
- Auto-negotiation
- Start-of-packet (SOP) and end-of-packet (EOP) detection and generation
- Collision and Carrier sense generation

5.2.1.1 Transmit

During the Transmit process, the MAC performs the encapsulation function according to the following rules:

- The first byte of the preamble is replaced with the /S/ code group
- All data in the packet are encoded according to 8B/10B encoding standard
- The /T/R/R/ or /T/R/ code group is inserted after the last byte of FCS
- An idle code group /I/ is transmitted between packets
- Collision and Carrier sense are generated
- /V/ Error insertion is also supported

5.2.1.2 Receive

During the Receive process, the MAC performs the de-encapsulation function according to the following rules:

- An /I/S/ code group sequence results in assertion of the internal data valid signal
- The /S/ code group is replaced by a preamble byte
- The data of the received packet is decoded according to 10B/8B decoding standard
- The /T/R/R/ or /T/R/ code group sequence results in de-assertion of the internal data valid signal

5.2.1.3 Synchronization

The synchronization process determines whether the underlying receive channel is ready for operation. The MAC implements synchronization of the received code groups according to the IEEE 802.3z. The MAC acquires the synchronization when the PHY achieves bit synchronization and the MAC detects three consecutive valid /COMMA/D/ patterns sent by the PHY in the even code group position. After the MAC acquires the synchronization, auto-negotiation can start.

If the MAC receives four consecutive invalid code groups or commas in odd clock after acquiring synchronization, the synchronization process restarts automatically. A built-in hysteresis prevents any restarting of the synchronization process happening because of invalid code groups reception.

5.2.1.4 Auto-Negotiation

The DWC_ether_qos supports a subset of the management registers (MAC_AN_Control–MAC_AN_Expansion). These registers can be accessed through MCI. The MAC supports only base page exchange and does not support any next page exchanges. Auto-negotiation (AN) is enabled by setting the ANE bit of MAC_AN_Control register.When auto-negotiation is enabled, any one the following events triggers the AN process:

- Request from another device (transmitting configuration code groups)
- Loss of synchronization for more than 10 ms
- Error condition detected while receiving /C/ or /I/ ordered set
- Auto-negotiation restart bit is enabled

When AN is initiated, the MAC initially sends configuration words with all zeros for a duration determined by the value in the link timer (10 ms). After the link timer duration expires, the contents of the AN Advertisement register are transmitted within the configuration words. The AN Advertisement register should be loaded before AN is enabled.

The AN process is completed after the base page exchange. When the AN is complete, the ANC bit in MAC_AN_Status register is set and the following registers are valid:

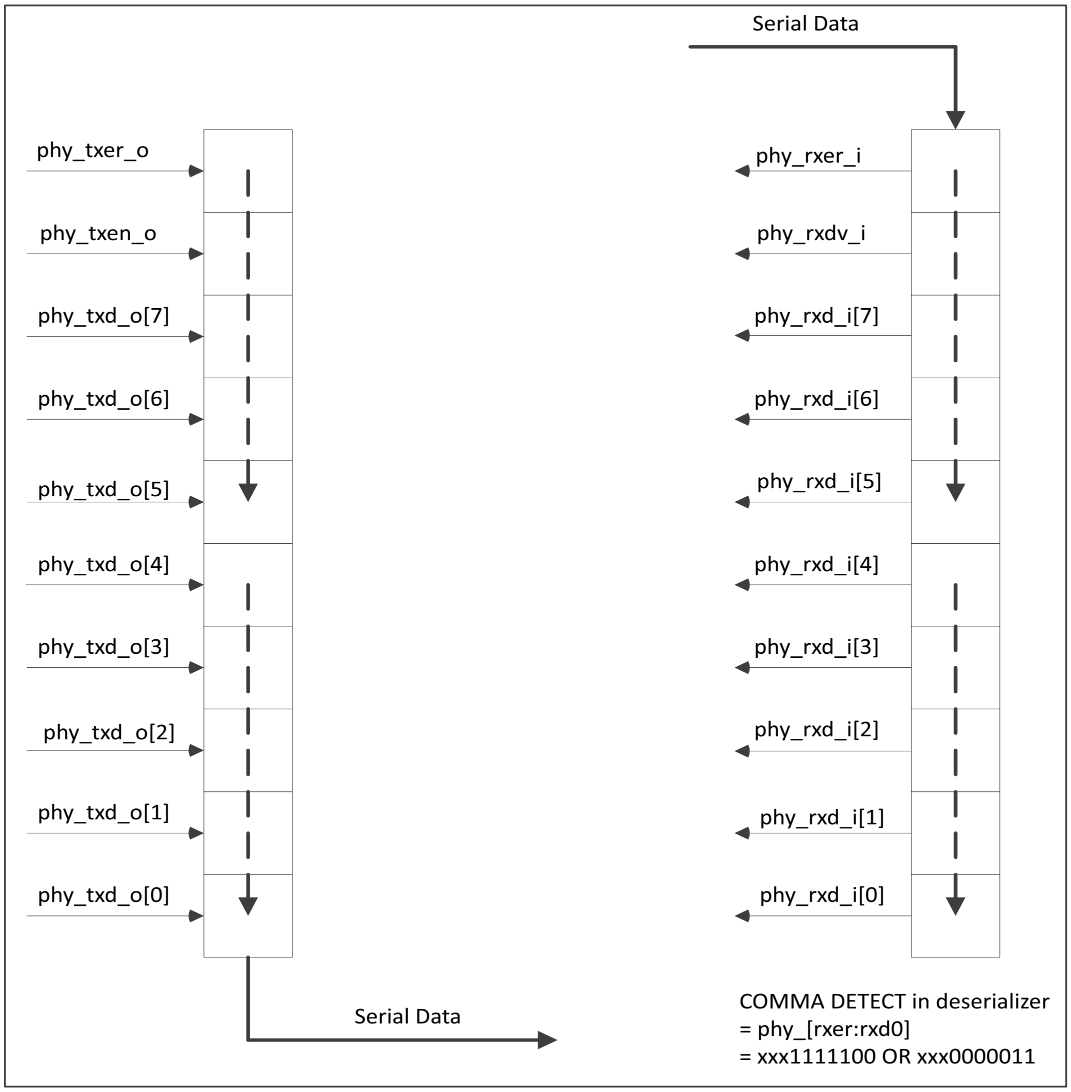
- MAC_AN_Expansion: This register indicates the new base page received from the link partner
- MAC_TBI_Extended_Status: This register indicates the modes of operation of the MAC, 1000-Mbps full-duplex and 1000-Mbps half-duplex

  The MAC is programmed automatically for the speed, duplex mode, and flow control mode. Highest common denominator (Maximum Common Mode) is selected. When the AN is completed, the MAC can transmit packets.

5.2.1.5 PMA SerDes

The 10-bit TBI signals from the PCS block are connected to the Physical Medium Attachment (PMA) layer of the PHY chip. The connection of the 10-bit transmit data signals with the serializer and the 10-bit receiver data signals from the de-serializer of the PMA are shown in Figure 5-4.

Figure 5-4 PMA Serializer and De-Serializer



5.2.2 Comma Detection

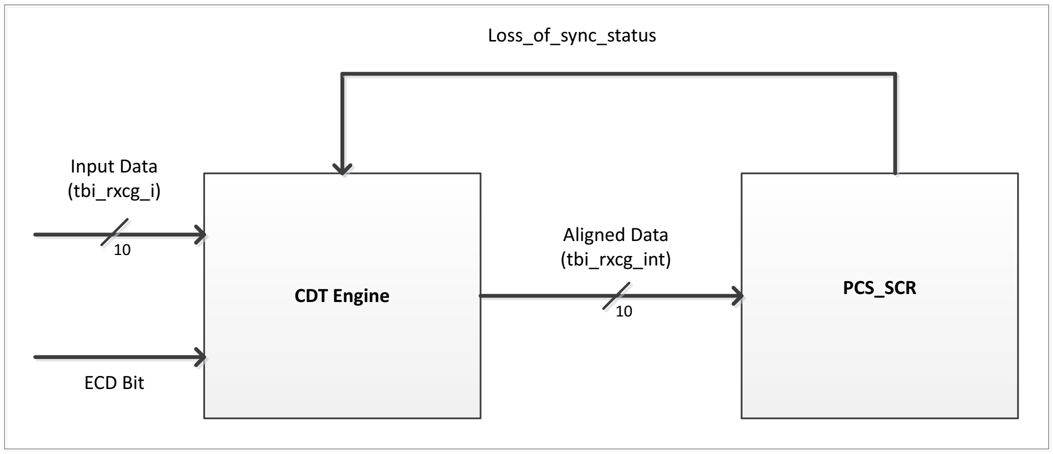
The physical coding sublayer (PCS) in DWC_ether_qos supports code group alignment based on the comma.

To enable comma detection and word resynchronization in DWC_ether_qos PCS, set the ECD bit of MAC_AN_Control register to 0.

For more details, see the "Register Descriptions" on page 605

Based on programmed ECD bit and the current state of synchronization, input data is aligned at the appropriate boundary. When the ECD bit is set, the input data is redirected to PCS Synchronizer (PCS_SCR) block without code group alignment.

Figure 5-5 Block Diagram of Comma Detect Mechanism



5.3 Reduced Gigabit Media Independent Interface

5.3.1 Introduction to Reduced Gigabit Media Independent Interface (RGMII)

The Reduced Gigabit Media Independent Interface (RGMII) module is implemented in DWC_ether_qos to reduce the pin count as mentioned in the RGMII specification.

The Reduced Gigabit Media Independent Interface (RGMII) specification reduces the pin count of the interconnection between the MAC and the PHY for GMII and MII interfaces. To achieve this, the data path and control signals are reduced and multiplexed together with both edges of the Transmit and Receive clocks. For Gigabit operation, the clocks operate at 125 MHz; for 10/100 operation, the clock rates are 2.5 MHz/25 MHz.

The RGMII module is instantiated between the GMII of the MAC and the PHY to translate the control and data signals between the GMII and RGMII protocols.

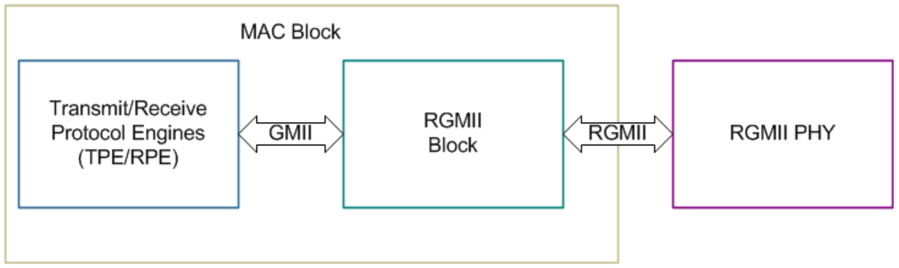
The RGMII block has the following characteristics:

- Supports 10 Mbps, 100 Mbps, and 1000 Mbps operation rates
- Requires no extra clock because both edges of the incoming clocks are used. To simplify back-end implementation of the MAC, there are separate clock inputs (180 degrees out-of-phase with the associated transmit/receive clocks) for logic that uses the falling edges. For guidelines for clock connections when the MAC operates with an RGMII, see *DesignWare Cores Ethernet Quality-of-Service User Guide*.
- Extracts the in-band (link speed, link status, and duplex mode) status signals from the PHY and provides them to the MAC for link detection

5.3.2 Description of Reduced Gigabit Media Independent Interface (RGMII)

Figure 5-6 shows the position of the RGMII block relative to the MAC and RGMII PHY. The RGMII block is placed between the GMII and the PHY to translate the GMII signals to RGMII signals.

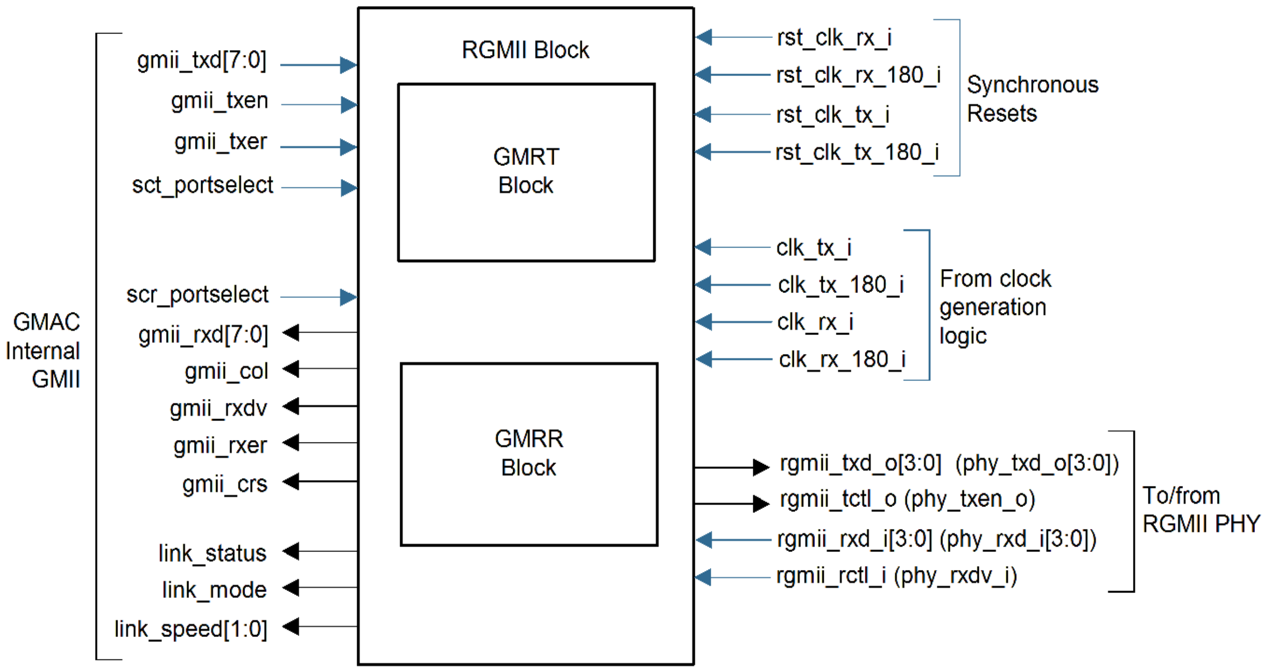
Figure 5-6 RGMII Block Diagram



The following list describes the RGMII components shown in Figure 5-7:

- **GMII-RGMII Transmit (GMRT) Block**: This block translates all GMII transmit signals to RGMII signals. The GMRT registers all GMII input signals at the rising edge of clk_tx_i and generates all RGMII transmit signals at the rising edge of either clk_tx_i or clk_tx_180_i.
- **GMII-RGMII Receive (GMRR) Block**: This block translates all RGMII receive signals to GMII receive signals. The GMRR registers all RGMII input signals at the rising edge of either clk_rx_i or clk_rx-_180_i and generates all GMII receive signals at the rising edge of clk_rx_i.

Figure 5-7 RGMII Block Pinout Diagram



5.3.3 RGMII Clocks

The RGMII block uses the transmit and receive clocks, and two additional clocks (clk_tx_180_i and clk_rx-_180_i) from the clock generation logic to reduce the pin count of the interconnection between the MAC and the PHY for GMII and MII interfaces.

The RGMII operation uses both edges of the RGMII transmit and receive clocks (clk_tx_i and clk_rx_i), which run at 125 MHz for gigabit operation or 2.5 MHz/25 MHZ for 10/100 operation. These clocks come from a separate PLL logic block. To simplify implementation of the MAC, the RGMII block provides two additional inputs to drive the logic that uses the falling edges of clk_tx_i and clk_rx_i:

- clk_tx_180_i must be 180 degrees out-of-phase with respect to clk_tx_i
- clk_rx_180_i must be 180 degrees out-of-phase with respect to clk_rx_i

Separate synchronous reset inputs are present to synchronously reset the logic driven by each of the four RGMII clocks.

5.3.4 Signal Conversion

The RGMII block performs the following conversions:

- "Transmit Data Conversion" on page 179
- "Transmit Control Signal Conversion" on page 179
- "Receive Data Conversion" on page 180
- "Receive Control Signal Conversion" on page 180
- "Receive Status Signal Conversion" on page 181

5.3.4.1 Transmit Data Conversion

As defined in the RGMII specification, multiplexing of data and control in Gigabit mode is accomplished by using both edges of the Transmit clock. The RGMII block accepts the 8-bit GMII transmit data from the MAC GMII (gmii_txd[7:0]) on the rising edge of clk_tx_i. For Gigabit operation, the RGMII block then converts gmii_txd[7:0] to two 4-bit nibbles and transmits the data on the RGMII transmit data port (rgmii_txd_o[3:0]) as follows:

- At the rising edge of clk_tx_i, rgmii_txd_o[3:0] contains gmii_txd[3:0]
- At the falling edge of clk_tx_i (rising edge of clk_tx_180_i), rgmii_txd_o[3:0] contains gmii_txd[7:4]

For 10/100 mode, the Transmit data to the PHY is 4 bits. Therefore, for 10/100 mode, the RGMII block drives the gmii_txd[3:0] data to the PHY on rgmii_txd_o[3:0] at the rising edge of clk_tx_i.

5.3.4.2 Transmit Control Signal Conversion

The RGMII block accepts the GMII transmit control signals (gmii_txen and gmii_txer) from the MAC GMII on the rising edge of clk_tx_i. The RGMII block then multiplexes gmii_txen and gmii_txer onto the rgmii_tctl_o output (as required by the RGMII specification) as follows:

- At the rising edge of clk_tx_i, rgmii_tctl_o is gmii_txen
- At the falling edge of clk_tx_i (rising edge of clk_tx_180_i), rgmii_tctl_o is the logical XOR of gmii_txen and gmii_txer.

Table 5-3 shows the mapping of the PLS_DATA.request parameters from the MAC to the corresponding signaling on the GMII and RGMII control and data signals.

Table 5-3 Encoding of gmii_txen, gmii_txer, rgmii_tctl_o, and rgmii_txd_o Signals

gmii_txen	gmii_txer	gmii_txd[a]	rgmii_tctl_o[b]	Description	PLS_DATA.request Parameter
0	0	0x00–0xFF	0,0	Normal inter-packet	TRANSMIT_COMPLETE
0	1	0x00	0,1	Reserved	ZERO, ONE (8 bits)
0	1	0x01	0,1	LPI pattern	TRANSMIT_LPI in EEE mode
0	1	0x02–0x0E0,1	0,1	Reserved	
0	1	0x0F	0,1	Carrier extend	EXTEND (8 bits)
0	1	0x10–0x1E	0,1	Reserved	

gmii_txen	gmii_txer	gmii_txd[a]	rgmii_tctl_o[b]	Description	PLS_DATA.request Parameter
0	1	0x1F	0,1	Carrier extend error	EXTEND_ERROR (8 bits)
0	1	0x20–0xFF	0,1	Reserved	
1	0	0x00–0xFF	1,1	Normal data transmission	ZERO, ONE (8 bits)
1	1	0x00–0xFF	1,0	Transmit error propagation	No applicable parameter

a. If the RGMII interface is configured to transmit the configuration during the IPG, rgmii_txd[3:0] reflects the Duplex Mode, Port Select, Speed (encoded as 00 for 10 Mbps, 01 for 100 Mbps, and 10 for 1000 Mbps), and Link Up/Down bits of the MAC_Configuration register during this period.
b. Values at rising, falling edges of clk_tx_i

5.3.4.3 Receive Data Conversion

For 1000 Mbps mode, the RGMII block registers the 4-bit data on the rgmii_rxd_i[3:0] signal at both edges of the Receive clock (clk_rx_i) and transfers the resulting 8-bit data to the DWC_ether_qos GMII on the rising edge of clk_rx_i clock, as follows:

- gmii_rxd[3:0] is the value received on rgmii_rxd_i[3:0] at the rising edge of clk_rx_i
- gmii_rxd[7:4] is the value received on rgmii_rxd_i[3:0] at the falling edge of clk_rx_i (rising edge of clk_rx_180_i)

For 10/100 mode, the RGMII block registers the 4-bit data on rgmii_rxd_i[3:0] only at the rising edge of clk_rx_i and transfers the data to the DWC_ether_qos GMII on the rising edge of clk_rx_i, as follows:

- gmii_rxd[3:0] is the value received on rgmii_rxd_i[3:0] at the rising edge of clk_rx_i
- gmii_rxd[7:4] is 0x0

5.3.4.4 Receive Control Signal Conversion

The RGMII block samples the rgmii_rctl_i signal from the PHY at both edges of the clk_rx_i clock and drives the resulting GMII receive control signals to the DWC_ether_qos GMII as follows:

- gmii_rxdv is the value received on rgmii_rctl_i at the rising edge of clk_rx_i.
- gmii_rxer is the logical XOR of the following two signals
  - The value received on rgmii_rctl_i at the rising edge of clk_rx_i (gmii_rxdv).
  - The value received on rgmii_rctl_i at the falling edge of clk_rx_i (rising edge of clk_rx_180_i).

5.3.4.5 Receive Status Signal Conversion

To generate the GMII link status signals, the RGMII block decodes the rgmii_rxd_i[3:0] value sampled at the rising edge of clk_rx_i when the value of rgmii_rctl_i is 0 for both the rising and falling edges of clk_rx_i (normal inter-packet value). The following are the decoded status signals:

- link_status is the value of rgmii_rxd_i[0]
- link_speed is the value of rgmii_rxd_i[2:1]
- link_mode is the value of rgmii_rxd_i[3]

Table 5-4 shows the mapping of the PLS_DATA.indicate parameters from the DWC_ether_qos to the corresponding signaling on the GMII and RGMII control and data signals. In addition, the RGMII block asserts gmii_crs to the DWC_ether_qos GMII when any of the following conditions is true:

- The gmii_rxdv is true
- The gmii_rxdv is false, gmii_rxer is true, and the value of gmii_rxd is 0xFF
- A carrier extend, carrier extend error, or false carrier occurs in Gigabit mode (see Table 5-4).
- A false carrier occurs in 10/100 mode.
- The RGMII is transmitting data, carrier extend, or carrier extend error on the RGMII transmit outputs.

The RGMII block asserts gmii_col to the MAC GMII when both of the following conditions are true:

- The RGMII is transmitting data, carrier extend, or carrier extend error on the RGMII transmit outputs
- The RGMII is asserting either gmii_crs or gmii_rxdv signal

Table 5-4 Decoding of rgmii_rctl_i and rgmii_rxd_i

rgmii_rctl_i	gmii_rxd[7:0]([3:0] in 10/100 mode)[a]	gmii_rxdv	gmii_rxer	Description	PLS_DATA.indicate or PHY_status Parameter
0,0	XXXXXXX0 or XXXXXXX1	0	0	Normal inter-packet	Indicates link status: ■ 0: down ■ 1: up
0,0	XXXXX00X or XXXXX01X or XXXXX10X or XXXXX11X	0	0	Normal inter-packet	Indicates Receive clock frequency: ■ 00: 2.5 MHz ■ 01: 25 MHz ■ 10: 125 MHz ■ 11: Reserved
0,0	XXXX1XXX or XXXX0XXX	0	0	Normal inter-packet	Indicates duplex status: ■ 0: half-duplex ■ 1: full-duplex
0,1	0x01 (or 0x1 in 10/100 mode)	0	1	Low Power Idle (LPI)	Remote MAC in LPI in the Energy Efficient Ethernet (EEE) mode
0,1	0x0E (or 0xE in 10/100 mode)	0	1	False carrier indication	False carrier present
0,1	0x0F	0	1	Carrier extend	EXTEND (8 bits)

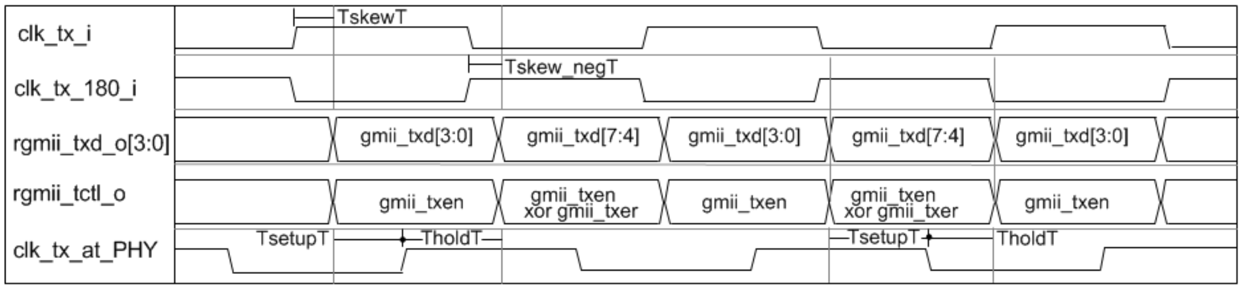
rgmii_rctl_i	gmii_rxd[7:0]([3:0] in 10/100 mode)[a]	gmii_rxdv	gmii_rxer	Description	PLS_DATA.indicate or PHY_status Parameter
0,1	0x1F	0	1	Carrier extend error	ZERO, ONE (8 bits)
0,1	0xFF (or 0xF in 10/100 mode)	0	1	Carrier sense	PLS_Carrier.Indicate
1,1	0x00–0xFF	1	0	Normal data reception	ZERO, ONE (8 bits)
1,0	0x00–0xFF	1	1	Data reception error	ZERO, ONE (8 bits)

a. Values at rising, falling edges of clk_rx_i

5.3.5 RGMII Transmit Timing

Figure 5-8 shows the timing for RGMII transmission. The RGMII specification defines a transmit skew (TskewT) of ± 0.5 ns. As shown in Figure 5-8, the trace delay on the PC board for clk_tx_i going to a receiving PHY must satisfy the setup and hold time requirements of the RGMII Transmit signals at the receiving PHY.

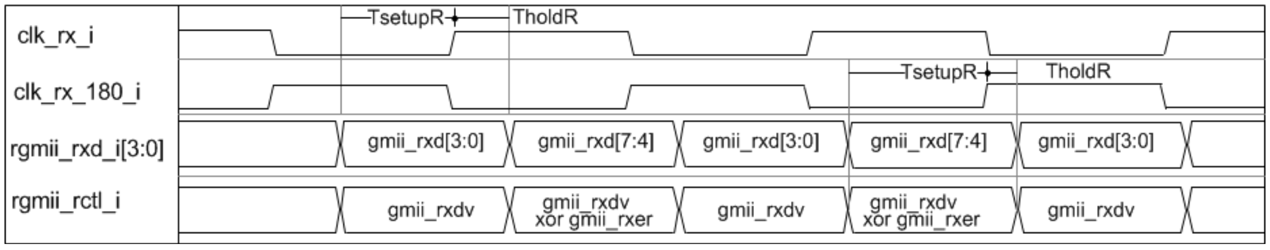
Figure 5-8 RGMII Transmit Timing



5.3.6 RGMII Receive Timing

Figure 5-9 shows the receive timing for RGMII. The TsetupR and TholdR values are required to be a minimum of 1 ns.

Figure 5-9 RGMII Receive Timing



5.3.7 Enabling Reduced Gigabit Media Independent Interface

To enable this interface while configuring the controller in the coreConsultant GUI:

1. Select the **Enable RGMII** option under the PHY Interface section during the **Specify Configuration** activity in coreConsultant.

   For details about this option, see PHY Interface Parameters in "Parameter Descriptions" on page 419.

Note See the "Enabled" field of the parameter descriptions to understand the dependencies.

5.4 Reduced Media Independent Interface

5.4.1 Introduction to Reduced Media Independent Interface

The Reduced Media Independent Interface (RMII) specification reduces the pin count between Ethernet PHYs and Switch ASICs (only in 10/100 mode). According to the IEEE 802.3u, an MII contains 16 pins for data and control. In devices incorporating multiple MAC or PHY interfaces (such as switches), the number of pins adds significant cost with increase in port count. The RMII specification addresses this problem by reducing the pin count to 7 for each port — a 62.5% decrease in pin count

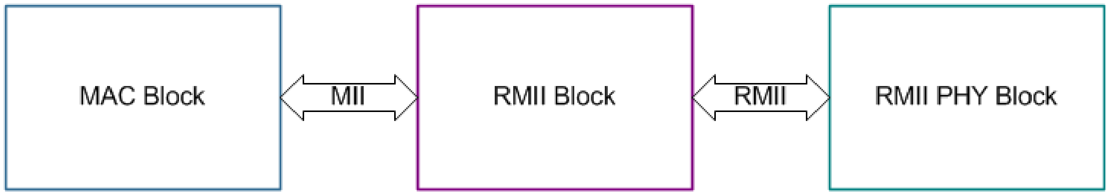
The RMII module is instantiated between the MAC and the PHY. This helps in translating the MII of the MAC into the RMII. The RMII block has the following characteristics:

- Supports 10 Mbps and 100 Mbps operating rates. It does not support the 1000 Mbps operation
- Provides independent 2-bits wide Transmit and Receive paths by sourcing two clock references externally.

5.4.2 Description of Reduced Media Independent Interface

Figure 5-10 shows the position of the RMII block relative to the DWC_ether_qos and RMII PHY. The RMII block is placed in front of the DWC_ether_qos to translate the MII signals to RMII signals.

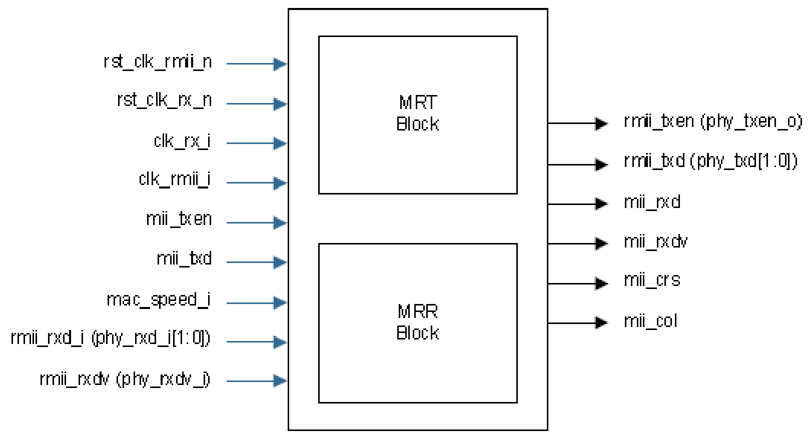
Figure 5-10 RMII Block Diagram



The following list describes the components of RMII shown in Figure 5-11:

- **MII-RMII Transmit (MRT) Block**: This block translates all MII transmit signals to RMII transmit signals. All RMII signals are synchronous to clk_rmii_i.
- **MII-RMII Receive (MRR) Block**: This block translates all RMII receive signals to MII receive signals. All MII signals are synchronous to clk_rx_i. You must ensure that the same clock is connected to both clk_tx_i and clk_rx_i clock input ports in RMII mode. This is required because clock-MUXing logic is avoided inside the MAC.

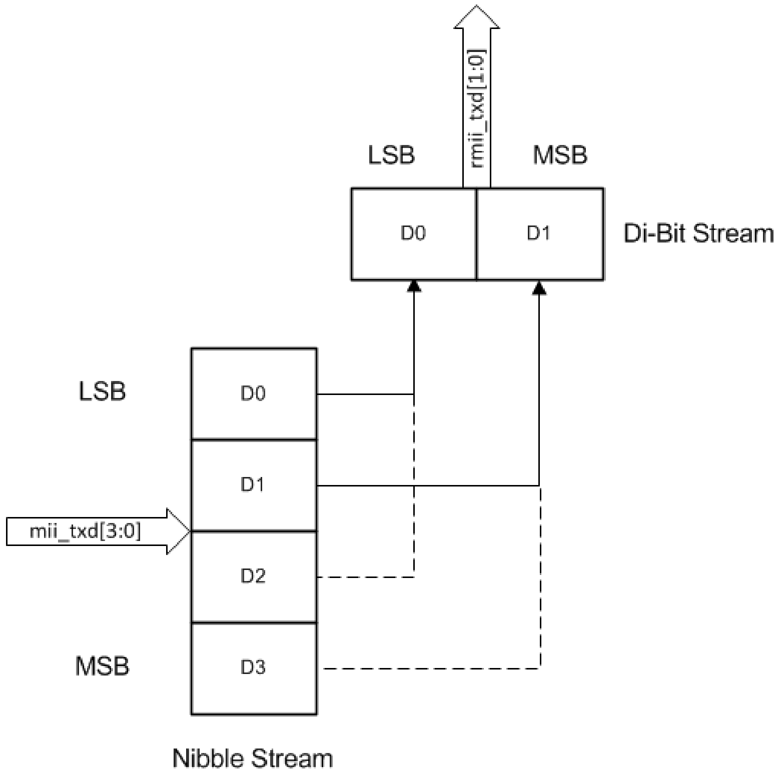
Figure 5-11 RMII Pinout



5.4.2.1 Transmit Bit Ordering

Each nibble from the MII interface must be transmitted on the RMII interface di-bit at a time with the order of di-bit transmission shown in Figure 5-12. The lower order bits (D1 and D0) are transmitted first followed by higher order bits (D2 and D3).

Figure 5-12 Transmit Bit Ordering



5.4.2.2 RMII Transmit Timing Diagrams

RMII transmit timing diagrams depict the various MII transmissions.

Figure 5-13 through Figure 5-16 show MII-to-RMII transaction timing. Figure 5-13 shows the start of MII transmission and the following RMII transmission in 100-Mbps mode.

Figure 5-13 Start of MII and RMII Transmission in 100-Mbps Mode

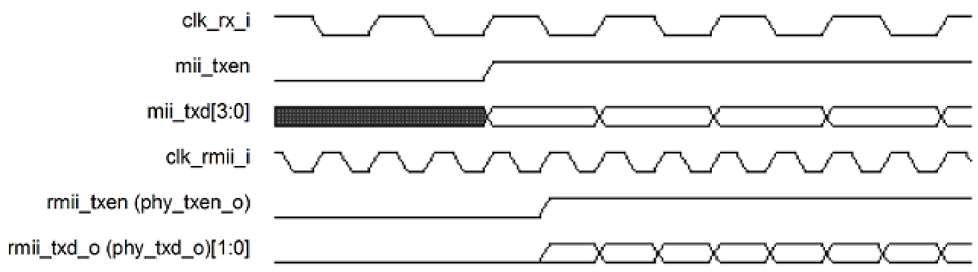


Figure 5-14 shows the end of packet transmission for MII and RMII in 100-Mbps mode.

Figure 5-14 End of MII and RMII Transmission in 100-Mbps Mode

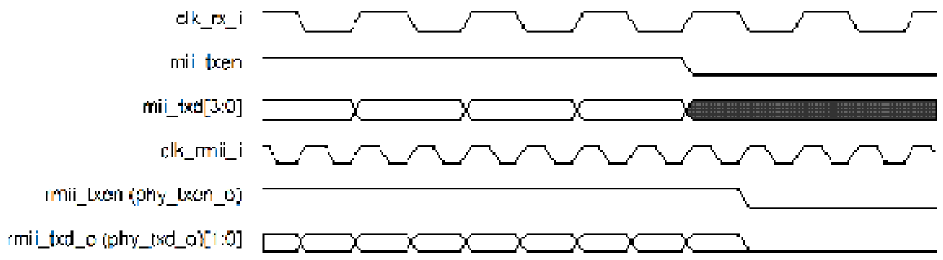


Figure 5-15 shows the start of MII transmission and the following RMII transmission in 10-Mbps mode.

Figure 5-15 Start of MII and RMII Transmission in 10-Mbps Mode

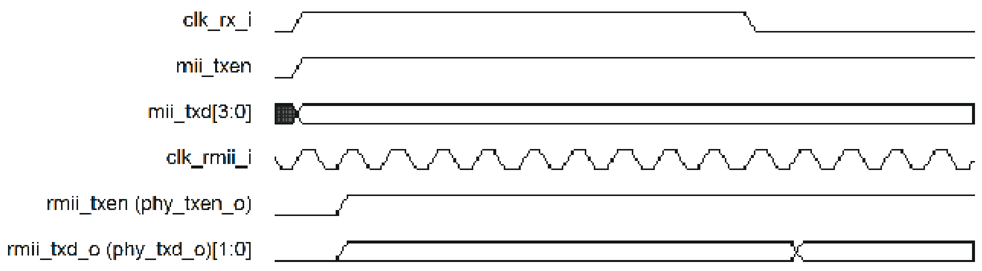
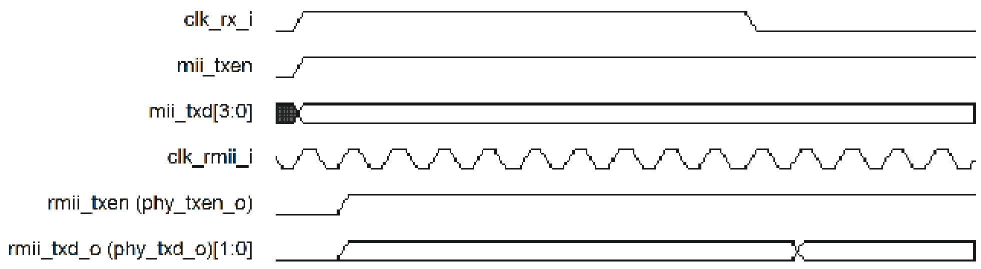


Figure 5-16 shows the end of MII transmission and RMII transmission in 10-Mbps mode.

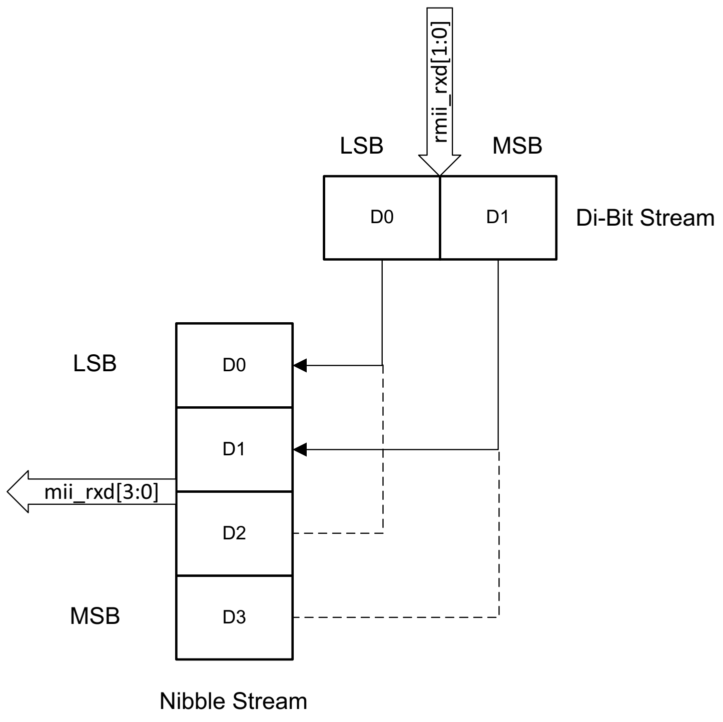
Figure 5-16 End of MII and RMII Transmission in 10-Mpbs Mode



5.4.2.3 Receive Bit Ordering

Each nibble is transmitted to the MII interface from the di-bit received from the RMII interface in the nibble transmission order shown in Figure 5-17. The lower order bits (D0 and D1) are received first, followed by the higher order bits (D2 and D3).

Figure 5-17 Receive Bit Ordering



5.4.3 Enabling Reduced Media Independent Interface

To enable this interface while configuring the controller in the coreConsultant GUI:

1. Select the **Enable RMII** option under the PHY Interface section during the **Specify Configuration** activity in coreConsultant.

   For details about this option, see PHY Interface Parameters “Parameter Descriptions” on page 419

Note See the “Enabled” field of the parameter descriptions to understand the dependencies.

5.5 Serial Media Independent Interface

5.5.1 Introduction to Serial Media Independent Interface

The SMII block is designed to fully comply with the SMII specification, revision 2.1, from Cisco. The SMII has the following characteristics:

- Conveys complete MII information between a 10/100 PHY and MAC with two pins (TX and RX) per port, one global 125 MHz System clock (CLOCK) and one global Synchronization signal (SYNC)

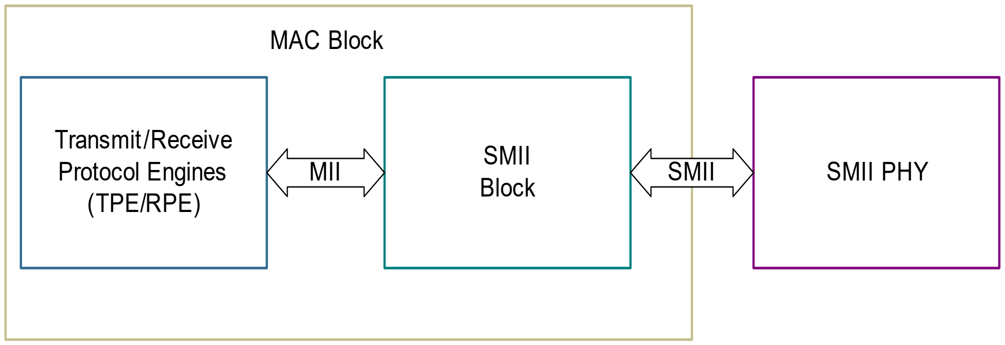
  For information about clock connections when the MAC operates with an SMII, see *DesignWare Cores Ethernet Quality-of-Service User Guide*
- Allows a multi-port MAC/PHY communication with one system clock
- Allows per packet switching between 10 Mbps and 100 Mbps data rates
- Allows direct MAC to MAC communication
- Operates in half-duplex and full-duplex modes
- Provides optional source synchronous mode in which four signals TX_CLK, TX_SYNC, RX_CLK, and RX_SYNC replace SYNC signal
- Supports optional selection of TX_SYNC as input in source synchronous mode to synchronize the transmit data
- Transmits status between packets (if enabled)

  This feature is useful in MAC to MAC connection

5.5.2 Description of Serial Media Independent Interface

Figure 5-18 shows the position of the SMII block relative to the MAC and SMII PHY. The SMII block is placed between the GMII and the PHY to translate the GMII signals to SMII signals. The pinout is shown in .

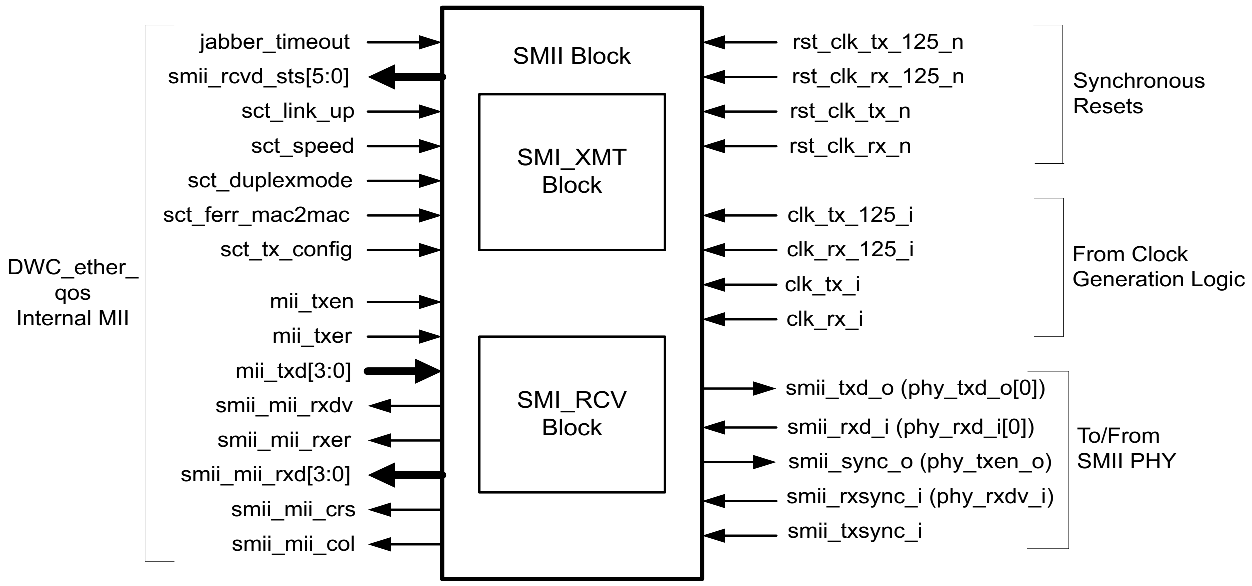
Figure 5-18 SMII Block Diagram



The following list describes the SMII components:

- **GMII-SMII Transmit (SMI_XMT)**: This block converts the two data nibbles and control signals received (at 2.5 MHz and 25 MHz for 10 Mbps and 100 Mbps, respectively) from MII transmit interface into one 10-bit segment transmitted serially at 125 MHz on the SMII transmit interface. The single 10-bit segment is repeated 10 times in 10 Mbps mode. This block generates the SYNC signal (TX_SYNC signal in source synchronous mode) once every 10 clocks of 125 MHz indicating the beginning of a 10-bit segment. In source synchronous mode with TX_SYNC as input, it synchronizes the transmit data to the received TX_SYNC.
- **GMII-SMII Receive (SMI_RCV)**: This block converts the 10-bit segment received serially at 125 MHz from SMII receive interface into two data nibbles and control signals (at 2.5 MHz and 25 MHz for 10 Mbps and 100 Mbps, respectively) on the MII receive interface. In 10 Mbps mode, although each 10-bit segment is received 10 times, only one sample is passed to the MII receive interface.

Figure 5-19 SMII Pinout



5.5.3 SMII Timing

This section describes the SMII timing in the following modes:

- "100 Mbps Mode" on page 189
- "10 Mbps Mode" on page 190

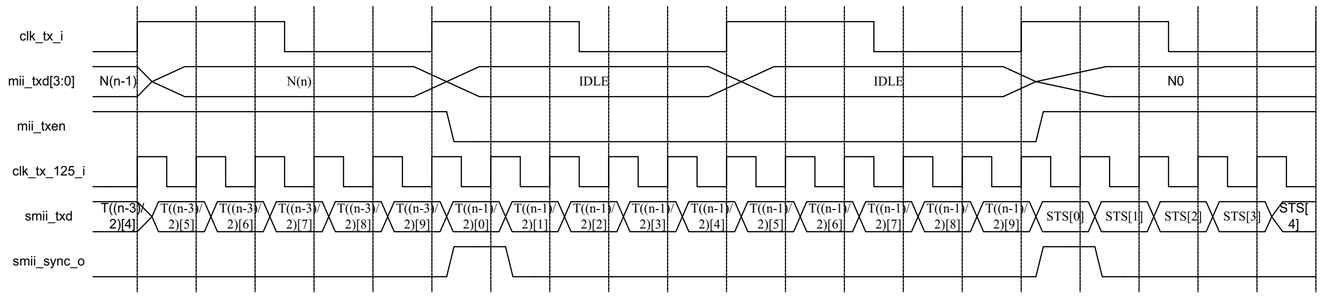
5.5.3.1 100 Mbps Mode

The SMI_XMT block converts the two data nibbles and control signals received at 25 MHz from 100 Mbps MII transmit interface into one 10-bit segment transmitted serially at 125 MHz clock on SMII transmit interface. The SMII transmit block sends 10-bit status segments in the inter packet gap as defined in the SMII specification. This status is sent only if enabled by setting Bit 0 of the MAC_PHYIF_Control_Status register. This feature is useful for sending status only in the MAC to MAC connection. The status includes the following:

- SFTERR and LUD bits from the MAC_PHYIF_Control_Status register
- FES and DM bits from the MAC_Configuration register
- Jabber timeout error signal from the TPE block

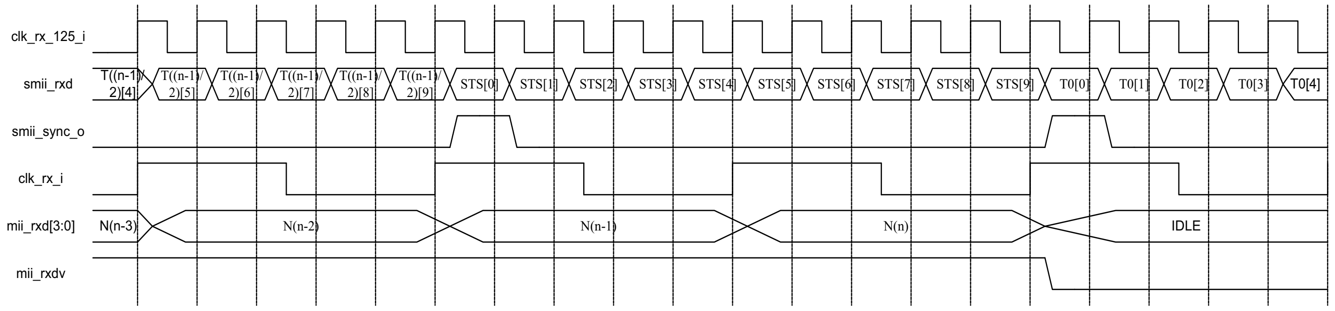
Note "T" and "N" in the timing diagrams indicate 10-bit segment and nibble, respectively. The relation between the T(m) and N(n) number is given by m = (n-1)/2 for a packet, that is, when there are 128 nibbles (minimum sized packet) to be transmitted, then N is from N(0) to N(127). Correspondingly, the T is from T(0) to T(63) in the waveform in Figure 5-20

Figure 5-20 SMII Transmit in 100 Mbps Mode



The SMI_RCV block converts the 10-bit segment received serially at 125 MHz on SMII receive interface into two data nibbles and control signals at 25 MHz in 100 Mbps mode on the MII receive interface. The data nibble given to MII receive interface is delayed by one cycle so that receive error (if indicated in the 10-bit status segment after the packet) can be signaled to the MAC with the last nibble.

Figure 5-21 SMII Receive Block in 100 Mbps Mode



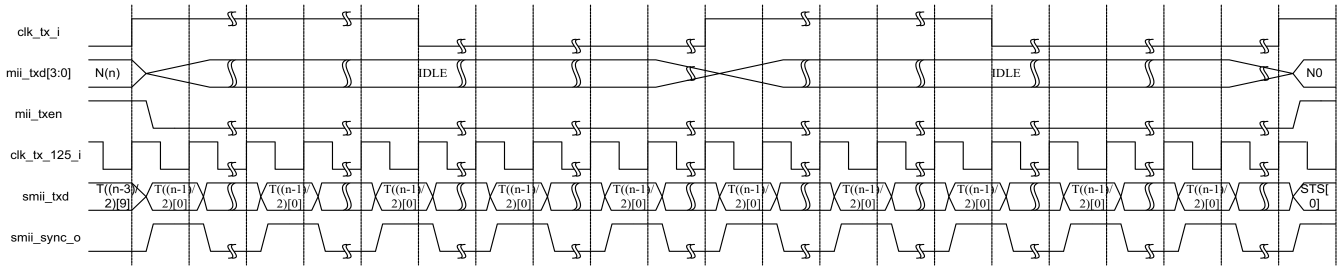
5.5.3.2 10 Mbps Mode

The SMI_XMT block converts the two data nibbles and control signals received at 2.5 MHz from 10 Mbps MII transmit interface into one 10-bit segment transmitted serially at 125 MHz clock on SMII transmit interface. The single 10-bit segment is repeated 10 times in 10 Mbps mode. The SMII transmit block sends 10-bit status segments in the inter packet gap as defined in the SMII specification. This status is sent only if enabled by setting Bit 0 of the MAC_PHYIF_Control_Status register. This feature is useful for sending status only in the MAC to MAC connection. The status includes the following:

- SFTERR and LUD bits from the MAC_PHYIF_Control_Status register
- FES and DM bits from the MAC_Configuration register
- Jabber timeout error signal from the TPE block

The SMI_XMT block generates the SYNC signal (TX_SYNC signal in source synchronous mode) once every 10 clocks of 125 MHz indicating the beginning of a 10-bit segment. In source synchronous mode with TX_SYNC as input, it synchronizes the transmit data to the received TX_SYNC.

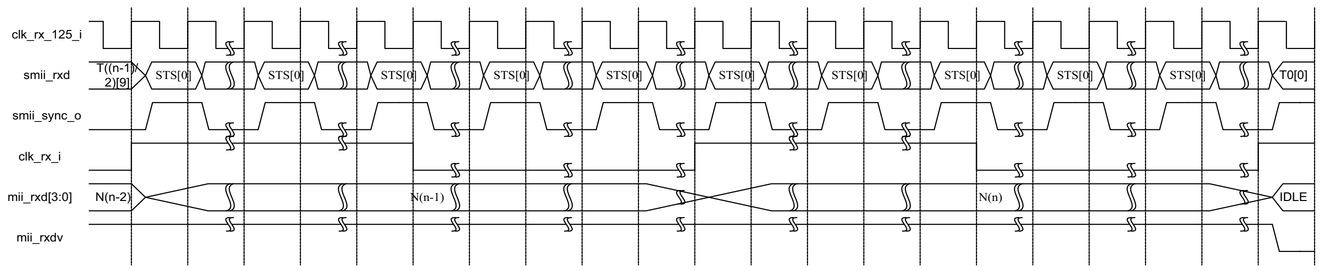
Figure 5-22 SMII Transmit Block in 10 Mbps Mode



The SMI_RCV block converts the 10-bit segment received serially at 125 MHz on SMII receive interface into two data nibbles and control signals at 2.5 MHz in 10 Mbps mode on the MII receive interface. In 10 Mbps mode, although each 10-bit segment is received ten times, only one sample is passed to the MII receive interface. The data nibble given to MII receive interface is delayed by one cycle so that receive error (if indicated in the 10-bit status segment after the packet) can be signaled to the MAC with the last nibble.

Note When Bit 4 of the MAC_PHYIF_Control_Status register is set, the MAC considers the timing of the received data delayed by one clock, with respect to the transmitted TX_SYNC, in non-source synchronous SMII configuration. The STS[0] and T0[0] bits in Figures 5-22 and 5-21 are taken as the data bit (smii_rxd) sampled one clock after the smii_sync_o signal.

Figure 5-23 SMII Receive Block in 10 Mbps Mode



When the TXSYNC signal is selected as input in Source Synchronous Mode, the smi_xmt block uses it to synchronize the 10-bit segments being transmitted.

5.5.4 Enabling Serial Media Independent Interface

To enable this interface during the Specify Configuration activity in the coreConsultant GUI:

1. Select the **Enable SMII option** under the PHY Interface section.
2. Select **Enable SMII Source Synchronous Mode**, if required.
3. Select **Enable TXSYNC as Input in SMII Source Synchronous Mode**, if required.

   For details about these options, see PHY Interface Parameters "Parameter Descriptions" on page 419.

Note See the "Enabled" field of the parameter descriptions to understand the dependencies.

5.6 Reverse Media Independent Interface

5.6.1 Introduction to Reverse Media Independent Interface

The Reverse Media Independent Interface (RevMII) connects two Ethernet MACs with a point-to-point link. The RevMII link provides a simple and low-cost alternative to using the Ethernet PHY in the system.The RevMII has the following features:

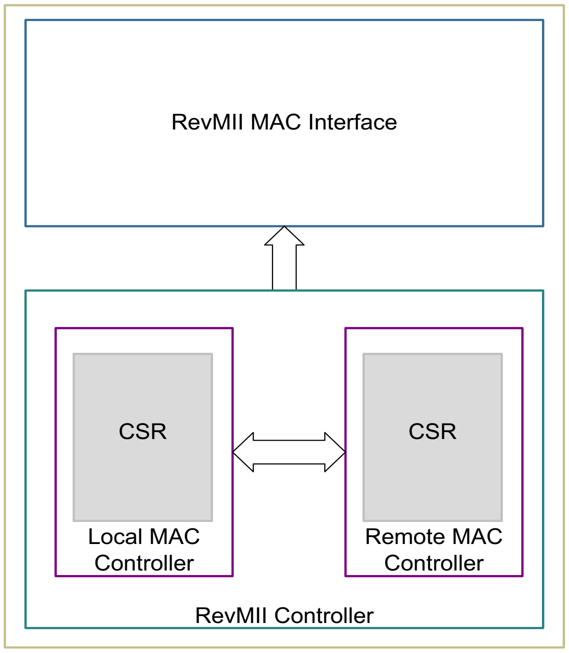
- Connects two Ethernet MACs directly as defined in the IEEE 802.3u standard
- Supports configuration through Serial Management Interface (SMI)
- Supports basic register set. This includes the Control and Status registers
- Supports extended register set
- Supports the following MII test modes:
  - Loopback
  - Isolate
  - Collision test
  - Power down
- Supports the half-duplex or full-duplex mode
- Supports 10 Mbps, 100 Mbps, and 1000 Mbps modes
- Supports carrier sense and collision detection

The RevMII does not support auto-negotiation and jabber detection.

5.6.2 Description of Reverse Media Independent Interface

Figure 5-24 shows the block diagram of RevMII.

Figure 5-24 RevMII Block Diagram



The RevMII block consists of the following sub-blocks:

- "RevMII MAC Interface" on page 193
- "RevMII Controller" on page 194

5.6.2.1 RevMII MAC Interface

The RevMII MAC interface block connects two Ethernet MACs (local MAC and Remote MAC). The RevMII MAC interface uses the following GMII/MII signals to connect two MAC:

- Transmit/Receive Data (TXD/RXD)
- Transmit/Receive Enable (TXEN/RXDV)
- Transmit/Receive Error (TXER/RXER)
- Carrier Sense (CRS)
- Collision Detection (COL)

The Transmit and Receive data paths (TXD/RXD) are 8-bit wide. If the MAC and remote MAC are working in the 10/100 Mbps mode, then the lower 4 bits of the data path contain valid data. The upper four bits are tied to zero. Similarly, if MAC and remote MAC are working in the full-duplex mode, then the CRS and COL signals are set to zero unless you have selected the collision test mode.

Note

- The Tx Bus (TXD, TXEN, and TXER) from the Remote MAC and the Rx Bus (RXD, RXDV, and RXER) to the Remote MAC are registered at the IOs to meet the timing constraints.
- The link between the MAC and remote MAC is up only when both operate at the same speed and in same mode. If speed or mode is not same, the link goes down. In addition, the link goes down if MAC or remote MAC is operating in any of test modes specified in RevMII MAC Interface Modes.

5.6.2.1.1 RevMII MAC Interface Modes

The MAC interface supports the following data transfer modes:

- Normal Mode

  In normal mode, the data MUXes are switched so that the Tx bus from the MAC goes to the Rx bus input of the Remote MAC and the Tx bus from the Remote MAC goes to the Rx bus input of the MAC. This is the default mode of operation. This mode is valid when no other mode is selected. In the half-duplex mode, when a MAC transmits the data to a remote MAC, the CRS signal becomes high on both sides. If MAC and remote MAC are in the half-duplex mode and both transmit at the same time, then there will be a collision. The COL signal is set to high to indicate a collision. When COL signal is high, the CRS signal also remains high.

- Power-Down Mode

  In power-down mode, the link between the MAC and remote MAC is shut down by switching the data multiplexers to port 2 so that the Rx buses of both MACs read all zeros. The RevMII controller remains active during this mode. The link status is set to 0 (Bit 2 of the Status register in RevMII Controller block) during the power down mode. In addition, when you enable the power down mode, the CRS and COL signals are set to low. You can select the power down mode by setting Bit 11 to 1 in any RevMII Control register.

- Isolate Mode

  In RevMII, the isolate mode is similar to the power down mode in functionality. You can select the isolate mode by setting Bit 10 to 1 in either RevMII Control register.
- Loopback Mode

  In loopback mode, the link between the MAC and remote MAC is shut down. The MAC going to the loopback mode switches the Tx bus to its Rx bus and the other MAC stops receiving any data. The RevMII controller sets the link status to 0 (Bit 2 of the Status register in RevMII Controller block) during this mode. When the TX_EN signal of a MAC is high, the CRS signal to that MAC is asserted high. The COL signal remains low throughout the process unless you have selected the collision test mode.

  You can select the loopback mode by setting Bit 14 to 1 in any RevMII Control registers.
- Collision Test Mode

  In collision test mode, the COL signal is asserted in response to the assertion of corresponding TX_EN signal. The COL signal is set to low after the TX_EN signal goes low. The link status is set to 0 (Bit 2 of Status register in RevMII Controller block) to indicate that the link is down.

  You can select the collision test mode by setting the Bit 7 to 1 in any RevMII Control register.

5.6.2.2 RevMII Controller

The RevMII controller block handles the station management functionality and controls the RevMII MAC interface. The RevMII Controller block consists of the registers to control and monitor the RevMII bus. In addition, it provides the serial interface (SMI) to the remote MAC and a parallel interface to the MAC to access these registers. The RevMII controller block consists of the following components:

- "Local MAC Controller" on page 194
- "Remote MAC Controller" on page 195

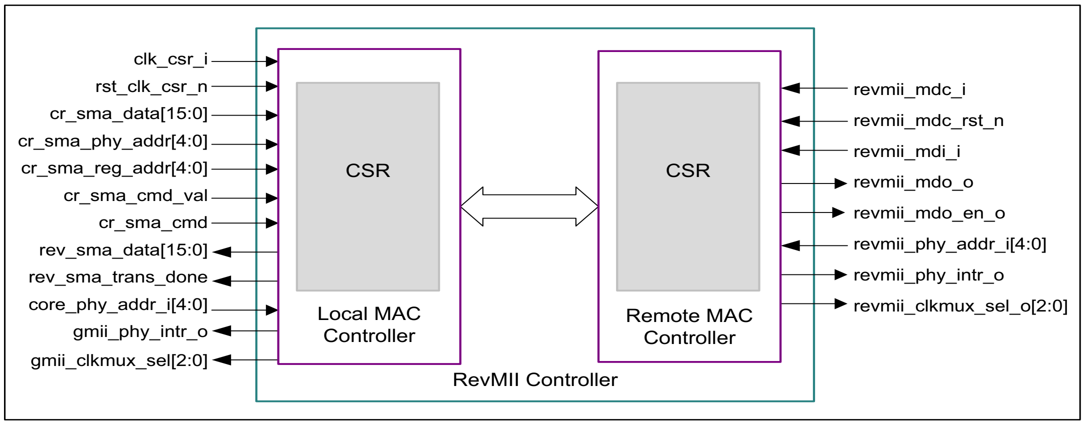
5.6.2.2.1 Local MAC Controller

The RevMII controller provides a parallel control interface with the MAC. The local MAC controller controls the data flow between the MAC and RevMII MAC interface. It performs the following tasks:

- Verifies the PHY address and Register address
- Writes the data into the register
- Returns the register read-back data along with transaction completion signal

Figure 5-25 shows the input and output signals of the local and remote MAC controller.

Figure 5-25 RevMII Controller



The CSR module inside the local MAC controller contains the registers described in "RevMII Registers" on page 231. The parallel data transfer occurs in the CSR clock domain. A valid transaction starts when the CSR module of the MAC asserts the cr_smacmdvld signal. When the transaction is complete, the rev_sma_transdone_n signal goes low. In response, the cr_smacmdvld signal is reset to low, and the sma_transdone_n goes high, completing the handshaking process.

5.6.2.2.2 Remote MAC Controller

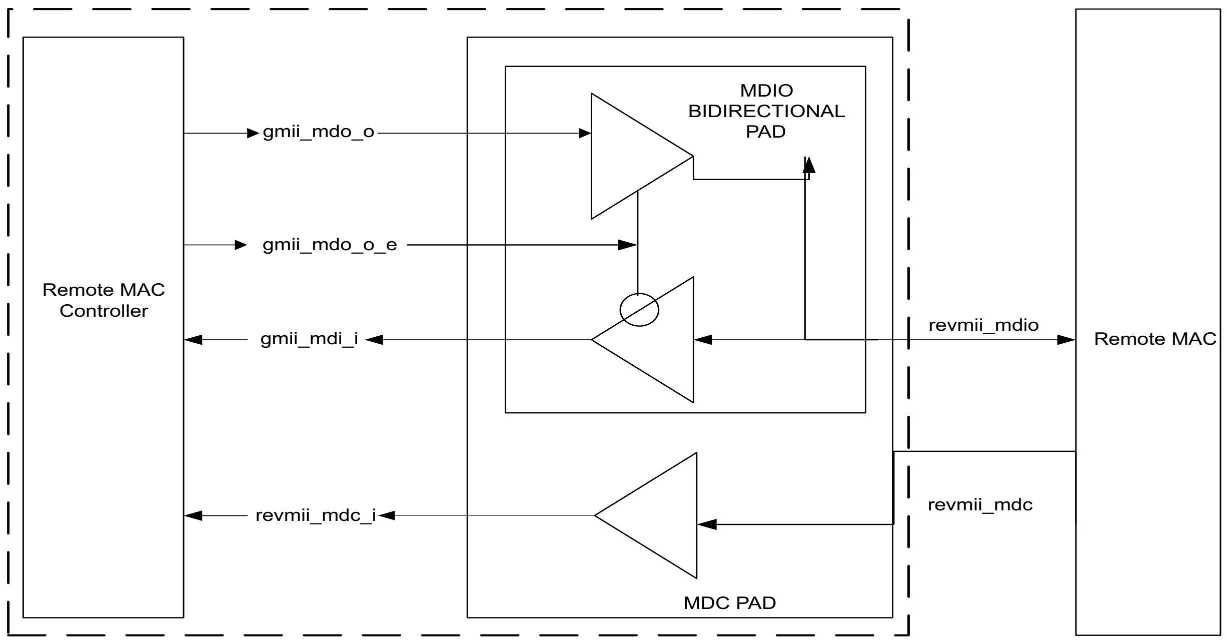
The RevMII controller provides a conventional serial interface (SMA slave port) with the Remote MAC. The serial interface consists of serial management clock (MDC) and serial data input and output (MDIO) signals shown in Figure 5-25. The remote MAC controller performs the following tasks:

- De-serializes the input data
- Checks the PHY and register address for validity
- Writes data to the register
- Serializes the register read-back data

The CSR module inside the remote MAC controller contains the registers described in "RevMII Registers" on page 231. To connect the bidirectional MDIO signal to the remote MAC, you need to implement a three-state buffer outside the MAC as shown in Figure 5-26.

Note When you select the SMA interface and the RevMII PHY interface during configuration, the MDIO interface signals from the SMA module and RevMII modules are multiplexed internally. This selection is controlled by the phy_intf_sel_i sampled at reset. For the RevMII mode, the phy_intf_sel_i signal has value 3'b111. For SMA mode, the phy_intf_sel_i signal can have any value specified in phy_intf_sel_i signal description.

Figure 5-26 Remote MAC to RevMII MDIO Connectivity



If the remote MAC tries to perform a read transaction to a register in the extended register set that is not implemented by the RevMII, then the RevMII does not drive the MDIO line in response to the read transaction. This means that gmii_mdo_o and gmii_mdo_o_e signals remain low. Similarly, if the remote MAC tries to perform a write transaction to a register in the extended register set that is not implemented by the RevMII, then the RevMII ignores the write transaction.

If a MAC tries to perform a read transaction on a register that is not implemented, then the RevMII reads back a zero value. Similarly, if MAC tries to perform a write transaction, then the RevMII does not write anything and ignores the write transaction.

5.6.3 RevMII Interrupts

The RevMII controller contains two interrupt signals, one for MAC and one for remote MAC. These interrupt signals are asserted high when the link status changes, that is, link goes up or down. After receiving the interrupt, the software should clear the MAC_RevMII_Interrupt_Status_Mask Register by reading this register. The Link Status Change interrupt is not generated if Bit 0 (LSIM) is set.

5.6.4 RevMII Registers

The CSR module of the local MAC controller and remote MAC controller contains the control, status, and interrupt registers for RevMII. This section provides the address maps of these registers and describes these registers.

Note The application indirectly accesses the RevMII registers by using the MAC_MDIO_Address and MAC_MDIO_Data registers.

5.6.4.1 RevMII Register Maps

Table 5-5 provides the address map and high-level summary of the RevMII registers for MAC.

Table 5-5 RevMII Register Maps - MAC

Register Number	Address Offset	Register Name and Description
0	5'b00000	MAC_RevMII_PHY_Control Controls the MAC side of the RevMII Controller.
1	5'b00001	MAC_RevMII_Common_Status Shows the status of the RevMII. This is a shared register for both MACs.
2–14	5'b00010–5'b01110	Reserved
15	5'b01111	MAC_RevMII_Common_Ext_Status Shows the status of the RevMII supporting 1000 Mbps speed. This is a common register for both MACs.
16	5'b10000	MAC_RevMII_Interrupt_Status_Mask Shows the status of the interrupt generated for the MAC and also provides interrupt masking signal for the generated interrupts.
17	5'b10001	MAC_RevMII_Remote_PHY_Status Shows the status of speed and duplex-mode programmed in the remote PHY Control register.
18–31	5'b10010–5'b11111	Reserved

Table 5-6 provides the address map and high-level summary of the RevMII registers of the remote MAC.

Table 5-6 RevMII Register Map - Remote MAC

Register Number	Address Offset	Register Name and Description
0	5'b00000	MAC_RevMII_RemotePHY_Control Controls the remote side of RevMII Controller.This register is similar to the MAC_RevMII_PHY_Control register.
1	5'b00001	MAC_RevMII_Common_Status Shows the status of the RevMII. This is a common register for both MACs.
2–14	5'b00010–5'b01110	Reserved
15	5'b01111	MAC_RevMII_Common_Ext_Status Shows the status of the RevMII supporting 1000 Mbps speed. This is a common register for both MACs.
16	5'b10000	MAC_RevMII_RemotePHY_Interrupt_Status_Mask Shows the status of the interrupt generated for remote MAC and also provides interrupt masking signal for the generated interrupts. This register is similar to MAC_RevMII_Interrupt_Status_Mask.
17	5'b10001	MAC_RevMII_PHY_Status Register Shows the status of speed and duplex-mode programmed in the RevMII PHY Control register.

Register Number	Address Offset	Register Name and Description
18–31	5'b10010–5'b11111	Reserved

5.6.4.2 MAC_RevMII_PHY_Control

The RevMII PHY Control register controls the RevMII operations for the MAC and enables the RevMII modes.

15	14	13	12	11	10	9	8	7	6	5:0
REVRST	REVLPBCK	REVSSL	REVANEN	REVPWRDN	REVISOL	REVREAN	REVDM	REVC OLTST	REV SSH	Rsvd

Table 5-7 MAC_RevMII_PHY_Control Register

Field	Name	Description	Reset	Access
15	REVRST	Reset When this bit is set, it configures the PHY Control register to its default values. This bit is cleared after the reset operation is complete.	0	R_W_SC
14	REVLPBCK	Loopback When this bit is set, it enables the loopback mode. When this bit is reset, it disables the loopback mode.	0	R/W
13	REVSSL	Speed Selection (LSB) This bit along with Bit 6 (MSB) indicates the link speed as described in the following table. **Bit 6 Bit 13 Speed** 1 1 Reserved 1 0 1000 Mbps 0 1 100 Mbps 0 0 10 Mbps When you select 10/100 Mbps as the **Mode of Operation**, the reset value of this bit is 1.	0	R/W
12	REVANEN	Auto-Negotiation Enable This bit is not used in RevMII.	0	RO
11	REVPWRDN	Power Down When this bit is set, it enables the power down mode. When this bit is reset, it disables the power-down mode.For more information, see “Power-Down Mode” on page 193.	0	R/W
10	REVISOL	Isolate When this bit is set, it enables the isolate mode. When this bit is reset, it disables the isolate mode.For more information, see “Isolate Mode” on page 194.	0	R/W

Field	Name	Description	Reset	Access
9	REVREAN	Restart Auto-Negotiation This bit is not used in RevMII.	0	RO
8	REVDM	Duplex Mode When this bit is set, it configures the RevMII PHY for the full-duplex operation. When this bit is reset, it configures the RevMII PHY for the half-duplex operation.When you select the Disable Half-Duplex Operation option, the reset value of this bit is 1.	0	R/W
7	REVCOLTST	Collision Test When this bit is set, it enables the collision test mode. When this bit is reset, it disables the collision test mode.For more information, see "Collision Test Mode" on page 194.	0	R/W
6	REVSSH	Speed Selection (MSB) When set, this bit along with Bit 6 (LSB) indicates the link speed. For more information, see REVSSL.When you select 10/100 Mbps as the Mode of Operation, the reset value of this bit is 0.	1	R/W
5:0	Rsvd	Reserved	0	RO

5.6.4.3 MAC_RevMII_Common_Status

This register provides the RevMII status. This register is common for the MAC and the remote MAC.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
100T4	100XFD	100XHD	10FD	10HD	100T2FD	100T2HD	EXT STS	Rsvd	PR ES UP	ANC	RM TF LT	AN A	LN KS TS	JA BD ET	EXT CAP

Table 5-8 MAC_RevMII_Common_Status Register

Field	Name	Description	Reset	Access
15	100T4	100BASE-T4 When this bit is set, it indicates that the RevMII PHY can perform the link transmission and reception using the 100BASE-T4 signaling specification.	1	RO
14	100XFD	100BASE-X Full-Duplex When this bit is set, it indicates that the RevMII PHY can perform the full-duplex link transmission and reception using the 100BASE-X signaling specification.	1	RO
13	100XHD	100BASE-X Half-Duplex When this bit is set, it indicates that the RevMII PHY can perform the half-duplex link transmission and reception using the 100BASE-X signaling specification.When you select the **Disable Half-Duplex Operation** option, the reset value of this bit is 0.	1	RO

Field	Name	Description	Reset	Access
12	10FD	10 Mbps Full-Duplex When this bit is set, it indicates that the RevMII PHY can perform the full-duplex link transmission and reception while operating in 10 Mbps mode.	1	RO
11	10HD	10 Mbps Half-Duplex When this bit is set, it indicates that the RevMII PHY can perform the half-duplex link transmission and reception while operating in 10 Mbps mode.When you select the **Disable Half-Duplex Operation** option, the reset value of this bit is 0.	1	RO
10	100T2FD	100BASE-T2 Full-Duplex When this bit is set, it indicates that the RevMII PHY can perform full-duplex link transmission and reception using the 100BASE-T2 signaling specification.	1	RO
9	100T2H D	100BASE-T2 Half-Duplex When this bit is set, it indicates that the RevMII PHY can perform the half-duplex link transmission and reception using the 100BASE-T2 signaling specification.When you select the **Disable Half-Duplex Operation** option, the reset value of this bit is 0.	1	RO
8	EXTSTS	Extended Status When this bit is set, it indicates that the base register status information is extended into the MAC_RevMII_Common_Ext_Status register. This bit should be set for 1000 Mbps mode.When you select 10/100 Mbps as the **Mode of Operation**, the reset value of this bit is 0.	1	RO
7	Rsvd	Reserved	0	RO
6	PRESUP	MF Preamble Suppression This Bit indicates that the RevMII can receive management frames irrespective of the Preamble length.	1	RO
5	ANC	Auto-Negotiation Complete This bit is not used in RevMII.	0	RO
4	RMTFLT	Remote Fault This bit is not used in RevMII.	0	RO
3	ANA	Auto-Negotiation Ability This bit is not used in RevMII.	0	RO
2	LNKSTS	Link Status When this bit is set, it indicates that a valid link has been established. When this bit is reset, it indicates that the link is not valid.	0	R_SS_SC_LLO
1	JABDET	Jabber Detect This bit is not used in RevMII.	0	RO
0	EXTCAP	Extended Capability This bit is always set because RevMII supports extended register capability.	1	RO

5.6.4.4 MAC_RevMII_Common_Ext_Status

This register is common for the MAC and the remote MAC. This register is implemented for RevMII supporting 1000 Mbps speed. It is not present when the MAC supports only 10/100 Mbps operations.

15	14	13	12	11:0
1000XFD	1000XHD	1000TFD	1000THD	Rsvd

Table 5-9 MAC_RevMII_Common_Ext_Status Register

Field	Name	Description	Reset	Access
15	1000XFD	1000BASE-X Full-Duplex When set, this bit indicates that the RevMII PHY can perform the full-duplex link transmission and reception using the 1000BASE-X signaling specification. When you select 10/100 Mbps as the **Mode of Operation**, the reset value of this bit is 0.	1	RO
14	1000XHD	1000BASE-X Half-Duplex When set, this bit indicates that the RevMII PHY can perform the half-duplex link transmission and reception using the 1000BASE-X signaling specification. When you select either the **Disable Half-Duplex Operation** option or 10/100 Mbps as the **Mode of Operation**, the reset value of this bit is 0.	1	RO
13	1000TFD	1000BASE-T Full-Duplex When set, this bit indicates that the RevMII PHY can perform the full-duplex link transmission and reception using the 1000BASE-T signaling specification. When you select 10/100 Mbps as the **Mode of Operation**, the reset value of this bit is 0.	1	RO
12	1000THD	1000BASE-T Half Duplex When set, this bit indicates that the RevMII PHY can perform the half-duplex link transmission and reception using the 1000BASE-T signaling specification. When you select either the **Disable Half-Duplex Operation** option or 10/100 Mbps as the **Mode of Operation**, the reset value of this bit is 0.	1	RO
11:0	Rsvd	Reserved	0	RO

5.6.4.5 MAC_RevMII_Interrupt_Status_Mask

This register provides the status of the interrupts and enables you to mask the interrupt signal. The status bits are cleared when this register is read.

15:9	8	7:1	0
Rsvd	LSI	Rsvd	LSIM

Table 5-10 MAC_RevMII_Interrupt_Status_Mask Register

Field	Name	Description	Reset	Access
15:9	Rsvd	Reserved	0	RO

Field	Name	Description	Reset	Access
8	LSI	Link Status Change Interrupt When this bit is set, it indicates that the link status has changed. This bit is cleared on read (or this bit is written to 1 when RWCE bit of MAC_CSR_SW_Ctrl register is set)	0	R_SS_RC_W1C
7:1	Rsvd	Reserved	0	RO
0	LSIM	Link Status Change Interrupt Mask When this bit is set, it disables the assertion of the interrupt signal because of the setting of the LSI bit.	0	R/W

5.6.4.6 MAC_RevMII_Remote_PHY_Status

This register shows the status of speed and duplex-mode programmed in the remote PHY Control register. You can use this register for MAC to MAC handshake when the link between the MAC and remote MAC is down because of the speed or duplex-mode mismatch.

15:3	2	1	0
Rsvd	RMACDM	RMACSSH	RMACSSL

Table 5-11 MAC_RevMII_Remote_PHY_Status Register

Field	Name	Description	Reset	Access
15:3	Rsvd	Reserved	0	RO
2	RMACDM	Remote MAC Duplex Mode When this bit is set, it indicates the duplex mode configured in Bit 8 of the MAC_RevMII_RemotePHY_Control register. When you select the **Disable Half-Duplex Operation** option, the reset value of this bit is 1.	0	RO
1	RMACSSH	Remote MAC Speed Select MSB When this bit is set, it indicates the link speed specified in Bit 6 of the MAC_RevMII_RemotePHY_Control register. When you select 10/100 Mbps as the **Mode of Operation**, the reset value of this bit is 0.	1	RO
0	RMACSSL	Remote MAC Speed Select LSB When this bit is set, this bit indicates the link speed specified in Bit 13 of the MAC_RevMII_RemotePHY_Control register. When you select 10/100 Mbps as the **Mode of Operation**, the reset value of this bit is 1.	0	RO

Table 5-12 describes the RevMII PHY Status Register of a remote MAC.

15:3	2	1	0
Rsvd	MACDM	MACSSH	MACSSL

Table 5-12 MAC_RevMII_PHY_Status Register

Field	Name	Description	Reset	Access
15:3	Rsvd	Reserved	0	RO
2	MACDM	MAC Duplex mode When set, this bit indicates the duplex mode configured in Bit 8 of the MAC_RevMII_PHY_Control register.When you select the **Disable Half-Duplex Operation** option, the reset value of this bit is 1.	0	RO
1	MACSSH	MAC Speed Select MSB When set, this bit indicates the link speed specified in Bit 6 of the MAC_RevMII_PHY_Control register.When you select 10/100 Mbps as the **Mode of Operation**, the reset value of this bit is 0.	1	RO
0	MACSSL	MAC Speed Select LSB When set, this bit indicates the link speed specified in Bit 13 of the MAC_RevMII_PHY_Control register.When you select 10/100 Mbps as the **Mode of Operation**, the reset value of this bit is 1.	0	RO

5.6.5 Enabling Reverse Media Independent Interface

To enable this interface while configuring the controller in the coreConsultant GUI:

1. Select the **Enable REVMII** option under the PHY Interface section during the **Specify Configuration** activity in coreConsultant.

   For details about this option, see PHY Interface Parameters in "Parameter Descriptions" on page 419

Note See the "Enabled" field of the parameter descriptions to understand the dependencies.

5.7 Serial Gigabit Media Independent Interface

5.7.1 Introduction to Serial Gigabit Media Independent Interface

The Serial Gigabit Media Independent Interface (SGMII) defines the interface from the Gigabit MAC to the Gigabit PHY. Using SGMII lowers the pin count required for GMII. The interface definition supports all speed modes (10 Mbps, 100 Mbps, and 1 Gbps). The reduced pin count comes at the cost of higher power consumption, mainly because the SGMII interface maintains a constant 1250-MHz clock rate, regardless of the operating speed of the MAC.The SGMII block has the following characteristics:

- Supports 10-Mbps, 100-Mbps, and 1000-Mbps operations
- Provides in-band (link speed, duplex mode, and link status) status signals from the PHY provided to MAC for link detection.
- Requires separate 125-MHz clock inputs for Transmit and Receive paths. For guidelines for clock connections when the MAC operates with an SGMII, see *DesignWare Cores Ethernet Quality-of-Service User Guide*.

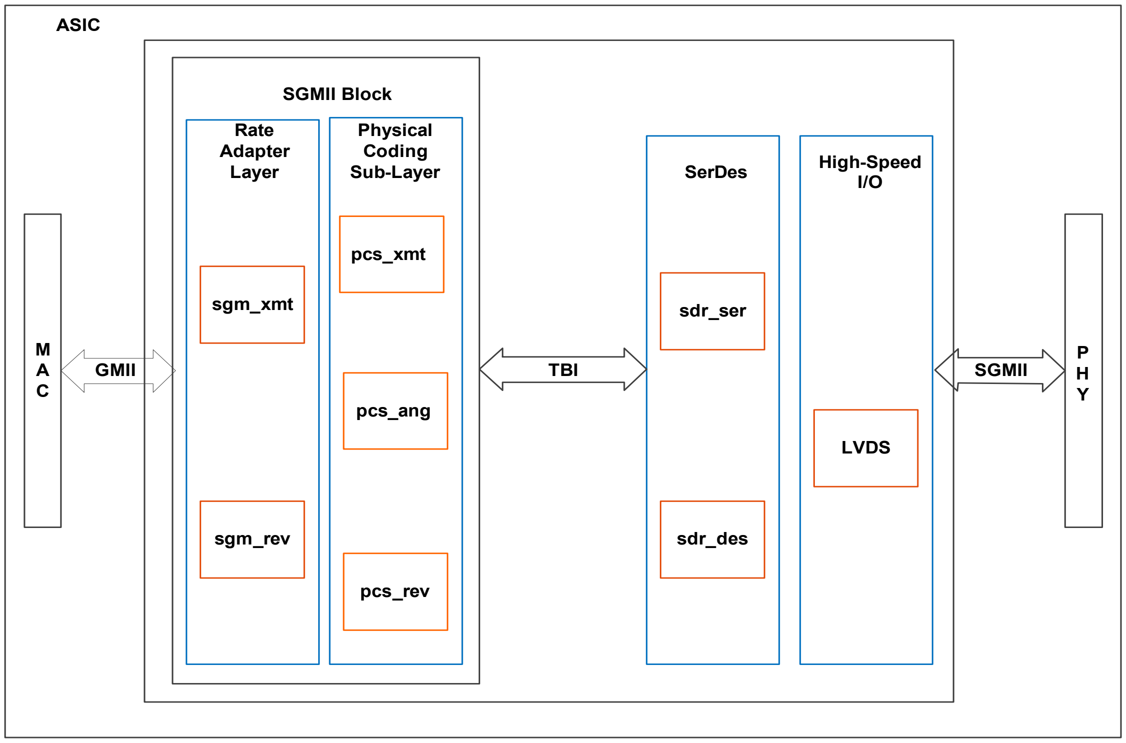
The Synopsys RTL deliverable does not include the SerDes and high-speed I/O blocks. You must develop these modules to suit the technology and ASIC manufacture process.

5.7.2 Description of Serial Gigabit Media Independent Interface

While the SGMII block is placed in the ASIC in such a way that it interacts with both the MAC and PHY blocks. However, you must implement the SerDes and high-speed I/O modules to implement the actual SGMII.

Figure 5-27 shows the placement of the SGMII block with respect to the MAC and PHY blocks.

Figure 5-27 SGMII Block Diagram



The SGMII block comprises the Rate Adapter Layer (RAL) and the Physical Coding Scheme (PCS) modules as shown in Figure 5-27. You must implement the SerDes and high-speed I/O modules to implement the actual SGMII interface.

The SGMII block consists of the following blocks:

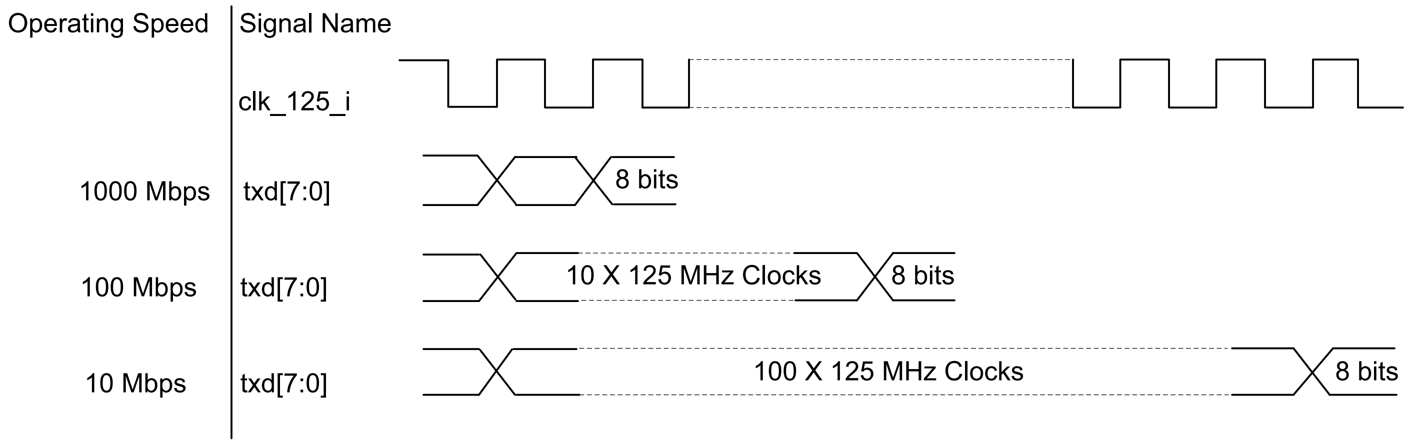
- Rate Adapter Layer (RAL)
- Physical Coding Sub-Layer (PCS)
- SerDes
- High-Speed I/O

5.7.2.1 Rate Adapter Layer (RAL)

The PCS layer runs on a 125 MHz clock whereas the GMII interface may run on a 2.5 MHz, 25 MHz, or 125 MHz clock. Therefore, the data to or from the GMII must be stretched in the Transmit path based on the MAC data rate. Figure 5-28 shows how a single 8-bit data stream is stretched with respect to the 125-MHz clock.

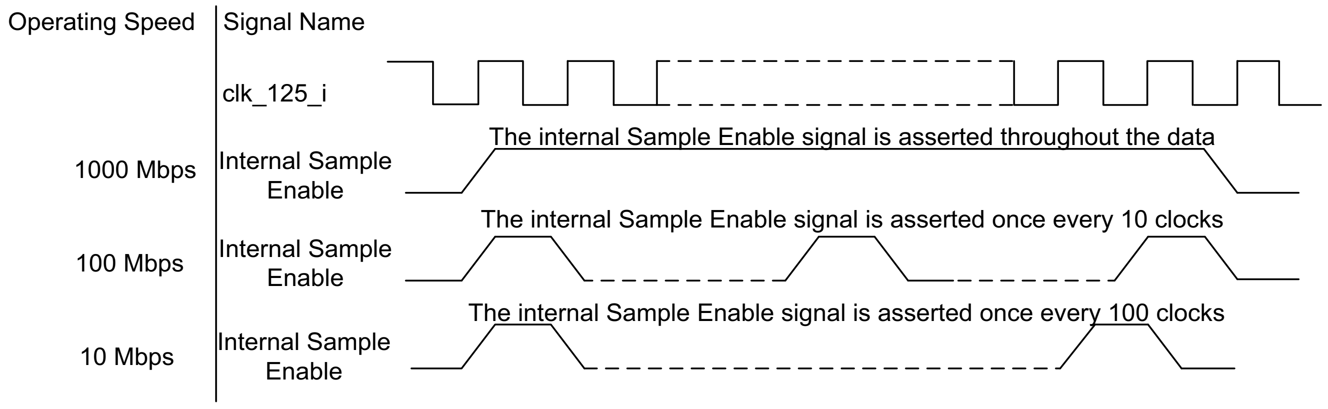
In the Transmit path, if the MAC is in the Gigabit mode, the data is not stretched at all, and it takes only one clock cycle. However, if the MAC is in 100 Mbps or 10 Mbps mode, the data is repeated for 10 or 100 consecutive 125 MHz clock cycles, respectively. In Figure 5-28, data is repeated 10 or 100 times in the Transmit path of the RAL to match the speed mode of the MAC layer. The txd[7:0] signal is an internal signal.

Figure 5-28 Data Stretched in the Tx Path With a 125-MHz Clock



In the Receive path, the SGMII PHY stretches the data according to the speed of the MAC before it sends the data out to the SGMII block. The RAL receives the data repeated 10 or 100 times depending on the mode of the MAC. Therefore, the RAL does not have to repeat that data as in the case of the Transmit path. It only samples the data once every 10 or 100 clocks, as shown in Figure 5-29. For additional information, refer to the Serial GMII Specification, Revision 1.7.

Figure 5-29 Data Stretched in the Rx Path With a 125-MHz Clock


Operating Speed
Signal Name
clk_125_i
1000 Mbps
Internal Sample Enable
The internal Sample Enable signal is asserted throughout the data
100 Mbps
Internal Sample Enable
The internal Sample Enable signal is asserted once every 10 clocks
10 Mbps
Internal Sample Enable
The internal Sample Enable signal is asserted once every 100 clocks

5.7.2.2 Physical Coding Sub-Layer (PCS)

In the Transmit path, the Physical Coding Sub-Layer takes the 8-bit-wide GMII data from the RAL and encodes it using the 8-bit/10-bit algorithm to supply the SerDes block with a 10-bit-wide data stream. In the Receive block, the Physical Coding sublayer takes 10-bit-wide data from the SerDes block and decodes it to supply the RAL block with an 8-bit-wide data stream.

The PCS module is common for SGMII, TBI, and RTBI PHY interfaces. However, there are some changes in the PCS module specific to SGMII as given in the SGMII Specification. These changes are with respect to the Link Timer duration (1.6 ms instead of 10 ms) and the Configuration Register codes transmitted during auto-negotiation.

The tx_config_reg[15:0] bits sent by the MAC during auto-negotiation depend on whether the Transmit Configuration register bit is enabled for the SGMII interface as described in Table 5-13.

Table 5-13 tx_config_reg[15:0] Setting When Transmit Configuration Register Bits are Enabled or Disabled

	Transmit Configuration Bit	
Bit	Disabled (Default)	Enabled
15	0	Link Up/Down (Bit 1 of the MAC_PHYIF_Control_Status register)
14	1	1
13	0	0
12	0	Duplex (Bit 13 of the MAC_Configuration register)
11:10	00	■ 10 in 1000 Mbps mode ■ 01 in 100 Mbps mode ■ 00 in 10 Mbps mode The speed is determined by the Port Select and Speed Control bits in the MAC_Configuration register.
9:1	00H	00H

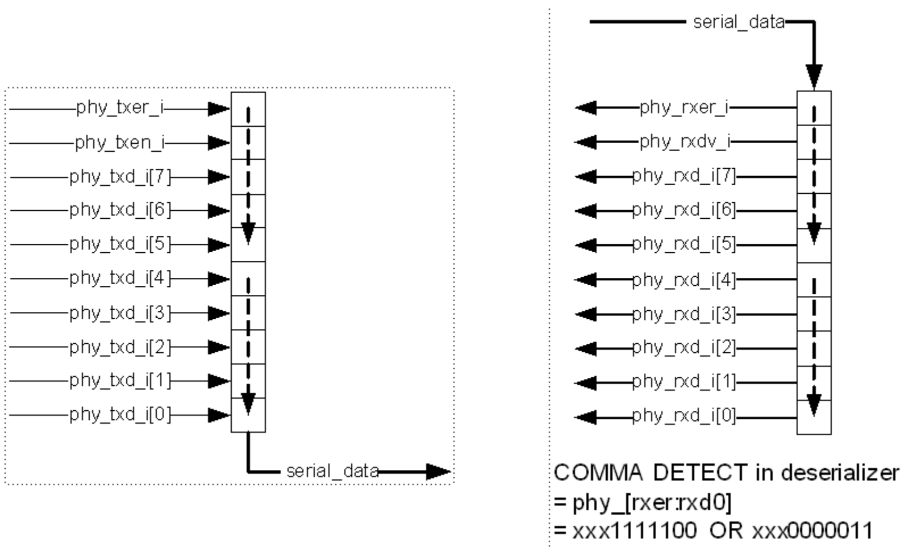
	Transmit Configuration Bit	
Bit	**Disabled (Default)**	**Enabled**
0	1	1

5.7.2.3 Serializer and Deserializer (SerDes) in the SGMII Block

SerDes is a serializer and de-serializer module in the SGMII block. It its connected by a Ten-Bit Interface (TBI) with the PCS module.

In the Transmit path, the SerDes block takes the 10-bit-wide parallel data from the PCS and converts it to a serial format. In the Receive path, the SerDes block takes the serial SGMII input and converts it to a parallel 10-bit-wide data format for the PCS layer.The connection of the 10-bit transmit data signals with the serializer and the 10-bit receiver data signals from the de-serializer are shown in Figure 5-30.

Figure 5-30 SGMII Serializer and De-Serializer



Note The provided RTL should be used only for simulation. It should not be used for synthesis. The SerDes and LVDS I/O blocks are custom blocks of technology-specific library parts. See 8 for a block diagram of the MAC (DUT) with support for an SGMII interface.

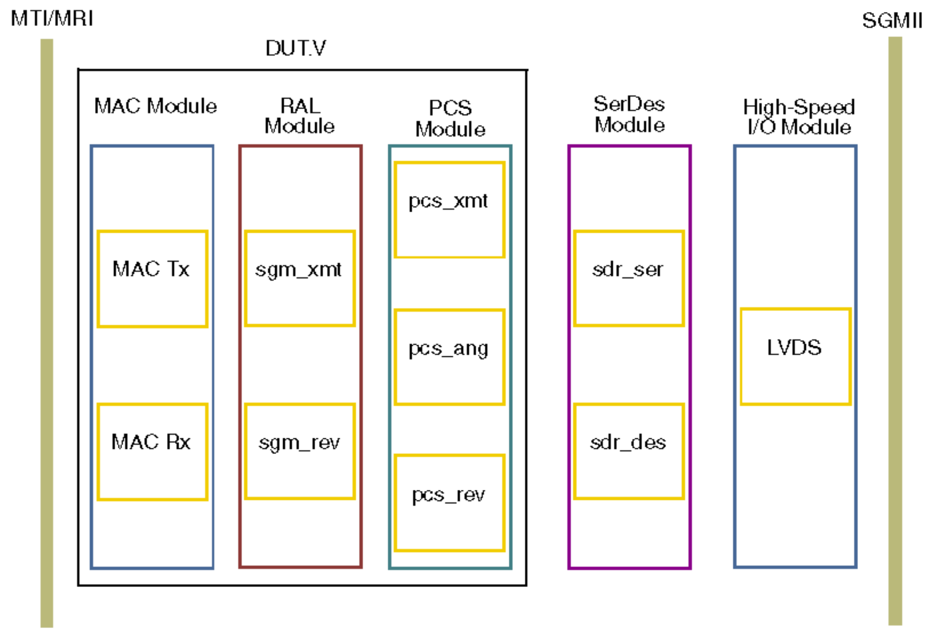
5.7.2.4 High-Speed I/O

The I/O block used in the SGMII is an LVDS type I/O. For additional information, refer to the Serial GMII Specification, Revision 1.7.

Note The provided RTL should be used only for simulation. It should not be used for synthesis. The SerDes and LVDS I/O blocks are custom blocks of technology-specific library parts.

Figure 5-31 shows a block diagram of the MAC (DUT) with support for an SGMII interface.

Figure 5-31 MAC (DUT) With Support for an SGMII Interface



5.7.3 Enabling Serial Gigabit Media Independent Interface

To enable this interface while configuring the controller in the coreConsultant GUI:

1. Select the **Enable SGMII** option under the PHY Interface section during the **Specify Configuration** activity in coreConsultant

   For details about this option, see PHY Interface Parameters "Parameter Descriptions" on page 419.

Note See the "Enabled" field of the parameter descriptions to understand the dependencies.

5.8 Reduced Ten-Bit Interface

5.8.1 Introduction to Reduced Ten-Bit Interface

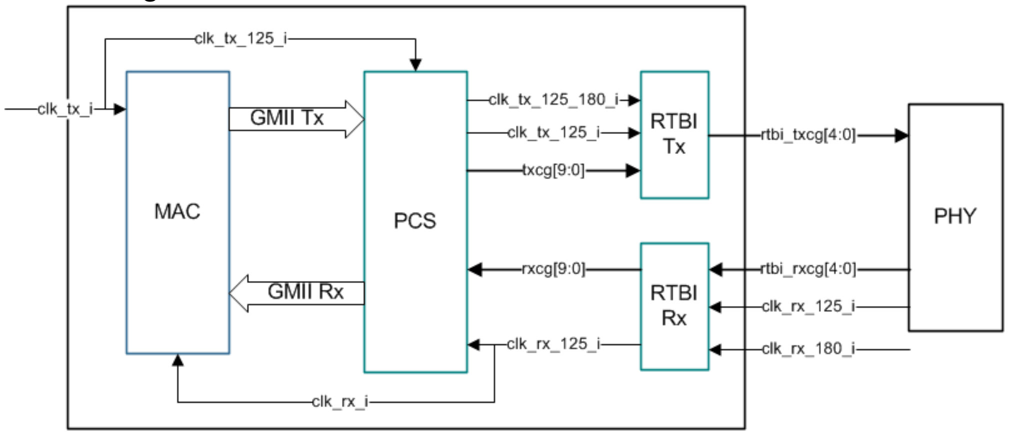
The Reduced Ten-Bit Interface (RTBI) specification reduces the pin count of the interconnection between the MAC and the PHY for TBI interface. To achieve this, the Transmit and Receive code-groups are multiplexed together with both edges of the Transmit and Receive clocks. In the MAC, the optional RTBI module is instantiated between the PCS module of the MAC and the PHY to translate the code-group signals between the TBI and RTBI protocols. The TBI of MAC supports only 1000BASE-X. Therefore, the RTBI also supports only 1000BASE-X, and it is not valid for 10/100 Mbps operation.For the RTBI block, no extra clock is required because both edges of the incoming clocks are used. To simplify back-end implementation of the MAC, there are separate clock inputs (180 degrees out-of-phase with the associated Transmit or Receive clocks) for logic that uses the falling edges. For guidelines for clock connections when the MAC operates with RTBI, see *DesignWare Cores Ethernet Quality-of-Service User Guide*.

5.8.2 Description of Reduced Ten-Bit Interface

An RTBI block contains a Transmit block and Receive block to implement the logic to reduce pin count of the interconnection between the MAC and the PHY for TBI interface.

Figure 5-32 shows the position of the RTBI block and its I/O signals relative to the MAC and RTBI PHY.

Figure 5-32 RTBI Block Diagram



The following list describes the RTBI components shown in Figure 5-32:

- **Transmit (RTBI Tx) Block**: This block translates the 10-bit Transmit code-group signal from PCS module to 5-bit transmit code-group. The RTBI Tx registers the input signals at the rising edge of clk_tx_125_i clock and generates the RTBI transmit signals at the rising edge of either clk_tx_125_i or clk_tx_125_180_i clock.
- **Receive (RTBI Rx) Block**: This block translates the 5-bit code-group signals received from the PHY to 10-bit code-group towards the PCS. The RTBI Rx registers the input signals at the rising edge of either clk_rx_125_i or clk_rx_125_180_i clock and generates the rxcg[9:0] signal output at the rising edge of clk_rx_125_i clock.

Note The clk_tx_125_180_i and clk_rx_125_180_i clocks should be 180 degrees out-of-phase with respect to the clk_tx_125_i and clk_rx_125_i clocks, respectively.

5.8.3 RTBI Transmit Timing

In the default mode, the RTBI transmitter outputs the lower 5 bits of the input txcg[9:0] (from PCS) at the rising edge of clk_tx_125_i followed by the upper 5 bits of txcg[9:0] on the falling edge of clk_tx_125_i.

5.8.4 Enabling Reduced Ten-Bit Interface

The Reduced Ten-Bit Interface (RTBI) specification reduces the pin count of the interconnection between the MAC and the PHY for TBI interface. You can use coreConsultant to confiugre the RTBI.

To enable this interface, select the **Enable RTBI** option under the PHY Interface section during the **Specify Configuration** activity in coreConsultant.

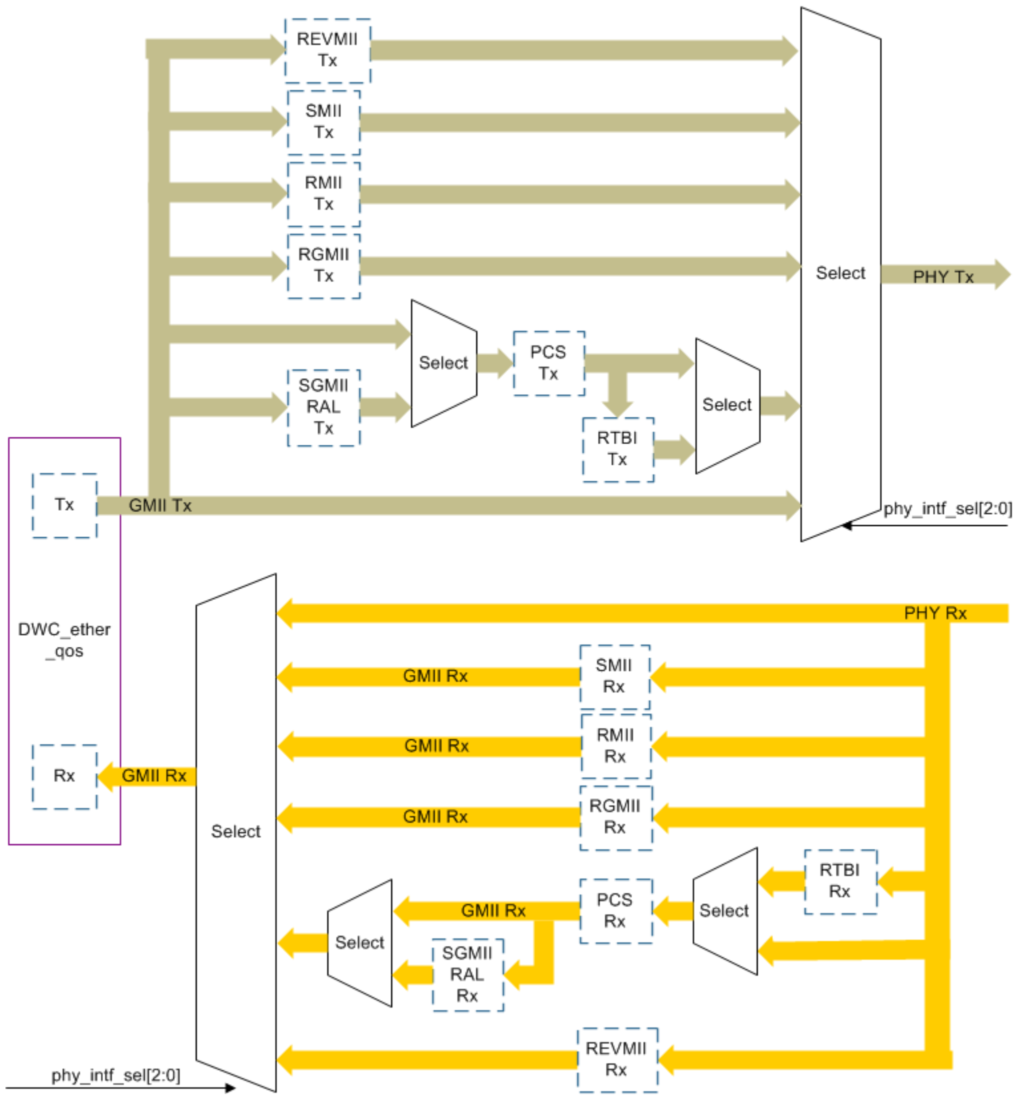
5.9 Multiple PHY Interfaces

DWC_ether_qos support multiple PHY interfaces by using a multiplexing log.

Figure 5-33 shows the multiplexing logic implemented to support multiple programmable PHY interfaces.

Note When the MAC is configured for a single PHY interface, all MUX logic is removed and the corresponding PHY interface is connected directly to the ports.

Figure 5-33 Multiplexing Logic for Multiple Programmable PHY Interfaces



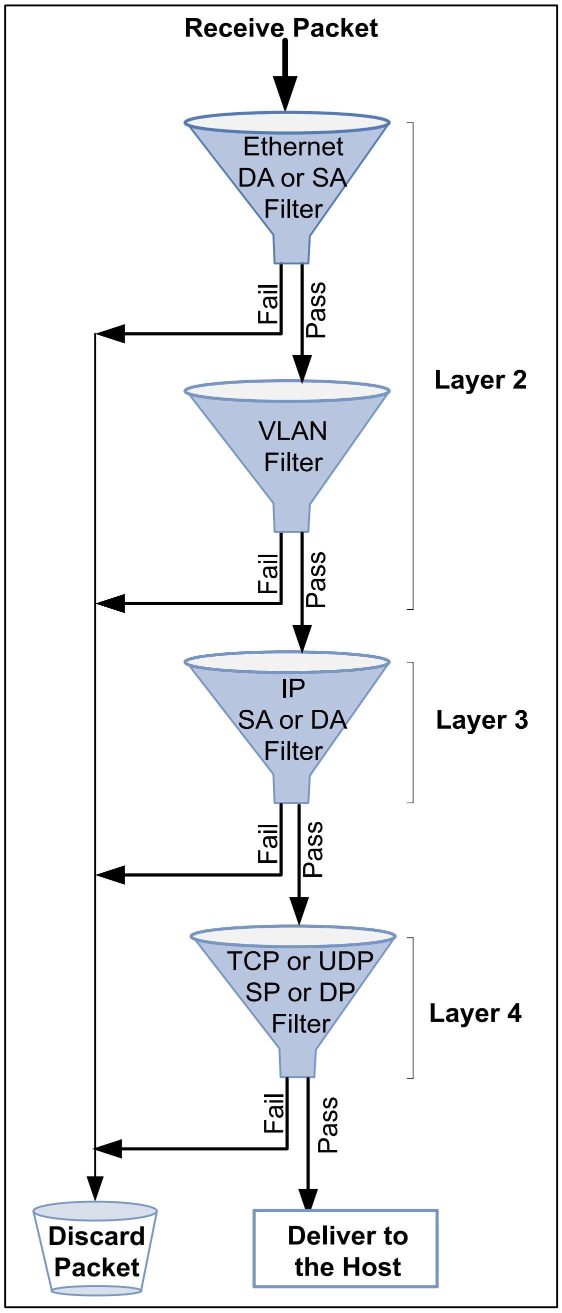
6 Packet Filtering

This chapter describes the various packet filtering schemes available in the MAC receiver. It contains the following sections:

6.1 Packet Filtering Sequence

Figure 6-1 shows the filtering sequence for Rx packets.

Figure 6-1 Packet Filtering Sequence



The sequence shown in Figure 6-1 is valid when all the filters (L2, VLAN, L3, L4) are active. If any of the Layer filters are not enabled, that filter is bypassed and the subsequent filter is applied. A packet that fails any of the filters is discarded. However, the discarded packet can be forwarded to the host based on the register control. For example, when Bit RA of MAC_Packet_Filter register is set to 1, all discarded packets are forwarded to the host but with its packet status indicating the specific filter-failure. If RA=0, Bits VTFE and IPFE of MAC_Packet_Filter register controls if the packets that fail the VLAN filter and Layer 3-4 filter should be discarded or forwarded to the host.

Note When Flexible Receive Parser feature is enabled in your configuration (DWC_EQOS_FRP_EN =1) and Flexible Receive feature (FRPE bit in MTL_Operation_mode register) is enabled, all the MAC packet filtering features should not be enabled. For more details, see "MAC Packet Filtering/Drop/Error Handling" on page 238.

6.2 Source Address or Destination Address Filtering

6.2.1 Introduction to Source or Destination Address Filtering

The Address Filtering Module of the MAC checks the source address (SA) and destination address (DA) fields of each incoming packet.

6.2.2 Enabling Additional MAC Address Registers

To enable additional MAC address registers while configuring the controller in the coreConsultant GUI:

1. Select the number of registers in **Enable Additional 1-31 MAC Address Registers** option under the Filtering section during the Specify Configuration activity in coreConsultant.
2. Select the **Enable Additional 32 MAC Address Registers (32-63)**option, if required.
3. Select the **Enable Additional 64 MAC Address Registers (64-127)** option, if required.

For details about this option, see Filtering Parameters in the "Parameter Descriptions" on page 419.

Note See the "Enabled" field of the parameter descriptions to understand the dependencies

6.2.3 Configuring Address Filter Hash Table

To configure Address Filter Hash Table while configuring the controller in the coreConsultant GUI:

1. Select the Enable Address Filter Hash Table option under the Filtering section during the Specify Configuration activity in coreConsultant.
2. Specify the hash table size in the Hash Tale Size option.

For details about this option, see Filtering Parameters in the "Parameter Descriptions" on page 419.

Note See the "Enabled" field of the parameter descriptions to understand the dependencies

6.2.4 Programming Different Types of Address Filtering

6.2.4.1 Unicast Destination Address Filtering

The MAC supports up to 128 MAC addresses for unicast perfect filtering. If perfect filtering is selected (HUC bit of MAC_Packet_Filter register is reset), the MAC compares all 48 bits of received unicast address with the programmed MAC address for any match. The default MacAddr0 is always enabled.

The MacAddr1 to MacAddr127 addresses are selected with an individual enable bit. For MacAddr1 to MacAddr31 addresses, you can mask each byte during comparison with corresponding received DA byte by setting the corresponding Mask Byte Control bit in the register. This enables group address filtering for the DA. The MacAddr32 to MacAddr127 addresses do not have mask control and all 6-bytes of the MAC address are compared with the received 6-bytes of DA.

In hash filtering mode (when HUC bit is set), the MAC performs imperfect filtering for unicast addresses using a 64-bit Hash table. For hash filtering, the MAC uses the upper 6 bits CRC of the received destination address to index the content of the Hash table. A value of 00000 selects Bit 0 of selected register, and a value of 11111 selects Bit 63 of Hash Table register. If the corresponding bit (indicated by the 6-bit CRC) is set to 1, the unicast packet is considered to have passed the Hash filter; otherwise, the packet is considered to have failed the Hash filter.

6.2.4.2 Multicast Destination Address Filtering

To program the MAC to pass all multicast packets, set the PM bit in MAC_Packet_Filter register. If the PM bit is reset, the MAC performs the filtering for multicast addresses based on the HMC bit of the MAC_Packet_Filter register.

The multicast address is compared with the programmed MAC Destination Address registers (1–31). Group address filtering is also supported.

In Hash filtering mode, the MAC performs imperfect filtering using a 64-bit Hash table. The MAC uses the upper 6-bits CRC of received multicast address to index the content of the Hash table. A value of 000000 selects Bit 0 of selected register and a value of 111111 selects Bit 63 of the Hash Table register. If the corresponding bit is set to 1, the multicast packet is considered to have passed the Hash filter. Otherwise, the packet is considered to have failed the Hash filter.

6.2.4.3 Hash or Perfect Address Filtering

To configure the DA filter to pass a packet when its DA matches either the Hash filter or the Perfect filter, set the HPF bit and the corresponding HUC or HMC bits in MAC_Packet_Filter register. This is applicable to both unicast and multicast packets. If the HPF bit is reset, only one of the filters (Hash or Perfect) is applied to received packet.

6.2.4.4 Broadcast Address Filtering

The MAC does not filter any broadcast packets by default. To program the MAC to reject all broadcast packets, set the DBF bit in MAC_Packet_Filter register.

6.2.4.5 Unicast Source Address Filtering

The MAC can perform perfect filtering based on the source address field of received packets. By default, the MAC compares the SA field with the values programmed in the SA registers. You can configure the MAC Address registers[1–31] to use SA instead of DA for comparison by setting Bit 30 of corresponding register.

The MAC also supports group filtering with SA. You can filter a group of addresses by masking one or more bytes of the address. The MAC drops the packets that fail the SA filter if the SAF bit is set in MAC_Packet_Filter register. Otherwise, the result of the SA filter is given as a status bit in the Receive Status word. When the SAF bit is set, the SA filter and DA filter result is AND'ed to decide whether the packet needs to be forwarded. This means that the packet is dropped if either filter fails. The packet is forwarded to the application only if the packet passes both filters in-order.

6.2.4.6 Inverse Filtering

For DA and SA filtering, you can invert the filter-match result at the final output by setting the DAIF and SAIF bits of MAC_Packet_Filter register. The DAIF bit is applicable for both Unicast and Multicast DA packets. The result of the unicast or multicast destination address filter is inverted in this mode. Similarly, when the SAIF bit is set, the result of unicast SA filter is reversed.

Tables 6-1 and 6-2 summarize the DA and SA filtering based on the type of packets received.

Note When the RA bit of MAC_Packet_Filter register is set, all packets are forwarded to the system along with the correct result of the address filtering in the Rx Status.

Table 6-1 Destination Address Filtering

Packet Type	PR	HPF	HUC	DAIF	HMC	PM	DBF	DA Filter Operation
Broadcast	1	X	X	X	X	X	X	Pass
	0	X	X	X	X	X	0	Pass
	0	X	X	X	X	X	1	Fail
Unicast	1	X	X	X	X	X	X	Pass all packets
	0	X	0	0	X	X	X	Pass on Perfect/Group filter match
	0	X	0	1	X	X	X	Fail on Perfect/Group filter match
	0	0	1	0	X	X	X	Pass on Hash filter match
	0	0	1	1	X	X	X	Fail on Hash filter match
	0	1	1	0	X	X	X	Pass on Hash or Perfect/Group filter match
	0	1	1	1	X	X	X	Fail on Hash or Perfect/Group filter match

Packet Type	PR	HPF	HUC	DAIF	HMC	PM	DBF	DA Filter Operation
Multicast	1	X	X	X	X	X	X	Pass all packets
	X	X	X	X	X	1	X	Pass all packets
	0	X	X	0	0	0	X	Pass on Perfect/Group filter match and drop Pause packets if PCF = 0x
	0	0	X	0	1	0	X	Pass on Hash filter match and drop Pause packets if PCF = 0x
	0	1	X	0	1	0	X	Pass on Hash or Perfect/Group filter match and drop Pause packets if PCF = 0x
	0	X	X	1	0	0	X	Fail on Perfect/Group filter match and drop Pause packets if PCF = 0x
	0	0	X	1	1	0	X	Fail on Hash filter match and drop Pause packets if PCF = 0x
	0	1	X	1	1	0	X	Fail on Hash or Perfect/Group filter match and drop Pause packets if PCF = 0x

Table 6-2 Source Address Filtering

Packet Type	PR	SAIF	SAF	SA Filter Operation
Unicast	1	X	X	Pass all packets.
	0	0	0	Pass status on Perfect or Group filter match but do not drop packets that fail
	0	1	0	Fail status on Perfect or Group filter match but do not drop packet
	0	0	1	Pass on Perfect or Group filter match and drop packets that fail
	0	1	1	Fail on Perfect or Group filter match and drop packets that fail

6.3 VLAN Filtering

6.3.1 VLAN Tag Perfect Filtering

In VLAN tag perfect filtering, the MAC compares the VLAN tag of received packet and provides the VLAN packet status to the application. Based on the programmed mode, the MAC compares the lower 12 bits or all 16 bits of received VLAN tag to determine the perfect match.

If VLAN tag perfect filtering is enabled, the MAC forwards the VLAN-tagged packets along with VLAN tag match status and drops the VLAN packets that do not match. You can also enable the inverse matching for VLAN packets by setting the VTIM bit of MAC_VLAN_Tag register. In addition, you can enable matching of S-VLAN tagged packets along with the default C-VLAN tagged packets by setting the ESVL bit of MAC_VLAN_Tag register. For more details, see "TCP/IP Offloading Features" on page 339. The VLAN packet status bit (Bit 10 of RDES0) indicates the VLAN tag match status for the matched packets.

Note The source or destination address (if enabled) has precedence over the VLAN tag filters. This means that a packet that fails the source or destination address filter is dropped irrespective of the VLAN tag filter results. By default, the VLAN tag-based perfect filter is available in all configurations.

6.3.2 VLAN Tag Hash Filtering

The 16-bit VLAN Hash Table is used for group address filtering based on the VLAN tag. The VLAN Tag Hash Filtering feature can be enabled using the VTHM (VLAN Tag Hash Table Match Enable) bit of the MAC_VLAN_Tag register.

The MAC provides VLAN tag hash filtering with a 16-bit Hash table.

The MAC performs the VLAN hash matching based on the VTHM of the MAC_VLAN_Tag register. If the VTHM bit is set, the most significant four bits of CRC-32 of VLAN tag are used to index the content of the VLAN Hash Table register. A value of 1 in the VLAN Hash Table register, corresponding to the index, indicates that the VLAN tag of the packet matched and the packet should be forwarded. A value of 0 indicates that VLAN-tagged packet should be dropped.

Note

- The 16 or 12 bits of VLAN Tag are considered for CRC-32 computation based on ETV bit in MAC_VLAN_TAG register.
- When ETV bit is reset, most significant four bits of CRC-32 of VLAN Tag are inverted and used to index the content of MAC_VLAN_Hash_Table register.
- When ETV bit is set, most significant four bits of CRC-32 of VLAN Tag are directly used to index the content of MAC_VLAN_Hash_Table register.

The MAC also supports the inverse matching for VLAN packets. In the inverse matching mode, when the VLAN tag of a packet matches the perfect or hash filter, the packet should be dropped. If the VLAN perfect and VLAN hash match are enabled, a packet is considered as matched if either the VLAN hash or the VLAN perfect filter matches. When inverse match is set, a packet is forwarded only when both perfect and hash filters indicate mismatch.

Table 6-3 shows the different possibilities for VLAN matching and the final VLAN match status. When the RA bit of MAC_Packet_Filter register is set, all packets are received and the VLAN match status is indicated in the VF bit of RDES2 Normal Descriptor (Write-Back Format). When the RA bit is not set and the VTFE bit is set in MAC_Packet_Filter register, the packet is dropped if the final VLAN match status is Fail. In Table 6-3, value X means that this column can have any value.

When VLAN VID is programmed to 0 in the VL field of MAC_VLAN_Tag register, all VLAN-tagged packets are considered as perfect matched but the status of the VLAN hash match depends on the VTHM and VTIM bits in MAC_VLAN_Tag register.

Table 6-3 VLAN Match Status

VID	VLAN Perfect Filter Match Result	VTHM Bit	VLAN Hash Filter Match Result	VTIM Bit	Final VLAN Match Status
VID = 0	Pass	0	X	X	Pass
	Pass	1	X	0	Pass
	Pass	1	Fail	1	Pass
	Pass	1	Pass	1	Fail
VID!= 0	Pass	X	X	0	Pass
	Fail	0	X	0	Fail
	Fail	1	Fail	0	Fail
	Fail	1	Pass	0	Pass
	Fail	0	X	1	Pass
	Pass	X	X	1	Fail
	Fail	1	Pass	1	Fail
	Fail	1	Fail	1	Pass

In Table 6-3, X represents any value.

6.3.2.1 Enabling VLAN Tag Hash Filtering

To enable VLAN tag hash filtering while configuring the controller in the coreConsultant GUI:

1. Select the **Enable VLAN Hash Table Based Filtering** option under the Filtering section during the **Specify Configuration** activity in coreConsultant

   For details about this option, see Filtering Parameters in the "Parameter Descriptions" on page 419.

Note See the "Enabled" field of the parameter descriptions to understand the dependencies

6.4 VLAN Filter Fail Packets Queue

When VLAN filtering is enabled, the VLAN Filter Fail Packets can be routed to a programmable Queue (VFFQ) when Bit RA = 1 or VTFE = 0 and the enable bit (VFFQE) for the queue is set.

1. RA and VTFE bits are implemented in the MAC_Packet_Filter register
2. VFFQ and VFFQE fields are implemented in the MAC_RxQ_Ctrl4 register

The packets that pass the VLAN filtering are routed based on the VLAN TAG priority field. The VLAN Tag priorities can be assigned to Rx Queues by programming the PSRQ field in the corresponding MAC_RxQ_Ctrl2 and MAC_RxQ_Ctrl3 registers. The packets that fail the VLAN filter are discarded if RA=0 or VTFE=1. However when RA=1 or VTFE=0, the VLAN filter fail packets are still forwarded to the application. In such a scenario, when the VLAN Filter Fail Queue Enable (VFFQE) bit is set, the VLAN filter fail packets are forwarded to the Rx queue number programmed in the VFFQ bit. If VFFQE=0, the Rx queue number is determined by the VLAN priority mapping as per the PSRQ fields.

Table 6-4 shows the Rx Queue routing table for Unicast tagged packets, with DA/SA Filter enabled.

Table 6-4 Rx Queue Routing table for Unicast Tagged Packets

RA	VFTE	SA/DA Filter Result	VLAN Filter Result	VFFQE	Queue Routing
X	X	PASS	PASS	X	PSRQ
0	0	PASS	FAIL	0	PSRQ
0	0	PASS	FAIL	1	VFFQ
0	X	FAIL	X	X	DROPPED
0	1	PASS	FAIL	X	DROPPED
1	X	FAIL	X	0	UFFQ*/PSRQ
1	X	FAIL	X	1	UFFQ*/VFFQ
1	X	PASS	FAIL	0	PSRQ
1	X	PASS	FAIL	1	VFFQ

X : Don't care condition

* : When UFFQE is enabled else PSRQ

6.5 Extended Receive VLAN Filtering and Routing

The MAC Receiver can classify the received packets based on VLAN Tag and steer them to a specific Rx DMA channel. The MAC compares the received frame's VLAN Tag with all the enabled and relevant filters and provides a filtering result. If any of the perfect filters give a pass result and if the respective filter's DMA channel Number is enabled, the frame is routed to that DMA Channel.

In addition to filtering, routing can also be done. For more details about routing, see ““Extended VLAN Based DMA Selection” on page 289.

6.5.1 Comparison Modes

For each VLAN Tag Filter, the application has the following comparison options:

- It can program the MAC to compare an outer VLAN Tag or an inner VLAN Tag with the programmed VID.
- It can choose if 12 or 16 bits of the VID field need to be compared.
- Type check can be disabled or enabled for each filter; if enabled, the application can choose if the VID comparison is for SVLAN or CVLAN type frames only.

  For example, if a filter is enabled for 16 bit comparison, SVLAN Type, and Outer VLAN Tag, any single or double VLAN Tagged frames with Outer SVLAN Tags are compared with this filter, and a pass or fail result is obtained.

Note The inner VLAN Tag comparison is applicable only if Double VLAN Tag processing is enabled through the parameter and the MAC VLAN Control Register bit.

6.5.2 Filtering

When Extended RX VLAN Filtering & Routing feature is enabled, the application can enable both Perfect and Hash Filtering. The overall VLAN Filter Result is based on the perfect filter result and the Hash Filter result (if enabled). The filter result is passed to the application as part of the status bits.

Perfect Filtering is done based on the MAC_VLAN_Tag_Filter registers. For each VLAN Tag Filter, the MAC will compare the relevant VLAN Tag ID and gives a result. If any one of the VLAN Tag Filters gives a match then the frame is considered to have passed the VLAN Tag Filters. If the frame mismatches all the filters, then the frame is considered to have failed the VLAN filter. This behavior is applicable only when the Inverse Filtering is not enabled in MAC_VLAN_Tag_Ctrl Register.

If inverse Filtering is enabled and the frame has mis-matched all the relevant filters then it is considered to have passed the VLAN filter. If the frame matches any one of the relevant filters then it is considered as a fail. If none of the enabled filters can perform a comparison or if none of the filters are enabled, then the frame is bypassed to the application.

The overall filter result and the programming on the VTFE and RA bits of the MAC Filter Register determine if the frame will be dropped or forwarded to the application. If RA = 1 or VTFE = 0, it does not matter if the filter result is a pass or fail. The frame is always forwarded. If RA = 0 and VTFE = 1, only then, if the VLAN Tag Filter result is a pass does the MAC forward the frame. If the frame is forwarded to the application, then the relevant filter result is indicated through the Status bits.

6.5.3 Filter Status

The Extended Receive VLAN Filtering & Routing feature provides two status bits to indicate the comparison result of the VLAN tags.

By default, the MAC indicates the VLAN Filter Status through one bit in the status – VF in RDES2. When Extended RX VLAN Filtering & Routing is enabled, two status bits are used to indicate the comparison result of VLAN tags. The Outer VLAN Tag Filter Pass and Inner VLAN Tag Filter Pass bits are defined in the following positions in various configurations. The status indicated through these bits is highly dependent on the programming as explained below.

In RDES2:

- Bit 15 – Outer VLAN Tag Filter Status
- Bit 14 – Inner VLAN Tag Filter Status

In ARI status: MAC Filter Status:

- Bit 15 – Outer VLAN Tag Filter Status
- Bit 14 – Inner VLAN Tag Filter Status

In MRI Status:

- Bit 47 – Outer VLAN Tag Filter Status
- Bit 46 – Inner VLAN Tag Filter Status

Outer VLAN Tag Filter Status (OTS)

- In perfect Filtering, without inverse filtering enabled, if this bit is set, it indicates that the frame's Outer VLAN Tag has matched one of the VLAN Tag Filters.
- If this bit is reset, it indicates that the frame's Outer VLAN Tag has either failed the relevant Outer VLAN Tag Filters or bypassed them.
- If none of the filters are enabled for Outer VLAN Tag Comparison, then this bit is reset.
- If Inverse Filtering is enabled and this bit is set, then the frame's VLAN Tag has passed all the relevant VLAN Tag Filters. If it is reset, then it has failed at least one of or bypassed all the Filters programmed for Outer VLAN Tag Comparison.
- This bit is valid for both Single and Double VLAN Tagged frames.

Inner VLAN Tag Filter Status (ITS)

- In perfect Filtering, without inverse filtering enabled, if this bit is set, it indicates that the frame's Inner VLAN Tag has matched one of the VLAN Tag Filters.
- If this bit is reset, it indicates that the frame's Inner VLAN Tag has either failed the relevant Inner VLAN Tag Filters or bypassed them.
- If none of the filters are enabled for Inner VLAN Tag Comparison, then this bit is reset.
- If Inverse Filtering is enabled and this bit is set, then the frame's VLAN Tag has passed all the relevant VLAN Tag Filters. If it is reset, then it has failed at least one of or bypassed all the Filters programmed for Inner VLAN Tag Comparison.
- This bit is valid for only Double VLAN Tagged frames, when Double VLAN Processing is enabled.

The application must look at the status bits and the programming to determine if the Frame has passed or failed the VLAN Filter.

Table 6-5 and Table 6-6 show the possible Filter combinations and the corresponding filter results. These tables explain the scenarios when Double VLAN Processing and Hash VLAN filter are enabled in the design.

Legend for the Table 6-5

- VTIM: VLAN Tag Inverse Match Enable – bit 17 in VLAN_Tag_Ctrl Register.
- HFO: Hash Filter enabled for Outer VLAN Tag Comparison . bit 25 and bit 27 in VLAN_Tag_Ctrl Register.
- HFI: Hash Filter enabled for Inner VLAN Tag Comparison – bit 25 and bit 27 in VLAN_Tag_Ctrl Register.
- PFO – Perfect Filter comparison enabled for Outer VLAN Tag - Any of the MAC_VLAN_Tag_Filter Registers is enabled (bit 16 is set) and programmed for Outer VLAN Tag comparison (bit 20 is set to 0).
- PFI – Perfect Filter comparison enabled for Inner VLAN Tag - Any of the MAC_VLAN_Tag_Filter Registers is enabled (bit 16 is set) and programmed for Inner VLAN Tag comparison (bit 20 is set to 1).
- OTS – Outer VLAN Tag Filter Status
- ITS – Inner VLAN Tag Filter Status

Table 6-5 shows the possible values of status bits (OTS and ITS) when at least one Perfect filter is enabled.

Table 6-5 OTS and ITS Bit Values with At Least 1 Perfect Filter Enabled

VTIM	HFO	HFI	PFO	PFI	OTS	ITS
0	0	0	0	1	0	1/0
0	0	0	1	0	1/0	0
0	0	0	1	1	1/0	1/0
0	1	0	1	1	1/0	1/0
0	1	0	1	0	1/0	0
0	1	0	0	1	1/0	1/0
0	0	1	1	1	1/0	1/0
0	0	1	1	0	1/0	1/0
0	0	1	0	1	0	1/0
1	0	0	0	1	0	1/0
1	0	0	1	0	1/0	0
1	0	0	1	1	1/0	1/0
1	1	0	1	1	1/0	1/0
1	1	0	1	0	1/0	0
1	1	0	0	1	1/0	1/0
1	0	1	1	1	1/0	1/0
1	0	1	1	0	1/0	1/0

VTIM	HFO	HFI	PFO	PFI	OTS	ITS
1	0	1	0	1	0	1/0

Table 6-6 shows the possible values of status bits (OTS and ITS) when none of the perfect filters are enabled and only the VLAN Hash Filter is enabled.

Table 6-6 OTS and ITS Bit Values with Only VLAN Hash Filter Enabled

VTIM	HFO	HFI	OTS	ITS
0	0	0	0	0
0	1	0	1/0	0
0	0	1	1/0	1/0
1	0	0	1/0	0
1	1	0	1/0	0
1	0	1	1/0	1/0

With no perfect filters enabled, any VLAN packet is considered to have bypassed the perfect filter. If Hash Filter is enabled for one of the Tags, then the respective Status bit depends on the Filter's result. The Status bits are set to 0 if VLAN Hash Filter is also not enabled.

If the value of ITS/OTS is shown as 1/0; then it indicates that the final result is dependent on the enabled relevant filter's result.

Example 1: The second row of table 1-1 indicates that at least one Perfect Filter is enabled for Outer VLAN tag comparison and none of the filters are enabled for Inner VLAN tag comparison. Inverse VLAN Filtering is not enabled. The bit OTS is given as 1/0. If the received frame passes atleast one of the enabled Outer VLAN Tag filters then the bit is set to 1. If the frame doesn't pass any of the enabled Outer VLAN Tag filters, then the bit is set to 0.

Example 2: Last Row of Table 1-1 indicates that Inverse Filtering is enabled, Hash Filter and at least one perfect filter is enabled for Inner VLAN Tag comparison, then if the received frame's Inner VLAN tag mismatches with both the Hash Filter and all the enabled Perfect filters, then the frame will have the ITS bit set to 1 else it is set to 0. OTS will be set to 0 as no comparison is performed.

6.5.4 Stripping

Each of the VLAN Tags has individual control over stripping. The programming options of Always strip, never strip, strip on pass and strip on fail are available. Inner or Outer VLAN Tag Stripping is based on the pass or fail results of the individual tag. If a tag is bypassed by all the relevant filters, stripping is not applicable for the tag.

- If strip on Pass is enabled for the outer VLAN Tag, then the stripping occurs only if the Outer VLAN tag has passed the relevant Filters. The Outer VLAN Tag Filter Result bit will be set.
- If strip on Fail is enabled for the outer VLAN Tag, then stripping occurs only if the Outer VLAN Tag has failed relevant filters. The Outer VLAN Tag Filter Result Bit will be reset.
- If the Outer VLAN tag of the received frame is bypassed by the entire filter (no comparison has been made), then the tag is not stripped, though the Status Bit is still 0.

- As multiple filters are enabled, it is possible that the received VLAN frame could have matched any one or more of the filters. The VLAN Tag's value is not always deterministic from the filter status bits.
- If the application strips the VLAN Tag based on the filter result, it might lose the VID. So the suggested use is, if Stripping is enabled for any of the tags, the tag can be put in the status. For this the application will have to enable the respective "VLAN Tag in Status" bit - 24 or 31 in the MAC VLAN Tag Control Register.

6.5.5 Enabling Extended Receive VLAN Filtering

To enable the extended receive VLAN filtering feature using the coreConsultant GUI.

1. Select the **Extended Rx VLAN Filter Enable** option under the Filtering section during the **Specify Configuration** activity in coreConsultant.

   For details about this option, see Filtering Parameters in the "Parameter Descriptions" on page 419.

Note See the "Enabled" field of the parameter descriptions to understand the dependencies

The number of VLAN perfect filters available depends on the value of the Number of VLAN Tag Filters (DWC_EQOS_NRVF) parameter. Using this feature,

- The application can enable a minimum of 4 and a maximum of 32 VLAN Tag filters.
- Every VLAN tag filter can be controlled by programming, to check the type and tag value of the received VLAN frame.
- The MAC can route the received VLAN traffic, based on the VLAN filter result and the enabled DMA channel number.

6.5.6 Programming Guidelines for Extended VLAN Filtering and Routing on Receive

See "Programming Guidelines for Extended VLAN Filtering and Routing on Receive" on page 1379

6.6 Layer 3 and Layer 4 Filtering

6.6.1 Introduction to Layer 3 and Layer 4 Filtering

The DWC_ether_qos supports Layer 3 and Layer 4 based packet filtering. The Layer 3 filtering refers to the IP Source or Destination Address filtering in the IPv4 or IPv6 packets whereas Layer 4 filtering refers to the Source or Destination Port number filtering in TCP or UDP.

6.6.2 Description of Layer 3 and Layer 4 Filtering

When Layer 3 and Layer 4 filtering is enabled, the packets are filtered in the following way:

- **Matched Packets**: The MAC forwards the packets that match all enabled fields to the application along with the status. The MAC gives the matched field status only if the IPC bit of MAC_Configuration register is set and one of the following conditions is true:
  - All enabled Layer 3 and Layer 4 fields match
  - At least one of the enabled field matches and other fields are bypassed or disabled

  When multiple Layer 3 and Layer 4 filters are enabled, any filter match is considered as a match. If more than one filter matches, the MAC provides the status of the lowest filter where Filter 0 is the lowest filter and Filter 3 is the highest filter. For example, if Filter 0 and Filter 1 match, the MAC gives the status corresponding to filter 0.

Note The source or destination address and VLAN tag filters (if enabled) have precedence over Layer 3 and Layer 4 filter. This means that a packet which fails the source or destination address or VLAN tag filter is dropped irrespective of the Layer 3 and Layer 4 filter results.

- **Unmatched Packets**: The MAC drops the packets that do not match any of the enabled fields. You can use the inverse match feature to block or drop a packet with specific TCP or UDP over IP fields and forward all other packets.

  When a packet is dropped, a partial packet with appropriate abort status may be received on MRI in the EQOS-CORE configuration. This is because there is no buffer in the MAC receiver for storing the data until the TCP/IP fields are received and matched. However, in the EQOS-AHB, EQOS-AXI, EQOS-DMA, and EQOS-AHB configurations, the aborted or partial packets can be dropped in the MTL Rx FIFO. If the Rx FIFO operates in the Threshold (cut-through) mode and the threshold is programmed to a small value, such that packet transfer to application starts before the failed Layer 3 and Layer 4 filter results are available, the application may receive a partial packet with appropriate abort status.
- **Non-TCP or UDP IP Packets**: By default, all non-TCP or UDP IP packets are bypassed from the Layer 3 and Layer 4 filters. You can optionally program the MAC to drop all non-TCP or UDP over IP packets.

6.6.2.1 Layer 3 Filtering

The DWC_ether_qos supports perfect matching or inverse matching for IP Source Address and Destination Address. In addition, you can match the complete IP address or mask the lower bits matching, that is, compare all bits of the address except the specified lower mask bits.

For IPv6 packets filtering, you can enable the last four data registers of a register set to contain the 128-bit IP Source Address or IP Destination Address. The IP Source Address or Destination Address should be programmed in the order defined in the IPv6 specification, that is, the first byte of the IP Source Address or Destination Address in the received packet is in the higher byte of the register and the subsequent registers follow the same order.

For IPv4 packet filtering, you can enable the second and third data registers of a register set to contain the 32-bit IP Source Address and IP Destination Address. The remaining two data registers are reserved. The IP Source Address or Destination Address should be programmed in the order defined in the IPv4 specification, that is, the first byte of IP Source Address and Destination Address in the received packet in the higher byte of the respective register.

6.6.2.2 Layer 4 Filtering

The DWC_ether_qos supports perfect matching or inverse matching for TCP or UDP Source and Destination Port numbers. However, you can program only one type (TCP or UDP) at a time. The first data register contains the 16-bit Source and Destination Port numbers of TCP or UDP, that is, the lower 16 bits for Source Port number and higher 16 bits for Destination Port number.

The TCP or UDP Source and Destination Port numbers should be programmed in the order defined in the TCP or UDP specification, that is, the first byte of TCP or UDP Source and Destination Port number in the received packet is in the higher byte of the register.

6.6.3 Layer 3 and Layer 4 Filters Registers

The MAC implements a set of registers for Layer 3 and Layer 4 based packet filtering. In a register set, there is a control register, such as MAC_L3_L4_Control(#i) (for i = 0; i <= DWC_EQOS_L3_L4_FILTER_NUM-1), to control the packet filtering. In addition, there are five address registers to program the Layer 3 and Layer 4 fields to be matched, such as:

- MAC_Layer4_Address(#i) (for i = 0; i <= DWC_EQOS_L3_L4_FILTER_NUM-1)
- MAC_Layer3_Address0_Reg(#i) (for i = 0; i <= DWC_EQOS_L3_L4_FILTER_NUM-1)
- MAC_Layer3_Address1_Reg(#i) (for i = 0; i <= DWC_EQOS_L3_L4_FILTER_NUM-1)
- MAC_Layer3_Address2_Reg(#i) (for i = 0; i <= DWC_EQOS_L3_L4_FILTER_NUM-1)
- MAC_Layer3_Address3_Reg(#i) (for i = 0; i <= DWC_EQOS_L3_L4_FILTER_NUM-1)

You can configure the MAC to have up to four such independent set of registers. You can also synchronize the address registers to the Rx clock domain by selecting the Synchronize Layer 3 and Layer 4 Address Registers to Rx Clock Domain option in coreConsultant.

6.6.4 Enabling Layer 3 and Layer 4 Filtering

To enable Layer 3 and Layer 4 filtering while configuring the controller in the coreConsultant GUI:

1. **Select the Enable Layer 3 and Layer 4 Packet Filter** option under the Filtering section during the **Specify Configuration** activity in coreConsultant.

   For details about this option, see Filtering Parameters in the "Parameter Descriptions" on page 419.

Note See the "Enabled" field of the parameter descriptions to understand the dependencies

You can enable this feature by selecting the **Enable Layer 3 and Layer 4 Packet Filter** option in core-Consultant. The Layer 3 and Layer 4 packet filtering feature automatically enables the IPC Full Checksum Offload Engine on the Receive side. For Layer 3 or Layer 4 filtering operation, you must set the IPC bit of the MAC_Configuration register to enable the Rx Checksum Offload Engine.

6.7 Flexible Receive Parser

6.7.1 Overview of Flexible Receive Parser

Flexible Rx Parser feature is supported in EQOS-DMA, EQOS-AHB, and EQOS-AXI configurations when SPRAM option is selected (DWC_EQOS_FRP_EN = 1) and number of RxDMA channels > 1. When this feature is enabled, all incoming packets are parsed as per the programmable instructions in the memory.

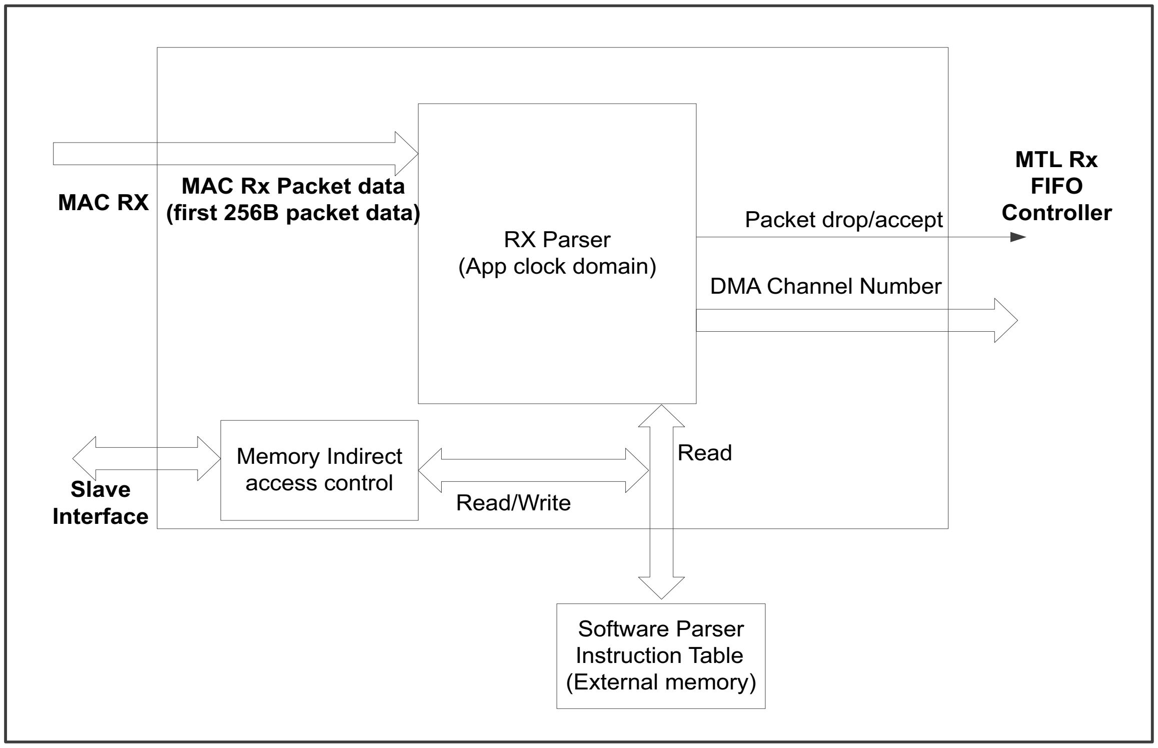
6.7.2 Description of Flexible Receive Parser

The Flexible Receive Parser operates over the first 256 bytes, 128 bytes, or 64 bytes of data as defined by DWC_EQOS_FRP_BUF_SIZE. The parser can process up to 256 bytes, 128 bytes, or 64 bytes (DWC_EQOS_FRP_ENTRIES) of instructions on each packet.

The parser operates after the MAC receiver and before the MTL receiver FIFO controller.

Figure 6-3 shows the functional block diagram of Flexible Receive Parser.

Figure 6-2 Functional Block Diagram of Flexible Receive Parser



6.7.2.1 Instruction Table Format of the Flexible Receive Parser

Before you enable the Flexible Receive Parser, ensure that the software has built the 128-bit wide instruction table for the parser. Each instruction is programmed as an entry in the Instruction memory. The format of each entry in the Instruction table is shown in Table 6-7.

Table 6-7 Rx Parser Instruction Entry Format

Field	Field Name	Description
31:0	MATCH_DATA	The 4-byte data. This data is used for comparing with incoming packet data starts at the frame offset as defined in [79:72] of the entry. The comparison is done only on those bits whose corresponding mask bits are set to 1 (MATCH_EN bits of [63:32]). See the Notes (below this table) on the byte order relative to Ethernet arrival data and endianness supported by QoS IP on the slave interface. The programming of this instruction table should adhere to the same.
62:32	MATCH_EN	When the MATCH_EN is set to 1, the corresponding Packet data bit is used for comparing. Otherwise, corresponding data bit is don't care.

64	AF	AF = Accept frame,
		RF = Reject Frame,
		IM = Inverse match,
		NIC= Next Instruction Control
		match = MATCH_DATA & MATCH_EN Bits ==
		        (Packet data[frame offset*8 +: 8*4]) & MATCH_EN bits
		If IM=0
		  ENTRY_MATCH = match
		If IM=1
		  ENTRY_MATCH = !match
		If ENTRY_MATCH ==1
		        If AF = 1,
		            then put frame into DMA Chanel (as indicated in
		                DMA CH Number, the [95:88] of this table)
		        else if RF = 1,
		            then reject/drop frame
		        else if !(AF|RF)
		          if(NIC==0)
		              continue parse from OK Index
		          else // (NIC ==1)
		              continue sequentially (parse the next entry in the instruction table)
		
		If ENTRY_MATCH ==0
		          if(NIC==1)
		              continue parse from OK Index
		          else //(NIC ==0)
		              continue sequentially (parse the next entry in the instruction table)
		Note: AF and RF should not be programmed together. However, if both AF and RF are programmed together, AF gets the priority in case of ENTRY_MATCH

65	RF	
66	IM	
67	NC	
71:68	Reserved	

Field	Field Name	Description
79:72	Frame Offset	[77:72] indicates the frame offset in terms of 4 bytes. (Here [79:78] always 0) Frame offset in the packet data to be compared for Match. This is in terms of 4 bytes. Value Actual Frame offset in bytes 0 0 1 4 2 8 ….. 63 252 The max value programmed in this Instruction Table should adhere to DWC_EQOS_FRP_BUF_SIZE parameter. The 'Actual Frame Offset' should not cross this limit.

87:80	OK Index	If(NIC==0)
		Memory Index to be used next when ENTRY_MATCH==1 and neither AF or RF is set to 1
		
		If(NIC==1)
		Memory Index to be used next when ENTRY_MATCH==0

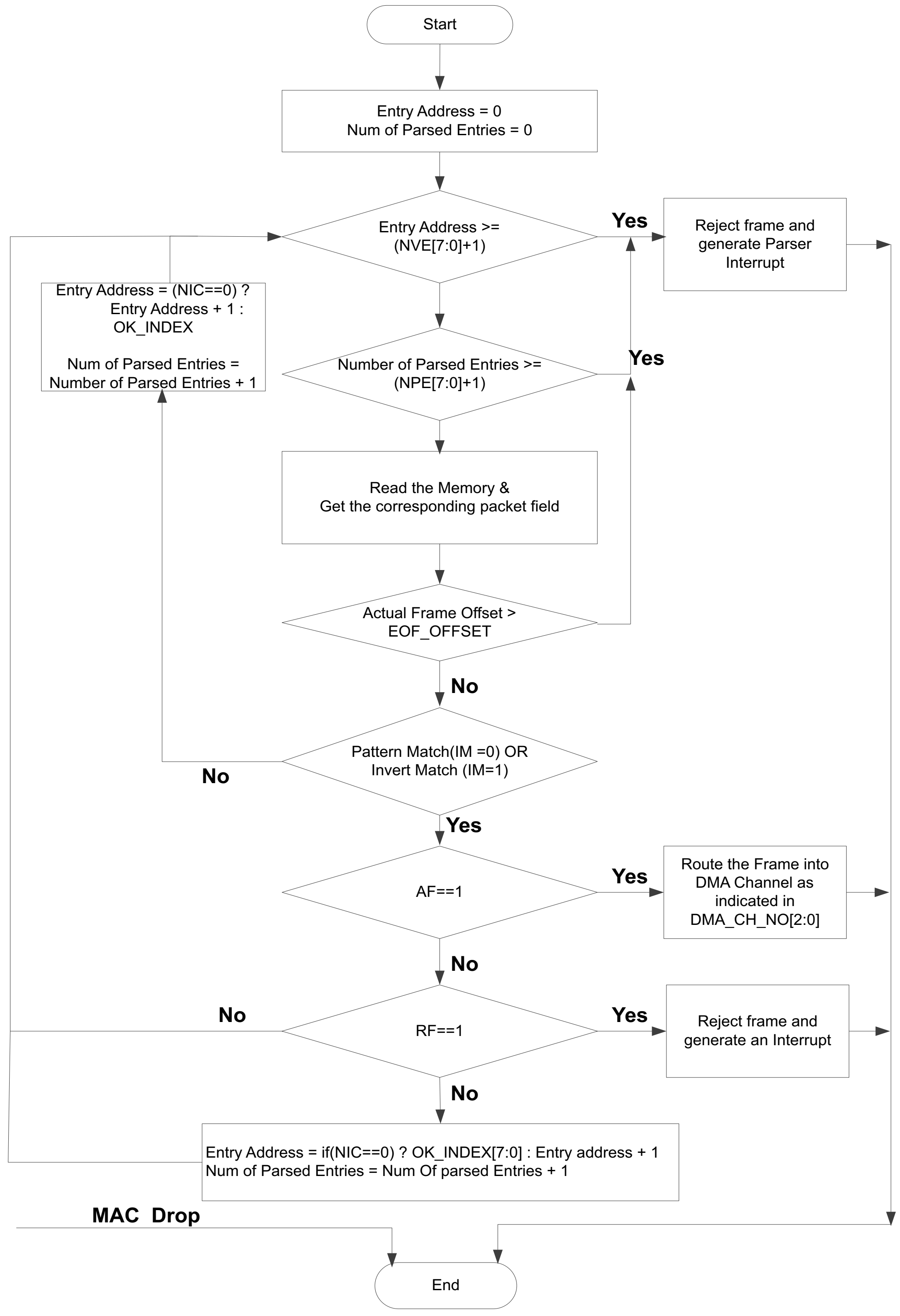
95:88	DMA CH NO	Indicate the DMA Channel Number (1-bit for each). This will be used when ENTRY_MATCH ==1 and AF = 1 (Accept frame) The encoding is given by the following. bit[0]- DMA Channel Number 0 bit[1]- DMA Channel Number 1 bit[2]- DMA Channel Number 2 … bit[7]- DMA Channel Number 7 **Note:** The DMA Channel number is encoded using bit wise as QoS IP support multicasting/broadcast across DMA Channels. Refer section 3.3.12 for more details and proper usage.
128:96	Reserved	0 (This field is only for software view and reserved for future enhancements. The memory width remains as 96-bits for now)

Note

- **Memory Width**: 96 and **Memory Depth**: Configurable - 256, 128, or 64.
- **Byte Ordering**: The first byte received on the line is compared with MATCH_DATA[7:0] of the instruction table.

The parser begins parsing from the 0th entry, for each received packet. The subsequent parsing entry location (next entry or OK_INDEX[]) depends on the parser result of the current entry. Figure 6-3 shows the detailed flow.

Figure 6-3 Flexible Receive Parser Flow Chart


Start
Entry Address = 0
Num of Parsed Entries = 0
Entry Address >= (NVE[7:0]+1)
Yes
Reject frame and generate Parser Interrupt
Entry Address = (NIC==0) ? Entry Address + 1 : OK_INDEX
Num of Parsed Entries = Number of Parsed Entries + 1
Number of Parsed Entries >= (NPE[7:0]+1)
Yes
Read the Memory & Get the corresponding packet field
Actual Frame Offset > EOF_OFFSET
No
Pattern Match(IM =0) OR Invert Match (IM=1)
No
Yes
AF==1
Yes
Route the Frame into DMA Channel as indicated in DMA_CH_NO[2:0]
No
RF==1
No
Yes
Reject frame and generate an Interrupt
No
Entry Address = if(NIC==0) ? OK_INDEX[7:0] : Entry address + 1
Num of Parsed Entries = Num Of parsed Entries + 1
MAC Drop
End

The EOF_OFFSET in Figure 6-3 is equal to the least among the following:

- Actual Packet size
- DWC_EQOS_FRP_BUF_SIZE, when frame pre-emption feature is not enabled/configured
- DWC_EQOS_FRP_BUF_SIZE, when frame pre-emption feature is enabled and packet is pre-emptible.
- DWC_EQOS_FRM_BUF_SIZE/2, when frame pre-emption feature is enabled and packet is Express Packet

6.7.2.2 Number of Valid Entries (NVE)

Indicates the number of valid entries in the instruction table. The default value is set to DWC_EQOS_FRP_ENTRIES. Software can override this value by writing into MTL_RXP_Control_Status registers field NVE[7:0].

While parsing, if the Entry address is found to be is greater than NVE[7:0]+1, an error is flagged in the MTL_RXP_Interrupt_Control_Status register bit[0], NVEOVIS (Number of Valid Entries overflow status). DWC_ether_qos generates an interrupt status (MTL_Interrupt_Status, bit[23], MTLPIS (MTL Parser Interrupt Status).

6.7.2.3 Number of Parsable Entries (NPE)

Indicates the number of entries that can be parsed on an incoming packet. The default value is set to DWC_EQOS_FRP_ENTRIES. Software can override this value by writing into MTL_RXP_Control_Status registers field NPE[7:0] (Number of Parsable Entries) bits [23:16].

While parsing, if the number of parsed entries is more than NPE[7:0]+1, an error is flagged in the MTL_RXP_Interrupt_Control_Status register bit[1], NPEOVIS (Number of Parsable Entries overflow status). DWC_ether_qos generates an interrupt status (MTL_Interrupt_Status, bit[23], MTLPIS (MTL Parser Interrupt Status).

While NVE[] indicates is the number of valid entries in the Rx parser memory built by software, NPE[] indicates the worst-case parsable entries considering the various possible paths that the parser can take. (NVE[] >= NPE[]).

6.7.2.4 Frame Offset

Indicates the frame offset, in terms of 4 bytes; used while comparing against the current table entry.

In an entry, an error is flagged in the MTL_RXP_Interrupt_Control_Status register bit[2], FOOVIS (Frame Offset Overflow Interrupt Status) in any of the following cases:

When Frame Offset is more than

- Packet size
- DWC_EQOS_FRP_BUF_SIZE if pre-emption is disabled
- DWC_EQOS_FRP_BUF_SIZE if pre-emption is enabled and packet is pre-empitble
- DWC_EQOS_FRP_BUF_SIZE/2 if pre-emption is enabled and packet is an Express packet

DWC_ether_qos generates an interrupt status (MTL_Interrupt_Status bit[23]) and MTLPIS (MTL Parser Interrupt Status).

Note If the frame ends after the (frame_offset *4) bytes but before ((frame_offset *4)+4) bytes, the Receive parser declares Frame Offset Error if comparison is enabled (via MASK bits) for the non-received bytes; from (frame_offset *4) to ((frame_offset *4)+4).

6.7.2.5 Frame Reject

In cases where an Entry matches (considering the Inverse match, IM bit in the instruction table) and Frame Accept bit is not set, and Frame Reject bit (RF) is set, the frame is dropped.

When a frame is dropped due to RF = 1, DWC_ether_qos generates an interrupt status (MTL_Interrupt_Status, bit[23], MTLPIS (MTL Parser Interrupt Status).

6.7.2.6 Out of Order Processing

The Rx parser flow can be out of order. The decision tree need not be based on the order in which packet arrives. This implies that the next entry and next frame offset (OK_INDEX[]/FRAME_OFFSET[]) can be less than the current entry address and current Frame Offset.

For example, one can look for matching/mismatching IP header and then look for Ethernet Header to decide on Filtering/Channel Selection. Also, it can skip from higher address entry to lower address entry in the table.

6.7.2.7 Ethernet Line Speed Dependency

To enable the maximum parsing ability the Rx parser block uses the application clock domain. It is assumed that the application clock frequency is higher enough than the MAC line rate clock frequency.

Example: 1G Ethernet (GMII working at MHz clock) and @400MHz application clock

Worst Case Processing Rate: 64B back-to-back (with 8B preamble and 12B IPG). A total of 84 cycles in 125MHz(8ns) clock. So, the requirement is one packet to be processed every 84 cycles of 8ns clock (672ns)

Number of Application Cycles Available at 400MHz: 672ns/2.5ns = ~268 cycles

Similarly for 2.5G Ethernet (GMII working @312.5MHz (3.2ns) clock) the number of available cycles @400MHz application clock is ~107 cycles.

Table 6-8 Line Rate Versus Clock cycles Available for Minimum Packet Size

Line Rate	Number of Cycles (@400MHz) Available to Process Minimum Packet Size
1G	268 cycles (~4 clocks overhead)
2.5G	107 cycles (~4 clocks overhead)

So, the number of available cycles is limited by Ethernet line rate. As the line rate increases the number of available cycles to parse reduces.

Note **Limitations:**

- As the line rate increases the ability to parse the number of entries reduces for a given application clock frequency.
- As the application clock frequency increase the ability to parse number of entries also increases for a given link speed.

When the link speed changes its assumed that software does not program the number of valid entries beyond the intended limit as per the calculations provided below. If programmed incorrectly, the Rx Parser drops the packet especially for short packets (This drops will be treated as NPE[] overflow).

In general, the number of available cycles = (64 + 8 + 12) * (GMII clock period in ns)/Application clock period in nanoseconds.

Note Consider 4 cycles as overhead. This is for the initial memory read latency and other overheads.

If there are too many JUMP instructions in the Instruction Table, the parsing ability of a minimum sized packet gets further reduced. So, the ability to parse might change based on the number of JUMP instructions that an incoming packet encounters. This is because every JUMP instruction adds 3 cycles of memory read latency. The following table shows the best case and worst case number of entries that Rx Parser can parse, when the application clock is 400MHz.

Table 6-9 Line Rate Versus Clock cycles Available for Minimum Packet Size

Line Rate	Best Case - 400MHz Application Clock with Back-to-Back 64byte Packet	Worst Case - 400MHz Application Clock with Back-to-Back 64byte Packet	Comments
1G	268 cycles (256 is the maximum entries in the table with ~4 clock cycle of overhead)	89 cycles (with ~4 clock cycle of overhead)	The programmed entries should adhere to this limit. While programming, the software should evaluate the number of possible jump instructions and limit the termination condition accordingly. If programmed incorrectly, the Rx Parser drops the packet (due to NPE[] overflow) especially for short packets
2.5G	107 cycles (~4 clock cycle of overhead)	35 cycles (with ~4 clock cycle of overhead)	

Number of available cycles = ((64 + 8 + 12) * (GMII clock period in ns))/Application clock period in ns.

Number of entries that can be processed (Best Case) = (Number of available cycles) - 4

Number of entries that can be processed (Worst Case) = (Number of available cycles/3) - 4

Note If packet parsing is not complete even after reaching the immediate next packet's end of the packet (EOP), the packet is dropped and considered as NPE[] overflow.

6.7.2.8 DMA Channel Selection

The MAC/MTL also determines the DMA Channel number based on incoming packet fields.

When the Rx parser is enabled (FRPE bit[16] is set to 1 in MTL_Operation_Mode register) the Rx parser overrides the DMA selection criteria of the MAC.

When the Rx parser is disabled (FRPE bit[16] is set to 0 in MTL_Operation_Mode register) the MAC/MTL determines the DMA channel number.

6.7.2.9 Receive Queue Selection

The Rx Queue is determined by the existing MAC functionality, irrespective of the enabling/disabling of the Rx Parser. This is because the Rx parser is mainly designed for selecting the DMA channel and Rx queuing is decided by the MAC to enable the proper functioning of the Pause/PFC feature.

Note When there are multiple queues and multiple DMA Channels are enabled, a particular DMA channel might receive out of order packets. This is because queue selection is based on MAC criteria (VLAN Priority and other criteria) and Rx DMA channel selection is based on Rx parser. If in-order packet reception is necessary, you must program Single Queue configuration.

6.7.2.10 MAC Packet Filtering/Drop/Error Handling

The MAC does the packet filtering based on the received packet fields. Disable the Packet Filtering features inside the MAC when the Rx parser is enabled. To disable the Packet Filtering features inside the MAC,

1. Set Promiscuous Mode (PR, bit[0] of MAC_Packet_Filter register)
2. Set all other bits in the MAC_Packet_Filter register to its default values

When MAC decides to drop the packet (due to Rx MAC dependent Error) the packet is dropped irrespective of the Rx parser decision. The MAC issues the following error:

- GMII Error
- Receive Watch Dog Error
- CRC Error
- Giant Frame Error

6.7.2.11 PAD Strip/CRC Strip Handling

The pad strip and CRC strip are controlled by the software and applicable to all the received packets. When these features are enabled, software must build the Rx parser instructions accordingly.

Note Enable PAD and CRC stripping in real use case.

6.7.2.12 VLAN Strip Handling

The MAC supports stripping the VLANs, outer as well as inner VLAN. When MAC is programmed to strip the VLAN, the Rx parser considers those VLAN tags to be part of the incoming packet.

6.7.2.13 Multicast and Broadcast Support

DWC_ether_qos supports multicast and broadcast only when PDC (packet duplication control) is enabled and the packet is routed to the highest queue number.

When Rx Parser is enabled, the packet routing is decided by the DMA Channel Numbers (DMA CH NO, Bits[95:88]) in the Instruction Table.

When Rx Parser is disabled, the packet routing is determined by DCS (DMA Channel Select) field of the MAC_AddressX_High register. For more details, see "Broadcast/Multicast Packet Duplication" on page 290.

Note DCS/DMA CH NO field is per DMA Channel control and therefore, multiple DMA channels can be selected for routing the packet.

6.7.2.14 Pre-emption Support

The Rx parser supports 256 bytes, 128 bytes, or 64 bytes configurable parsing on all the incoming packets. So, it has 2x256 bytes, 2x128 bytes or 64 bytes storage. This is because, while parsing the current packet, the next packet needs to be stored.

When pre-emption is enabled, the pre-emptible packets might get fragmented just after 64 bytes in the middle of the parsing and the remaining data is expected to arrive after Express packet gets over; also, multiple back to back Express packets need to be processed till then. So, the context of the pre-emptible parser and associated data buffers need to be maintained while DWC_ether_qos is still able to process back to back Express packets. This demands another 256 bytes, 128, or 64 bytes storage. However, by keeping the total storage limit to 2x256 bytes, 2x128 bytes, or 2x64 bytes. Following limitations apply to the parsable number of bytes for the Express Traffic.

Table 6-10 Limitations to the Parsable Number of Bytes in Express Traffic

DWC_EQOS_FRP_BUF_SIZE	Number of Bytes the Rx Parser Can Operate On the Express Traffic	Number of Bytes the Rx Parser Can Operate On the Pre-emptible Traffic
256	128	256
128	64	128
64	32	64

The reduction of the parsing ability of the Express packet is done considering packet parameters that are well known in advance (via stream reservation protocol). The packets are generated and consumed within the LAN, and does not need deep frame parsing.

Note The Express packet queue is always different from the pre-emptible queue.

6.7.2.15 Software Access to the Flexible Receive Parser Memory

Software can read/write into the Rx parser memory, using the following registers via indirect addressing.

- MTL_RXP_Indirect_Acc_Control_Status
- MTL_RXP_Indirect_Acc_data

DWC_ether_qos collects 4x32 bit data (parser table width) before initiating the writes. It also collects 4x32-bit (parser table width) data in advance before responding to the first read.

For more details, see "Register Descriptions" on page 605.

6.7.2.16 Statistical Counters

Following counters are available when flexible receive parser is enabled.

1. MTL_RXP_Drop_Cnt

   This counter is incremented when the Rx parser drops a packet due to RF = 1.

   For more details, see "Register Descriptions" on page 605.
2. DMA_RXP_Error_Cnt

   This counter is incremented when Rx parser encounters an error. The error scenarios are:

   - Entry address >= NVE[]
   - Number of parsed entries >= NPE[]
   - Frame offset in bytes > EOF/DWC_EQOS_FRP_BUF_SIZE(for pre-emptible packet) / (DWC_EQOS_FRP_BUF_SIZE/2) (for Express packet) data entry address
   - Parsing is not complete even after receiving next packet's EOP

   For more details, see "Register Descriptions" on page 605.
3. DMA_CH(#i)_Rxp_Accept_cnt (i=0; i<DWC_EQOS_NUM_DMA_RX_CH)

   This is a per DMA channel counter. This counter is incremented when Rx parser accepts a packet due to AF = 1.

   For more details, see "Register Descriptions" on page 605.

6.7.2.17 Changing the Instruction Table by Software

Software can update the instruction table in the memory in the following ways

- Disable the MAC Receiver and Receive Parser by setting the RE field of the MAC_Configuration register to 0 and FRPE bit of MTL_Operation_mode register to 0. Wait for Rx Parser to become inactive (RXPIA, MTL_RXP_Control_Status register bit[31])
- (Optional) Program in the Entry address 0 to unconditionally (all MATCH_EN bit 0) skip to OK_INDEX[] (to certain location in the memory) where it is programmed to reject or accept all packets.

6.7.2.18 Receive Cut-Through Functionality

In the case of cut-through mode, the Rx controller attempts to transfer the received packets to DMA just after the cut-through threshold amount of packet data is received (RTC field in the MTL_RxQx_Operation_Mode register. The RTC can take values 16, 32, 64, and 128).

However due to Rx parser's result worst case arrival time can be more than the programmed cut-through threshold. So, the Rx controller delays the packet transfer to DMA accordingly. So, the cut-through functionality is re-defined as follows

The Rx controller attempts to transfer packet to DMA after at least RTC number of bytes have been received and Rx parser results are available.

6.7.2.19 Receive Packet Drop Indication

In most cases, the packet drops internally in the Rx Queue. However, back-to-back packets can arrive for the same queue and first packet's Rx parser result is not available when the second packet arrives. In such cases, the packet cannot be flushed internally and forwarded to DMA or the Application Interface and status is indicated accordingly. For more details, see RDES2, bit 16(RXPD) in "Receive Normal Descriptor (Write-Back Format)" on page 1340.

6.7.2.20 Runt Packet Handling

The Rx parser is designed for worst case minimum packet size to be of 64B.

If the runt packet is received (should be indicated by MAC), the Rx parser assumes that it gets dropped in the MAC.

This is because, back-to-back runt packets (< 64B) cannot be handled in the Rx parser.

6.7.2.21 Uncorrectable ECC Error Handling

When the parser detects an uncorrectable ECC error while parsing the parser memory, the packet is not dropped. Instead, the packet is sent to the application with an Error Indication. The RXPI (RX Parser Incomplete) is set in the packet status along with Error Summary (ES) bit. For more details, see "Receive Normal Descriptor (Write-Back Format)" on page 1340.

6.7.3 Enabling/Disabling the Flexible Receive Parser

To enable the Flexible Receive Parser, set bit[16] (FRPE, Flexible Rx Parser Enable) to 1, in the MTL_Operation_Mode register.

You must disable the MAC receiver (block the traffic on the receiver) before enabling or disabling the flexible receive parser. To disable the MAC Receiver, set the RE (bit[0]) bit of MAC_Configuration Register to 0.

For details about these options, see **Filtering Parameters** in "Parameter Descriptions" on page 419.

Note See the "Enabled" field of the parameter descriptions to understand the dependencies.

6.7.4 Signals Related to Flexible Receive Parser

- rxp_addr_o
- rxp_wdata_o
- rxp_wdata_ecc_o
- rxp_csn_o
- rxp_we_o
- rxp_rdata_i
- rxp_rdata_ecc_i
- rxp_oe_o

For a description of signals related to Flexible Receiver Parser, see "Signal Descriptions" on page 471.

6.7.5 Registers Related to Flexible Receive Parser

- MTL_RXP_Interrupt_Control_Status
- MTL_RXP_Control_Status
- MTL_RXP_Drop_Cnt
- MTL_RXP_Error_Cnt
- MTL_RXP_Indirect_Acc_Control_Status
- MTL_RXP_Indirect_Acc_Data
- MTL_Operation_Mode
- MTL_Interrupt_Status
- DMA_CH_RXP_Accept_Cnt

For more details, see the "Register Descriptions" on page 605.

6.7.6 Programming Guidelines for Flexible Receive Parser

The programming guidelines is mentioned in the respective sections of the feature.

7 IEEE 1588 Timestamp Support

The IEEE 1588 defines a Precision Time Protocol (PTP) which enables precise synchronization of time in measurement and control systems. This protocol enables heterogeneous systems that include clocks of varying inherent precision, resolution, and stability to synchronize. The protocol supports system-wide synchronization accuracy in the sub-microsecond range with minimal network and local clock computing resources.

This chapter contains the following sections:

7.1 IEEE 1588 Timestamp Support

7.1.1 Introduction to IEEE 1588 Timestamp Support

The DWC_ether_qos supports the IEEE 1588-2002 (version 1) and IEEE 1588-2008 (version 2). The IEEE 1588-2002 supports PTP transported over UDP/IP and IEEE 1588-2008 supports PTP transported over Ethernet. The DWC_ether_qos provides programmable support for both standards.

The controller supports the following features:

- Provides an option to take snapshot of all packets or only PTP type packets
- Provides an option to take snapshot of only event messages
- Provides an option to take the snapshot based on the clock type: ordinary, boundary, end-to-end transparent, and peer-to-peer transparent
- Provides an option to select the node to be a master or slave for ordinary and boundary clock
- Identifies the PTP message type, version, and PTP payload in packets sent directly over Ethernet and sends the status
- Provides an option to measure sub-second time in digital or binary format

7.1.2 Description of IEEE 1588 Timestamp Support

7.1.2.1 Clock Types

DWC_ether_qos supports different clock types defined in IEEE 1588-2008.

The DWC_ether_qos supports the following clock types:

- Ordinary Clock
- Boundary Clock
- End-to-End Transparent Clock
- Peer-to-Peer Transparent Clock

7.1.2.1.1 Ordinary Clock

The ordinary clock has a single PTP state and a single physical port. In a domain, an ordinary clock supports a single copy of the protocol.

The ordinary clock in a domain supports a single copy of the protocol. The ordinary clock has a single PTP state and a single physical port. In typical industrial automation applications, an ordinary clock is associated with an application device such as a sensor or an actuator. In telecom applications, the ordinary clock can be associated with a timing demarcation device.

The ordinary clock can be a grandmaster or a slave clock. It supports the following features:

- Sends and receives PTP messages. The timestamp snapshot can be controlled as described in MAC_-Timestamp_Control.
- Maintains the data sets such as timestamp values

Table 7-1 shows the messages for which you can take the timestamp snapshot on the receive side for master and slave nodes.

Table 7-1 Ordinary Clock: PTP Messages for Snapshot

Master	Slave
Delay_Req	SYNC

For an ordinary clock, you can take the snapshot of either of the following PTP message types: version 1 or version 2. You cannot take the snapshots for both PTP message types. You can take the snapshot by setting the TSVER2ENA bit and selecting the snapshot mode in MAC_Timestamp_Control.

7.1.2.1.2 Boundary Clock

The boundary clock typically has several physical ports communicating with the network. The messages related to synchronization, master-slave hierarchy, and signaling terminate in the protocol engine of the boundary clock and such messages are not forwarded. The PTP message type status given by the MAC helps you to identify the type of message and take appropriate action.

The boundary clock is similar to the ordinary clock except for the following features:

- The clock data sets are common to all ports of the boundary clock.
- The local clock is common to all ports of the boundary clock.

7.1.2.1.3 End-to-End Transparent Clock

The end-to-end transparent clock supports the end-to-end delay measurement mechanism between the slave clocks and the master clock. The end-to-end transparent clock forwards all messages like normal bridge, router, or repeater. The residence time of a PTP packet is the time taken by the PTP packet from the Ingress port to the Egress port.

The residence time of a SYNC packet inside the end-to-end transparent clock is updated in the correction field of the associated Follow_Up PTP packet before it is transmitted. Similarly, the residence time of a Delay_Req packet, inside the end-to-end transparent clock, is updated in the correction field of the associated Delay_Resp PTP packet before it is transmitted. Therefore, the snapshot must be taken at both Ingress and Egress ports only for the messages mentioned in Table 7-2. You can take the snapshot by setting the SNAPTYPSEL bits to 10 in the MAC_Timestamp_Control register.

Table 7-2 End to End Transparent Clock: PTP Messages for Snapshot

PTP Messages
SYNC
Delay_Req

7.1.2.1.4 Peer-to-Peer Transparent Clock

In the peer-to-peer transparent clock, the computation of the link delay is based on an exchange of Pdelay_Req, Pdelay_Resp, and Pdelay_Resp_Follow_Up messages with the link peer.

The peer-to-peer transparent clock differs from the end-to-end transparent clock in the way it corrects and handles the PTP timing messages. In all other aspects, it is identical to the end-to-end transparent clock.

In the peer-to-peer transparent clock, the computation of the link delay is based on an exchange of Pdelay_Req, Pdelay_Resp, and Pdelay_Resp_Follow_Up messages with the link peer. The residence time of the Pdelay_Req and the associated Pdelay_Resp packets is added and inserted into the correction field of the associated Pdely_Resp_Followup packet. Therefore, support for taking snapshot for the event messages related to Pdelay is added as shown in Table 7-3.

Table 7-3 Peer-to-Peer Transparent Clock: PTP Messages for Snapshot

PTP Messages
SYNC
Pdelay_req
Pdelay_Resp

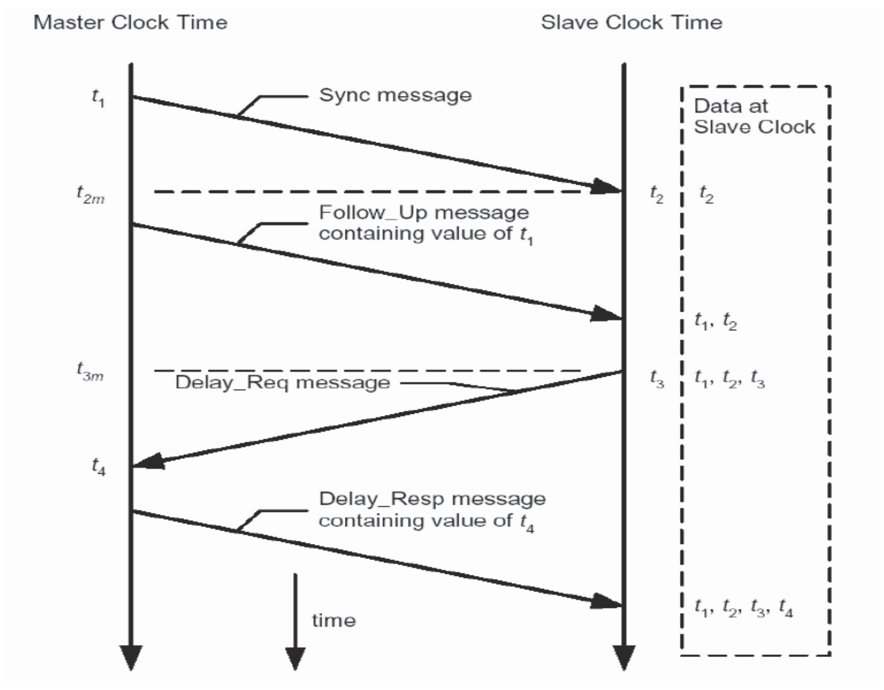
You can take the snapshot by setting the SNAPTYPESEL bit to 11 in MAC_Timestamp_Control register.

7.1.2.2 Delay Request-Response Mechanism

The system or network is classified into the master and slave nodes for distributing the timing and clock information.

The system or network is classified into the master and slave nodes for distributing the timing and clock information. Figure 7-1 shows the process that PTP uses for synchronizing a slave node to a master node by exchanging PTP messages.

Figure 7-1 Networked Time Synchronization



As shown in Figure 7-1, PTP uses the following process:

1. The master broadcasts the PTP Sync messages to all its nodes. The Sync message contains the reference time information of the master. This message leaves the system of the master at t1. This time must be captured for Ethernet ports at GMII or MII.
2. The slave receives the Sync message and also captures the exact time, t2, using its timing reference.
3. The master sends a Follow_up message to the slave, which contains t1 information for later use.
4. The slave sends a Delay_Req message to the master and notes the exact time, t3, at which this packet leaves the GMII or MII interface.
5. The master receives the message, capturing the exact time t4, at which the message enters its system.
6. The master sends the t4 information to the slave in the Delay_Resp message.
7. The slave uses the four values of t1, t2, t3, and t4 to synchronize its local timing reference to the timing reference of the master.

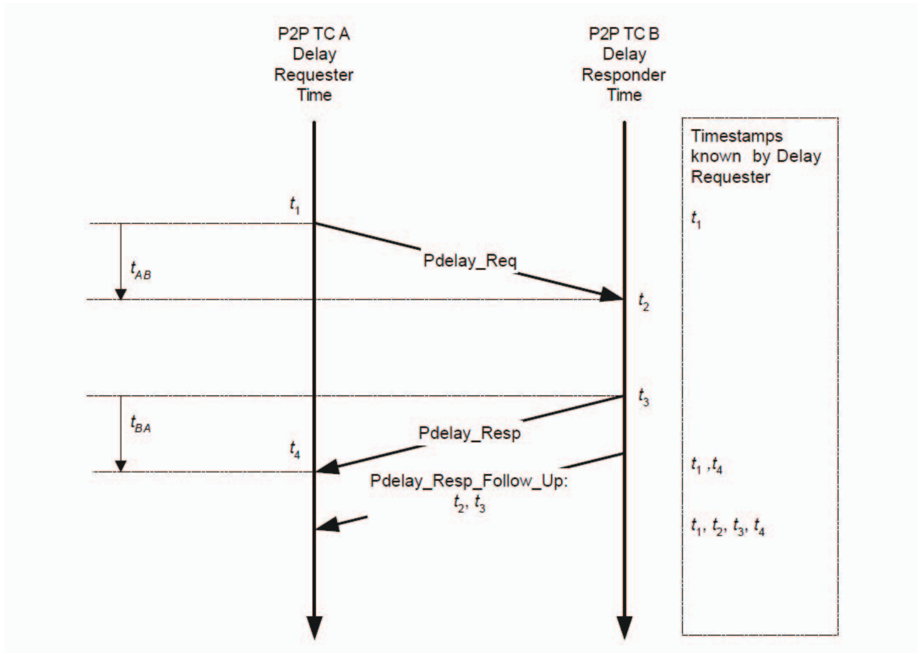
Most of the PTP implementation is done in the software above the Ethernet/UDP layer. However, the hardware support is required to capture the exact time when specific PTP packets enter or leave the Ethernet port at the GMII or MII interface. This timing information must be captured and returned to the software for proper implementation of PTP with high accuracy.

7.1.2.3 Peer-to-Peer PTP Transparent Clock (P2P TC) Message Support

The IEEE 1588-2008 supports peer-to-peer PTP (Pdelay) message in addition to the SYNC, Delay Request, Follow-up, and Delay Response messages.

The IEEE 1588-2008 supports peer-to-peer PTP (Pdelay) message in addition to the SYNC, Delay Request, Follow-up, and Delay Response messages. Figure 7-2 shows the method to calculate the propagation delay in clocks supporting peer-to-peer path correction.

Figure 7-2 Propagation Delay Calculation in Clocks Supporting Peer-to-Peer Path Correction


P2P TC A Delay Requester Time
P2P TC B Delay Responder Time
Timestamps known by Delay Requester
t_1
t_{AB}
Pdelay_Req
t_2
t_3
t_{BA}
Pdelay_Resp
t_4
Pdelay_Resp_Follow_Up: t_2, t_3
t_1
t_1 ,t_4
t_1, t_2, t_3, t_4

As shown in Figure 7-2, the propagation delay is calculated in the following way:

1. Port 1 issues a Pdelay_Req message and generates a timestamp (t1) for the Pdelay_Req message.
2. Port 2 receives the Pdelay_Req message and generates a timestamp (t2) for this message.
3. Port 2 returns a Pdelay_Resp message and generates a timestamp (t3) for this message.

   To minimize errors because of any frequency offset between the two ports, Port 2 returns the Pdelay_Resp message as quickly as possible after the receipt of the Pdelay_Req message. Port 2 returns any one of the following:

   - Difference between the timestamps t2 and t3 in the Pdelay_Resp message
   - Difference between the timestamps t2 and t3 in the Pdelay_Resp_Follow_Up message
   - Timestamps t2 and t3 in the Pdelay_Resp and Pdelay_Resp_Follow_Up messages, respectively
4. Port 1 generates a timestamp (t4) on receiving the Pdelay_Resp message.
5. Port 1 uses all four timestamps to compute the mean link delay.

7.1.2.4 Timestamp Correction

According to the IEEE 1588 specification, a timestamp must be captured when the PTP message timestamp point (leading edge of the first bit of the octet immediately following the Start Frame Delimiter octet) crosses the boundary between the node and the network. As the MAC takes the timestamp at an internal point far from the actual boundary of the node and network, this captured timestamp is corrected/updated for the ingress/egress path latency (including the delay in the PHY layers). Further correction is done for the inaccuracies/errors introduced due to the clock (GMII Tx, Rx clock) being different at the capture point as compared to the PTP clock (clk_ptp_ref_i) that is used to generate the time. The resultant CDC (Clock Domain Crossing) circuits add error depending on the clock period of the GMII and PTP clocks.

7.1.2.4.1 Ingress Correction

In the Receive side the timestamp captured at the internal snapshot point is delayed (later in time) as compared to the time at which that packet's SFD bit is received at the port's boundary. Therefore, the captured timestamp must be reduced by the ingress latency and the errors in CDC sampling. This correction value must be determined/calculated by the software and written into the MAC_Timestamp_Ingress_Corr_* registers.

The correction value consists of the following 3 components:

1. External latency in the PHY layer between boundary point and the input of the core.

   If the PHY is compliant to the IEEE 802.3 Clause 45 MMD registers, it has a register indicating the maximum and minimum ingress latency. The software can read these registers and determine the average ingress latency in the PHY. Alternatively (if the PHY does not support these registers), the ingress latency must be determined from its datasheet or timing characteristics.
2. Internal latency from the input of the core to the internal capture point

   The internal ingress latency can be read from the MAC_Ingress_Timestamp_Latency register. This is a read-only register and gives the latency in scaledNanoseconds format defined in IEEE 1588 Clause 5.3.2. The latency differs based on the active PHY interface (RGMII, RMII, so on) and the operating speed. Therefore, the software must read this register after any speed change in the MAC to determine the current internal latency.

3. CDC Synchronization

   The CDC synchronization error is almost equal to 2 times the clock-period of the PTP clock (clk_ptp_ref_i).

The values determined from these 3 components should be added by the software and must be written into the TSIC & TSICSNS fields of the MAC_Timestamp_Ingress_Corr_* registers.

Note The value written to the register must be negative (two's complement as explained below), because it has to be subtracted from the captured timestamp. The MAC Receiver adds the value in this register to the captured timestamp and then gives the resultant value as the timestamp of the received packet.

When TSCTRLSSR bit in MAC_Timestamp_Control register is set, the nanoseconds field of the captured timestamp is in decimal format with a granularity of 1ns. So the Bit31 of TSIC must be set to 1 (for negative value) and bits[30:0] must be written with "10^9 - total ingress_correction_value[nanosecond part]" represented in binary. For example, if the required correction value is -5 ns, then the value is 0xBB9A_C9FB.

When TSCTRLSSR bit in MAC_Timestamp_Control register is reset, the nanoseconds field of the captured timestamp is in binary format with a granularity of ~0.466ns. Therefore, bits[30:0] must be written with "2^31 - total ingress_correction_value" represented in binary with bit[31] = 1.

7.1.2.4.2 Egress Correction

In the Transmit side the timestamp captured at the internal snapshot point is earlier (advanced in time) as compared to the time at which that packet's SFD bit is output at the port's boundary. Therefore, the captured timestamp must be compensated by the egress latency and the errors in CDC sampling. This correction value must be determined/calculated by the software and written into the MAC_Timestamp_Egress_Corr_* registers.

The correction value consists of the following 3 components:

1. External latency in the PHY layer between the output of the core and the boundary of the port and the network

   If the PHY is compliant to the IEEE 802.3 Clause 45 MMD registers, it has a register indicating the maximum and minimum egress latency. The software can read these registers and determine the average egress latency in the PHY. Alternatively (if the PHY does not support these registers), the egress latency must be determined from its datasheet or timing characteristics.
2. Internal latency from the internal capture point and the output of the core

   This internal egress latency can be read from the MAC_Egress_Timestamp_Latency register. This is a read-only register and gives the latency in scaledNanoseconds format defined in IEEE 1588 Clause 5.3.2. The latency will differ based on the active PHY interface (RGMII, RMII, so on) and the operating speed. Hence the software must read this register after any speed change in the MAC to determine the current internal latency.
3. CDC synchronization error

   The CDC synchronization error value differs depending on the One-step timestamping mode.

   When One-step timestamping is enabled, the value = (1 * period of clk_ptp_ref_i + 4 * period of clk_tx_i).

   Otherwise (Two-step timestamping mode), the value = -(2 * period of clk_ptp_ref_i).

7.1.2.4.3 Frequency Range of Reference Timing Clock

The timestamp information is transferred across asynchronous clock domains, from the MAC clock domain to the application clock domain. Therefore, a minimum delay is required between two consecutive timestamp captures. This delay is 4 clock cycles of GMII or MII and 3 clock cycles of PTP clocks. If the delay between two timestamp captures is less than this delay, the MAC does not take a timestamp snapshot for the second packet.

Maximum PTP Clock Frequency

The maximum PTP clock frequency is limited by the maximum resolution of the reference time (1 ns resulting in 1 GHz) and the timing constraints achievable for logic operating on the PTP clock, whichever is lesser. In addition, the resolution or granularity of the reference time source determines the accuracy of the synchronization. Therefore, a higher PTP clock frequency gives better system performance.

Minimum PTP Clock Frequency

The minimum PTP clock frequency depends on the time required between two consecutive SFD bytes and the time taken for synchronizing the time to the GMII or MII clock domain. This relationship is given in the following equation:

```
    3 * PTP clock period + 4 * GMII/MII clock period <= Minimum gap between two SFDs
```

The GMII or MII clock frequency is fixed by IEEE specification. Therefore, the minimum PTP clock frequency required for proper operation depends on the operating mode and operating speed of the MAC as shown in Table 7-4.

Note

- It is recommended that you use a clock that has constant frequency, preferably a divided application clock.
- When IEEE 1588 timestamp feature is enabled with internal timestamp, use a PTP clock frequency which is greater than 5 MHz. This is because the 8-bit MAC_Sub_Second_Increment register limits the minimum PTP frequency that can be used to ~4 MHz.

Table 7-4 Minimum PTP Clock Frequency Example

Mode	Minimum Gap Between Two SFDs	Minimum PTP Frequency with External Timestamp Input	Minimum PTP Frequency with Internal Timestamp
10 Mbps full duplex	168 MII clocks (128 clocks for a 64-byte packet + 24 clocks of min IFG + 16 clocks of preamble)	~45 KHz	5 MHz
10 Mbps half duplex	48 MII clocks (8 clocks for a JAM pattern sent just after SFD because of collision + 24 IFG + 16 preamble)	170 KHz	5 MHz
100 Mbps full duplex	168 MII clocks (128 clocks for a 64-byte packet + 24 clocks of min IFG + 16 clocks of preamble)	~0.5 MHz	5 MHz

Mode	Minimum Gap Between Two SFDs	Minimum PTP Frequency with External Timestamp Input	Minimum PTP Frequency with Internal Timestamp
100 Mbps half duplex	48 MII clocks (8 clocks for a JAM pattern sent just after SFD because of collision + 24 IFG + 16 preamble)	4.55 MHz	5 MHz
1000 Mbps full duplex	84 GMII clocks (64 clocks for a 64-byte packet + 12 clocks of min IFG + 8 clocks of preamble)	~4.7 MHz	5 MHz
1000 Mbps half duplex	24 GMII clocks (4 clocks for a JAM pattern sent just after SFD because of collision + 12 IFG + 8 preamble)	18.75 MHz	18.75 MHz
2.5G Full Duplex	24 GMII clocks (4 clocks for a JAM pattern sent just after SFD because of collision + 12 IFG + 8 preamble)	85.3 MHz	11.72MHz
2.5G Half Duplex	24 GMII clocks (4 clocks for a JAM pattern sent just after SFD because of collision + 12 IFG + 8 preamble)	21.33 MHz	46.875MHz

7.1.2.4.4 PTP Processing and Control

Table 7-5 shows the common message header for the PTP messages. This format is taken from the IEEE 1588-2008.

Table 7-5 Message Format Defined in IEEE 1588-2008

Bits									
7	6	5	4	3	2	1	0	Octet	Offset
transportSpecific				messageType				1	0
Reserved				versionPTP				1	1
messageLength								2	2
domainNumber								1	4
Reserved								1	5
flagField								2	6
correctionField								8	8
Reserved								4	16

Bits									
7	6	5	4	3	2	1	0	Octet	Offset
sourcePortIdentity								10	20
sequenceId								2	30
controlField (*)								1	32
logMessageInterval								1	33

(*) – control Field is used in version 1. In version 2, message Type field is used for detecting different message types.

There are some fields in the Ethernet payload that you can use to detect the PTP packet type and control the snapshot to be taken. These fields are different for the following PTP packets:

- PTP Packets Over IPv4
- PTP Frames Over IPv6
- PTP Packets Over Ethernet

PTP Packets Over IPv4

Table 7-6 provides information about the fields that are matched to control snapshot for the PTP packets sent over UDP over IPv4 for IEEE 1588 version 1 and 2. The octet positions for the tagged packets are offset by 4. This is based on the IEEE 1588-2008, Annex D and the message format defined in Table 7-5.

Table 7-6 IPv4-UDP PTP Packet Fields Required for Control and Status

Field Matched	Octet Position	Matched Value	Description
MAC Packet Type	12, 13	0x0800	IPv4 datagram
IP version and Header Length	14	0x45	IP version is IPv4
Layer 4 Protocol	23	0x11	UDP
IP Multicast Address (IEEE 1588 version 1)	30, 31, 32, 33	0xE0, 0x00, 0x01, 0x81 (or 0x82 or 0x83 or 0x84)	Multicast IPv4 addresses allowed: ■ 224.0.1.129 ■ 224.0.1.130 ■ 224.0.1.131 ■ 224.0.1.132
IP Multicast Address (IEEE 1588 version 2)	30, 31, 32, 33	0xE0, 0x00, 0x01, 0x81 (Hex) 0xE0, 0x00, 0x00, 0x6B (Hex)	■ PTP Primary multicast address: 224.0.1.129 ■ PTP Pdelay multicast address: 224.0.0.107
UDP Destination Port	36, 37	0x013F, 0x0140	■ 0x013F: PTP event messages ■ 0x0140: PTP general messages

Field Matched	Octet Position	Matched Value	Description
PTP Control Field (IEEE 1588 version 1)	74	0x00, 0x01, 0x02, 0x03, 0x04	■ 0x00: SYNC ■ 0x01: Delay_Req ■ 0x02: Follow_Up ■ 0x03: Delay_Resp ■ 0x04: Management
PTP Message Type Field (IEEE 1588 version 2)	42 (nibble)	0x0, 0x1, 0x2, 0x3, 0x8, 0x9, 0xB, 0xC, 0xD	■ 0x0: SYNC ■ 0x1: Delay_Req ■ 0x2: Pdelay_Req ■ 0x3: Pdelay_Resp ■ 0x8: Follow_Up ■ 0x9: Delay_Resp ■ 0xA: Pdelay_Resp_Follow_Up ■ 0xB: Announce ■ 0xC: Signaling ■ 0xD: Management
PTP Version	43 (nibble)	0x1 or 0x2	■ 0x1: Supports PTP version 1 ■ 0x2: Supports PTP version 2

a. PTP event messages are SYNC, Delay_Req (IEEE 1588 version 1 and 2) or Pdelay_Req, Pdelay_Resp (IEEE 1588 version 2 only)

PTP Frames Over IPv6

Table 7-7 provides information about the fields that are matched to control the snapshots for the PTP packets sent over UDP over IPv6 for IEEE 1588 version 1 and 2. The octet positions for the tagged packets are offset by 4. This is based on the IEEE 1588-2008, Annex D and the message format defined in Table 7-5.

Table 7-7 IPv6-UDP PTP Packet Fields Required for Control and Status

Field Matched	Octet Position	Matched Value	Description
MAC Packet Type	12, 13	0x86DD	IP datagram
IP Version	14 (Bits [7:4])	0x6	IP version is IPv6
Layer 4 Protocol	20[a] (*)	0x11	UDP
PTP Multicast Address	38 – 53	FF0x:0:0:0:0:0:0:181 (Hex) FF02:0:0:0:0:0:0:6B (Hex)	■ PTP Primary multicast address: FF0x:0:0:0:0:0:0:0:181 (Hex) ■ PTP Pdelay multicast address: FF02:0:0:0:0:0:0:0:6B (Hex)
UDP Destination Port	56, 57a	0x013F, 0x140	■ 0x013F: PTP event message ■ 0x0140: PTP general messages

Field Matched	Octet Position	Matched Value	Description
PTP Control Field (IEEE 1588 version 1)	94a	0x00, 0x01, 0x02, 0x03, or 0x04	■ 0x00: SYNC ■ 0x01: Delay_Req ■ 0x02: Follow_Up ■ 0x03: Delay_Resp ■ 0x04: Management (version1)
PTP Message Type Field (IEEE 1588 version 2)	62a (nibble)	0x0, 0x1, 0x2, 0x3, 0x8, 0x9, 0xB, 0xC, or 0xD	■ 0x0: SYNC ■ 0x1: Delay_Req ■ 0x2: Pdelay_Req ■ 0x3: Pdelay_Resp ■ 0x8: Follow_Up ■ 0x9: Delay_Resp ■ 0xA: Pdelay_Resp_Follow_Up ■ 0xB: Announce ■ 0xC: Signaling ■ 0xD: Management
PTP Version	63 (nibble)	0x1 or 0x2	■ 0x1: Supports PTP version 1 ■ 0x2: Supports PTP version 2

a. The Extension Header is not defined for PTP packets.

PTP Packets Over Ethernet

Table 7-8 provides information about the fields that are matched to control the snapshots for the PTP packets sent over Ethernet for IEEE 1588 version 1 and 2. The octet positions for the tagged packets are offset by 4. This is based on the IEEE 1588-2008, Annex D and the message format.

Table 7-8 Ethernet PTP Packet Fields Required for Control And Status

Field Matched	Octet Position	Matched Value	Description
MAC Destination Multicast Address[a]	0–5	01-1B-19-00-00-00 01-80-C2-00-00-0E	All PTP messages can use any of the following multicast addresses[b]: ■ 01-1B-19-00-00-00 ■ 01-80-C2-00-00-0E[c]
MAC Packet Type	12, 13	0x88F7	PTP Ethernet packet
PTP Control Field (IEEE 1588 version 1)	46	0x00, 0x01, 0x02, 0x03, or 0x04	■ 0x00: SYNC ■ 0x01: Delay_Req ■ 0x02: Follow_Up ■ 0x03: Delay_Resp ■ 0x04: Management

Field Matched	Octet Position	Matched Value	Description
PTP Message Type Field (IEEE 1588 version 2)	14 (nibble)	0x0, 0x1, 0x2, 0x3, 0x8, 0x9, 0xB, 0xC, or 0xD	■ 0x0: SYNC ■ 0x1: Delay_Req ■ 0x2: Pdelay_Req ■ 0x3: Pdelay_Resp ■ 0x8: Follow_Up ■ 0x9: Delay_Resp ■ 0xA: Pdelay_Resp_Follow_Up ■ 0xB: Announce ■ 0xC: Signaling ■ 0xD: Management
PTP Version	15 (nibble)	0x1 or 0x2	■ 0x1: Supports PTP version 1 ■ 0x2: Supports PTP version 2

a. The unicast address match of destination addresses (DA), programmed in MAC address 0 to 31, is used if the TSENMACADDR bit of MAC_Timestamp_Control register is set.
b. IEEE 1588-2008, Annex F
c. The MAC does not consider the PTP version 1 messages with Peer delay multicast address (01-80-C2-00-00-0E) as valid PTP messages.

7.1.2.5 Transmit Path Functions

The MAC captures a timestamp when the Start Packet Delimiter (SFD) of a packet is sent on the GMII or MII interface. The packets, for which the timestamps has to be captured can be controlled on per-packet basis. Each Transmit packet can be marked to indicate whether a timestamp should be captured for it.

The MAC does not process the transmitted packets to identify the PTP packets. You need to specify the packets for which you want to capture timestamps. You can do this by using the following method based on your configuration:

- **EQOS-CORE Configuration**

  You can specify the packets through input signals. To return the captured timestamps to the application, the MAC uses a separate 64-bit bus. The MAC gives the timestamp, along with the Tx status of the packet, on this bus.

- **EQOS-MTL Configuration**

  You can specify the packets through ATI Control Word. To return the captured timestamp to the application, the core provides the additional status on ati_txstatus_o on Bits [81:18].

- **EQOS-AHB, EQOS-AXI, and EQOS-DMA Configurations**

  You can specify the packets by using the control bits in the Transmit descriptor. The MAC returns the timestamp to the software inside the corresponding Transmit descriptor, thus connecting the timestamp automatically to the specific PTP packet.

  The 64-bit timestamp information is written to the TDES0 and TDES1 fields. The TDES0 field holds the 32 least significant bits of the timestamp.

7.1.2.6 Receive Path Functions

The MAC can be programmed to capture the timestamp of all packets received on the GMII or MII interface or to process packets to identify the valid PTP messages. Use the following options of the MAC_Timestamp_Control register to control the snapshot of the time to be sent to the application:

- Enable snapshot for all packets
- Enable snapshot for IEEE 1588 version 1 or version 2 timestamp
- Enable snapshot for PTP packets transmitted directly over Ethernet or UDP-IP-Ethernet
- Enable timestamp snapshot for the received packet for IPv4 or IPv6
- Enable timestamp snapshot only for EVENT messages (SYNC, DELAY_REQ, PDELAY_REQ, or PDELAY_RESP)
- Enable the node to be a master or slave and select the snapshot type

  This feature controls the type of messages for which snapshots are taken.

Note The DWC_ether_qos also supports the PTP messages over VLAN packets.

Table 7-9 Timestamp Snapshot Dependency on Register Bits

SNAPTYPSEL	TSMSTRENA	TSEVNTENA	PTP Messages
00	x	0	SYNC, Follow_Up, Delay_Req, Delay_Resp
00	0	1	SYNC
00	1	1	Delay_Req
01	x	0	SYNC, Follow_Up, Delay_Req, Delay_Resp, Pdelay_Req, Pdelay_Resp, Pdelay_Resp_Follow_Up
01	0	1	SYNC, Pdelay_Req, Pdelay_Resp
01	1	1	Delay_Req, Pdelay_Req, Pdelay_Resp
10	x	x	SYNC, Delay_Req
11	x	x	Pdelay_Req, Pdelay_Resp

The following list describes how the timestamp is provided, depending on the configuration:

- **EQOS-CORE Configurations**

  The MAC provides the timestamp and the corresponding status, along with EOP, on a 64-bit bus (mri_timestamp_o) on MAC Receive Interface (MRI). An additional signal (mri_timestamp_val_o) validates the presence of timestamp for the receive packet.

- **EQOS-MTL Configurations**

  The MTL provides the timestamp on the data bus (ari_data_o[N - 1:0]) after the EOP data and receive status has been transferred. Bit 30 in the first receive status word indicate the availability of timestamp. The MTL provides the additional status related to the timestamp on the data bus.

- **EQOS-AHB, EQOS-AXI, and EQOS-DMA Configurations**

  The DMA returns the timestamp to the software inside the corresponding Receive Descriptor. The extended status, containing the timestamp message status and the IPC status, is written in normal descriptor RDES1 and the snapshot of the timestamp is written in RDES0 and RDES1 fields of context descriptor. The RDES0 field holds the 32 least significant bits of the timestamp.

7.1.3 Enabling IEEE 1588 Timestamp Support

To enable this feature while configuring the controller in the coreConsultant GUI:

1. Select the **Enable IEEE 1588 Timestamp Support** option under the IEEE 1588 Timestamp section during the **Specify Configuration activity** in coreConsultant.

   For details about this option, see Filtering Parameters in the "Parameter Descriptions" on page 419.

Note See the "Enabled" field of the parameter descriptions to understand the dependencies

7.1.4 Programming Guidelines for IEEE 1588 Timestamping (System Time Correction)

See "System Time Correction" on page 1367.

7.2 IEEE 1588 System Time Source

7.2.1 Introduction to IEEE 1588 System Time Source

To get a snapshot of the time, the MAC requires a reference time in 64-bit format as defined in the IEEE 1588-2002 (80-bit format as defined in the IEEE 1588-2008). The DWC_ether_qos provides the following options for using the reference timing source in a node:

- External Timestamp Input

  This option takes an external 64-bit timing reference and its clock as input. The clock input is used to synchronize the timing reference to the MAC clock domain.

- Internal Reference Time (80-bit)

  This option takes only the reference clock input and uses it to internally generate the Reference time (also called the system time) and capture timestamps.

7.2.2 Description of IEEE 1588 System Time Source

7.2.2.1 External Timestamp Input

The 64-bit timing reference is split into the following two 32-bit signals:

- Upper 32-bits providing the time in seconds
- Lower 32-bits providing the time in nanoseconds

This timing reference is used to timestamp the packets.

7.2.2.2 Internal Reference Time

The timestamp has the following fields:

- UInteger48 seconds Field

  The seconds field is the integer portion of the timestamp in units of seconds. It is 48-bits wide. For example, 2.000000001 seconds are represented as seconds Field = 0x0000_0000_0002.

- UInteger32 nano secondsField

  The nanoseconds field is the fractional portion of the timestamp in units of nanoseconds. For example, 2.000000001 seconds are represented as nanoSeconds = 0x0000_0001.

  The nanoseconds field supports the following two modes:

  - **Digital rollover mode**: In this mode, the maximum value in the nanoseconds field is 0x3B9A_C9FF, that is, (10e9-1) nanoseconds.
  - **Binary rollover mode**: In this mode, the nanoseconds field rolls over and increments the seconds field after value 0x7FFF_FFFF. Accuracy is ~0.466 ns per bit.

7.2.2.3 System Time Register Module

The system time generator module is an optional module. It is not available if external time updating is enabled. The 80-bit time is maintained in this module and updated using the input reference clock (clk_ptp_ref_i). This time is the source for taking snapshots (timestamps) of Ethernet packets being transmitted or received at the GMII interface.

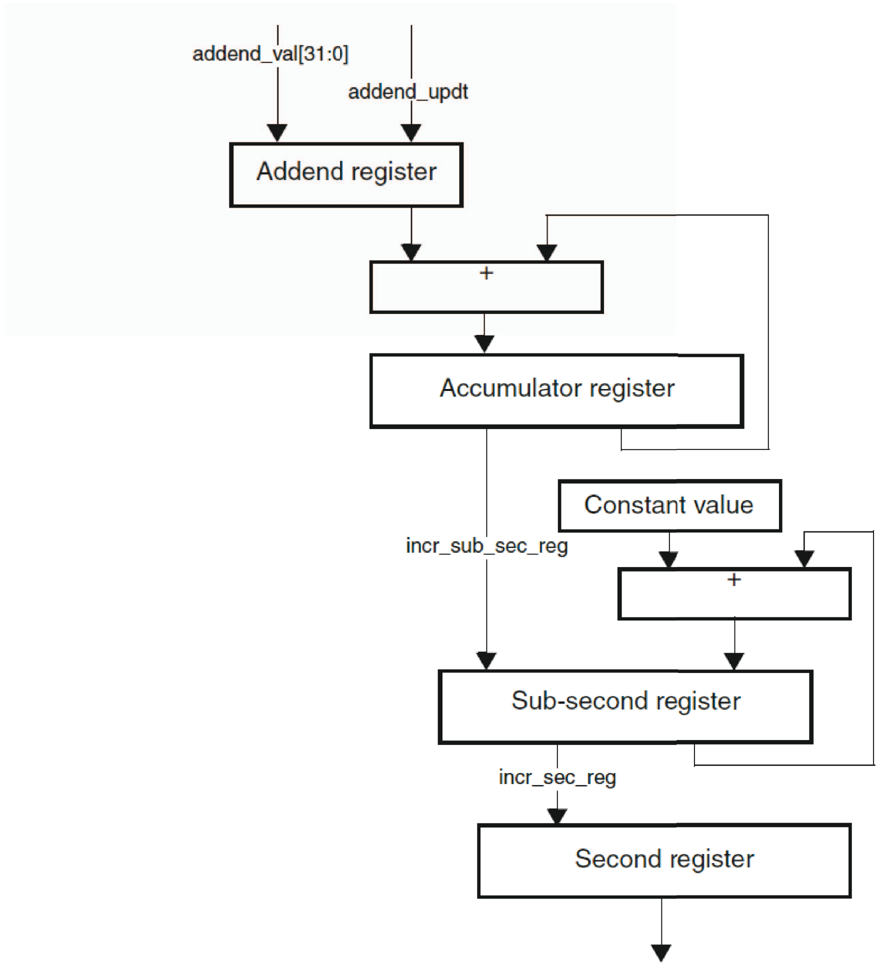
The system time counter can be initialized or corrected using the coarse correction method. In this method, the initial value or the offset value is written to the Timestamp Update register. For initialization, the system time counter is written with the value in the Timestamp Update register. For system time correction, the offset value is added to or subtracted from the system time.

In the fine correction method, the frequency offset and/or frequency drift of a slave clock (clk_ptp_ref_i) with respect to the master clock (as defined in IEEE 1588-2002) is corrected over a period of time instead of in one clock, as in coarse correction. This helps maintain linear time and does not introduce drastic changes (or a large jitter) in the reference time between PTP Sync message intervals. In this method, an accumulator sums up the contents of the Addend register, as shown in Figure 7-3. The arithmetic carry that the accumulator generates is used as a pulse to increment the system time counter. The accumulator and the addend are 32-bit registers. The accumulator acts as a high-precision frequency multiplier or divider.

Note You must connect a PTP clock with a frequency higher than the frequency required for the specified accuracy.

This algorithm is shown in Figure 7-3.

Figure 7-3 System Time Update Using Fine Method



The System Time Update logic requires a 50-MHz clock frequency to achieve 20-ns accuracy. The frequency division is the ratio of the reference clock frequency to the required clock frequency. For example, if the reference clock (clk_ptp_ref_i) is 66 MHz, this ratio is calculated as 66 MHz / 50 MHz = 1.32. Therefore, the default addend value to be set in the register is 2^{32}/ 1.32, 0xC1F07C1F.

If the reference clock drifts lower, for example, to 65 MHz, the ratio is 65 / 50, or 1.3 and the value to set in the addend register is 2^{32} / 1.30, or 0xC4EC4EC4. If the clock drifts higher, for example, to 67 MHz, the addend register must be set to 0xBF0B7672. When the clock drift is nil, the default addend value of 0xC1F07C1F (2^{32} / 1.32) must be programmed.

In Figure 7-3, the constant value used to accumulate the sub-second register is decimal 43, which achieves an accuracy of 20 ns in the system time (in other words, it is incremented in 20 ns steps). When External Time Update is enabled, the optional System Time module is not available.

The software must calculate the drift in frequency based on the Sync messages and accordingly update the Addend register.

Initially, the slave clock is set with FreqCompensationValue0 in the Addend register. This value is as follows:

$\text{FreqCompensationValue0} = 2^{32} / \text{FreqDivisionRatio}$

If MasterToSlaveDelay is initially assumed to be the same for consecutive Sync messages, the algorithm given in this section must be applied. After a few Sync cycles, frequency lock occurs. The slave clock can then determine a precise MasterToSlaveDelay value and re-synchronize with the master using the new value.

The algorithm is as follows:

- At time MasterSyncTime_n the master sends the slave clock a Sync message. The slave receives this message when its local clock is SlaveClockTime_n and computes MasterClockTime_n as

$\text{MasterClockTime}_n = \text{MasterSyncTime}_n + \text{MasterToSlaveDelay}_n$

- The master clock count for current Sync cycle, $\text{MasterClockCount}_n$ is

$\text{MasterClockCount}_n = \text{MasterClockTime}_n - \text{MasterClockTime}_n - 1$ (assuming that MasterToSlaveDelay is the same for Sync cycles n and n – 1)

- The slave clock count for current Sync cycle, SlaveClockCount_n is

$\text{SlaveClockCount}_n = \text{SlaveClockTime}_n - \text{SlaveClockTime}_n - 1$

- The difference between master and slave clock counts for current Sync cycle, ClockDiffCount_n is

$\text{ClockDiffCount}_n = \text{MasterClockTime}_n - \text{SlaveClockTime}_n$

- The frequency-scaling factor for slave clock, FreqScaleFactor_n is

$\text{FreqScaleFactor}_n = (\text{MasterClockCount}_n + \text{ClockDiffCount}_n) / \text{SlaveClockCount}_n$

- The frequency compensation value for Addend register, $\text{FreqCompensationValue}_n$ is

$\text{FreqCompensationValue}_n = \text{FreqScaleFactor}_n * \text{FreqCompensationValue}_n - 1$

In theory, this algorithm achieves lock in one Sync cycle. However, it may take several cycles, because of changing network propagation delays and operating conditions. This algorithm is self-correcting. If the slave clock is initially set to an incorrect value from the master, the algorithm corrects it at the cost of more Sync cycles.

7.2.3 Selecting IEEE 1588 Time Source Type

To select the IEEE 1588 time source type, select the **Time source type** in the **IEEE 1588 System Time Source** option under the IEEE 1588 Timestamp section during the Specify Configuration activity in coreConsultant.

1. Select the IEEE 1588 time source type in the Time source type in the IEEE 1588 System Time Source option under the IEEE 1588 Timestamp section during the Specify Configuration activity in coreConsultant.

   For details about this option, see IEEE 1588 Timestamp Parameters in the "Parameter Descriptions" on page 419.

Note See the "Enabled" field of the parameter descriptions to understand the dependencies

Note Modes of "Internal Reference Time" can be set through TSCTRLSSR bit in MAC_Timestamp_Control Register.

7.3 Programming Guidelines for IEEE 1588 Timestamping (For Internal Timestamp Source Configuration)

See “Initialization Guidelines for System Time Generation” on page 1367

7.4 IEEE 1588 Higher Word Register

7.4.1 Introduction to IEEE 1588 Higher Word Register

The timestamp maintained in the MAC is 64-bit wide. The overflow to upper 16-bits of seconds register happens once in 130 years. The values of the upper 16-bits of the seconds field can be read from the CSR register.

This optional feature is provided to invoke an additional Higher Word Register.

7.4.2 Enabling IEEE 1588 Higher Word Register

To enable higher word register while configuring the controller in the coreConsultant GUI:

1. Select the **Add IEEE 1588 Higher Word Register** option under the IEEE 1588 Timestamp section during the **Specify Configuration** activity in coreConsultant.

   For details about this option, see IEEE 1588 Timestamp Parameters in the “Parameter Descriptions” on page 419.

Note See the “Enabled” field of the parameter descriptions to understand the dependencies

Note This Higher Word Register can be invoked only when the “System Time Source” is “Internal”.

7.5 IEEE 1588 Auxillary Snapshot

7.5.1 Overview of IEEE 1588 Auxillary Snapshot

The auxiliary snapshot feature allows you to store a snapshot of the system time based on an external event. The event is considered to be the rising edge of the ptp_aux_ts_trig_i sideband signal.

This feature is independent of whether the system time is generated internally or given as input (on ptp_timestamp_i[63:0] bus).

When this feature is enabled, you can configure up to four auxiliary snapshot inputs and also specify the depth (4, 8, or 16) of a single common auxiliary snapshot FIFO.

7.5.2 Description of IEEE 1588 Auxillary Snapshot

The snapshots taken for any input are stored in a common FIFO. The application can read the MAC_Timestamp_Status register to know the timestamp of which input is available for reading at the top of this FIFO.

The MAC stores these snapshots in a FIFO. Only 64-bits of the timestamp are stored in the FIFO. You can read the upper 16-bits of seconds from the MAC_System_Time_Higher_Word_Seconds register when it is present. When a snapshot is stored, the MAC indicates this to the application with an interrupt. The value of the snapshot is read through a FIFO register access. If the FIFO becomes full and an external trigger to take the snapshot is asserted, a snapshot trigger-missed status (ATSSTM) is set in the MAC_Timestamp_Status register. This indicates that the latest auxiliary snapshot of the timestamp is not stored in the FIFO.
The latest snapshot is not written to the FIFO when it is full.

When an application reads the 64-bit timestamp from the FIFO, the space becomes available to store the next snapshot. You can clear a FIFO by setting the ATSFC bit in MAC_Auxiliary_Control register. When multiple snapshots are present in the FIFO, the count is indicated in Bits[27:25] of MAC_Timestamp_Status register.

7.5.3 Enabling IEEE 1588 Auxillary Snapshot

1. Select the **Add IEEE 1588 Auxillary Snapshot** option under the **IEEE 1588 Timestamp** section during the **Specify Configuration** activity in coreConsultant.
2. Select the required number in the **Number of IEEE 1588 Auxiliary Snapshot Inputs** option.
3. Select the required FIFO depth in the **FIFO Depth for IEEE 1588 Auxiliary Snapshots** option.

   For details about these options, see IEEE 1588 Timestamp Parameters in the "Parameter Descriptions" on page 419.

Note See the "Enabled" field of the parameter descriptions to understand the dependencies.

7.6 Flexible Pulse-Per-Second Output

7.6.1 Overview of Flexible Pulse-Per-Second Output

The DWC_ether_qos provides the flexibility to program the start or stop time, width, and interval of the pulse generated on the ptp_pps_o output.

Note

- By default, DWC_ether_qos is in the “Fixed Pulse-Per-Second Output” mode and indicates 1 second interval.
- The frequency of the PPS output can be changed by setting the PPSCTRL0 field in the MAC_PPS_Control register.

7.6.2 Description of Flexible Pulse-Per-Second (PPS) Output

DWC_ether_qos provides features such as programming the start or stop time, with the flexible PPS output option.

DWC_ether_qos supports the following features with the flexible PPS output option:

- Programming the start or stop time in terms of system time.
- Programming the start point of the single pulse and start and stop points of the pulse train in terms of 64-bit system time. The Target Time registers are used to program the start and stop time.
- Programming the stop time in advance, that is, you can program the stop time before the actual start time has elapsed.
- Programming the width between the rising edge and corresponding falling edge of PPS signal output in terms of number of units of sub-second increment value programmed in the MAC_Sub_Second_Increment register. You can program the width of pulse from 1 to 2^{32}-1 units of sub-second increment value.
- Programming the interval, between the rising edges of PPS signal, in terms of number of units of sub-second increment value. You can program the interval between pulses from 1 to 2^{32}-1 units of sub-second increment value.
- Option to cancel the programmed PPS start or stop request.
- Error if the start or stop time being programmed has already elapsed.

Note

The PTP Reference clock mentioned in the following sections is the clock at which the system time gets updated. When the TSCFUPDT bit of MAC_Timestamp_Control register is set to 0, this clock is similar to the clk_ptp_ref_i clock. In the Fine Correction mode, this is the clock tick at which the system time gets updated (using incr_sub_sec_reg (as shown in Figure 7-3).

7.6.2.1 PPS Start or Stop Time

The start time in the Target Time registers can be programmed initially.

You can initially program the start time in the Target Time registers. If you have enabled additional flexible PPS outputs, you need to program the following registers corresponding to an enabled flexible PPS output:

- MAC_PPS1_Target_Time_Seconds and MAC_PPS1_Target_Time_Nanoseconds registers for second Flexible PPS output
- MAC_PPS2_Target_Time_Seconds and MAC_PPS2_Target_Time_Nanoseconds registers for third Flexible PPS output
- MAC_PPS3_Target_Time_Seconds and MAC_PPS3_Target_Time_Nanoseconds registers for fourth Flexible PPS output

If required, you can again program the start or stop time but you can do it only after the earlier programmed value is synchronized to the PTP clock domain. Bit 31 of MAC_PPS#_Target_Time_Nanoseconds register indicates that the synchronization is complete. This enables you to program the start or stop time in advance even before the earlier stop or start time has elapsed.

To ensure proper PPS signal output, you should program advanced system time for the start or stop time. If the application programs a start or stop time that has already elapsed, the MAC sets an error status bit indicating the programming error. If enabled, the MAC also sets the Target Time Reached interrupt event. The application can cancel the start or stop request only if the corresponding start or stop time has not elapsed. If the time has elapsed, the cancel command has no effect.

7.6.2.2 PPS Width and Interval

The PPS width and interval are programmed in terms of granularity of system time, that is, number of the units of sub-second increment value.

The PPS width and interval are programmed in terms of granularity of system time, that is, number of the units of sub-second increment value. For example, to have a PPS pulse width of 40 ns and interval of 100 ns with PTP reference clock of 50 MHz, you should program the width and interval to values 2 and 5, respectively. You can achieve smaller granularity by using a faster PTP reference clock.

Before giving the command to trigger a pulse or pulse train on the PPS output, you should program or update the interval and width of the PPS signal output.

7.6.3 Enabling Flexible Pulse-Per-Second Output

To enable this feature while configuring the controller in the coreConsultant GUI:

1. Select the **Enable Flexible Pulse-Per-Second Output** option under the IEEE 1588 Timestamp section during the Specify Configuration activity in coreConsultant.
2. Select the required number in the **Number of Pulse-Per-Second Outputs** option.

For details about these options, see IEEE 1588 Timestamp Parameters in "Parameter Descriptions" on page 419.

Note See the "Enabled" field of the parameter descriptions to understand the dependencies.

7.7 Media Clock Generation and Recovery

7.7.1 Overview of Media Clock Generation and Recovery

DWC_ether_qos supports dedicated hardware for media clock recovery. This hardware generates a reference clock, which is multiplied to get the desired media clock. The hardware clock multiplier is external to DWC_ether_qos.

The media clock generation and recovery process is mainly hardware based and involves very little software.

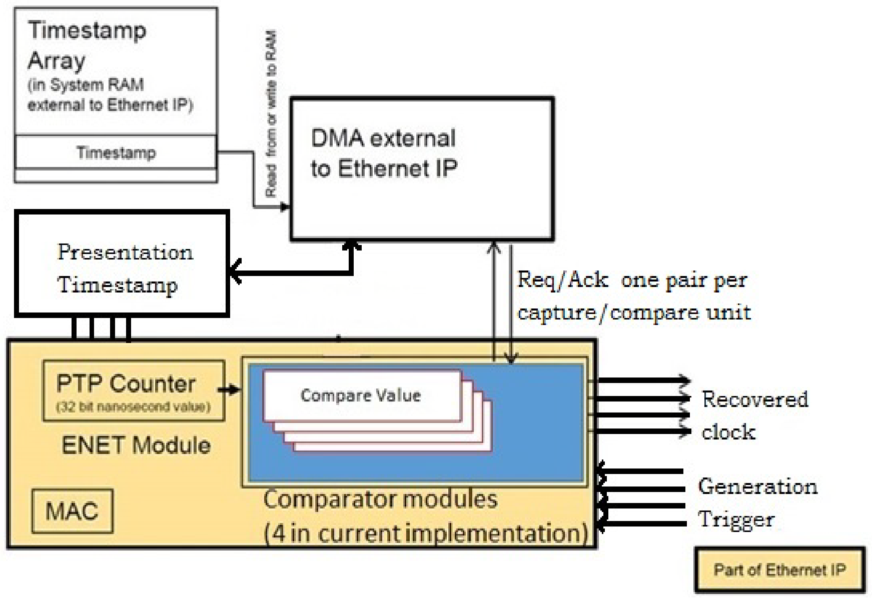
7.7.2 Description of Media Clock Generation and Recovery

The media clock generation and recovery support involves a dedicated external hardware and a thin layer of software. The dedicated hardware generates the reference clock (for example, 6KHz or 8KHz for AVB class A stream) which is then multiplied to get the desired media clock (for example, audio MCLK of 12.88 MHz).

The media clock recovery process involves host software extracting the presentation time from the 1722 frames and programming them into a CSR register, which is read by the DUT. When the timer expires, the previous value is either toggled or a pulse is sent. The output at the recovered clock pin is generally a wave of 750Hz ranging up to 8KHz depending on the application that is synced with, at the other end of the network.

As shown in Figure 7-4, the timestamp array and the DMA module are external to DWC_ether_qos.

Figure 7-4 Block Diagram of Media Clock Generation and Recovery


Timestamp Array
(in System RAM external to Ethernet IP)
Timestamp
Read from or write to RAM
DMA external to Ethernet IP
Presentation Timestamp
Req/Ack one pair per capture/compare unit
PTP Counter
(32 bit nanosecond value)
ENET Module
Compare Value
Recovered clock
MAC
Comparator modules
(4 in current implementation)
Generation Trigger
Part of Ethernet IP

For the operation of the media clock generation and recovery, DWC_ether_qos supports,

- A 32-bit Presentation Time nanosecond counter that wraps at the full 32-bit value (and not after one second) to match up with the 1722 presentation time format.

- A mechanism to handshake with external application (such as DMA) to program the presentation time into the CSR register. The hardware compares the programmed timestamp value with the Presentation time and generates a wave based on the programmable settings.

7.7.2.1 Presentation Time Counter

The Presentation Time Counter follows the 1722 presentation time format. In this format, time is represented as a 32-bit counter in nanoseconds that does a binary rollover on reaching the maximum value of 32'hFFFF_FFFF. This counter is another view of the PTP System time (1588 timer). For a given PTP System time (PTP[63:32] represents time in seconds and PTP[31:0] represents time in nanoseconds), there exists a corresponding Presentation Time (32-bit value in nanosecond). The Presentation time computed is referred to as Current Presentation Time (CPT).

The Current Presentation Time (CPT) can be viewed as derived from 64bit PTP System Time.

PTPNS[63:0] = (PTP[63:32] * 32'd1,000,000,000) + PTP[31:0]

where, PTPNS is the PTP system time converted into a 64bit nanosecond value.

Current Presentation Time[31:0] = PTPNS[31:0]

Media Clock Recovery is possible when the System Time is internally generated in DWC_ether_qos, and CPT is computed in the same way as the Internal PTP System Time generation. Because CPT and System Time values are in nanoseconds, and must be synchronous, the increment cycle of both the timers are same. The timer is updated at the same instance, but the updated value could be different. The software must compute a separate 32-bit update value in nanoseconds for the CPT update. CPT sampled at different edges of a triggering input results in the generation of Media Clock timestamps which are inserted in 1722 based AVBTP packets, to recover the clock at the destination.

Note

- The value of CPT can be obtained by reading the MAC_Presn_Time_ns register.
- DWC_ether_qos supports both fine-grain and course-grain correction modes.
- Only the digital rollover mode must be used for the PTP time. This is because, CPT also uses the same increment value and CPT requires the time to be in nanoseconds.

7.7.2.2 Comparator Modules

As shown in the Figure 7-4, DWC_ether_qos supports four comparator modules to handle multiple media clocks that might be required.

The Comparator modules hand shake with the application to

- Program the timestamps into the MAC_PPS(#i)_Target_Time_Seconds register when PTGE field of MAC_Timestamp_Control register is set.
- Set the Presentation Time Control bits of the particular instance for recovery mode.

The timestamps received from the application are referred as Target Presentation Time (TPT).

The MCGR and PPS modes are mutually exclusive. When MCGREN#i field of MAC_PPS_Control register is set to 1, the DUT operates in MCGR mode. Otherwise, by default, the DUT operates in PPS mode.

The comparator module sends up to two requests to the application for TPT write, when PTGE field of MAC_Timestamp_Control register is set and the Presentation time control bits of the particular instance is set for MCGR mode. Subsequent requests are generated each time a Presentation Time match occurs. (CPT transitions from a value less than TPT, to a value greater than or equal to CPT). The first request is asserted when a specific comparator instance is set to MCGR mode with a non-zero Presentation Time Control and an additional request is made when the first data is received into the Comparator. This allows the application to write the next TPT value while DUT processes the previous TPT value for a match.

The TPT read from the MAC_PPS(#i)_Target_Time_Second is considered as a future time. A toggle/pulse is generated in the next cycle when a match is detected. When a match is detected, the mcgr_dma_req_o#i request for that comparator is asserted (to obtain next TPT) until the corresponding application acknowledgment is set.

The presentation control bits (PPSCMD#i field of MAC_PPS_Control register) determine the shape of the generated waveform. Also, these bits can be programmed to either toggle or generate a high/low pulse for one PTP clock cycle, upon match.

Clock Recovery: A match is when the free-running CPT value matches the received TPT value. A output signal ptp_pps_o is asserted (toggle, low pulse, or high pulse) based on the presentation control vaue programmed.

Clock Generation: Based on the presentation control value programmed in MAC_PPS_Control register, the particular comparator captures the presentation time and programs it into MAC_PPS(#i)_Target_Time_Second register. A request is raised to the application to read the captured timestamp. Until the acknowledgment is received from the application, acknowledging that the read operation is complete, no new timestamps are captured.

7.7.2.3 Media Clock Generation and Recovery Flow

Figure 7-5 shows the media clock generation and recovery flow.

For both media clock generation and recovery, set the PTGE field of MAC_Timestamp register for CPT generation and set the MCGREN#i field of MAC_PPS_Control Register for the corresponding instance to be enabled in the MCGR mode.

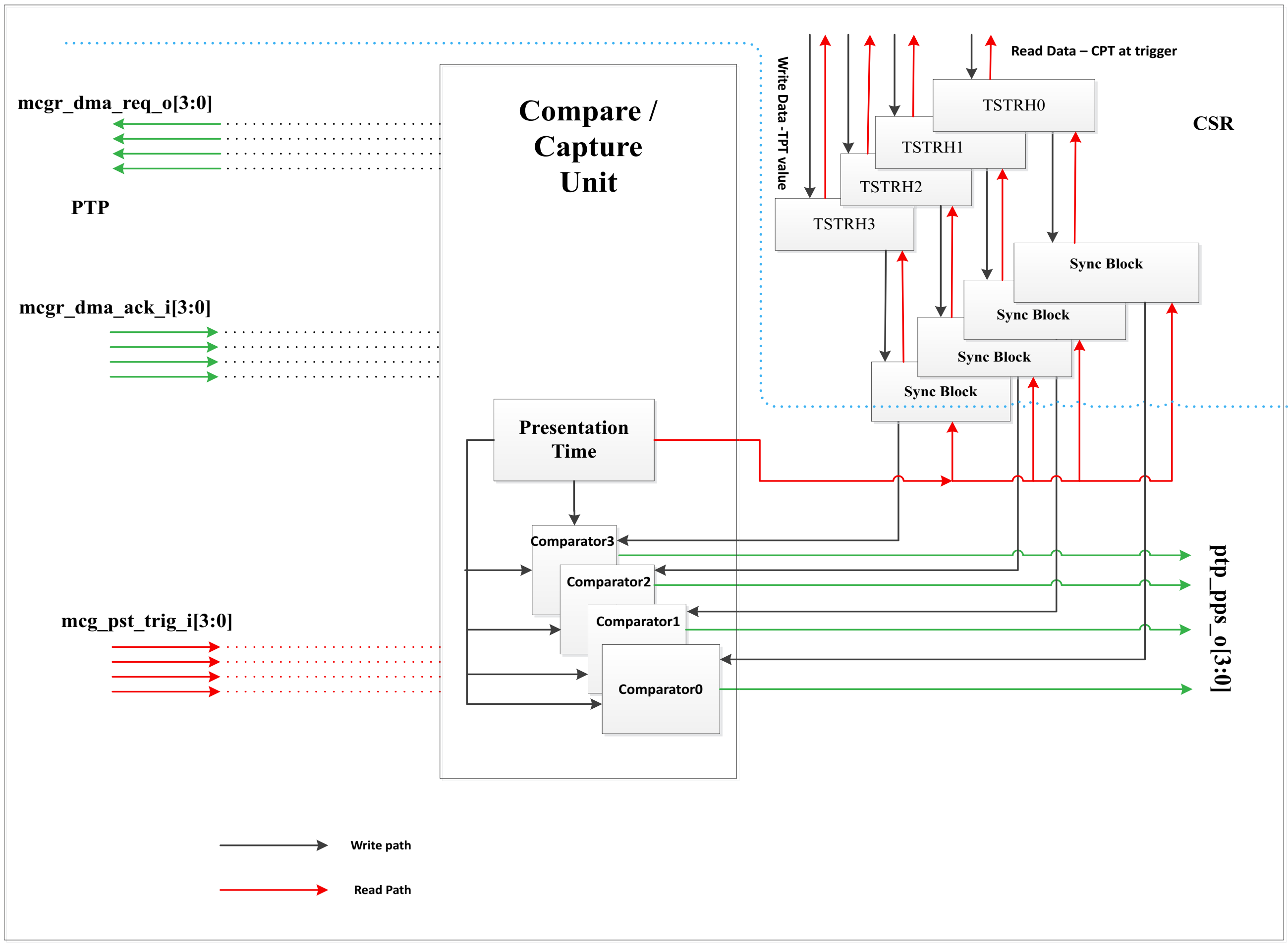
For Media Clock Generation, the generated presentation time is sampled at positive, negative or at both the edges of trigger inputs (mcg_pst_trig[i]) based on the mode programmed in MAC_PPS_Control register. The sampled value is captured in MAC_PPS(#i)_Target_Time_Seconds register. After the MAC_PPS(#i)_Target_Time_Seconds register is updated with the presentation time, the DUT asserts mcgr_dma_req_o#i to request the application to read the presentation time. Application acknowledges that it has read the captured timestamp by asserting the corresponding mcgr_dma_ack_i[#i].

For Media Clock Recovery, PPS instance asserts the mcgr_dma_req_o#i register to indicate the application to program the MAC_PPS(#i)_Target _Time_Seconds with TPT value and acknowledge the corresponding request using mcgr_dma_ack_i. When the application programs the MAC_PPS(#i)_Target_Time_Seconds with the TPT value, the particular comparator compares the programmed TPT value with the CPT and generates the waveform on ptp_pps_o[3:0] based on the presentation time control value programmed in MAC_PPS_Control.

Note

- The consecutive triggers to sample the presentation time should always be asserted after a few PTP clock cycles to allow for synchronization delays. (This is not an issue when the input trigger maximum frequency is 8KHz and the PTP clock is running at least at 1MHz)
- DMA acknowledgment can be both posted as well as non-posted, in the Media Clock Recovery Mode.

Figure 7-5 Media Clock Generation and Recovery Flow



7.7.3 Enabling Media Generation and Recovery

When you enable the Flexible Pulse Per Second Output feature (by selecting DWC_EQOS_FLEXI_PPS_OUT_EN), the media generation and recovery feature is also enabled. The number of comparator instances and the clock outputs is determined by value of the DWC_EQOS_PPS_OUT_NUM parameter.

For details about these options, see **IEEE 1588 Timestamp Parameters** in "Parameter Descriptions" on page 419.

Note See the "Enabled" field of the parameter descriptions to understand the dependencies.

7.7.4 Signals Related to Media Generation and Recovery

- mcg_pst_trig_i
- mcg_dma_req_o

- mcgr_dma_ack_i
- ari_data_p_o

For a description of signals related to Media Generation and Recovery, see "Signal Descriptions" on page 471.

7.7.5 Registers Related to Media Clock Generation and Recovery

- MAC_PPS_Control
- MAC_PPS_Target_Time_Seconds
- MAC_Presn_Time_ns
- MAC_Presn_Time_updt

For more details, see "Register Descriptions" on page 605.

7.7.6 Programming Guidelines for Media Clock Generation and Recovery

For more details about programming guidelines for ECC protection for memories, see "Programming Guidelines for ECC Protection for Memories" on page 1384.

7.8 PTP Timestamp Offload Function

7.8.1 Introduction to PTP Timestamp Offload Function

PTP Timestamp Offload Function is an optional feature that enables the automatic generation of specific PTP packets when the MAC is working as a specific node in the PTP network.

These packets may be generated periodically or triggered by the host software. In other modes, this feature can parse the incoming PTP packets on the receiver, and automatically generate and respond to the required PTP packets. This helps in offloading certain functions of a PTP node with better accuracy and lower latency of responses.

7.8.2 Description of PTP Offload Function

Based on the programmed mode, the MAC generates PTP Ethernet messages periodically or from the application, or based on reception of a particular PTP message

Table 7-10 indicates the PTP message generation criteria.

Table 7-10 PTP Message Generation Criteria

Programming					
SNAPTYPSEL	**TSMSTRENA**	**TSEVNTENA**	**Mode**	**Criteria for Generation of PTP Messages**	**PTP Message Type Generated**
2’b00	0	1	Ordinary or Boundary Slave	SYNC message reception	Delay_Req
2’b00	1	1	Ordinary or Boundary Master	Periodic or on trigger from application	SYNC
				Delay_Req message reception	Delay_Resp
2’b01	0	1	Transparent Slave	Periodic or on trigger from application	Pdelay_Req
				Pdelay_Req message reception	Pdelay_Resp
				SYNC message reception	Delay_Req
2’b01	1	1	Transparent Master	Periodic or on trigger from application	Pdelay_Req
				Pdelay_Req message reception	Pdelay_Resp
				Periodic or on trigger from application	SYNC
				Delay_Req message reception	Delay_Resp

Programming					
SNAPTYPSEL	**TSMSTRENA**	**TSEVNTENA**	**Mode**	**Criteria for Generation of PTP Messages**	**PTP Message Type Generated**
2’b11	X	X	Peer-to-Peer Transparent	Periodic or on trigger from application	Pdelay_Req
				Pdelay_Req message reception	Pdelay_Resp
All other programming combinations are invalid for PTP Offload feature.					

Note Clocks supporting peer delay mechanism must not generate delay request/delay response messages, according to IEEE 1588-2008 specification. However, the DWC_ether_qos controller supports this for flexibility, with a programmable control bit (DRRDIS).

You can use the DRRDIS bit to control the response generation for delay request/delay response message. For example, in transparent slave mode, delay request is generated in response to received sync only when the bit is reset.

For example, when the MAC is set as an Ordinary or Boundary Slave clock in the PTPT network, it can respond to the reception of SYNC messages with an automatic generation and transmission of the corresponding Delay_Req message. Similarly, various other modes of operation are explained in Table 7-10.

The MAC supports the multicast communication model for the generation of SYNC and Pdelay_Req PTP messages. For instance, the Destination Address field of the generated PTP over Ethernet packet is the defined special multicast addresses (0x011B19000000 for all except peer delay mechanism messages and 0x0180C200000E for peer delay mechanism messages).

When the MAC responds to received SYNC, Delay_Req and Pdelay_Req PTP messages with special multicast destination address, it also uses the corresponding special multicast address in the DA field of the automatically generated Delay_Req, Delay_Resp, and Pdelay_Resp PTP messages, respectively.

When the MAC responds to received SYNC, Delay_Req and Pdelay_Req PTP messages with unicast destination address, it takes the SA field of the received packets and makes them as the DA field of the automatically generated Delay_Req, Delay_Resp, and Pdelay_Resp PTP messages, respectively.

At the same time, all the received PTP messages are forwarded to the application along with Rx status, indicating whether the response was generated by the MAC, if it satisfies the packet filtering logic of the MAC receiver

When the MAC automatically generates a PdelayReq or responds with a Delay_Req, the egress timestamp of these two PTP messages are provided in the Tx TS status (Tx Timestamp Status Register and interrupt generated).

In addition to messageType and versionPTP fields match for basic PTP over Ethernet message detection, the following additional fields are matched to qualify the received PTP message type:

1. The domainNumber field is checked for a match against the value programmed in the CSR.
2. The twoStepFlag in flagField field is checked for one-step indication (1'b0).
3. The transportSpecific field is checked for Default PTP over Ethernet (4'h0) or 802.1AS mode (4'h1) when enabled.

7.8.2.1 PTP Packet Generation

This section explains the format and content of the automatically generated PTP packets by the MAC when this mode is enabled. The template of the Common PTP Message Header is provided, as well as the detailed description of the fields of specific PTP packets generated.

Table 7-11 Common PTP Message Header Fields

Bits									
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**	**Octets**	**Offset**
transportSpecific				messageType				1	0
Reserved				versionPTP				1	1
messageLength								2	2
domainNumber								1	4
Reserved								1	5
flagField								2	6
correctionField								8	8
Reserved								4	16
sourcePortIdentity								10	20
sequenceId								2	30
sequenceId								2	30
controlField								1	32
logMessageInterval								1	33

messageType

The following encoded values are used for PTP message types:

- SYNC - 4'h0
- Delay_Req - 4'h1
- Pdelay_Req - 4'h2
- Pdelay_Resp - 4'h3
- Delay_Resp - 4'h9

transportSpecific

The following transport protocol encoding is used:

- Default PTP over Ethernet - 4'h0
- 802.1AS mode - 4'h1

versionPTP

It is always set to 2 because PTP version 2 is supported.

domainNumber

This contains the value from the MAC_PTO_Control Register.

flagField

The following values are used:

- alternateMasterFlag (Octet 0 bit 0) - 1b0 for SYNC and Delay_Resp
- twoStepFlag (Octet 0 bit 1) - 1'b0 for SYNC and Pdelay_Resp
- unicastFlag (Octet 0 bit 2) - 1'b0 for Multicast Address, 1'b1 for Unicast Address

correctionField

For more information, see Table 7-12.

sourcePortIdentity

This field takes the value programmed in the MAC_Source_Port_Identityx registers.

sequenceId

Pdelay_Resp and Delay_Resp use the same sequenceId field from received Pdelay_Req and Delay_Req PTP messages. For SYNC/Delay_Req, Pdelay_Req, a separate sequenceId counter is maintained. These sequenceId counters get incremented by 1 every time the corresponding message is generated and transmitted.

controlField

The following encoded values are used for controlField:

- SYNC - 8'h00
- Delay_Req - 8'h01
- Pdelay_Req - 8'h02
- Pdelay_Resp - 8'h05
- Delay_Resp - 8'h03

logMessageInterval

- SYNC - This contains logSyncInterval from the corresponding MAC_Log_Message_Interval register.
- Delay_Resp - This contains the sum of DRSYNCR and logSyncInterval value taken from the MAC_Log_Message_Interval register for a multicast PTP message and 8'h7F for unicast PTP message.
- Delay_Req, Pdelay_Req and Pdelay_Resp - 8'h7F

  where logSyncInterval = log2 (Mean Value of Interval in seconds)

The MAC supports values of -15 to 15 for logSyncInterval fields, which translates to a range from 32.768 ?s (2-15) to 215 sec. For a given value of log sync interval (N), the time interval between two SYNC packets is given by the following:

- 2(30+N) nanoSec, when N is negative (-1 to -15)
- 2N Sec, when N is positive (0 to 15)

For example:

- When logSyncInterval is programmed to 1, the interval is 21; therefore, the SYNC message is sent once every 2 seconds.
- When logSyncInterval is programmed to -1, the interval is 2-1 = 0.536 seconds; therefore, the SYNC message is sent once every 536 milliseconds. The value is 0.536 seconds, because 2-30 = 1 ns.

- When logSyncInterval is programmed to -5, the interval is 2-5 = 33.55ms; therefore, the SYNC message is sent once every 33.55 milliseconds.

Note The MAC uses the PTP system time to generate the intervals for periodic packet transmission. For negative values of log message interval programmed, the generated period may deviate from the value given by the equation 2(30+N), because of the non-binary nature of the nanoseconds field of the system time.

7.8.2.2 PTP Message-Specific Fields

messageLength

There is no suffix supported, so this field contains the length of the PTP message that includes 34 byte PTP common header and the body specific to the message type.

For SYNC and Delay_Req packets, this field contains 44, whereas for Delay_Resp, Pdelay_Req and Pdelay_Resp, it contains 54.

originTimestamp

This field is the captured egress timestamp for SYNC, Delay_Req, and Pdelay_Req PTP messages.

receiveTimestamp

For Delay_Resp PTP message, this is the ingress timestamp of the corresponding received Delay_Req PTP message.

requestingPortIdentity

For Delay_Resp and Pdelay_Resp PTP messages, this is the sourcePortIdentity field taken from the corresponding received Delay_Req and Pdelay_Req PTP messages.

requestReceiptTimestamp

For the Pdelay_Resp PTP message, this field is set to 0.

7.8.3 Enabling PTP Timestamp Offload Function

1. Select the **Enable PTP Timestamp Offload Feature** option under the IEEE 1588 Timestamp section during the **Specify Configuration** activity in coreConsultant.

   For details about this option, see IEEE 1588 Timestamp Parameters in "Parameter Descriptions" on page 419.

   Note See the "Enabled" field of the parameter descriptions to understand the dependencies.

7.9 One-Step Timestamp

7.9.1 Introduction to One-Step Timestamp

The DWC_ether_qos supports the one-step timestamp feature. When the One step timestamp feature is enabled, the MAC identifies the offset in the packet and inserts the timestamp received from the application at that offset.

7.9.2 Description of One-Step Timestamp

7.9.2.1 MAC Transmit PTP Mode

Depending upon the type of message and its mode, the MAC updates the following fields of Transmit PTP packets:

- correctionField in the PTP header of messages
- originTimestamp in SYNC, Delay_Req, and Pdelay_Req messages

Table 7-12 shows how the PTP mode is selected based on the settings of SNAPTYPSEL, TSMSTRENA, and TSEVNTENA bits of the MAC_Timestamp_Control register and the fields that are updated for the incoming PTP packets based on the message type in that mode, during the one-step timestamping operation.

Table 7-12 MAC_Transmit_PTP_Mode_and_One-Step_Timestamping_Operation

Programming				**Per Packet Control**[a]			
SNAPTYP SEL	**TSMSTR ENA**	**TSEVN TENA**	**Mode**	**TTSE**[b]	**OSTC**[c]	**TTS**[d]	**Messages Processed on Tx**
X	X	X	N/A	1	X	X	Timestamp is captured and returned to application
X	X	X	N/A	X	0	X	OST operation is not performed (PTP packet is not modified)
2'b00	X	0	End-to-end transparent	0	1	Ingress TS	Sync (correction field for residence time and Ingress Asym cor)
							Delay_Req (correction field for residence time and Egress Asym Cor)
2'b00	0	1	Ordinary or Boundary Slave	1	1	X	Delay_Req (originTimestamp field) Delay_Req (correction field for Egress Asym cor)
2'b00	1	1	Ordinary or Boundary Master	0	1	X	Sync (originTimestamp field) Sync (correction field for sub-nanosecond cor)

Programming				Per Packet Control[a]			
SNAPTYPSEL	TSMSTRENA	TSEVNTENA	Mode	TTSE[b]	OSTC[c]	TTS[d]	Messages Processed on Tx
2’b01	X	0	End-to-End Transparent with support for peer delay mechanism	0	1	Ingress TS	Sync (correction field for residence time and Ingress Asym cor)
						Ingress TS	Pdelay_Req (correction field for residence time and Egress Asym Cor)
						Ingress TS	Pdelay_Resp (correction field for residence time and Ingress Asym Cor)
2’b01	0	1	Ordinary or Boundary Slave with support for peer delay mechanism or Peer to Peer Transparent	0	1	Ingress TS	Sync (correction field for residence time and Ingress Asym cor) (applicable only for Peer to Peer transparent clock operation)
				1	1	X	Delay_Req (originTimestamp field) Delay_Req (correction field for Egress Asym cor)
				1	1	X	Pdelay_Req (originTimestamp field) Pdelay_Req (correction field for Egress Asym Cor)
				0	1	Ingress TS for Pdelay_Req	Pdelay_Resp (correction field for turnaround time and Ingress Asym Cor)
2’b01	1	1	Ordinary or Boundary Master with support for peer delay mechanism	0	1	X	Sync (originTimestamp field) Sync (correction field for sub-nanosecond cor)
				1	1	X	Pdelay_Req (originTimestamp field) Pdelay_Req (correction field for Egress Asym Cor)
				0	1	Ingress TS for Pdelay_Req	Pdelay_Resp (correction field for turnaround time and Ingress Asym Cor)
2'b10	X	X	End-to-End Transparent	0	1	Ingress TS	Sync (correction field for residence time and Ingress Asym cor)
						Ingress TS	Delay_Req (correction field for residence time and Egress Asym Cor)

Programming				Per Packet Control[a]			
SNAPTYP SEL	TSMSTR ENA	TSEVN TENA	Mode	TTSE[b]	OSTC[c]	TTS[d]	Messages Processed on Tx
2'b11	X	X	Peer-to-Peer Transparent	0	1	Ingress TS	Sync (correction field for residence time and Ingress Asym cor)
				1	1	X	Pdelay_Req (originTimestamp field) Pdelay_Req (correction field for Egress Asym Cor)
				0	1	Ingress TS for Pdelay_Req	Pdelay_Resp (correction field for turnaround time and Ingress Asym Cor)

a. The per packet control values provided here are the recommended settings used by devices in typical PTP operation, for the programmed mode.
b. TTSE represents TTSE bit of transmit descriptor in EQOS-AXI configuration, TTSE bit in TX control word in EQOS-MTL configuration, mti_ena_timestamp_i signal in EQOS-CORE configuration. The TTSE function is independent of the OST function and the programmed operation mode for OST. The MAC captures and returns the timestamp when the TTSE bit is set.
c. OSTC represents OSTC bit of transmit descriptor in EQOS-AXI configuration, OSTC bit in TX control word in EQOS-MTL configuration, mti_ost_en_i signal in EQOS-CORE configuration.
d. TTS represents the timestamp value provided in the TTSH, TTSL fields of transmit descriptor in EQOS-AXI configuration, timestamp provided in the TTSH, TTSL fields in the TX control word in EQOS-MTL configuration, mti_ost_i signal in EQOS-CORE configuration.

Note Residence time/ turnaround time is calculated as the difference between the captured timestamp (egress timestamp) and the ingress timestamp.When sub-nanosecond feature is enabled, residence time calculation includes sub-nanosecond accuracy.Clocks supporting peer delay mechanism do not use delay request or response, but it is included in OST for flexibility.

7.9.3 Enabling One-Step Timestamp

One-step timestamp feature can be enabled using the coreConsultant GUI.

1. Select the **Enable One-Step Timestamp Feature** option under the IEEE 1588 Timestamp section during the **Specify Configuration** activity in coreConsultant.

   For details about this option, see IEEE 1588 Timestamp Parameters.

Note See the “Enabled” field of the parameter descriptions to understand the dependencies.

You can enable the one-step timestamp feature for a packet by setting Bit 20 (OSTC) in ATI Control Word. For updating the correction field in certain PTP packets, the ingress timestamp must be given in the TSSL and TSSH fields. If the **Add IEEE 1588 Higher Word Register** option is selected, the timestamp is 80-bit and it also contains the content of higher 16 bits of Higher Timestamp Word Register.

7.10 One-Step Time Stamping for PTP Over UDP

7.10.1 Introduction to One-Step Timestamping for PTP Over UDP

One step timestamping for PTP over UDP feature is available in configuration where IEEE 1588 Timestamp Support option is selected. DWC_ether_qos supports one-step timestamp operation for PTP messages sent over UDP/IPv4 and UDP/IPv6. Because origin timestamp and/or correction field (based on the PTP message type and the programmed PTP mode) in the PTP message are updated as part of the one-step timestamp operation, it also requires update to the checksum field of the PTP message sent over UDP/IP.

7.10.2 Checksum Update for One-Step Timestamping for PTP Over UDP

In DWC_ether_qos, checksum is updated for a packet, in the MTL before the packet is sent to the MAC where the PTP message is updated for one step timestamp operation. PTP requires the timestamp to be captured while the packet (SFD of the packet) is on the line and one step timestamp requires the origin timestamp and/or correction field to be updated in the PTP header while the packet is being transmitted on the line. Because the PTP message is present in the UDP payload, it is not possible to update the checksum field in the UDP header for modifications made to the PTP message, because it is already transmitted by then. Instead, the following options are supported that are in line with specification mentioned in Annex D & E of IEEE 1588-2008:

- When PTP message is sent over UDP/IPv4, the UDP checksum may be set to 0. The application must set the UDP checksum to 0 and set CIC to 0 or 1 while providing the packet to the MAC and the MAC does not update checksum for changes in origin timestamp or correction field.
- When PTP message is sent over UDP/IPv6, a transmitting node extends the UDP payload of all PTP messages by 2 octets beyond the end of the PTP message. The contents of the UDP checksum field or the final 2 octets of the UDP payload might be modified to ensure that the UDP checksum remains intact after modification of PTP fields. The application must add the two extra bytes at the end of the PTP message while providing the packet to the MAC and the MAC updates these two bytes to keep the checksum correct.

A programmable option (CSC field in MAC_Timestamp_Control register) supports UDP with non-zero checksum for IPv4. When set, the two bytes at the end of the PTP message are updated to keep the UDP checksum correct, similar to PTP over UDP/IPv6. So, the application must add two bytes at the end of the PTP message sent to the DWC_ether_qos and DWC_ether_qos updates these two bytes to keep the checksum correct.

Updating the pad bytes to keep the checksum correct requires the old value of correction field and/or origin timestamp in the packet. To get better latency and area, the application can set the origin timestamp field to zero, for PTP message in particular modes in which DWC_ether_qos overwrites the origin timestamp. For example, DWC_ether_qos operating in Ordinary/Boundary master mode overwrites the origin timestamp field of the sync message, when OSTC is set. Because the sync message is originated by the port operating in Ordinary/Boundary master mode, it is possible for the application generating the message to set the origin timestamp field to zero.

Transmit checksum engine is automatically enabled when this feature is selected. DWC_ether_qos uses the transmit checksum engine to identify the start of PTP message in the UDP/IP packet. Checksum engine has the capability to detect some kind of errors in the header (L3) and payload (L4). Also is not computed for some type of packets, which is indicated by "No" in Table 9-1. For one step, operation the packet is free of those errors and have "Yes/Not applicable" for the corresponding packet type in Table 9-1. If an error is reported by the checksum engine or there is a "No" in Table 9-1, for PTP messages sent over UDP/IP, no update is performed on the PTP message. See "Transmit Checksum Offload Engine" on page 340 for more details about the errors detected by transmit checksum engine.

Note

- The packet is sent with specific IP address and UDP port numbers and does not have IPv4 options/IPv6 extension headers. Applications must take care of these requirements before giving the packet to the DWC_ether_qos. DWC_ether_qos does not validate them. As PTP is identified by specific value in the UDP destination port and as DWC_ether_qos does not look into these fields, it assumes an IP packet with OSTC set as a PTP packet, and it is the responsibility of the application to set the OSTC bit only for IP packets which contains PTP messages.
- For PTP over UPD/IPv6 and PTP over UDP/IPv4 with CSC set, the DWC_ether_qos does not verify the presence of the two pad bytes, instead it overwrites the two bytes after the PTP. message
- When Enable One step timestamp for PTP over UDP/IP feature is selected, Enable Transmit TCP/IP Checksum offload feature is automatically enabled and this is enabled only for Queue 0 in multiple transmit queue configurations. If the application requires OST support for PTP messages sent over UDP/IP for packets sent from queues other than 0, transmit checksum engine must be manually enabled for those transmit queues. If checksum engine is not enabled, no updates will be done for PTP messages sent overUDP/IP even though OSTC is set.
- One-Step timestamp operation is not supported for tunneled packets

7.11 IEEE 1588 Sub Nanoseconds Timestamp

The ingress and egress correction can have sub-nanoseconds accuracy.

7.11.1 Description of IEEE 1588 Sub Nanoseconds Timestamp

The sub-nanosecond part of the ingress and egress correction is programmed in the TSICSNS and TSECSNS fields of the MAC Timestamp Ingress Correction Subnanosecond and MAC Timestamp Egress Correction Subnanosecond registers, respectively.

In this case, the correction has an unit of nanosecond, multiplied by 2^8. The least significant 8 bits of the value should be programmed in the sub-nanoseconds register.

For example, if the required egress correction is 3.6 nS and TSCTRLSSR bit is set, then TSEC must be 0x3 and TSECSNS must be 0x99 (0.6 * 256). Similarly, if the required ingress correction is -3.6 nS and TSCTRLSSR bit is set, then TSIC must be 0xBB9A_C9FC and TSICSNS must be 0x66 (fractional part of (10^9 - 3.6) * 256).

7.11.2 Enabling IEEE 1588 Sub Nanoseconds Timestamp

To enable this feature while configuring the controller in the coreConsultant GUI:

1. Select the **Enable IEEE 1588 Sub Nanoseconds timestamp support** option under the IEEE 1588 Timestamp section during the **Specify Configuration** activity in coreConsultant.

   For details about this option, see IEEE 1588 Timestamp Parameters in "Parameter Descriptions" on page 419.

Note See the "Enabled" field of the parameter descriptions to understand the dependencies.

8 Multiple Channels and Queues Support

This chapter describes how the DWC_ether_qos core supports multiple queues and channels.

This chapter contains the following sections:

Note In multiple queue/channel configurations, for half-duplex operation, enable only the Q0/CH0 on Tx and Rx. For single queue/channel in full-duplex operation, any queue/channel can be enabled.

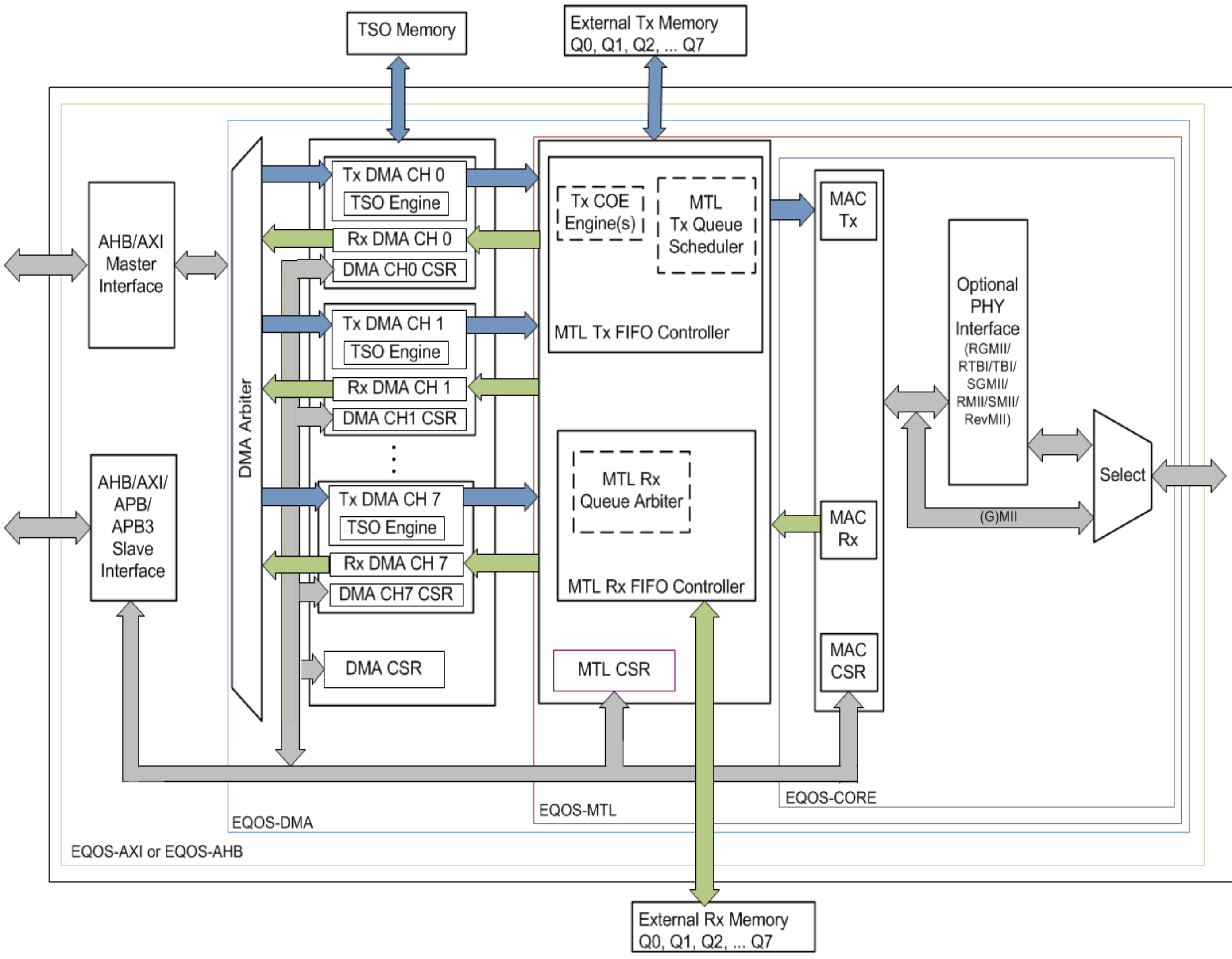
8.1 Block Diagram

The DWC_ether_qos supports up to eight queues and channels on Tx and Rx paths.

Figure 8-1 shows the architecture of DWC_ether_qos with multiple queues and channels.

Note In multi queue/channel configurations, enable only Q0/CH0 on Tx and Rx for half-duplex operation. If you want to enable single queue/channel in full-duplex operation, any queue/channel can be enabled.

Figure 8-1 DWC_ether_qos with Multiple Channels and Queues Block Diagram



8.2 Multiple Queues and Channels Support in the Transmit Path (for EQOS-AHB, EQOS-AXI, and EQOS-DMA)

8.2.1 Introduction to Multiple Queues Support in the Transmit Path

The DWC_ether_qos supports up to eight Transmit queues. The number of Tx channels is equal to the number of Tx queues. This section describes how multiple queues and channels are supported in EQOS-AHB, EQOS-AXI, and EQOS-DMA.

8.2.2 Description of Fixed Priority Scheme in EQOS-AHB and EQOS-DMA

The fixed priority scheme is the default priority scheme for the DMA channels. In fixed priority scheme, the channel with highest priority always wins the arbitration when it requests the bus. Table 8-1 provides information about the priority levels of DMA channels.

Table 8-1 Fixed Priority Scheme for DMA Channels

Priority Level	Channel
0 (low)	Channel 0
1	Channel 1
2	Channel 2
3	Channel 3
4	Channel 4
5	Channel 5
6	Channel 6
7 (high)	Channel 7

8.2.3 Description of Fixed Priority Scheme in EQOS-AXI

In EQOS-AXI configuration, the Fixed Priority scheme does not necessarily cause “starving” of lower priority channels for the AXI bus because of the capabilities of the AXI protocol and characteristics of the TxDMA, described as follows:

- Read command requests from a TxDMA for a burst of data transfers is driven on a separate AXI channel and gets executed in one clock cycle itself. The actual data transfer transfers starts on a separate AXI channel only after the request is accepted and it takes multiple clock cycles to complete with relatively large read access latencies.
- AXI master is free to drive out subsequent requests even before the previous data transfers are completed.
- A TxDMA requests for another data transfer only after the data transfer for its previous request is completed and transferred to the MTL TxQ and space is available in TxQ to take in the next request.
- The EQOS-AXI can be programmed to control the maximum number of outstanding requests, Requested Burst sizes by DMA, and the burst sizes of each data transfer on AXI.

Considering the previous points, the EQOS-AXI can be programmed such that the requests of all the TxDMAs are placed and executed on the AXI bus before the data transfer of the first TxDMA is completed and it raises the next request to the arbiter. For example, in a 64-bit AXI data bus, set the following:

1. TxDMA PBL = 64 (512 bytes)
2. Number of Outstanding AXI Read requests = 16
3. Maximum Burst Length on AXI (BLEN) = 32 (256 bytes)
4. Number of active TxDMA = 8

Assuming that all TxDMAs place a request simultaneously and get serviced by the Fixed priority algorithm, all the requests are placed on the AXI bus in a window of 32 clocks (two clocks per AXI request, two AXI burst requests per TxDMA request). The data transfer of the first request takes 32 clock cycles + the read-access latencies (assuming that there are tens of cycles). So, the TxDMA arbiter is already IDLE and has completed servicing all the TxDMA requests before this data transfer completes.

Effectively, all the TxDMA channels get equal access to AXI bus and the corresponding TxQs get filled up at the same rate. The data gets read out from the TxQs as per the Traffic Class scheduler in the MTL. So at a steady state (assuming that AXI bandwidth available to fetch data from system memory is always more than the operating line rate), the bandwidth for each channel is controlled by the priorities and settings of the Traffic Scheduler in the MTL.

8.2.4 Description of Weighted Strict Priority in EQOS-AHB and EQOS-DMA

In Weighted Strict Priority (WSP), the weight corresponds to the number of burst transfers given to a channel in one arbitration cycle. The unused burst transfers of one or more channels are reallocated based on the priority. The channel with highest priority gets the unused burst transfer time before it is allocated to a channel with next highest priority.

The channel priorities are fixed. Channel 7 has the highest priority and Channel 0 has the lowest priority. When a channel uses the allocated burst transfers, the channel with next lower priority is processed. After processing the allocated bandwidth of all channels that had packets to transmit, any unused burst transfer time is allocated to the channel of the highest priority (if required), and then next highest priority (if required), and so on.

8.2.5 Description of Weighted Round Robin in EQOS-AHB and EQOS-DMA

EQOS-AHB and EQOS-DMA configurations support a Weighted Round Robin (WRR) priority scheme, in which the weight can be programmed through the DMA_CH#_Tx_Control register.

In Weighted Round Robin (WRR), all channels are serviced in round-robin order according to the weights settings. The TCW field of the DMA_CH#_Tx_Control register provides the weight for each Transmit channel as shown in Table 8-2.

Table 8-2 Weight for DMA Channels

TCW Field	Transmit Channel Weight
000	1
001	2
010	3
011	4

TCW Field	Transmit Channel Weight
100	5
101	6
110	7
111	8

The configured weights correspond to the number of burst transfers given to a channel in one arbitration cycle. The unused or excess burst transfers are distributed equally to all channels.

8.2.6 Selecting the Number of Transmit Queues

To select the number of transmit queues for the DWC_ether_qos controller, complete the following steps:

1. Select the number of Transmit queues in the **Number of Transmit Queues** option under the Multiple Queues Support section during the Specify Configuration activity.

Note See the “Enabled” field of the parameter descriptions to understand the dependencies

2. Perform the relevant steps of “Programming Guidelines for Multi-Channel Multi-Queuing” on page 1364

For details, see the “Parameter Descriptions” on page 419.

8.2.7 Programming Guidelines for Multiple Queues and Channels in the Transmit Path

See, “Programming Guidelines for Multi-Channel Multi-Queuing” on page 1364.

8.3 Multiple Queues and Channels Support in the Receive Path

The DWC_ether_qos supports up to eight Receive channels; the number of Rx channels is independent of the Rx queues.

8.3.1 Introduction to Multiple Queues in the Receive Path

The DWC_ether_qos supports up to eight Receive channels in EQOS-AHB, EQOS-AXI, and EQOS-DMA configurations. The number of Rx channels is independent of the Rx queues.

8.3.2 Description of Multiple Queues in the Receive Path

In the Rx direction, the MTL Rx Controller selects the Rx DMA for which it is transferring or reading the data from the Rx FIFO memory. This scheduling is based on the programming done in the respective MTL_RxQ[x]_Control register.

Each Rx DMA indicates when it is ready to transfer data and the size of the burst-length (number of beats) that it has to transfer. The scheduler checks whether sufficient data (of requested burst length) is available to be transferred to these DMAs and then selects the Rx DMA that gets serviced using the programmed priorities.

8.3.2.1 Priority Scheme for Tx DMA and Rx DMA

The DWC_ether_qos DMA arbiter supports two types of arbitration: fixed priority and weighted round-robin.

The DMA arbiter performs the arbitration between the Tx and Rx paths of DMA channels for accessing descriptors and data buffers. The DMA arbiter supports two types of arbitration: fixed priority and weighted round-robin. The DA bit of the DMA_Mode register specifies the arbitration scheme (fixed or weighted round-robin) between the Tx and Rx DMA of a channel.

If you have enabled the Tx DMA and Rx DMA of a channel, you can specify which DMA gets the bus when the channel gets the control of the bus. You can set the priority between the corresponding Tx DMA and Rx DMA by using the TXPR field of the DMA_Mode register. For round-robin arbitration, you can use the PR field of the DMA_Mode register to specify the weighted priority between the Tx DMA and Rx DMA.

Table 8-3 provides information about the priority scheme between Tx DMA and Rx DMA.

Table 8-3 Priority Scheme for Tx DMA and Rx DMA

Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Priority Schemes
x	x	x	0	1	Rx always has priority over Tx
0	0	0	0	0	Tx and Rx have equal priority. Rx gets the access first on simultaneous requests
0	0	1	0	0	Rx has priority over Tx in ratio 2:1
0	1	0	0	0	Rx has priority over Tx in ratio 3:1
0	1	1	0	0	Rx has priority over Tx in ratio 4:1
1	0	0	0	0	Rx has priority over Tx in ratio 5:1
1	0	1	0	0	Rx has priority over Tx in ratio 6:1

Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Priority Schemes
1	1	0	0	0	Rx has priority over Tx in ratio 7:1
1	1	1	0	0	Rx has priority over Tx in ratio 8:1
x	x	x	1	1	Tx always has priority over Rx
0	0	0	1	0	Tx and Rx have equal priority. Tx gets the access first on simultaneous requests
0	0	1	1	0	Tx has priority over Rx in ratio 2:1
0	1	0	1	0	Tx has priority over Rx in ratio 3:1
0	1	1	1	0	Tx has priority over Rx in ratio 4:1
1	0	0	1	0	Tx has priority over Rx in ratio 5:1
1	0	1	1	0	Tx has priority over Rx in ratio 6:1
1	1	0	1	0	Tx has priority over Rx in ratio 7:1
1	1	1	1	0	Tx has priority over Rx in ratio 8:1

Note Bits 14, 13, 12, 11, and 1 are not valid if one of the paths (Transmit or Receive) is not present in a channel. The Transmit and Receive paths are always present in Channel 0.

8.3.3 Selecting the Number of Receive Queues and Channels

Select the number of queues in the Receive path while configuring the controller in the coreConsultant GUI.

1. Select the number of Receive queues in the **Number of Receive Queues** option under the Multiple Queues Support section during the **Specify Configuration** activity.

   For details about this option, see Multiple Queues Support Parameters in the "Parameter Descriptions" on page 419.

   Note See the "Enabled" field of the parameter descriptions to understand the dependencies

2. Perform the relevant steps of "Programming Guidelines for Multi-Channel Multi-Queuing" on page 1364.

8.3.4 Programming Guidelines for Multiple Queues and Channels in the Receive Path

See "Programming Guidelines for Multi-Channel Multi-Queuing" on page 1364.

8.4 Multiple Queues and Channels Support in EQOS-MTL

8.4.1 Introduction to Multiple Queues and Channels Support in EQOS-MTL

The DWC_ether_qos supports up to eight Tx and Rx queues. This section describes how multiple queues and channels are supported in MTL.

You can control a Tx queue through the MTL_TxQ#_Operation_Mode register. You can control an Rx queue through the MTL_RxQ#_Operation_Mode register.

8.4.2 Queue Memory

All MTL Tx queues share one Tx memory and all Rx queues share one Rx memory with programmable individual queue sizes in blocks of 256 bytes. The application can program the Tx queue size in the TQS field of MTL_TxQ#_Operation_Mode register and the Rx queue size in the RQS field of the MTL_RxQ#_Operation_Mode register.

The DWC_ether_qos supports the total Tx memory size of 128 KB (16 KB * 8) and total Rx memory size of 256 KB (32 KB * 8). The total memory size can be selected in powers of 2, that is, 1 KB, 2 KB, 4 KB, 8 KB, 16 KB, 32 KB, 64 KB, 128 KB, and 256 KB. In configurations with a single queue, the maximum Tx memory size is 32 KB and the maximum Rx memory size is 64 KB. In configurations with a single queue, the FIFOs size of 128 bytes, 256 bytes, and 512 bytes is also supported.

8.4.3 Selecting the Number of Transmit Queues in EQOS-MTL

Select the number of transmit channels while configuring the controller in the coreConsultant GUI.

1. Select the number of Transmit queues in the **Number of Transmit Queues** option under the Multiple Queues Support section during the **Specify Configuration** activity.

   For details about this option, see Multiple Queues Support Parameters in the "Parameter Descriptions" on page 419.

Note See the "Enabled" field of the parameter descriptions to understand the dependencies

2. Perform the relevant steps of "Programming Guidelines for Multi-Channel Multi-Queuing" on page 1364.

8.5 Rx Queue to DMA Mapping

The packets in the MTL Rx Queues can be routed to any one of the multiple DMA channels by programming the following registers:

- MTL_RxQ_DMA_Map0 Register (for queues 0, 1, 2 and 3)
- MTL_RxQ_DMA_Map1 Register (for queues 4, 5, 6 and 7)

The following types of Rx Queue to DMA mapping is possible through programming:

- Static Mapping
- Dynamic (Per Packet) Mapping

8.5.1 Static Mapping

In Static Mapping mode, all packets of an Rx Queue are connected to a specific DMA channel.

In this mode, all of the packets of an Rx Queue are connected to a specific DMA channel. For example, all the packets from Rx Queue 0 can be routed to a DMA channel by programming Q0MDMACH (bit[3:0]) and Q0DDMACH (bit[7] = 0) of the MTL_RxQ_DMA_Map0 Register.

Similarly, packets from other Rx Queues can be routed to any DMA channel by programming register fields corresponding to each Queue.

8.5.2 Dynamic (Per Packet) Mapping

In Dynamic (per packet) Mapping mode, the destination DMA channel is decide by the MAC core receiver for each packet.

In this mode, the destination DMA channel of a packet being read from a Rx Queue is not constant but decided independently for each packet. For example, if you set the Q1DDMACH bit of the MTL_RxQ_DMA_Map0 register, the static mapping is disabled for Rx Queue 1 and the value in Q1MDMACH is ignored. The destination DMA channel is decided by the MAC receiver for each packet, depending on the following in decreasing order of priority:

1. L3-L4 filter Based DMA Selection

   When the Enable Layer 3 and Layer 4 Packet Filter feature is present and enabled in your configuration, the TCP/UDP and IP header fields of the received packet are matched against the corresponding values programmed and enabled for comparison in the MAC_L3_L4_Address_Control register. If the match is successful, the DMA channel number programmed in the DMCHN field of the MAC_L3_L4_Address_Control register is selected as the destination DMA channel number provided DMCHEN bit of the same register is set.

   If none of the L3-L4 Registers give a comparison match, then DWC_ether_qos proceeds to the next step below.

2. Extended VLAN Based DMA Selection

   Extended routing is applicable only if the VLAN Filter has passed. Routing is only based on Perfect Filter Result. Each perfect filter has a DMA Channel Enable and a DMA Channel Number field which can be programmed. Routing is applicable for a filter, only if the DMA Channel Enable bit is set.

   The frame is routed to the smallest Matching Filter's DMA Channel provided it is enabled. If that filter's DMA Channel number is not enabled, the frame gets routed to Channel 0. For example, if a frame's VLAN tag matches filters 7, 3, and 1. Then the MAC checks if Filter 1's DMA Channel Number is enabled through programming. If yes, the frame gets routed to the programmed value; otherwise it gets routed to DMA. When the inverse filter is enabled; is routed to the least mismatched filter's DMA channel number provided it is enabled.If the DMA Channel enable bit is not set, then the frame is routed based on DA based addressing or to Channel 0.

If Hash Filter is also enabled, it is used to determine the Filter result only. Routing will still depend on the Perfect Filters enabled. If none of the perfect filters are enabled or if all of them are bypassed, then the VLAN Filter based routing will not take place. The frame will be routed via DA Based addressing or to Channel 0. If all the perfect filters give a fail result and the Hash Filter has passed, then the VLAN Filter result is a pass but routing will be based on DA Based Addressing or to Channel 0. Similar behavior is seen when inverse Filtering is enabled as well.

3. Ethernet DA-Based DMA Selection

   The DA address of the received packet is compared against the programmed DA values in MAC Address Registers. If the address matches any of the programmed values, the corresponding DCS field (when enabled) determines the destination DMA channel number.

If none of the previous operations is able to make a successful match/decision, then the packet is routed to DMA Channel 0 by default.

8.5.2.1 Broadcast/Multicast Packet Duplication

Broadcast/Multicast Packet Duplication feature provides a mechanism to send the received Broadcast/Multicast packets to multiple DMA channels.

In a virtual system that supports multiple guest OS's (for instance, each OS owning a DMA), where each DMA could act as an independent Ethernet Node, there can be requirement to route the received multicast/broadcast packets to all/multiple DMA channels. This is implemented by having the MTL_RX_FIFO_read_controller to read the packet from the same queue multiple times and route the same packet to the specified multiple DMA channels.

For this enhancement, to maintain backward compatibility, the DMA channel selection (DCS) field inside the MAC Address Registers are redefined to denote multiple DMA channels through software control.

Note Only the MAC_Address(#i)_High (i=0 to 31) registers support Broadcast/Multicast packet duplication

8.5.2.1.1 Functional Changes to the MAC

The DCS field of the of MAC_Address0_High and other optional MAC_Address(#i)_High (i=1 to 127) registers determines the DMA channel number to which the received packet (that matches the MAC Address present in that register) must be routed.

With this enhancement, for MAC_Address0_High and other optional MAC_Address(#i)_High (i=1 to 31) registers, DCS is a programmable field that allows more than one DMA channel number to be selected.

The DCS field can hold any one of the following values:

- Binary value of the channel number when the PDC bit of the MAC_Ext_Configuration register is set to 0.
- One-hot value of the channel number when PDC bit of the MAC_Ext_Configuration register is set to 1.

To enable the Multicast/Broadcast packets to be routed to multiple DMA channels, use the DA based routing. Packet duplication is not supported in other DMA channel routing mechanisms such as Extended VLAN, or L3-L4 based routing. In other words, if the destination DMA channel is decided by the VLAN or

L3-L4 filter result, it is routed only to the specific channel even if the DA of the packet matches the multicast address in the MAC_Address register.

The Packet Duplication feature is supported only on the highest MTL Queue configured. Therefore, packet duplication for Broadcast/Multicast packets requires that the MCBCQ field of MAC_RxQ_Ctrl1 register is programmed to the highest RxQ present in the configuration.

8.5.2.1.2 Functional Changes to MTL

When more than one bit is set in the mri_dmach_sel_i (DMA channel number) signal on the MRI Interface, MTL can forward a packet to more than one DMA channel.

In a packet duplication, the MTL Read Controller sequentially transfers the same packet to all the required RxDMA. This packet is removed from the MTL FIFO only after the packet is transferred to all the selected DMA channels. Therefore this feature is suitable for smaller packet sizes or very low bandwidth traffic; to ensure that the MTL RxQ does not overflow due to the long period for which the packet remains in the queue during the packet duplication.

Note When more than one bit is set for the DMA channel number, a packet that is not routed to the highest receive queue is routed to only one DMA channel as indicated by the lowest bit set in the mri_dmach_sel_i signal.

8.6 Selection of Tag Priorities Assigned to Tx and Rx Queues

The DWC_ether_qos controller supports assigning tag priorities to Tx and Rx Queues through programming registers.

The VLAN Tag priorities can be assigned to Tx Queues by programming the corresponding PSTQ# field in the following registers:

- MAC_TxQ_Prty_Map0 (for Tx Queues 0 to 3)
- MAC_TxQ_Prty_Map1 (for Tx Queues 4 to 7)

The bit corresponding to the VLAN Tag priority can be set in the PSTQ# field for assigning that priority to the Tx Queue. For example, if you want to assign VLAN Tag priority of 3 to Tx Queue 0, set bit [3] in PSTQ0 field.

The same VLAN Tag priority can be set for multiple Tx queues. The Tx packet association to particular Tx Queue is governed by the application, so the MAC does not route the packet based on Tx Queue priority mapping; it is only used for Pause Flow Control (PFC). The settings in the PSTQ# field determines the Tx Queues blocked for transmission when the PFC packet is received with corresponding VLAN Tag priorities enabled.

The VLAN Tag priorities can be assigned to Rx Queues by programming the PSRQ field in the corresponding MAC_RxQ_Ctrl2 and MAC_RxQ_Ctrl3 registers. The bit corresponding to the VLAN Tag priority can be set in the PSRQ field for assigning that priority to the Rx Queue. For example, if you want to assign VLAN Tag priority of 3 to Rx Queue 0, set bit [3] in PSRQ field of MAC_RxQ_Ctrl2 register. The VLAN Tag priority assigned to particular Rx Queue must be unique, that is, more than one Rx Queue cannot be assigned the same VLAN Tag priority. However, more than one VLAN Tag priorities can be assigned to same Rx Queue.

The settings in the PSRQ field is used for VLAN Tagged Rx packet routing to Rx Queues as well as for PFC based Tx flow control. The received VLAN Tagged Rx packet is routed to Rx Queue that has the VLAN Tag priority match. In PFC based Tx flow control, PSRQ field corresponding to a particular Rx Queue is used for enabling VLAN Tag priorities in the PFC packet transmitted when corresponding Rx Queue threshold levels are reached.

8.7 Rx Side Routing from MAC to Queues

The DWC_ether_qos supports Rx Side Routing from the MAC to Queues.

The MAC routes the Rx packets to the Rx Queues based on following packet types in that order

- Unicast/Multicast destination address packets that fail the destination address filter
- Multicast/Broadcast destination address packets that pass the destination address filter
- VLAN Tag Priority field in VLAN Tagged AV data packets
- AV Control packets
- VLAN Tagged IEEE 1588 PTP over Ethernet packets
- Untagged IEEE1588 PTP over Ethernet packets
- VLAN Tag Priority field in VLAN Tagged packets
- Data Center Bridging (DCB) related Link Layer Discovery Protocol (LLDP) packets
- Untagged packets

The outer C-VLAN Tagged AV data Rx packets can be routed based on the priorities assigned to Rx Queues through PSRQ field in corresponding MAC_RxQ_Ctrl2 and MAC_RxQ_Ctrl3 registers and the corresponding Rx Queue is enabled for AV through RXQ#EN field in MAC_RxQ_Ctrl0 register. These packets may be single VLAN Tagged with C-VLAN type or Double VLAN Tagged with outer VLAN Tag of C-VLAN type when Double VLAN feature is enabled (EDVLP bit in MAC_VLAN_Tag register is set) with inner C-VLAN Tagged or inner S-VLAN Tagged when SVLAN processing is enabled (ESVL bit in MAC_VLAN_Tag register is set). This type of Rx packet routing is available when AV feature and Multiple Rx Queues are selected in the configuration.

The AV Control Rx packets can be routed based on the Rx Queue number specified in the AVCPQ field in MAC_RxQ_Ctrl1 register and corresponding Rx Queue is enabled for AV through RXQ#EN field in MAC_RxQ_Ctrl0 register. These packets may be single VLAN Tagged with C-VLAN type or Double VLAN Tagged with outer VLAN Tag of C-VLAN type when Double VLAN feature is enabled (EDVLP bit in MAC_VLAN_Tag register is set) with inner C-VLAN Tagged or inner S-VLAN Tagged when SVLAN processing is enabled (ESVL bit in MAC_VLAN_Tag register is set). This type of Rx packet routing is available when AV feature and Multiple Rx Queues are selected in the configuration.

The VLAN Tagged Rx packets can be routed based on the priorities assigned to Rx Queues through PSRQ field in corresponding MAC_RxQ_Ctrl2 and MAC_RxQ_Ctrl3 registers and the corresponding Rx Queue is enabled for DCB/generic through RXQ#EN field in MAC_RxQ_Ctrl0 register. This type of Rx packet routing is available when Multiple Rx Queues are selected in the configuration.

The Data Center Bridging (DCB) related Link Layer Discovery Protocol (LLDP) Rx packets can be routed based on the Rx Queue number specified in the DCBCPQ field in MAC_RxQ_Ctrl1 register and corresponding Rx Queue is enabled for DCB through RXQ#EN field in MAC_RxQ_Ctrl0 register. These packets may be single VLAN Tagged or Double VLAN Tagged when Double VLAN feature is enabled (EDVLP bit in MAC_VLAN_Tag register is set), or C-VLAN Tagged or S-VLAN Tagged when SVLAN processing is enabled (ESVL bit in MAC_VLAN_Tag register is set). This type of Rx packet routing is available when Data Center Bridging feature and Multiple Rx Queues are selected in the configuration.

The untagged IEEE 1588 PTP over Ethernet Rx packets can be routed based on the Rx Queue number specified in the PTPQ field in MAC_RxQ_Ctrl1 register and the corresponding Rx Queue is enabled through RXQ#EN field in MAC_RxQ_Ctrl0 register. The VLAN tagged IEEE 1588 PTP over Ethernet Rx packets can be routed based on the priorities assigned to Rx Queues through PSRQ field in corresponding MAC_RxQ_Ctrl2 and MAC_RxQ_Ctrl3 registers or the Rx Queue number specified in the PTPQ field in

MAC_RxQ_Ctrl1 register and the corresponding Rx Queue is enabled through RXQ#EN field in MAC_RxQ_Ctrl0 register. This is determined by programming in TPQC field of MAC_RxQ_Ctrl1 register. This type of Rx packet routing is available when IEEE 1588 Timestamp feature support and Multiple Rx Queues are selected in the configuration. The VLAN Tagged IEEE 1588 PTP over Ethernet Rx packets are detected only when 802.1AS mode is disabled (AV8021ASMEN bit in MAC_Timestamp_Control register is set to 0), otherwise this type of Rx packets are routed as generic VLAN Tagged Rx packets.

The multicast/broadcast destination address Rx packets that pass the destination address filter can be routed based on the Rx Queue number specified in the MCBCQ field of MAC_RxQ_Ctrl1 register when enabled through MCBCQEN bit of MAC_RxQ_Ctrl1 register and the corresponding Rx Queue is enabled through RXQ#EN field in MAC_RxQ_Ctrl0 register. This type of Rx packet routing is available when Multiple Rx Queues are selected in the configuration.

The untagged Rx packets can be routed based on the Rx Queue number specified in the UPQ field of MAC_RxQ_Ctrl1 register and the corresponding Rx Queue is enabled through RXQ#EN field in MAC_RxQ_Ctrl0 register. This type of Rx packet routing is available when Multiple Rx Queues are selected in the configuration.

The unicast destination address Rx packets that fail the destination address filter can be routed based on the Rx Queue number specified in the UFFQ field of MAC_RxQ_Routing_Ctrl register when enabled through UFFQE bit of MAC_RxQ_Routing_Ctrl register, RA bit of MAC_Packet_Filter register is set to 1 and the corresponding Rx Queue is enabled through RXQ#EN field in MAC_RxQ_Ctrl0 register. This type of Rx packet routing is available when Multiple Rx Queues are selected in the configuration.

The multicast destination address Rx packets that fail the destination address filter can be routed based on the Rx Queue number specified in the MFFQ field of MAC_RxQ_Routing_Ctrl register when enabled through MFFQE bit of MAC_RxQ_Routing_Ctrl register, RA bit of MAC_Packet_Filter register is set to 1 and the corresponding Rx Queue is enabled through RXQ#EN field in MAC_RxQ_Ctrl0 register. This type of Rx packet routing is available when Multiple Rx Queues are selected in the configuration.

If the Rx packet cannot be classified in any of the defined packet type for routing, it will be routed through Rx Queue 0.

8.8 Rx Side Arbitration Between DMA and MTL

The DWC_ether_qos controller supports Rx side arbitration between the DMA and the MTL.

After completing the current packet processing, the DMA Channel Controller fetches the next descriptor. After the descriptor fetching is complete, the DMA Channel Controller evaluates the amount of data to be to be transferred to the Rx Buffer based on the programmed PBL and Rx Buffer Length. Accordingly, it requests the MTL to transfer the data.

After servicing the current request, the MTL Rx Queue arbitration scheme selects Rx Queue based on the arbitration scheme (RAA field of the MTL_Operation_Mode register) and the weights programmed in the Queue <n> Receive Control Register. The arbitration is done among Queues for which DMA is ready to service. After the Rx Queue is selected, PBL (Programmable Burst Length) amount of data is read out from that Queue and is routed to the Rx DMA Channel based on the Rx Channel selection criteria.

The arbitration takes place when every PBL data transfer is completed and descriptors are ready for the processing from at least one DMA Channel.

8.9 Tx Side Arbitration between DMA and MTL

The DWC_ether_qos support Tx side arbitration between the DMA and the MTL.

Because the number of Tx DMA channels is always equal to the number of Tx Queues in the MTL, they are always mapped directly. For instance, each Tx DMA pushes data into its respective Tx Queue assigned to it. The main reason for this strategy is to optimize and increase the efficiency of the DMA data transfer, as well as to simplify the TX Checksum Engines in the MTL.

The data inside each Tx Queue is stored in packets. Hence, if two DMAs are allowed to transfer data into the same queue, when a Tx DMA starts a packet transfer, the other DMA cannot transfer data unless the previous packet is completely pushed-in. This means that the second DMA remains idle until the first packet is transferred. Hence, each DMA is always connected directly to its corresponding Tx Queue.

8.10 Data Centre Bridging

8.10.1 Introduction to Data Centre Bridging

8.10.2 Description of Data Center Bridging Feature

The Data Center Bridging feature supports three types of algorithms: weighted round robin, deficit weighted round robin, and weighted fair queuing.

For the Data Center Bridging, the following algorithms are used:

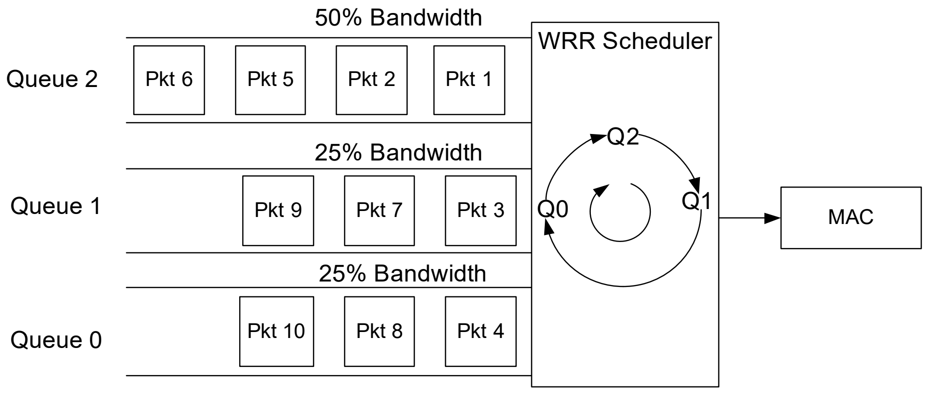
- Weighted Round Robin
- Deficit Weighted Round Robin
- Weighted Fair Queuing

8.10.2.1 Weighted Round Robin

In the Weighted Round Robin algorithm used in the Data Center Bridging feature, all queues are serviced in round-robin order based on weight setting.

In Weighted Round Robin (WRR) algorithm, each queue is assigned a weight based on the percentage of configured bandwidth. All queues are serviced in the round-robin order according to the weight settings. This algorithm is less complex and requires less hardware as compared to other algorithms. Large packets get more bandwidth in this algorithm.

Figure 8-2 Weighted Round Robin Example



In Figure 8-2, Queue 2, Queue 1 and Queue 0 are configured for 50%, 25%, and 25% bandwidth, respectively. The corresponding weights are 2, 1, and 1. Before the WRR scheduler is started, Queue 2 has four packets; Queue 1 has three packets; and Queue 0 has three packets. The packets are transmitted in the following order:

Table 8-4 Weighted Round Robin Sequence Example

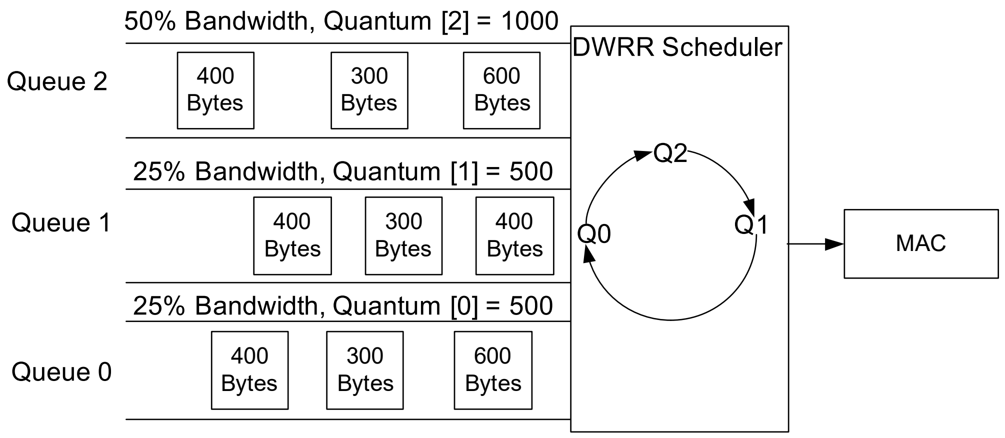
Packet Sequence	Queue Number
Packet 1	Queue 2
Packet 2	Queue 2
Packet 3	Queue 1
Packet 4	Queue 0
Packet 5	Queue 2
Packet 6	Queue 2
Packet 7	Queue 1
Packet 8	Queue 0
Packet 9	Queue 1
Packet 10	Queue 0

8.10.2.2 Deficit Weighted Round Robin

When the Data Center Bridging feature is supported, you can use the deficit weighted round robin algorithm to assign a weight based on the percentage of configured bandwidth. A deficit counter holds the credit for transmission.

In Deficit Weighted Round Robin (DWRR) algorithm, each queue is assigned a weight based on the percentage of configured bandwidth. A deficit counter holds the credit for transmission. The quantum value, proportional to the bandwidth, is added to the deficit counter every time a queue is scanned. The packet in the queue is selected for transmission only if the credit in the deficit counter is less than or equal to the size of the packet available at the head of the queue. A selected queue is serviced until the outstanding credit is lesser than the packet size at the head of the queue during a particular queue scan cycle. After this, the next lower priority queue is serviced. When the packets in a queue are over, the credit becomes zero for that queue scan cycle. This algorithm is less complex, and it provides more fairness in scheduling as compared to other algorithms.

Figure 8-3 Deficit Weighted Round Robin Example



In Figure 8-3, Queue 2, Queue 1, and Queue 0 are configured for 50%, 25%, and 25% bandwidth, respectively. The corresponding quantum values are 1000, 500, and 500. Before the DWRR scheduler is started, each queue has three packets.Table 8-5 on page 297 shows the queues scan cycle number and the packet size at the head of queues, deficit counter value before and after packet in selected queue is transmitted, and queue scheduled for transmission.

Table 8-5 Deficit Weighted Round Robin

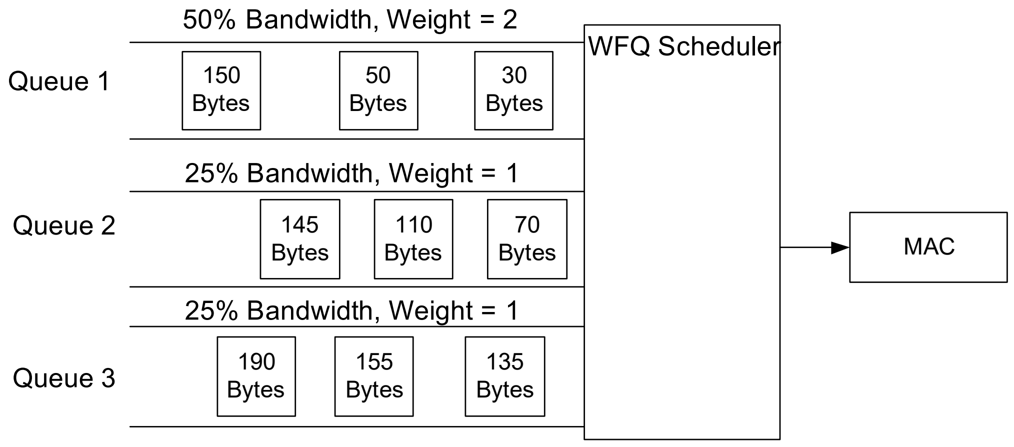
Queues Scan Cycle Number	Packet Size at the Head of Queue			Deficit Counter Value Before the Packet is Transmitted in Selected Queue			Deficit Counter Value after the Packet is Transmitted in Selected Queue			Queue Scheduled for Transmissi on
	Queue2	Q1	Q0	Q2	Q1	Q0	Q2	Q1	Q0	
1	600	400	600	1000	500	500	400	500	500	Q2
	300	400	600	400	500	500	100	500	500	Q2
	400	400	600	100	500	500	100	100	500	Q1
	400	300	600	100	100	500	100	100	500	None
2	400	300	600	1100	600	1000	700	600	1000	Q2
	N/A	300	600	0	600	1000	0	300	1000	Q1
	N/A	400	600	0	300	1000	0	300	400	Q0
	N/A	400	300	0	300	400	0	300	100	Q0
	N/A	400	400	0	300	100	0	300	100	None
3	N/A	400	400	0	800	600	0	400	600	Q1
	N/A	N/A	400	0	N/A	600	0	0	200	Q0
	N/A	N/A	N/A	0	N/A	N/A	0	0	0	None

8.10.2.3 Weighted Fair Queuing

The weighted fair queuing algorithm, used in the Data Center Bridging feature, assigns each queue a weight based on the percentage of the configured bandwidth and schedules the packet with the fastest finish time to start transmission first.

In Weighted Fair Queuing (WFQ) algorithm, each queue is assigned a weight based on the percentage of configured bandwidth. The algorithm computes the finish time of packets at the head of the queues based on the packet size and weights for the queue. The packets with earliest finish time is scheduled for transmission first. The algorithm is more complex and less scalable as compared to other algorithms.

Figure 8-4 Weighted Fair Queuing Example



In Figure 8-4, Queue 2, Queue 1, and Queue 0 are configured for 50%, 25%, and 25% bandwidth, respectively. The corresponding weights are 2, 1, and 1. Before the WFQ scheduler is started, each queue has three packets. Table 8-6 shows the packet size at the head of the queue, computed finish time, and the queue scheduled for transmission.

Table 8-6 Weighted Fair Queuing Algorithm Example Values

Packet size at the Head of the Queue			Computed Finish Time in Clock Cycles (packet size or weights)			Queue Scheduled for Transmission
Q2	Q1	Q0	Q2	Q1	Q0	
30	70	135	15	70	135	Q2
50	70	135	25	70	135	Q2
150	70	135	75	70	135	Q1
150	110	135	75	110	135	Q2
N/A	110	135	N/A	110	135	Q1
N/A	145	135	N/A	145	135	Q0
N/A	145	155	N/A	145	155	Q1
N/A		155	N/A	N/A	155	Q0
N/A		190	N/A	N/A	190	Q0

8.10.3 Enabling Data Center Bridging

To enable this feature while configuring the controller in the coreConsultant GUI:

1. Select the **Data Center Bridging** option under the Multiple Queues Support section during the **Specify Configuration** activity.

For details about these option, see Multiple Queues Support Parameters in the "Parameter Descriptions" on page 419.

Note See the "Enabled" field of the parameter descriptions to understand the dependencies

8.11 Audio Video Bridging

The DWC_ether-qos supports the Audio Video Bridging. The AV feature enables transmission of time-sensitive traffic over bridged local area networks (LANs).

This section includes the following topics:

- "Introduction to AV Feature" on page 300
- "Transmit Path Functions" on page 300
- "Receive Path Functions" on page 302
- "Credit-Based Shaper Algorithm" on page 302
- "Enabling AV Bridging" on page 304
- "Slot Number Function and Audio Video Bridging" on page 304

While the AVB standards were originally designed for Audio and Video over standard Ethernet, extending these to control streams would additionally benefit the industrial and automotive sectors. Time - Sensitive Networking (TSN) is a set of standards developed by the Time - Sensitive Networking Task Group with the intent to extend the future AVB enhancements to other sectors that could benefit.

Bridges are increasingly used to interconnect devices that support scheduled applications (for example, industrial automation, process control and vehicle control). The IEEE 802.1Qbv-2015 (Enhancements to Scheduling Traffic) provides performance assurances of latency and delivery variation to enable these applications in an engineered LAN while maintaining the existing guarantees for the credit-based shaper and best-effort traffic.

The IEEE 802.3br (Interspersing Traffic) and IEEE 802.1Qbu (Frame Preemption) will enable the suspension of a large preemptable packet being transmitted by the MAC layer to allow one or more express packets to be transmitted before the transmission of the preemptable packet is resumed. This will provide capability to schedule express traffic packets with minimal delays/latencies or at predictable times with efficient utilization of the line bandwidth.

As per the IEEE specification for 802.1Qbv, 802.1Qbu, and 802.3br, the TSN features are supported only in the following modes:

- Full duplex
- 100 Mbps or higher link speed
- Pause or no flow control

  When Qbv schedule is enabled for time-sensitive traffic, Pause or PFC is likely to interfere with the Qbv schedules. So, it is recommended not to enable PFC/PAUSE when Qbv scheduler is enabled.
- MTL or higher configurations
- Two or more Tx queues (for Qbv) configured and enabled
- One or more Express Queues and one or more Preemption queues on both transmit and receive (for Qbu/802.3br) configured and enabled.

8.11.1 Introduction to AV Feature

The DWC_ether_qos supports the AV data transfer in 100 Mbps and 1000 Mbps modes.

The DWC_ether_qos supports the AV data transfer in 100 Mbps and 1000 Mbps modes. The AV feature enables transmission of time-sensitive traffic over bridged local area networks (LANs). The following standards define various aspects of the AV feature implementation:

- IEEE 802.1Qav-2009: Allows the bridges to provide time-sensitive and loss-sensitive real-time audio video data transmission (AV traffic). It specifies the priority regeneration and controlled bandwidth queue draining algorithms that are used in bridges and AV traffic sources.
- IEEE 802.1Qat-2009: Allows the network resources to be reserved for specific traffic streams traversing a bridged local area network.
- IEEE 802.1AS-2011: Specifies the protocol and procedures used to ensure that the synchronization requirements are met for time-sensitive applications such as audio and video across bridged and virtual-bridged LANs consisting of LAN media where the transmission delays are fixed and symmetrical. For example, IEEE 802.3 full-duplex links include the maintenance of synchronized time during normal operation followed by addition, removal, or failure of network components and network reconfiguration.

Note In EQOS-CORE configurations, the AV feature is limited to identifying the AV Type payloads in the Ethernet packet and indicating it on the MAC Receive Interface (MRI) through mri_q_sel_o signal along with the SOP. For more information, see "Receive Path Functions" on page 302.

8.11.2 Transmit Path Functions

When the DWC_ether_qos controller supports the AV Bridging feature, you can enable and disable certain functions associated with the transmit path.

When you select the AV feature, the Transmit paths of Queue 0 and the highest enabled queue are enabled by default.

The Transmit path of Queue 0 supports the strict priority algorithm, and it is used for best-effort traffic. For a queue, the strict priority algorithm determines that a packet is available for transmission if the queue contains one or more packets. When the threshold mode for MTL Tx FIFO is enabled, the strict priority algorithm determines that a packet is available for transmission if the queue contains a partial packet of size equal to the programmed threshold limit.

The Transmit paths of additional queues support traffic management by using the credit-based shaper algorithm. For a queue, the credit-based shaper algorithm determines that a queue is available for transmission if the following conditions are true:

- The queue contains one or more packets.
- The credit for the queue is positive as per the algorithm.

You can disable the credit-based shaper algorithm for all queues or lower-priority queues. For example, you can either disable the credit-based shaper algorithm for Queue 1 and Queue 2 or only for Queue 1. You should not disable the credit-based shaper algorithm for Queue 2 and enable it for Queue 1. When you disable the credit-based shaper algorithm for a queue, the channel uses the default strict priority algorithm.

Each Transmit DMA has a separate descriptor chain for fetching the transmit data. The Transmit channel that gets the access to the system bus depends on the DMA arbiter.

The Transmit path has a shared FIFO (MTL layer) for each queue. The data fetched by the DMA is put in the respective part of the FIFO. The MTL Tx Queue Scheduler controls which part of the FIFO data is transmitted by the MAC. If the credit-based shaper algorithm is enabled for a queue, the corresponding queue is selected for transmission if the following conditions are true:

- If the packet is available in the channel and has a positive or zero credit.
- If the higher priority queue has no packet waiting in the FIFO.

If the credit-based shaper algorithm is disabled for all queues, the packet to be transmitted from a queue is selected based on the priority scheme described in Table 8-7.

Table 8-7 Weight for DMA Channels

TCW Field	Transmit Channel Weight
000	1
001	2
010	3
011	4
100	5
101	6
110	7
111	8

8.11.3 Receive Path Functions

When the DWC_ether_qos controller supports the AV Bridging feature, you can enable and disable certain functions associated with the receive path.

When you select the AV feature, the Receive path of Queue 0 is enabled by default. All traffic is received on this channel. You can enable the Receive paths of additional queues. By enabling the Receive paths of multiple channels, you can demultiplex the received data to send the packets into separate receive channels.

To differentiate between the AV and non-AV traffic, the MAC provides a status that indicates if it is an AV packet and its corresponding VLAN Priority tag value. This status is updated in the Extended Status field of the Receive descriptor as explained in "Receive Descriptor" on page 1337. All received packets with the EtherType field of 0x22F0 are detected as AV packets. The AV packets can be of the following two types:

- AV data packets: The AV data packets are always tagged. The tagged AV data packets are received based on the programmed priority value. To specify the channel to which an AV packet with a given priority must be sent, program Bits[15:8] in the Transmit Flow Control Register of the corresponding queue.
- AV control packets: The AV control packets can be either tagged or untagged. The untagged AV control packets are received on Queue 0 by default. To receive these packets on any other queue, you can program Bits[2:0] of the MAC_RxQ_Ctrl1 register. Similar to the AV data packets, the tagged AV control packets are received based on the programmed priority value.

In addition to the AV packets, you can receive the untagged PTP packets on any queue. By default, the PTP packets (tagged or untagged) are received on Queue 0. To receive these packets on any other queue, you need to program Bits[6:4] of the MAC_RxQ_Ctrl1 register.

8.11.4 Credit-Based Shaper Algorithm

The MTL Queue Scheduler (shown in Figure 1-1 on page 24) uses the credit-based shaper algorithm to arbitrate the AV traffic in all queues and the legacy Ethernet traffic in Queue 0. You can program the additional queues to use the credit-based shaper algorithm.

The following sections provide information about how you can implement the credit-based shaper algorithm:

- Credit Value
- idleSlopeCredit and sendSlopeCredit Values
- Bandwidth Status

8.11.4.1 Credit Value

The credit value is part of the credit-based sharper algorithm used by the MTL Queue Scheduler.

The credit value is accumulated every transmit clock cycle, that is, 40 ns for 100 Mbps and 8 ns for 1000 Mbps. The credit to be added or subtracted per cycle can be fractional based on the required idleSlope and sendSlope values, as described in Table 8-8.

Table 8-8 Credit Value Per Transmit Cycle Example

Mode	Values	Description
100 Mbps	■ portTransmitRate = 100 Mbps ■ idleSlope = 70 Mbps (assuming 70% bandwidth reserved for a higher priority traffic class) ■ sendSlope = 30 Mbps	■ credit = 2.8 bits accumulates per cycle (40 ns) for the higher priority traffic class when best-effort packet is being transmitted. ■ credit = 1.2 bits drains per cycle (40 ns) when higher priority traffic class packet is being transmitted.

The DMA stores the queue traffic in the respective part of the Tx FIFO based on the slot number in the transmit descriptor (if enabled) or the bandwidth availability on the AMBA AHB application bus.

The credit for a queue builds up only when the packet is available but it cannot be transmitted because the MAC is sending a packet from another queue. The DWC_ether_qos supports another mode in which the credit can build up in advance for a queue in which no packet is available in respective part of the FIFO. This enables sending a burst of high priority traffic in a queue as soon as the data is available. You can enable this mode with Bit 3 of the CBS Control registers of corresponding queues.

When reset, the accumulated credit parameter in the credit-based shaper algorithm is set to zero if there is positive credit, and there is no packet to transmit in a queue. The credit does not accumulate when there is no packet waiting in a queue and other queues are transmitting. When set, the accumulated credit parameter in the credit-based shaper algorithm is not reset to zero if there is positive credit and no packet to transmit in a queue. The credit accumulates even when there is no packet waiting in a queue and other queues are transmitting.

8.11.4.2 idleSlopeCredit and sendSlopeCredit Values

The idleSlopeCredit and sendSlopeCredit values are programmable.

The software must program the idleSlopeCredit and sendSlopeCredit values. The programmed values should be the credit accumulated or drained per clock cycle scaled by 1024, such as, 2.8 x 1024 = 2867 and 1.2 x 1024 = 1229. In addition, the software must program the hiCredit and loCredit values, scaled by 1024, to adjust for scaling of the idleSlopeCredit and sendSlopeCredit values. This means that if computed hiCredit and loCredit values are 12,000 bits and 3,036 bits respectively, the values to be programmed in the hiCredit and loCredit registers of the corresponding channel are 12000 x 1024 bits and two's complement of 3036 x 1024, respectively.

8.11.4.3 Bandwidth Status

The hardware maintains the status of the actual bandwidth consumed by each higher priority queue in the CBS status registers. This enables the software to estimate the average bandwidth consumed by numerically higher traffic classes as compared to the reserved bandwidth.

The CBS status register gives the average number of bits transmitted during the previous programmed slot interval (1, 2, 4, 8, or 16 slots of 125 µs) in a queue. The status register is updated even if the credit-based shaper algorithm is not enabled for a queue. The number of slots over which the average bits transmitted per slot are computed is programmed in Bits[6:4] of the CBS control register of the respective queue. For example, if you have programmed two slots, the average bits are computed over slot numbers 0–1, 2–3, 4–5, and so on.

The value programmed in the idleSlopeCredit register of a queue is proportional to the bandwidth reserved for the queue. The software can allocate any bandwidth that is not used by the higher priority queue to the reserved bandwidth of the lower priority queue.

A lower priority queue, which is using the credit-based shaper algorithm, cannot use the unused reserved bandwidth of any higher priority queue that is using the credit-based shaper algorithm. However, a lower priority queue, which is using the strict-priority algorithm, can use the unused reserved bandwidth of any higher priority queue that uses the credit-based shaper algorithm. For example, Queue 1 and Queue 2 use the credit-based shaper algorithm (with reserved bandwidth of 50% and 25%, respectively) and Queue 0 uses the strict-priority algorithm. If Queue 1 uses only 40% of reserved bandwidth, the remaining 10% is used by Queue 0. Queue 2 cannot exceed the reserved bandwidth of 25%.

8.11.5 Enabling AV Bridging

You can enable the AV Bridging feature in the coreConsultant GUI.

To enable this feature while configuring the controller in the coreConsultant GUI:

1. Select the **Enable Audio Video Bridging** option under the Multiple Queues Support section during the Specify Configuration activity.
2. Select the **Enable Support for AV in Tx Queue x options (x = 1 to 7)** under the Multiple Queues Support section.

For details about these option, see Multiple Queues Support Parameters in the "Parameter Descriptions" on page 419.

Note See the "Enabled" field of the parameter descriptions to understand the dependencies

8.11.6 Slot Number Function and Audio Video Bridging

8.11.6.1 Introduction to Slot Number Function

When the AV feature is enabled, you can use the slot number function to schedule the data fetching from the system memory by the DMA. This feature is useful when the source AV data needs to be transmitted at specific intervals.

You can program the slot number at which the DMA should fetch the data from system memory in the Transmit Descriptor Word 3 (TDES3). This 4-bit field allows the application to schedule data up to 16 slots; a programmable slot interval that ranges from 1µs to 4096µs and granularity of 1µs. This field is applicable only for the AV channels.

8.11.6.2 Description of Slot Number Function

When the DMA fetches a Tx descriptor, it compares the slot number of the Tx descriptor with the internally generated reference slot interval. The slot interval is a counter that is updated based on a programmable slot interval (with range from 1µs to 4096µs) of the IEEE 1588 system time. In addition, the slot interval counter is initialized to zero when the seconds field of the system time is incremented, that is, the sub-second counter rolls over. The DMA fetches the data only if it matches the current slot or the next slot. The DMA remains in the descriptor fetch state till there is a match.

To enable the DMA to fetch the data only if it matches the current slot or the next two slots, you can program Bit 1 of the Slot Function Control and Status register of the corresponding DMA channel.

Note If the slot number in the descriptor is less than the reference slot number, the DMA takes it as a future slot.

You can enable the check for slot number by setting Bit 0 of the Slot Function Control and Status register of corresponding DMA channel. When this check is not enabled, the packets are fetched immediately after the descriptor is read. In addition, Bits [19:16] indicate the value of the reference slot number in DMA.

8.11.7 Programming Guidelines for Initializing the DMA

See "Initializing DMA" on page 1357

8.11.8 Programming Guidelines for Enabling Slot Number Checking

See "Enabling Slot Number Checking" on page 1370

8.11.9 Programming Guidelines for Enabling Average Bits Per Slot Reporting

See "Enabling Average Bits Per Slot Reporting" on page 1370

8.12 Queue Modes

You can enable a Tx queue for generic/DCB, AV, or all types of traffic. When you select multiple Tx queues without selecting the DCB feature, the queues are enabled for generic queuing using WRR or WSP algorithms.

When you select the **Enable Data Center Bridging** feature, DCB is enabled for all selected Tx queues and queuing is based on WRR, WSP, DWRR, or WFQ algorithms.

To enable a Tx queue for AV, you need to select the **Enable Audio Video Bridging** option and then the Enable Support for AV in Tx Queue # option. The queuing is based on the CBS or SP algorithms.

You can enable an Rx queue for generic, DCB, AV, or all types of traffic. A particular queue enabled for generic/DCB or AV based routing is determined by the RXQ#EN field of corresponding queue. The following list explains how Rx queues are enabled based on the selected features:

- Multiple Rx queues without the DCB feature

  When you select multiple Rx queues without selecting DCB feature, all queues are enabled for generic queuing based on the VLAN Tag priority. The VLAN Tag priority should match the PSRQ field of the MAC_RxQ_Ctrl2 and MAC_RxQ_Ctrl3 registers. By default untagged packets are routed to Receive queue specified in UPQ field of MAC_RxQ_Ctrl1 register. Queue 0 is the default value of UPQ field. You can override the default value with any other value, for the UPQ field. The Rx packets can also be routed to a particular DMA channel based on the DCS field of perfectly-matched MAC Address register.

- Multiple Rx queues with DCB feature

  When you select the **Enable Data Center Bridging** feature, DCB is enabled for all selected Rx queues. The queuing is done based on the VLAN Tag priority for DCB data packets. The VLAN Tag priority should match the PSRQ field of the MAC_RxQ_Ctrl2 and MAC_RxQ_Ctrl3 registers. The Rx packets can also be routed to a particular DMA channel based on the DCS field of perfectly-matched MAC Address register. The DCB control packets are routed based on the DCBCPQ field of the MAC_Rx-Q_Ctrl1 register.

- Multiple Rx queues with AV feature

  When you select the **Enable Audio Video Bridging** option, AV is enabled for all selected Rx queues. The queuing is done based on the VLAN Tag priority for DCB data packets. The VLAN Tag priority should match the PSRQ field of the MAC_RxQ_Ctrl2 and MAC_RxQ_Ctrl3 registers. The Rx packets can also be routed to a particular DMA channel based on the DCS field of perfectly-matched MAC Address register. The AV control packets (tagged or untagged) are routed based on the AVCPQ field of the MAC_RxQ_Ctrl1 register. The PTP over Ethernet packets are routed based on the AVPTPQ field of MAC_RxQ_Ctrl1 register.

8.12.1 Programming Guidelines for Disabling Flow Control in AV Queues

The DWC_ether_qos supports programming of same priorities for AV and DCB enabled Rx queues, when both coexist. For example, priority 1 can be programmed in PSRQ[n] field corresponding to Rx Queue enabled for AV in MAC_RxQ_Ctrl2/3 register and same priority 1 can be programmed in PSRQ[n] field corresponding to Rx Queue enabled for DCB in MAC_RxQ_Ctrl2/3 register, at the same time. When AV and DCB operation coexists, it requires that the Flow Control is disabled for AV enabled queues. See, "Disabling Flow Control for AV Enabled Queues" on page 1371.

8.13 Queue Priorities

The Tx queue priorities can be programmed through the MAC_TxQ_Prty_Map0 and MAC_TxQ_Prty_Map1 registers. The Rx queue priorities are programmed through the Rx Flow Control register.

You can program the priority of a Tx queue in the MAC_TxQ_Prty_Map0 and MAC_TxQ_Prty_Map1 registers. You can program the priority of an Rx queue corresponding field of MAC_RxQ_Ctrl2 and MAC_RxQ_Ctrl3 registers. The priority should be assigned in the following order:

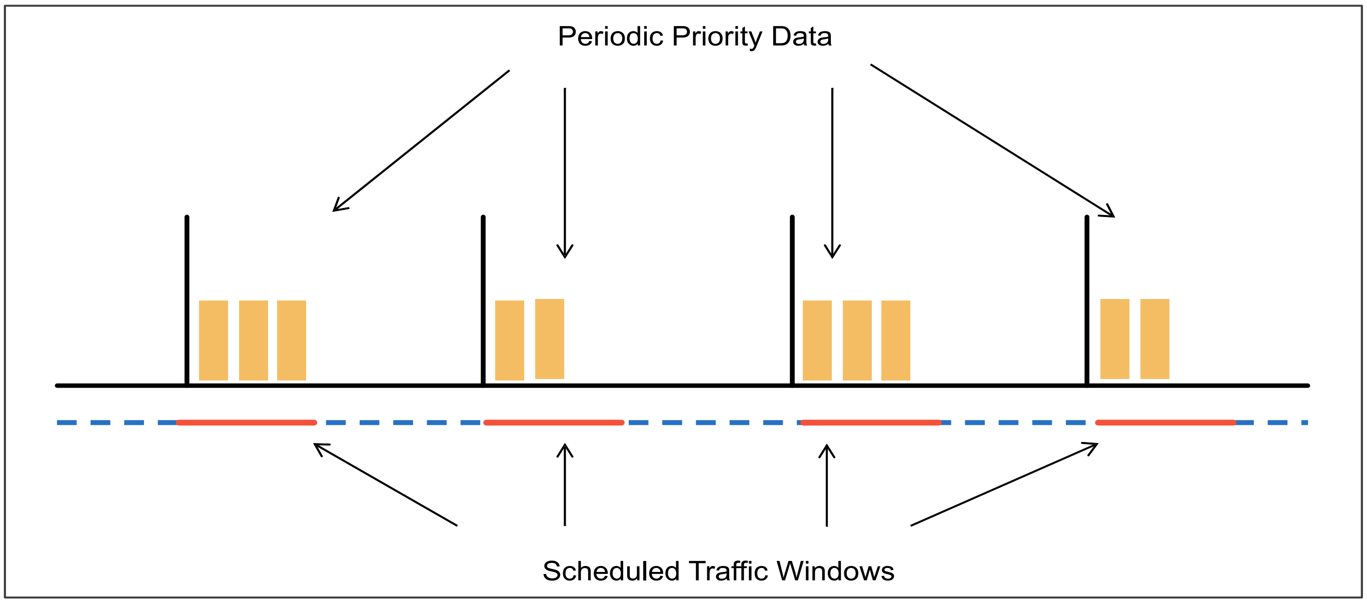
1. AV queue (high priority)
2. DCB queue (medium priority)
3. Best-effort queue (low priority)

The software should put packets with correct priorities in the respective programmed queue on Tx side. The MAC uses the programmed priorities for blocking the Tx queues when a PFC packet is received. If a single queue is selected for multiple priorities and PFC is enabled, the entire queue is paused if one or more priorities in the queue are paused.

8.14 Enhancements to Scheduled Traffic (EST)

The IEEE 802.1Qbv-2015 defines the schedule for each of the queues on every egress port which makes the implementation aware of traffic arrival schedule. This information can be used to block the lower priority traffic from transmission in this time window/slot. This ensures that scheduled traffic is forwarded from sender to receiver through all the network nodes with a deterministic delay.

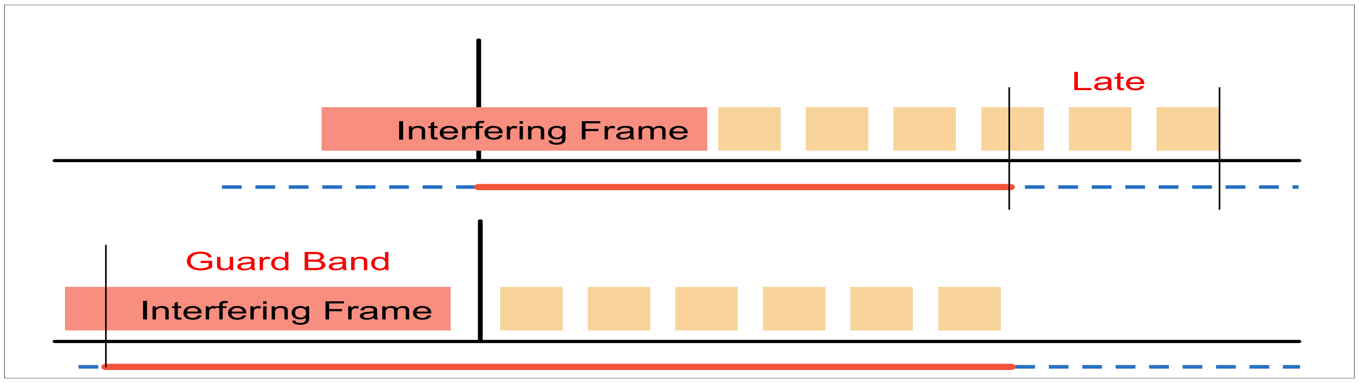
Figure 8-5 Time Aware Shaper Implementation for Traffic Scheduling



One of the important requirements to achieve a low latency is to ensure there are no interfering frames during the scheduled windows that are reserved for high priority traffic. The use of scheduled traffic imposes limitations while starting a transmission.

As shown in Figure 8-6, if an interfering frame begins transmission just before the start of a reserved time period, it can extend transmission into the reserved window, and potentially interfere with higher priority traffic. Therefore, a guard band whose size is equal to the largest possible interfering frame is required before the window starts.

Figure 8-6 Implementing a Guard Band to Avoid Delays Due to Interfering Frames



A larger guard band equates to a less efficient use of network bandwidth. However, with the implementation of IEEE802.1Qbu (frame preemption), this issue is addressed. Frame preemption breaks the interfering frame into smaller fragments. Therefore, the guard band needs to be only as large as the largest possible interfering fragment instead of the largest possible interfering frame.

During the guard band, only the frames that can complete the transmission of the entire frame before the next gate close event are permitted. This ensures that the high priority traffic can always start at the beginning of the window reserved for it.

8.14.1 Frequently Used Terms in EST

Following are some of the frequently used terms in the EST support and their definitions.

- **Gate Control List**

  The list in the hardware memory that holds the gate controls and the associated time intervals.
- **Gate Controls**

  For a given schedule (row in gate control list) there is a Gate Open (O) and Gate Close (C) state associated with each TC. The set of O or C values, whose width is same as configured TCs is called the Gate Controls. Example: CCOCOOCO means TC7=C, TC6=C, TC5=O and so on.
- **Time Interval**

  Time interval (in nano seconds) is a 16, 20, or 24-bit configurable field in the Gate Control list that indicates the time for which the associated gate controls are valid.
- **Base Time Register**

  Each Gate Control List is associated with a 64-bit Base Time Register that holds the start time (in PTP format) for the list.

8.14.2 Updates to the Transmit Scheduling to Support EST

To support EST, following updates are required to the transmit scheduling:

- Implementation of Gate Control List (GCL)
- Enforcing gate controls in the scheduler
- Accounting for Gate Open (O) duty cycle in the computation of idleSlope (CBS)

8.14.2.1 Implementation of Gate Control List (GCL)

Figure 8-7 shows block diagram from the IEEE 802.1Qbv spec that illustrates how the gate control list governs the gate close(C) and open(O) events based on the schedule provided for each event.

GCL has the following two parts:

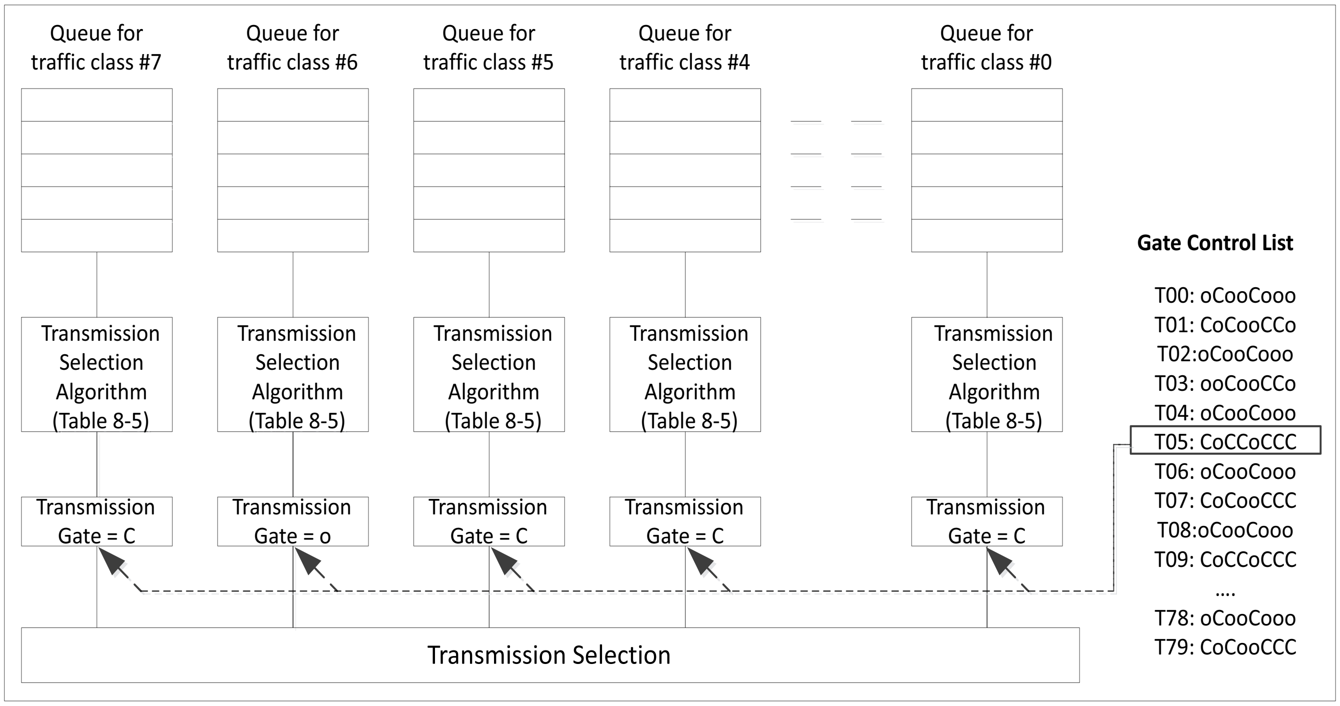
- **Time Interval**

  Defines the time in nano seconds for which the gate controls are valid and should be applied before reading the next gate controls from the list. Configurable width of 16, 20 or 24 to represent a maximum of 64us, 1ms, or 16ms schedule interval respectively. Supports left shift of the time interval upto 7 bits to be able to apply a multiplication factor from 1 to 128ns (in steps of 2^n). The maximum value (post shifting) of this field must be 999,999,999 ns.

- **Gate Control**

  Defines the Open (O) represented by logic 1 or Close (C) represented by logic 0 state for the gate of each TC.

Figure 8-7 Block Diagram from IEEE 802.1 Qbv Specification - GCL Governing Gate Close and Open Events



The implementation of GCL consists of the following two Gate Control Lists:

- HWOL - Hardware Owned List which is a list for hardware access.
- SWOL - Software Owned List which is a list for software access.

The access to these lists is mutually exclusive. The ownership to the list is set by hardware in the SWOL field of MTL_EST_Status register. Following table provides the implementation details of GCL.

Table 8-9 External Memory Used for Holding the Two Gate-Control Lists

Gate Control (up to 8 bits)	Time Interval (ns) 16, 20, or 24 bits	
OOCCCCCC	T0	HWOL or SWOL
OOOOCCCC	T1	
CCCCOOCC	T2	
CCCCCCOO	T3	
I	I	
I	I	
OCOCOCOO		
OOOOCCCC	T last	
OOOOCCCC	T0	SWOL or HWOL
OOOOCCCC	T1	
OOCCCCCC	T2	
OOOOCCCC	T3	
I	I	
I	I	
OCOCCCCC		
OOCCCCCC	T last	

8.14.2.2 Registers Related to Gate Control List

Following are the set of 4 registers (one for each GCL) related to GCL. These registers are implemented through Indirect Addressing using the GCL_Control and GCL_Data registers.

1. 64-bit Base Time Register (BTR)
2. 40-bit Cycle Time (CTR)
3. m-bit Time Extension (TER)
4. n-bit [Gate Control] List Length (LLR) (n depends of the configured GCL depth)

8.14.2.2.1 Base Time Register (BTR)

Base Time Registers is a 64-bit register that holds the start time to execute the GCL. The format of the BTR is same as the PTP format (upper 32-bits holds time in seconds and lower 32 bits hold time in nano seconds). Once the execution of a given list begins, the implementation can update the value in BTR to indicate the next list execution begin time.

8.14.2.2.2 Cycle Time Register (CTR)

Cycle Time Register is a 40-bit register that holds the time at which the execution of the GCL should be repeated. The Cycle Time Register consists of an 8-bit value in seconds, and a 32-bit value in nano seconds (similar to the PTP time format with truncated seconds register).

For a given Gate Control List the start time is "Base Time" + N * "Cycle Time" where N is an integer value representing the iteration number starting with 0 for first iteration. If the Gate Control List execution takes longer than the Cycle time, then the list is truncated at the Cycle Time and the subsequent loop begins at Cycle Time.

8.14.2.2.3 Time Extension Register (TER)

Time Extension Register (TER) is a m-bit (where m = Configured Time Interval width + 7) register that holds the amount of time (in nano seconds) the current Gate Control List can be extended before switching to the new Gate Control List. This will be useful in avoiding small fragments of the current list being executed before switching to a new list.

8.14.2.2.4 List Length Register (LLR)

List Length Register (LLR) is an n-bit register (when n is 7, 8, 9, 10, or 11 for a GCL configured depth of 64, 128, 256, 512, or 1024 respectively) that holds the integer value of the length of the GCL (that is the number of valid rows in each GCL). The processing of the GCL stops after the number of rows read equals to the LLR value.

8.14.2.3 Transmission Gating Implementation

A Bridge or an end station can be enhanced to allow transmission from each TC that is yet to be scheduled relative to a known timescale. To achieve this, a transmission gate is associated with each TC; the state of the transmission gate determines if the queued frames can be selected for transmission.

For a TC, the transmission gate can be in one of the following two states:

1. **OPEN**

   Queued frames are selected for transmission, in accordance with the definition of the transmission selection algorithm associated with the TC.
2. **CLOSED**

   Queued frames are not selected for transmission.

A frame on a traffic class queue is not available for transmission if the transmission gate is in the closed state or if there is insufficient time available to transmit the entirety of that frame before the next gate-close event associated with that queue.

The implementation has visibility to the current schedule of gate controls and the immediate next schedule of the gate controls. So the maximum Gate Open period does not exceed the sum of the two Time Intervals.

This is because, a frame is selected for transmission only if the gate is currently Open and the duration of gate open interval is greater than the time taken to transfer the entire frame.

The implementation must know the frame size before the transmission, so that the transmission overruns can be avoided and only the frames that can complete are scheduled at all times.

The implementation adequately compensates for the implementation delays in the data transfer from the buffer to the line by offsetting the current time with all the relevant delays (provided by the CTOV field of MTL_EST_Control register). This ensures that the schedule provided is always accurately implemented at the line.

Note

- A 20 Byte overhead is added to the packet size to account for the IFG and Preamble overheads on the line. Therefore for a 128 Byte frame to be transmitted, the Gate Open window should at least be able to accommodate148 Bytes.
- In 100 Mbps mode of operation, the rounding down error is about 0.4%. For example, the Gate open duration must be at least 1024 Bytes for transmitting 1000 Byte Frames at a speed of 100 Mbps (20 Bytes for line overheads, and 4 Bytes for rounding down error)

8.14.3 Idle Slope Computation Updates

When EST is enabled, credit is accumulated only when the gate is open; therefore, the effective data rate of the idleSlope must be increased to reflect the duty cycle for the transmission gate associated with the queue.

The idleSlope is computed based on the GateOpenTime and OperCycleTime values. Program the idleSlope registers (implemented one per CBS enabled TC) based on the following equation. The existing MTL register has sufficient bit width to accommodate the new values for idleSlope.

```
    idleSlope = (operIdleSlope(N) * OperCycle/GateOpenTime)
```

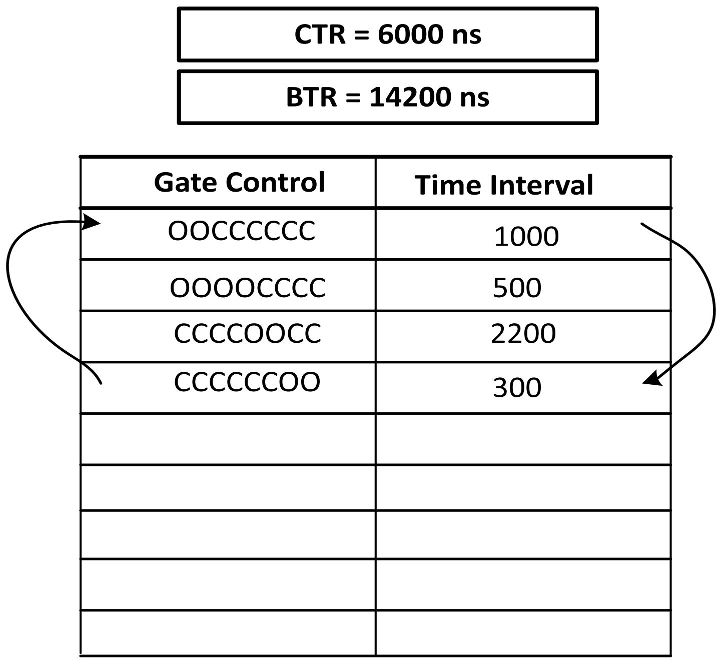
8.14.3.1 Operational Details of GCL

Set the switch to software List, the SSWL field of the MTL_EST_Control Register, so that hardware can access the programmed gate control list. The first set of gate controls are applied when the current time is equal to the value in the Base Time Register (BTR) and is held until the programmed "Time Interval" value.

To avoid transmission overruns, one additional gate control event is always read ahead from the list. This enables the GCL to determine the next gate close events (if any) for the TCs that are open.

The scheduling based on the Gate Open state and Time Interval of only the current and subsequent schedule. An internal accumulator is used to add the time intervals when gate controls are applied. BTR + Accumulator holds the time at which the next set of gate controls are to be applied.

Figure 8-8 GCL and Associated Registers - BaseTime and CycleTime



The GCL is read progressively from the first row adhering to the schedule. The read operations continue until the List Length (from LLR Register) is reached and the execution of the list restarts at BTR + CTR time. At this point the BTR is incremented by value in CTR to mark the beginning of a new cycle. In the absence of any gate controls, all the gates are deemed to be in Open state, during the execution of the list.

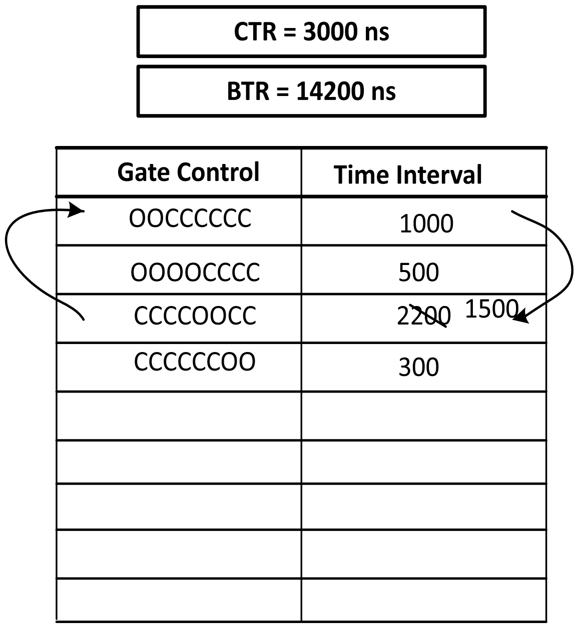
In cases where the execution time of the list is greater than the CycleTime, the list is truncated and the next iteration starts when the current time equals BTR + CTR.

Table 8-10 GCL and Associated Registers - BTR and CTR

Current Time	Gate Control Applied	Accumulator Value	BTR (with updates)
14200	OOCCCCCC	1000	14200
15200	OOOOCCCC	1500	14200
15700	CCCCOOCC	3700	14200
17900	CCCCCCOO	4000	14200
18200	OOOOOOOO	0	20200
20200	OOCCCCCC	1000	20200
21200	OOOOCCCC	1500	20200

In the example in Table 8-10, the execution starts at 14200 and the first set of Gate Controls "OOCCCCCC" are applied immediately. As the time interval is 1000ns the next set of gate controls are applied at 14200 (BTR) + 1000 (Accumulator) = 15200 ns as shown in Table 8-9. As there are no gate controls available after the execution of the last gate control and before the next iteration of the loop, the gates are deemed to be in open state during that time period as depicted at time 18200 in Table 8-10.

Figure 8-9 GCL and Associated Registers - BaseTime and CycleTime List Execution Time > CycleTime



As the list execution takes longer than the allocated CycleTime, the list is truncated and the list is started from the BTR+CTR as shown in Table 8-11.

Table 8-11 GCL and Associated Registers - BTR and CTR, Execution Time > Cycle Time

Current Time	Gate Control Applied	Accumulator Value	BTR (with updates)
14200	OOCCCCCC	1000	14200
15200	OOOOCCCC	1500	14200
15700	CCCCOOCC	3700	14200
17200	OOCCCCCC	1000	17200
18200	OOOOCCCC	1500	17200
18700	CCCCOOCC	3700	17200

While applying the third set of gate controls, the BTR + CycleTime (17200) < BTR + Accumulator (17900), so the list is truncated and execution switches to a new iteration at 17200.

8.14.3.2 Installing a New GCL

When a new software programmed GCL is available and must be executed at the new BTR value, the switch to the new GCL can happen in one of the following ways.

- New Base Time aligned with current schedule
- New Base Time unaligned with current schedule

8.14.3.2.1 New Base Time Aligned with Current Schedule

If the choice of cycle time for the new gating cycle is unchanged from the cycle time for the current gating cycle, and if the BTR chosen for the new gating cycle (new BTR) is an integer multiple of the current cycle time (+ current BTR), then the new gating cycle exactly lines up with the old gating cycle, that is, the cycle start times for the new gating cycle is same as they would have been for the old configuration. This could be considered to be the ideal case and allows the new gating cycle to be installed and executed with no timing issues. The implementation completes the execution of an iteration of the current list and switches to the new list at the beginning of the BaseTime listed in the new list.

If (New BaseTime > = Current Time)

  ConfigChangeTime = New BaseTime

Else If (New BaseTime < Current Time)

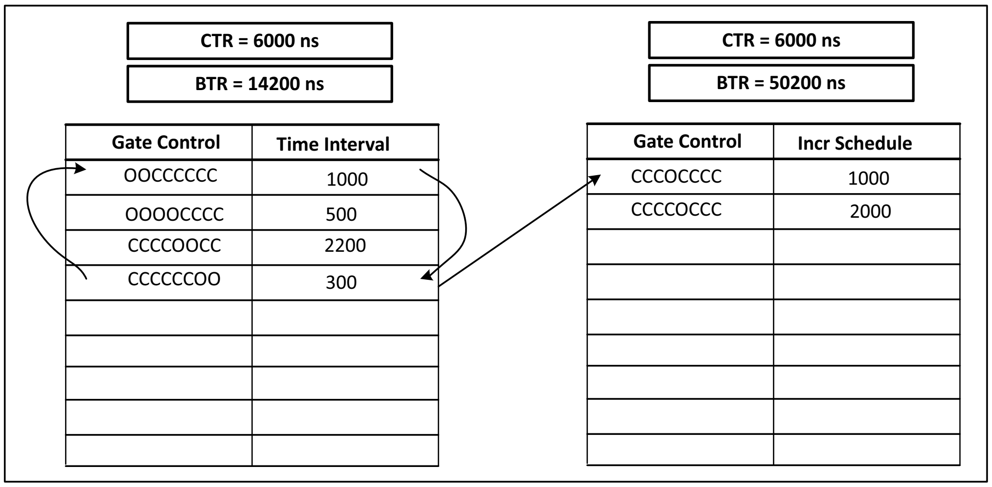
1. Set the BTRError
2. ConfigChangeTime = (New BaseTime + N* New CycleTime)

   where N is the smallest integer for which the relation ConfigChangeTime >= CurrentTime and (N =< 8) is TRUE.

   When N > 8 the hardware cannot auto recover and the loop count value in BTRError reporting is set to 1111 requiring the software to reprogram the New Base Time.

   Figure 8-10 illustrates the installation of the new GCL along with the timelines.

Figure 8-10 Switching to a New Configuration that is Aligned with the Existing Configuration



In the above example, after the sixth iteration of the first GCL, the BTR values of the old and new GCL are equal. At that point the new GCL is processed as a natural extension to the existing GCL.

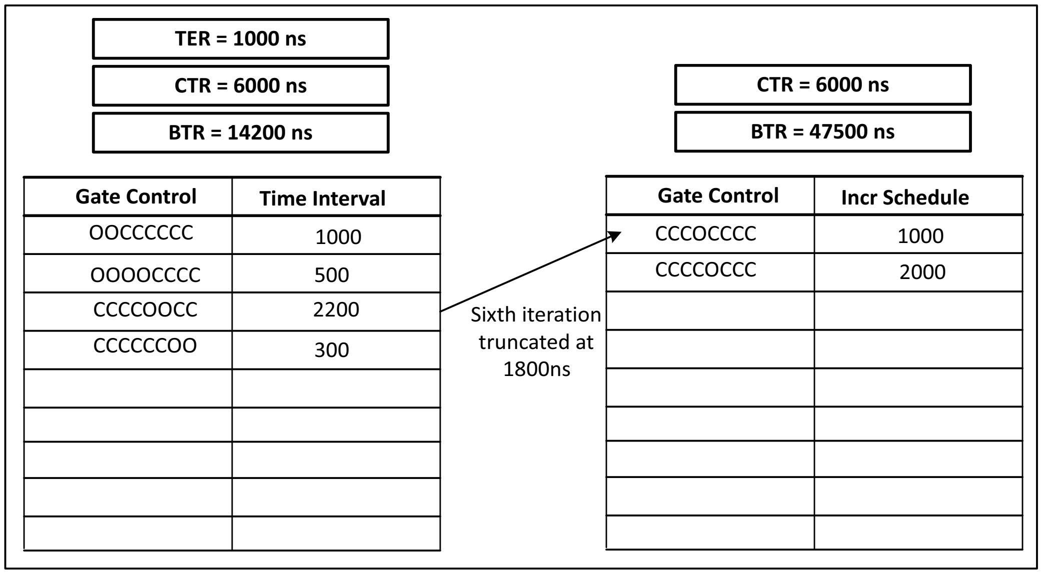
Table 8-12 GCL and Associated Registers - BTR and CTR

Current Time	Gate Control Applied	Accumulator Value	BTR (with updates)
44200	OOCCCCCC	1000	44200
45200	OOOOCCCC	1500	44200
45700	CCCCOOCC	3700	44200
47900	CCCCCCOO	4000	44200
48200	OOOOOOOO	0	50200
50200	CCCOCCCC	1000	50200
51200	CCCCOCCC	3000	50200
53200	OOOOOOOO	0	56200

8.14.3.2.2 New Base Time Unaligned with Current Schedule

If new CycleTime differs from current CycleTime or New BaseTime is in the future and is not an integer multiple of current CycleTime, then the old and new cycles do not line up; when new BaseTime is reached (when the new configuration is installed and starts to execute), the last old cycle is normally truncated to start the first new cycle. This could be undesirable if it results in a very short last old cycle; arguably it would be better to simply extend the penultimate old cycle by that small amount, rather than starting a very short cycle. The Cycle Time Extension Register (related to the current config list) allows this extension of the last old cycle to be done in a defined way; if the last complete old cycle ends normally in less than current Cycle Time Extension (TER) ns before the new base time, then the last complete cycle before new BaseTime is reached is extended so that it ends at new BaseTime.

Figure 8-11 Switching to New List Truncating the Current List



At the end of the fifth iteration the Current time + Cycle time extension (TER) < New BTR so the sixth iteration of current configuration is started. During the sixth iteration of the current list when the new BTR value is smaller than the next schedule in the current list, it switches to the new list.

Table 8-13 Extending to New List By Truncating the Current List

Current Time	Gate Control Applied	Accumulator Value	BTR (with updates)
44200	OOCCCCCC	1000	44200
45200	OOOOCCCC	1500	44200
45700	CCCCOOCC	3700	44200
47500	CCCOCCCC	4000	44200
48500	CCCCOCCC	0	50200
50500	OOOOOOOO	1000	50200

Below is an example where the current config list is extended instead of starting a new iteration as the extension time of 800ns is less than the allowed cycle extension time (TER) of 1000ns.

Figure 8-12 Switching to New List By Extending the Current List

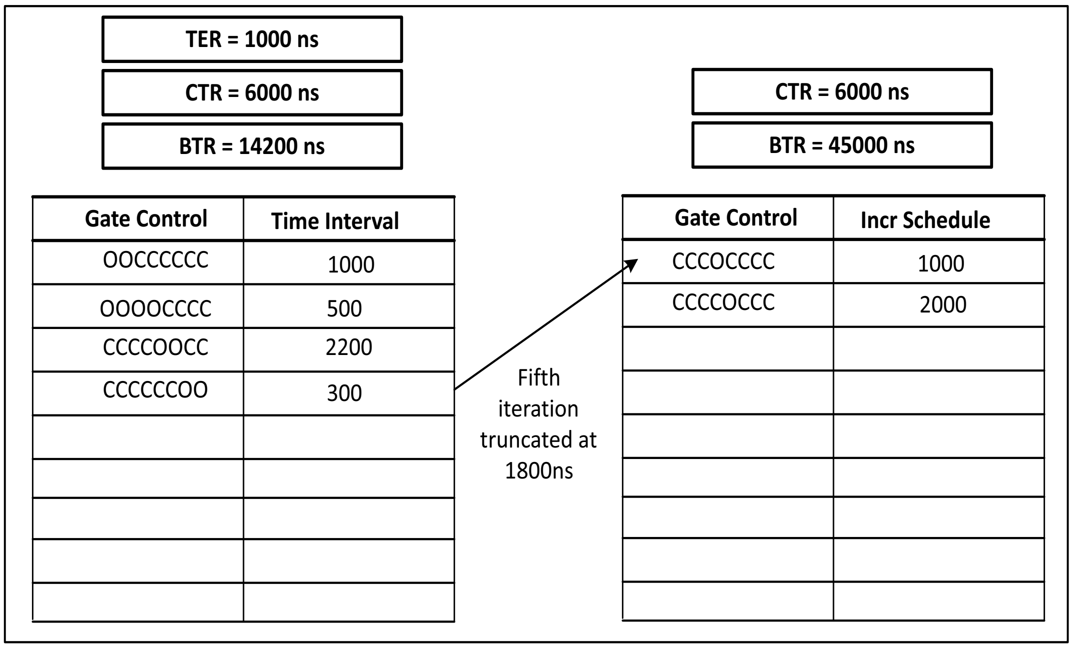


Table 8-14 Switching to New List By Extending the Current List

Current Time	Gate Control Applied	Accumulator Value	BTR (with updates)
38200	OOCCCCCC	1000	38200
39200	OOOOCCCC	1500	38200
39700	CCCCOOCC	3700	38200

Current Time	Gate Control Applied	Accumulator Value	BTR (with updates)
41900	CCCCCCOO	4000	38200
42200	OOOOOOOO	0	44200
45000	CCCOCCCC	1000	45000
46000	CCCCOCCC	3000	45000
48000	OOOOOOOO	0	51000

8.14.4 Impact of Transmit Scheduling Algorithms on EST

When EST is used in isolation, the Gate Control List manages the final open/close state of the Queues along with the checks carried out by the Transmission Selection Algorithm in MTL. As the Gate Controls operate on a predefined repetitive schedule, it is recommended to use Strict Priority or Credit Based Shaper (CBS) scheduling algorithms.

Other algorithms such as the Weighted Round Robin (WRR), Deficit Weighted Round Robin (DWRR) and Weighted Fair Queuing (WFQ) implement masking of the queues based on the current winning queue. The algorithm is applied only among the group of queues that open simultaneously. To ensure Queues whose gates are "Open" get priority, these algorithms are modified to treat "gate open" queues and "gate closed" queues as separate groups giving priority to gate open queues.

For example, consider 4 queues (Q3, Q2, Q1, Q0) with weights 4:3:2:1; Q3 & Q2 are in Open state in slot one slot, while Q1 & Q0 are in Open state in another slot. In this case, the scheduler works as follows:

1. In the first slot, the Q3 & Q2 are serviced in the ratio of 4:3 for the duration the slot is open.
2. In the second slot, the queues Q1 & Q0 are serviced in the ratio of 2:1.
3. Fresh arbitration is started every time a slot is opened.

In other words, the traffic does not get distributed in the intended ratio of 4:3:2:1; but as two groups with different ratios and only for the duration of the slot when the gates are open continuously

8.14.5 Programming Guidelines for EST

For details about programming guidelines for EST, see “Programming Guidelines for EST” on page 1380.

8.15 Frame Preemption (FPE)

8.15.1 Overview of the Frame Preemption

The IEEE 802.1Qbv-2015 defines the schedule for each of the queues on every egress port which makes the implementation aware of traffic arrival schedule. This information can be used to block the lower priority traffic from transmission in this time window/slot. This ensures that scheduled traffic is forwarded from sender to receiver through all the network nodes with a deterministic delay.

One of the important requirements to achieve a low latency is to ensure there are no interfering frames during the scheduled windows that are reserved for high priority traffic. The use of scheduled traffic imposes limitations while starting a transmission.

If an interfering frame begins transmission just before the start of a reserved time period, it can extend transmission into the reserved window, and potentially interfere with higher priority traffic. Therefore, a guard band whose size is equal to the largest possible interfering frame is required before the window starts.

A larger guard band equates to a less efficient use of network bandwidth. However, with the implementation of IEEE802.1Qbu (frame preemption), this issue is addressed. Frame preemption breaks the interfering frame into smaller fragments. Therefore, the guard band needs to be only as large as the largest possible interfering fragment instead of the largest possible interfering frame.

During the guard band, only the frames that can complete the transmission of the entire frame before the next gate close event are permitted. This ensures that the high priority traffic can always start at the beginning of the window reserved for it.

8.15.2 Description of the Frame Preemption

You can select the Frame Preemption feature (FPE) feature during RTL configuration of the core. FPE feature can be selected independently with or without selecting the EST feature.

Preemption allows one or more higher priority (express) frames to interrupt the transmission of a lower priority (preemptable) frame; the preemptable frame transmission is resumed and completed after the express frame transmission is complete. To support frame preemption, the following two abstractions of the MAC are used:

- A preemptable MAC, called pMAC, which carries the preemptable traffic.
- An express MAC, called eMAC, which carries the express traffic.

In the implementation, only parts of the MAC that holds the state during preemption is replicated and represented as pMAC and eMAC. The MAC uses the following two ways to puts on hold, the transmission of the preemtable traffic, in the presence of express traffic:

- The MTL scheduler interrupts the preemptable traffic that is currently being transmitted.

  When the preemption capability is active, the MAC interrupts the transmission and reception of preemptable frames. A preempted fragment can be followed by zero or more express frames, before the continuation fragments. The preemptable frame can be fragmented any number of times, however, the minimum final and non-final fragment length criterion must be is met.

  However, interleaving of more than 1 preemptable packet is not permitted. This implies that if a preemptable packet is fragmented by an express packet, another preemptable packet cannot be transferred until all the remaining fragments of the first preempted packet are transferred.

- The MTL scheduler prevents starting the transmission of preemptable traffic.

  When the preemption capability is inactive, the pMAC entity is disabled and only express traffic is transmitted or received.

8.15.3 Enabling the Frame Preemption

Enable the frame preemption feature by setting EFPE field of the MAC_FPE_CTRL_STS register.

8.15.4 GCL Modification to Support FPE

In the EST-only configuration, the GCL entry has up to 24 bits of Time Interval and up to 8 high order bits representing the Gate Open/Close state requirements as shown in Figure 8-13.

Figure 8-13 Gate Control List When FPE is Disabled

Gate Control (up to 8 bits)	**Time Interval(ns) 16, 20 , or 24 bits**
OOCCCCCC	T0
OOOOCCCC	T1
CCCCOOCC	T2
CCCCCCOO	T3
|	|
|	|
OCOCOCOO	T126
OOOOCCCC	T127

EST only supports SetGate operation, which implies that the gates are set to either open or close at a given time interval.

However, when both Frame Preemption (FPE) and EST are enabled, the GCL also supports Set-and-Hold-MAC and Set-and-Release-MAC operations, in addition to the SetGate operation.

To enable the support of hold and release operations, the format of the GCL is slightly changed while configuring and enabling the FPE. The first Queue (Q0) is always programmed to carry preemption traffic and therefore it is always Open. The bit corresponding to Q0 is renamed as Release/Hold MAC control.

The TX Queues whose packets are preemptable are indicated by setting the PEC field of the MTL_FPE_CTRL_STS register. The GCL bit of the corresponding Queue are ignored and considered as always "Open". So, even if the software writes a "0" ("C"), it is ignored for such queues.

Figure 8-14 Gate Control List When FPE is Enabled

Gate Control (up to 7 bits)	Release/Hold Indication	Time Interval(ns) 16, 20, or 24 bits
CCCOOOO	0	T0
CCCCOOO	0	T1
OCCCOOO	1	T2
COCCOOO	1	T3
CCOCOOO	1	T4
CCCOOOO	0	T5
CCCCOOO	0	T6

When the Release/Hold bit transitions from a '0' to '1', a Set-and-Hold-MAC operation is performed. This marks the cease of the preemptable traffic. This is achieved by sending a Hold request to MTL "ha" ns in advance (where ha is the time interval mentioned in the Hold Advance (HADV) field of the MTL_FPE_Advance register).

When the Release/Hold bit transitions from a '1' to '0' a Set-and-Release-MAC operation is performed. This marks the resuming of the preemptable traffic. This is achieved by sending a Release request to MTL "ra" ns in advance (where ra is the time interval mentioned in the Release Advance (RADV) field of the MTL_FPE_Advance register).

The preemptable traffic is blocked for the time duration the Release/Hold bit is set to a '1' in the Gate Control List.

8.15.5 Impact of Preemption on CBS

In CBS, the definition of "Transmit" is as follows:

- TRUE for the duration of frame transmission from the queue;
- FALSE when frame transmission from the queue is complete.

When CBS algorithm is used along with frame preemption, the value of "Transmit" is TRUE only while the frame is being transmitted by the MAC. If the frame transmission is delayed or interrupted (for instance, a preemptable frame transmission is interrupted to allow the transmission of an express frame from a different queue, or the start of express frame is delayed because a preemptable frame is being transmitted) the value of "Transmit" is FALSE until transmission of the frame commences or is resumed.

Also, the value of "Transmit" is FALSE during the transmission of any overhead that is a consequence of frame preemption. For example, additional frame overhead (mCRC, Fragment Count) that is added to the preemptable frame.

At any given time, if there are no frames in the queue, and the value of Transmit is FALSE, and credit is positive value, the credit value is set to zero if here is no preemptable frame from the queue for which transmission is in progress but has been interrupted.

8.15.6 mPacket Format

When the preemption capability is active, MAC sends mPackets to the PHY. An mPacket can be one of the following:

1. A express packet
2. A preemptable packet
3. An initial fragment of a preemptable packet
4. A continuation fragment of a preemptable packet

Figure 8-15 shows the format of the mPacket. It contains an express packet, a complete preemptable packet or the initial fragment of a preemptable packet. Figure 3.3(b) shows the format of an mPacket containing a continuation fragment of a packet.

Figure 8-15 mPacket Formats

Preamble

The preamble in the mPacket shown in Figure 8-15 (a) contains seven octets. The preamble in the mPacket shown in Figure 8-15 (b) contains six octets. Each octet contains the value of 0x55 (transmitted in order from left to right 10101010).

Start mPacket Delimiter (SMD)

The value of the SMD indicates whether the mPacket contains an express packet, the start of a preemptable packet (initial fragment or complete packet), or any of continuation fragments of a preemptable packet. Table 8-15 shows the valid SMD values.

Table 8-15 Possible SMD Values of mPacket

mPacket Type	Notation	Frame Count	Value
verify packet	SMD-V	-	0x07
respond packet	SMD-R	-	0x19
express packet	SMD-E	-	0xD5
preemptable packet start	SMD-S0	0	0xE6
	SMD-S1	1	0x4C
	SMD-S2	2	0x7F
	SMD-S3	3	0xB3
continuation fragment	SMD-C0	0	0x61
	SMD-C1	1	0x52
	SMD-C2	2	0x9E
	SMD-C3	3	0x2A

frag_count

A frag_count is a modulo-4 counter that increments for each continuation fragment of the preemptable packet. The frag_count protects against mPacket reassembly errors by enabling detection of the loss of up to 3 packet fragments.

The frag_count field is present only in mPackets with SMD-C notation (continuation fragment). The frag_count is zero in the first continuation fragment of each preemptable packet.

Table 8-16 Possible frag_count Values

frag_count	Value
0	0xE6
1	0x4C
2	0x7F
3	0xB3

mData

The contents of the packet from the MAC, starting with the first byte after the SFD to the last byte before the FCS are sent in the mData fields of one or more mPackets for that frame. The minimum size of the mData field is 60 bytes.

CRC

The CRC field contains a cyclic redundancy check (CRC) and has an indication of the final mPacket of a frame. In the final mPacket of a frame, the CRC field contains the last 4 octets of the MAC frame (the FCS field).

For other mPackets, the CRC field contains an mCRC value. The mCRC is calculated on the octets of the packet from the first octet of the frame (the octet following the SFD of preemption frames) to the last octet of the packet transmitted in that mPacket by performing an XOR of the calculated 32 bit CRC value of the fragment and a value of 0x0000FFFF.

Summary of Packet Formats

- Express Packet

  7bytes of PREAMBLE, SMD-E, Data, and CRC
- Complete Preemptable Packet

  7 bytes of PREAMBLE, <Current Preemptable packet SMD>, Data, CRC
- Initial Fragment (non-final) of Preemptable Packet

  7 bytes of PREAMBLE, <Current Preemptable packet SMD>, Data, mCRC
- Continuation fragments (non-final) of Preemptable packet

  6 bytes of PREAMBLE, <Current Preemptable continuation fragment SMD>, <Current Preemptable continuation fragment FC>, Data, mCRC
- Final fragment of Preemptable packet

  6 bytes of PREAMBLE, <Current Preemptable continuation fragment SMD>, <Current Preemptable continuation fragment FC>, Data, CRC

Note When Frame Preemption is enabled, the MAC receiver communicates the exceptions related to received preempted fragments (incorrect fragment sequencing, missed fragment, and so on) to MTL layer by reporting it as CRC error. Software should not set the DCRCC (Disable CRC Checking for Received Packets) bit of MAC_Ext_Configuration register to 1; otherwise the MTL layer might forward the unintended preempted fragments to the application.

Table 8-17 Current and Previous SMD Values

Previous Preemptable packet SMD	Current Preemptable packet SMD
SMD-S0	SMD-S1
SMD-S1	SMD-S2
SMD-S2	SMD-S3
SMD-S3	SMD-S0

Table 8-18 Current and Previous SMD Values

Previous Preemptable fragment SMD	Previous Preemptable fragment FC	Current Preemptable continuation fragment SMD	Current Preemptable continuation fragment FC
SMD-S0	NA	SMD-C0	FC0
SMD-S1	NA	SMD-C1	FC0
SMD-S2	NA	SMD-C2	FC0
SMD-S3	NA	SMD-C3	FC0
SMD-C0	FC0	SMD-C0	FC1
SMD-C0	FC1	SMD-C0	FC2
SMD-C0	FC2	SMD-C0	FC3
SMD-C0	FC3	SMD-C0	FC0
SMD-C1	FC0	SMD-C1	FC1
SMD-C1	FC1	SMD-C1	FC2
SMD-C1	FC2	SMD-C1	FC3
SMD-C1	FC3	SMD-C1	FC0
SMD-C2	FC0	SMD-C2	FC1
SMD-C2	FC1	SMD-C2	FC2
SMD-C2	FC2	SMD-C2	FC3
SMD-C2	FC3	SMD-C2	FC0

Previous Preemptable fragment SMD	Previous Preemptable fragment FC	Current Preemptable continuation fragment SMD	Current Preemptable continuation fragment FC
SMD-C3	FC0	SMD-C3	FC1
SMD-C3	FC1	SMD-C3	FC2
SMD-C3	FC2	SMD-C3	FC3
SMD-C3	FC3	SMD-C3	FC0

8.15.7 Transmit Preemption

8.15.7.1 MTL Tx Preemption

To support preemption MAC should have more than 1 TX queue with at least 1 Queue designated as Express Queue. (and 1 queue designated as preemption queue). When FPE is enabled (setting EFPE=1 in MAC_FPE_CTRL_STS register), the MTL preempts a preemptable frame, when a "hold" request is asserted (EST/Qbv configured and enabled) express frames are available for transmission, that is, frame is present in MTL FIFO and is qualified for arbitration, after ensuring that the minimum mPacket mData field size is met. Therefore, preemption occurs only if at least 60 bytes of the preemptable frame have been transmitted and at least 64 bytes (including the frame CRC) remain to be transmitted.

The earliest starting position of preemption is controlled by the AFSZ field of the MTL_ FPE_CTRL_STS Register. Preemption does not occur until at least 64 * (1+ AFSZ) - 4 bytes of the preemptable frame have been sent.

When preemption occurs, all the preemptable queues are blocked and only the express queues are allowed to arbitrate (if more than one express queue has traffic) and transmit.

Continuation fragment of the preempted frame is the first frame to be transmitted after "release" request is asserted (EST/Qbv configured and enabled) and all the express traffic transmission completes.

Note All the PTP packets should be transmitted as express packets.

MTL communicates the following frame-type information to the MAC using a 2-bit preemption control signal on the MTI interface qualified by SoF and EoF.

- Express Frame
- Preemption Frame (Full or Fragment)
- Continuation Fragment (non-Final or Final)

Table 8-19 Preemption Control Values on MTI Interface for Various Frame Types

Qualifier	Preemption Control Value	Frame Type
SoF	00	Start Express
SoF	01	Start Preemption

Qualifier	Preemption Control Value	Frame Type
SoF	10	Continuation Fragment
SoF	11	Reserved
EoF	00	End of Frame
EoF	01	End of Fragment

If the mti_sof_o and mti_eof_o are sent in the same cycle on the MTI interface, the mti_preemp_ctrl (Preemption Control Value) for EoF is taken as "00", End of Frame.

MTL should wait for the previous fragment status to be received before resuming the continuation fragments of a preempted frame. Fragment status is described in detail in the Tx Fragment Status section

8.15.7.2 MAC Tx Preemption

MAC supports preemption by implementing the functionality needed to generate the mPackets as described in "mPacket Format" on page 322.

Based on the Preemption Control value received on the MTI interface (qualified with SoF and EoF), MAC determines the frame type (shown in Table 8-19) and generates mPackets accordingly

When the preemption capability is active, MAC replaces the SFD of a preemption packet with an SMD-S value. A 2-bit rolling frame count is encoded in the SMD-S value.

The SMD-E value is the same as the SFD value so the SFD of an express packet does not need to be replaced.

8.15.7.2.1 Tx Fragment Status

MAC sends Tx fragment status to indicate successful transmission of fragmented mPackets. To indicate a fragment status, bit-31 of the mti_txstatus_o field is set and all other values in the status field is set to '0'.

In case of a transmission error (underflow, jabber, and so on) the frame status is sent (and not a fragment status) with an error indication along with all other relevant status fields. The bit-31 of mti_txstatus_o is set to '0' to indicate a Frame Status. In case of receiving an error status for a transmitted fragment, the MTL drops the remaining fragments and does not send any more continuation fragments.

8.15.8 Receive Preemption

8.15.8.1 MAC Receive Preemption

When FPE is enabled, the MAC Receiver pases the incoming packets and differentiates between Express packets and preemptable packets. An SMD containing an SMD-E indicates express packet, and SMD containing an SMD-S indicates the first mPacket of a preemptable packet.

If an mPacket containing an SMD-S is received when MAC has not completed receiving the previous preempted packet, MAC sets a CRC Error status for the previously received partial packet.

When an SMD-S is detected, MAC records the frame count indicated by the SMD and then begins sending data on the MRI interface.

The MAC checks the last four bytes of the mPacket . If the last four bytes of the mPacket do not match CRC, that indicates the end of the packet with or without a CRC error as per the CRC check result. If the last four octets of the mPacket match, that indicates that the packet was preempted.

An SMD containing an SMD-C indicates an mPacket that continues the data for a preempted packet. Upon receiving an SMD value of SMD-C, MAC checks the following:

1. A preempted packet is in progress
2. The frame count indicated by the SMD matches the frame count of the packet in progress
3. The frag_count value indicates the next fragment count.

If any of the above check fails, the mPacket is discarded and MAC sets a CRC Error status for the partially received packet.

If all the checks pass, the next fragment count is incremented modulo 4.

When a packet is preempted, the MAC saves the state of the partially received packet (filter check status, timestamp, length fields etc.) and will be able to process any received Express packets before the continuation fragment is received.

The MAC Receiver sends a "dummy status" (all zeros in mri_rxstatus_o) for all the mPacket fragments successfully received and sends the Rx status with the final fragment. If an error is detected during any of the fragments the Rx status is sent and the fragment is marked as final fragment. All subsequent continuation fragments received for this packet are dropped in the MAC.

The MAC communicates the following frame type information to the MTL using a 2-bit preemption control signal on the MRI interface qualified by mri_sof_o and mri_eof_o.

1. Express Frame
2. Preemption Frame (Full or Fragment)
3. Continuation Fragment (non-Final or Final)

Table 8-20 Preemption Control Values on MRI Interface for Various Frame Types

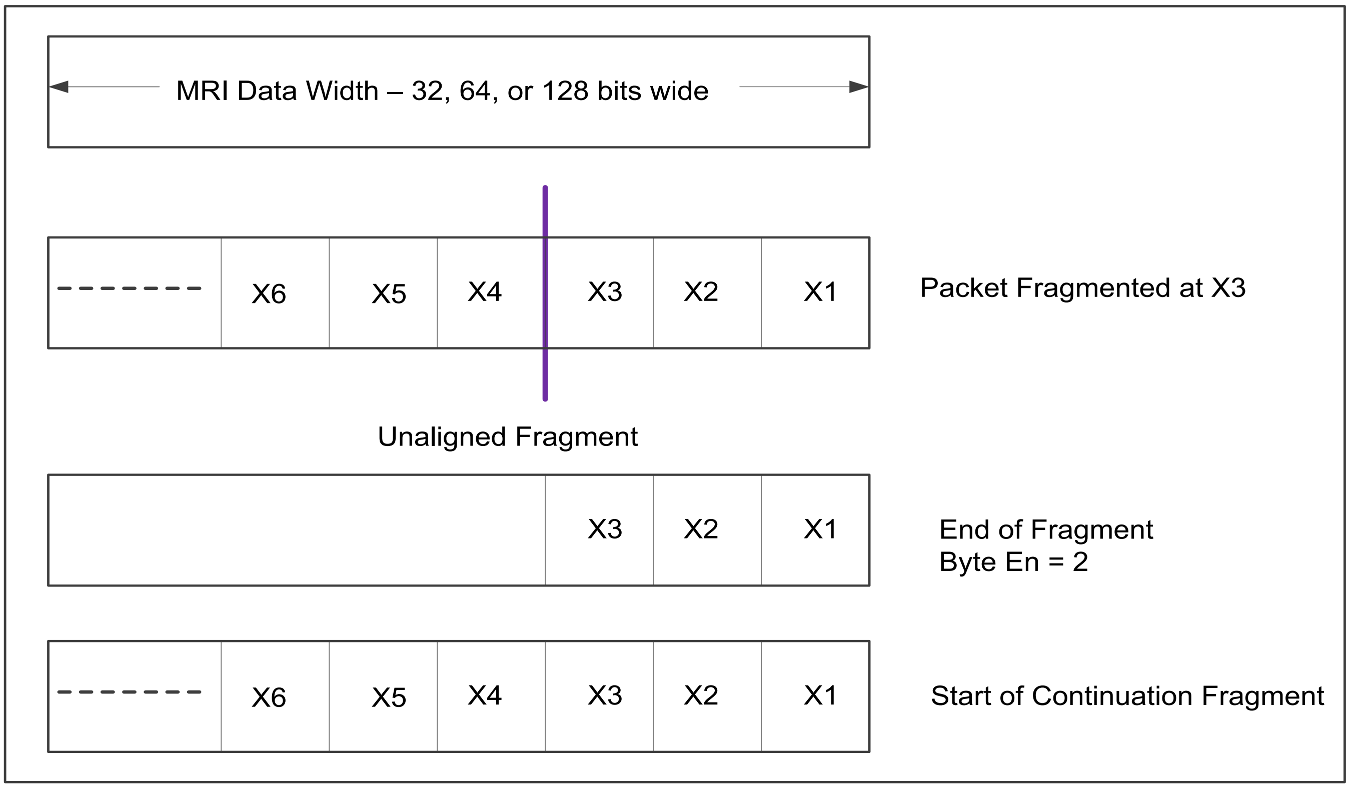
Qualifier	Preemption Control Value	Frame Type
SoF	00	Start Express
SoF	01	Start Preemption
SoF	10	Continuation Fragment
SoF	11	Reserved
EoF	00	End of Frame
EoF	01	End of Fragment

If the mri_sof_o and mri_eof_o are sent in the same cycle on the MRI interface, the mri_preemp_ctrl (Preemption Control Value) for EoF is taken as '0' indicating End of Frame.

8.15.8.1.1 Data Alignment

When a received frame cannot be fragmented on any byte boundary, MAC retains the unaligned bytes of data in the previous fragment and resends them with the next fragment as shown in Figure 8-16.

Figure 8-16 Data Alignment Feature of MAC



MTL can choose to ignore the partial data (based on Byte Enable value) that comes with End of Fragment and instead use the "data width aligned" data received in the first beat of the next continuation fragment.

8.15.8.1.2 MTL Receive Preemption

The MTL Rx must have at least 2 Rx queues to support FPE function as the preemptable packets and the express packets must be routed to separate Rx queues. The destination Rx queue of a received packet is controlled by MAC_RxQ_Ctrl[n] registers. Program these registers such that Preemptable traffic and express traffic are not routed to the same Rx queue. As the queue mapping for tagged packets is based on VLAN user-priority field, this implies that priority of preemptable and express packets are mutually exclusive. In other words, packets of a certain priority (traffic class) are either express or preemptable but cannot be both.

For the preemptable frames, in addition to the PSRQ (Priority Selected in Rx queue) based queue routing, a programmable "Frame Preemption Residue Queue" (FPRQ) is supported to route all other Preemption Packets received (Untagged, SA/ DA or VLAN Filter Fails forwarded due to RA being set or VTFE being reset).

Table 8-21 Preemption Control Values on MRI Interface for Various Frame Types

Preemption Packet Type	Queue Routing
AV Tagged Packets passing the SA/DA and VLAN filters	PSRQ*
DCB/Generic Tagged Packets passing the SA/DA and VLAN filters	PSRQ**

Preemption Packet Type	Queue Routing
Tagged Packets failing the SA/DA and VLAN filters without setting the Receive All (RA = 0) or setting VTFE = 1	Dropped
All other packets OR when RA = 1	FPRQ

The express frames use Queue 0 as a default queue. So, in case of filter failures, Queue0 must not be used for receiving preemption frames.

When a fragment is received, it is pushed into its destination Rx queue. When this packet is preempted, and is followed by an express packet, it is pushed/stored in a different Rx queue. The MTL saves the context of the preempted frame, and processes one or more express frames. When the continuation fragment arrives, it restores the saved context and pushes the remainder of the fragmented frame into its Rx queue.

8.15.8.1.3 MTL Receive Arbitration

On the ARI Interface, data is fetched from the MTL Rx FIFO based on the arbitration selected in the MTL_Operation_Mode register. Frame based arbitration can be used only when all the MTL Preemption Queues are operating in Store and Forward mode, else there will be loss of bandwidth on the ARI interface because all the fragments of a preemptable packet is not available. Therefore, any express packets received between the fragments are blocked until all the fragments are received and transferred; this defeats the purpose of express traffic.

When operating in either Threshold (cut through) or Store and forward modes of operation, PBL based arbitration is recommended over Frame based arbitration. In case of PBL based arbitration, the watermark check is always performed and the arbitration/transfer of data in terms of chunks of "PBL" size of data which is almost similar to the concepts of "fragments". Therefore the express queue packets get blocked for less time as well as the ARI interface transfers the data without loss of efficiency.

8.15.9 Received Preemption Frames Support in Offload Engines

When Frame Preemption is enabled, all the packets processed by the Offload Engines (ARP, PTO) and PMT, PTP, PAUSE, and PFC packets must be received as express/normal packets. These offload engine logic and functions do not recognize preemption frames.

8.15.10 Verify and Respond mPackets

When FPE function is present, the MAC can receive and detect the Verify and Respond mPackets, even when FPE is not enabled by software. When MAC detects valid Verify/Respond mPackets, it notifies the software by setting the RVER and RRSP fields of MAC_FPE_CTRL_STS register respectively. Optionally an interrupt can be generated. As such packets have empty (all-zero) data payload, they are dropped inside the MAC and not forwarded to the MRI.

Software can set the SVER and SRSP fields of MAC_FPE_CTRL_STS register to request MAC to transmit Verify and Respond mPackets respectively. Upon successful transmission of these frames, MAC clears the SVER/SRSP bits and sets the TVER & TRSP fields of MAC_FPE_CTRL_STS register. Optionally an interrupt can be generated when these events occur.

Note

1. For the preemption frames received with CRC error indication, ignore the following status fields:
   - Packet length
   - Dribble error indication
   - Runt frame indication
2. When FPE is configured, Forward Undersized Good Packets (FUP) bit of MTL_RxQ_Operation_Mode register should not be set.
3. When FPE is configured, do not disable CRC checking on receive. This is required because, internally, FPE error fragments are identified using CRC checks. For more details, see the DCRCC field in the MAC_Ext_Configuration Register in the "Register Descriptions" on page 605.

8.15.11 Frame Preemption and MMC Counter and Interrupt Registers

The following MMC counters and associated Interrupt registers are instantiated/present in the MAC when Frame Preemption feature and any one of the existing MMC counters are selected during RTL configuration.

Table 8-22 MMC Counters and Associated Interrupt Registers

Frame Assembly Error Counter	Description	Associated MMC Counter
Frame Assembly Error Counter	A 32-bit counter that provides the number of MAC frames with reassembly errors on the Receiver, due to mismatch in the Fragment Count value.	MMC_Rx_Packet_Assembly_Err_Cntr
Frame SMD Error Counter	A 32-bit counter that provides the number of received MAC frames rejected due to arriving with an SMD-C when there was no preceding preempted frame	MMC_Rx_Packet_SMD_Err_Cntr
Frame Assembly OK Counter	A 32-bit counter that provides the number of MAC frames that were successfully reassembled	MMC_Rx_Packet_Assembly_Ok_Cntr
MAC Rx Fragment Counter	A 32-bit counter that provides the number of additional mPackets received due to preemption	MMC_Rx_FPE_Fragment_Cntr
MAC Tx Fragment Counter	A 32-bit counter that provides the number of additional mPackets transmitted due to preemption	MMC_Tx_FPE_Fragment_Cntr
Hold Request Counter	A 32-bit counter that maintains the count of number of times a hold request is given to MAC	MMC_Tx_Hold_Req_Cntr

8.15.11.1 Additional Registers Associated With MMC Interrupts

Following are the additional registers associated with MMC interrupts for the MMC error counters:

- MMC_FPE_Tx_Interrupt
- MMC_FPE_Tx_Interrupt_Mask
- MMC_Rx_Interrupt
- MMC_FPE_Rx_Interrupt_Mask

8.16 Time-Based Scheduling

Time-based scheduling feature is suitable for traffic whose periodicity and rate are predictable. This is an optional feature. To improve the quality-of-service of such traffic,

- The transmit DMA fetches the packet from the host memory for transmission at designated time. This helps the software to setup the Transmit descriptors in advance even before packet is ready/available. It reduces the overhead on the software and avoids constant monitoring of the time and preparing descriptors just in time when the packet is targeted to be transmitted.
- The MAC transmits the packet only at the designated/pre-determined time even if the packets are fetched in advance. This helps in maintaining a constant transmission rate that can be consumed by the receiver station; therefore avoiding congestion and excessive buffering in the network.

8.16.1 Description of Time-Based Scheduling

The time-based scheduling feature supports fetching and launching an Ethernet packet at (or after) a pre-determined time. The time-based scheduling is supported only in the following modes/configurations:

- Full duplex mode
- Link speed is 100Mbps or higher
- MTL and higher configurations
- Configurations in which time-stamping is enabled.

Note
- Do not enable time-based scheduling for the channel for which TSO feature is enabled.
- In EST enabled configurations the ESTM mode bit setting impacts the time-based scheduling feature only when the DUT processes the GCL list.

8.16.2 Definitions Used

Following are the definitions specific to the time-based scheduling feature:

- **Launch Time**

  The time beyond which MTL can schedule the packet for transmission.
- **Fetch Time**

  The time beyond which the Tx DMA can schedule a packet-fetch from the host memory.
- **Expiry Time**

  The time beyond which the packet is dropped by MTL.

8.16.2.1 Launch Time

The launch time is specified in the Enhanced Normal Descriptor in the DMA configurations. For more details, see "Enhanced Descriptor for Time-Based Scheduling" on page 1350.

The launch time can also be specified as a Control Word for each packet in the MTL configurations.

When the launch time is specified in the DMA configuration, it is valid only for the specific frame that is scheduled for transmission. The launch time is reset after the transmission of one frame.

Following are the two formats of the launch time.

- Normal/Absolute Format
- EST/Offset Format

Normal/Absolute Format

In this format, the Launch time is an absolute time value at which the packet is launched for transmission. This format is supported in all configurations in which Time-Based Scheduling feature (DWC_EQOS_TBS) is enabled.

When both Time-Based scheduling (DWC_EQOS_TBS) and Enhancements to Scheduled Traffic, EST (DWC_EQOS_AV_EST) features are enabled, the launch time is interpreted in the Normal/Absolute format if the ESTM bit of the MTL_TBS_CTRL register is set to 0.

The launch time is a 32-bit value, where most-significant 8-bits represent the time in seconds and the rest 24 bits represent the time in 256 ns. The launch time is compared against the IEEE 1588 based System/PTP Time (bits[39:8]) and rolls over after 256 seconds.

The maximum value of the lower 32-bits of System Time is 999,999,999 decimal (0x3B9AC9FF) and it wraps to 0 when reaching this value (representing a full second). Therefore, the maximum value of the lower 24 bits of the Launch Time (after multiplying by 256) must be 0x3B9AC9.

As the maximum value of Launch time is 256 seconds,

- Launch Time is greater than current System Time when its value is between System Time[39:8] and System Time[39:8] + 128 sec
- Launch Time is less than current System Time when its value is between "System Time[39:8] + 128sec" and "System Time[39:8] + 256 sec", because this is a modulo 256 computation.

EST/Offset Format

In this format, the Launch time is a offset value relative to the time indicated by the Base Time Register (BTR) of the GCL list provided in the GCL Slot Number (GSN). The value in the BTR is always updated to the start time of the current loop of the GCL.

This format is enabled only when both Time-Based scheduling (DWC_EQOS_TBS) and Enhancements to Scheduled Traffic, EST (DWC_EQOS_AV_EST) features are enabled. For each packet, the GSN value and the Launch Time value are specified in the Enhanced Transmit Descriptor in the DMA configurations, or as Control Word in the MTL configurations.

The launch time offset is a 32-bit value; the upper 8-bits represent the time in seconds and the rest 24 bits represent the time in 256 ns, which is added to the BTR value corresponding to GCL Slot Number. The value of the Launch Time Offset must be smaller than the value of the Cycle Time (specified in the CTR register that is implemented when EST is enabled).

If the CTR is greater than or equal to 1s, the maximum value of the lower-24 bits of Launch Time offset should be 999,999,999 decimal (0x3B9AC9FF).

In this format, the launch time is a 64-bit value, which is interpreted as,

Launch Time = Launch GSN BTR[63:0] + (Launch Time Offset[31:0] << 8), which is compared with the System Time [63:0].

GSN BTR is the Base Time value at which the Launch GSN loop started execution.

GCL Slot Number (GSN)

When Time-Based scheduling (DWC_EQOS_TBS) and Enhancements to Scheduled Traffic, EST (DWC_EQOS_AV_EST) features are enabled, a modulo 16 count of the GCL loop count is implemented and known as GSN. The count is incremented for every new GCL loop; installation of new GCL list does not impact the count. The current GCL slot number can be obtained by reading the CGSN field of the MTL_EST_Status register.

The maximum value of GSN is 15. Therefore GSN values between Current GSN (CGSN field of MTL_EST_Status register) and CGSN+8 represent current or future slots; all other GSN values are interpreted as elapsed slots or past slots. So, for the correct interpretation of time, GSN value must be between CGSN and CGSN+8, for correct interpretation of time.

8.16.2.2 Launch Expiry Time

Normal/Absolute Mode

In the normal mode, when the LEOV (Launch Expiry Offset Valid bit of MTL_TBS_CTRL register) is set, the Launch Expiry Offset (LEOS field of MTL_TBS_CTRL register) determines the maximum amount of time a frame is eligible for launch, starting from the time the frame becomes eligible for launch.

The Launch Expiry Offset is a 24-bit value defined in 256ns units, with a maximum possible value of 999,999,999 ns (0x3B9AC9FF).

Launch Expiry Time = (Launch Time[39:8] + LEOS[32:8]) * 256ns

The packet with a specific Launch time is considered eligible for transmission when the Launch time is less than the System Time and (if LEOV is set) the System Time is less than the Launch Expiry time.

When the System time is greater than the Launch Expiry time the frame is categorized as expired and will be dropped from the MTL FIFO.

Note

1. For correct interpretation and meaningful operation, the Fetch, Launch, and Launch Expiry time should never be set to a value larger than Current System Time + 128 sec; such a value is interpreted as time that has already elapsed.
2. In full duplex mode, the frames dropped from the MTL FIFO have Error Summary (Bit 15) and Excessive Deferral (bit 3) of TxStatus set.

EST/Offset Mode

In the EST mode of operation, when the LEOV field of the MTL_TBS_CTRL register is set, the launch expiry GSN offset (LEGOS field of the MTL_TBS_CTRL register) and Launch Expiry Offset (LEOS field of MTL_TBS_CTRL register) determine the maximum amount of time a frame remains eligible for launch, starting from the time the frame becomes eligible for launch.

Launch Expiry Offset = (LEGOS: LEOS)

LEGOS holds the GSN offset (multiples of CTR time) and LEOS holds maximum value of CTR (sub CTR values) in ns.

Launch Expiry offset is a 24-bit value defined in the units of 256 ns, with a maximum possible value of the smaller of 999,999,999ns or CTR-1 ns.

The Launch Expiry GSN is computed as follows:

Launch Expiry GSN = (Launch GSN + LEGOS + CCMA)

Where,

CCMA is the CTR Carry due to Modulo addition. This value is 1 if ((Launch Time Offset + LEOS) << 8) is equal to or greater than CTR. When CCMA = 1,

Launch Expiry Offset = (Launch Time Offset+ LEOS) - CTR

When CCMA = 0,

Launch Expiry Offset = (Launch Time Offset + LEOS).

Launch Expiry Time = Launch Expiry GSN BTR[63:0] + Launch Expiry Time Offset.

When LEOV is not set, Launch Expiry Time is not checked.

When LEOV is set, and

- the system time is greater than the Launch Expiry time, the frame is dropped from MTL FIFO. The frame is considered as expired.
- the launch time is smaller than the System Time and Launch Expiry Time is greater than system time, the frame is considered eligible for launch.

Note

1. Max value of LEGOS is 7. This implies that when LEOV is set, the frame has a maximum life time of <8 GCL loop iterations after it becomes eligible for Launch.
2. The slot number of the First GCL List executed each time after EST is enabled, is zero.

8.16.2.3 Fetch Time

In configurations with DMA, the Fetch Time can also be indicated for each packet by the software. This is done by setting FTOV field of DMA_TBS_CTRL register.

If the FTOV filed is not set, the Fetch Time Offset is not valid and the DMA fetches packets without any time constraints.

The fetch time accounts for all possible delays in the DMA fetch operation and ensures that the frame is present in the MTL FIFO before the launch time.

Normal/Absolute Mode

In the Normal mode fetch time derived/calculated by reducing the time specified in the Fetch Time Offset (FTOS) field of DMA_TBS_CTRL register from the given Launch time.

Normal mode of operation, the Fetch Launch Time is computed as:

Fetch Time[39:8] = (Launch Time[39:8] - FTOS[31:8])

The Fetch time is 32-bits and is compared against System Time[39:8] to determine the Eligible for fetching the frame:

- The Fetch time is defined as "greater" than System Time if the Fetch Time is in the range of "System Time[39:8]" and "System Time[39:8] + 128 sec". The frame is considered as not-eligible for fetch.
- The Fetch time is defined as “smaller” than the System Time if the Fetch time is in the range of “System Time[39:8] + 128 sec” and “System Time[39:8] + 256 sec”. The frame is considered as eligible for fetch.

This is a modulo 256 computation.

EST/Offset Mode

In the EST/Offset mode, the Fetch GSN Offset (FGOS field of MTL_TBS_CTRL register) provides the Slot Number offset to be deducted from the Launch GSN. In this case the FTOS value should be the smaller of 999,999,999ns or CTR-1 ns.

If (Launch Time Offset >= FTOS):

Fetch Time Offset = ((Launch Time Offset - FTOS) * 256ns)

CBFS(CTR Borrow for Fetch Subtraction) = 0

If (Launch Time Offset < FTOS)

Fetch Time Offset = CTR + ((Launch Time Offset - FTOS) * 256ns)

CBFS (CTR Borrow for Fetch Subtraction) = 1

The Fetch GSN is computed as follows:

Fetch GSN = Launch GSN - FGOS - CBFS

Fetch Time = Fetch GSN BTR[63:0] + Fetch Time Offset

The frame is marked eligible for DMA fetch when the fetch time is smaller than the System Time.

Note Max value of FGOS is 7. This implies that when FTOV is set, the frame can be fetched at a maximum of <8 GCL loop iterations before it becomes eligible for Launch.

DMA Operations Sequence

Following is the sequence of operation when FTOV = 1.

1. Fetch the first Enhanced Normal Descriptor. (FD is set).
2. If LTV is set and Fetch enabled, compute the Fetch Time based on the Launch Time. Wait for the System Time to be greater than Fetch Time
3. Read the Frame (Data) from the host memory and transfer to MTL FIFO
4. Close the Normal Descriptor
5. Fetch the next Normal Descriptor (if the previous Descriptor was not the Last)
6. Repeat steps 4 to 6, until the last descriptor of the frame (LD is set)

After the Last Descriptor of a frame, the next enhanced normal descriptor should be programmed with a new Launch Time and with LTV bit set. Otherwise, the subsequent frames will be processed without any time restrictions.

Control Word (MTL Configurations)

Time-based scheduling ensures that a packet is allowed to arbitrate for transmission only if the System Time is greater than the Launch Time.

In MTL configuration,

- Launch Time and GSN, use the same field that is used to hold the 64-bit timestamp value for OSTC (One Step Timestamping Correction). If the LTV bit is set and OSTC bit is not set (bit 19 of first control word), the Time Stamp field is interpreted as Launch Time. Launch Time uses the fields meant for OSTC TimeStamp[39:8] and GSN uses OSTC TimeStamp[43:40] field.
- Bit-31 of Control-Word0 is the equivalent of the LTV field, and the last two Control D-Words is used for providing the Launch Time and GST.
- The last two Control D-words is considered to the Time Stamp for OSTC, when bit-19 of Control-Word0 (OSTC) is set.

When the MTL TXFIFO Read Controller reads the first control word, it checks bit-31 to determine if the frame has a valid Launch Time. If bit-31 is set, it reads the subsequent control words to get the Launch Time.

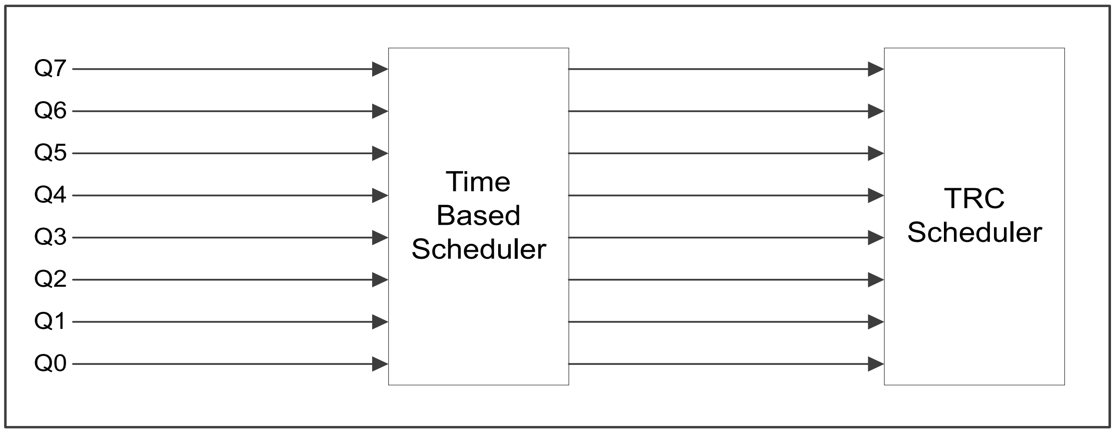
Impact on Transmission Selection Algorithm

Time-Based Scheduler does not directly influence the Transmission Selection Algorithm but has an indirect influence as it determines if a queue is eligible to participate in the scheduling.

If a frame of a specific queue has a valid Launch Time, the Time Based Scheduler ensures that the frame participates in Transmit Scheduling (TRC Scheduler) only after the Launch time has elapsed.

The gating done by Time-Base scheduler is only for the Scheduling (picking frame for Transmission). It might not alter any other characteristics and behavior of the frame. For example, the Frame waiting for the Launch time to lapse will continue to gain credits in CBS, and the CBS credits will not be lost as the frame is marked as available but not ready. Strict Priority or CBS are recommended Transmit Scheduling algorithms for the TBS implementation because, scheduling for TBS enabled queues are more predictable.

Figure 8-17 Time Based Scheduler Implementation



8.16.3 Enabling the Time-Based Scheduling Feature

To enable this feature while configuring the controller in the coreConsultant GUI:

1. Select the **Time Based Scheduler** feature during the **Specify Configuration** activity in coreConsultant.
2. Select **Enable Time Based Scheduling** option.

For details about this option, see Time Based Scheduling Parameters in the "Parameter Descriptions" on page 419.

Note See the "Enabled" field of the parameter descriptions to understand the dependencies.

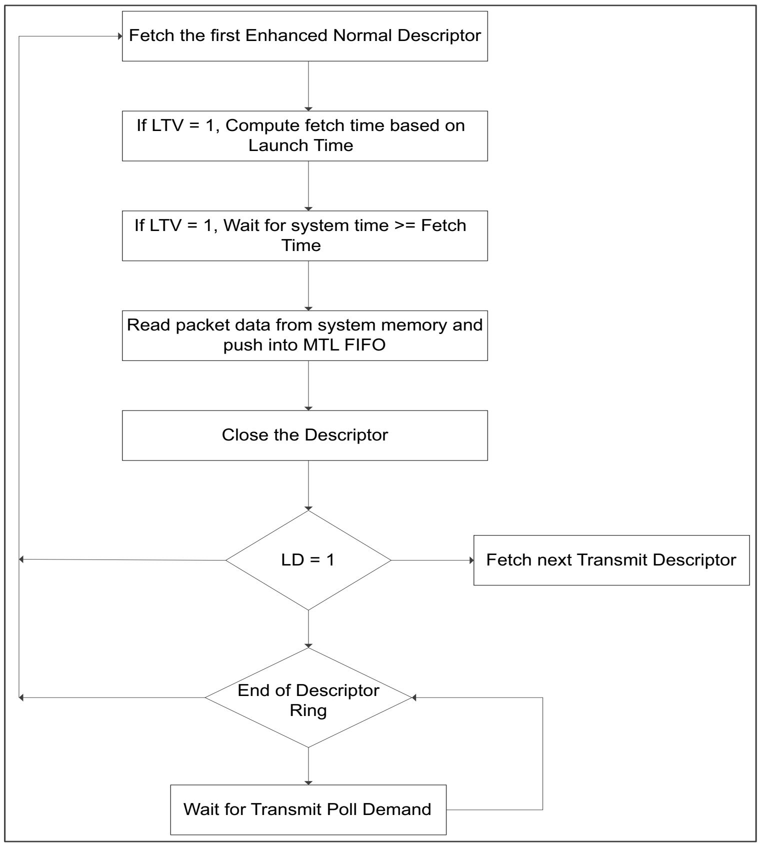
8.16.4 Programming Guidelines for Time-Based Scheduling

For details about programming guidelines, see "Programming the Launch Time in Time-Based Scheduling" on page 1382.

8.16.5 Time-Based Scheduling Flow

The following figure shows the time-based scheduling flow.

Figure 8-18 Time-Based Scheduling Flow


Fetch the first Enhanced Normal Descriptor
If LTV = 1, Compute fetch time based on Launch Time
If LTV = 1, Wait for system time >= Fetch Time
Read packet data from system memory and push into MTL FIFO
Close the Descriptor
LD = 1
Fetch next Transmit Descriptor
End of Descriptor Ring
Wait for Transmit Poll Demand

9 TCP/IP Offloading Features

Communication protocols such as TCP and UDP implement checksum fields, which help determine the integrity of data transmitted over a network. The most widespread use of Ethernet is to encapsulate TCP and UDP over IP datagrams. Therefore, the MAC has an optional Checksum Offload Engine (COE) to support checksum calculation and insertion in the Transmit path, and error detection in the Receive path.

The TCP Segmentation Offload (TSO) engine is useful in offloading the TCP segmentation functions to the hardware.

This chapter contains the following sections:

9.1 Transmit Checksum Offload Engine

9.1.1 Overview of Transmit Checksum Offload Engine

The MAC has a Checksum Offload Engine (COE) to support checksum calculation and insertion in the Transmit path, using which, the software can offload the task of checksum insertion to the hardware. In the transmit path MAC calculates the checksum and inserts it in the Tx packet. This feature helps in reducing the load on the software and can improve the overall throughput of the system.

9.1.2 Description of Transmit Checksum Offload Engine

The checksum offload engine module supports two types of checksum calculation and insertion. The checksum engine can be controlled for each packet by setting the CIC bits (TDES3 Bits[17:16]) in EQOS-AHB, EQOS-DMA, or EQOS-AXI configurations. In EQOS-MTL configuration, the ati_chksum_ctrl_i signal controls the operation.

Note The checksum for TCP, UDP, or ICMP is calculated over a complete packet, and then inserted into its corresponding header field. Because of this requirement, when this function is enabled, the Tx FIFO automatically operates in the store-and-forward mode even if the DWC_ether_qos is configured for Threshold (cut-through) mode.

Note You must make sure that the Tx FIFO is deep enough to store a complete packet before that packet is transferred to the MAC transmitter. The reason being that when space is not available to accept the programmed burst length of data, then the MTL Tx FIFO starts reading to avoid dead-lock. In such a case, the COE fails as the start of the packet header is read out before the payload checksum can be calculated and inserted. Therefore, you must enable the checksum insertion only in the packets that are less than the number of bytes, given by the following equation:

```
        Packet size < TxQSize - (PBL + N)*(DATAWIDTH/8),
```

Where,

- Datawidth = DWC_EQOS_DATAWIDTH parameter
- If Datawidth = 32, N = 7, elseif Datawidth != 32, N = 5.
- TxQSize is indicated by TQS field of MTL_TxQ#_Operation_Mode register
- PBL = value of ati_q#_pbl_i signal in EQS-MTL configuration
  OR
  PBL = TxPBL field in the DMA_CH#_TX_Control register in all DMA configurations.

Hint See IETF specifications RFC 791, RFC 793, RFC 768, RFC 792, RFC 2460, and RFC 4443 for IPv4, TCP, UDP, ICMP, IPv6, and ICMPv6 packet header specifications, respectively.

9.1.2.1 IP Header Checksum Engine

In IPv4 datagrams, the integrity of the header fields is indicated by the 16-bit Header Checksum field (the eleventh and twelfth bytes of the IPv4 datagram). The COE detects an IPv4 datagram when the Type field of

Ethernet packet has the value 0x0800 and the Version field of IP datagram has the value 0x4. The checksum field of the input packet is ignored during calculation and replaced with the calculated value.

Note IPv6 headers do not have a checksum field. Therefore, the COE does not modify the IPv6 header fields.

The result of this IP header checksum calculation is indicated by the IP Header Error status bit in the Transmit status (Bit 0 in Table 19-12 on page 1330). This status bit is set whenever the values of the Ethernet Type field and the Version field of IP header are not consistent, or when the Ethernet packet does not have enough data, as indicated by the IP header Length field. In other words, this bit is set when an IP header error is asserted under the following circumstances:

- For IPv4 datagrams:
  - The received Ethernet type is 0x0800, but the Version field of IP header is not equal to 0x4.
  - The IPv4 Header Length field indicates a value less than 0x5 (20 bytes).
  - The total packet length is less than the value given in the IPv4 Header Length field.
- For IPv6 datagrams:
  - The Ethernet type is 0x86dd but the IP header Version field is not equal to 0x6.
  - The packet ends before the IPv6 header (40 bytes) or extension header (as given in the corresponding Header Length field in an extension header) is completely received.

9.1.2.2 TCP/UDP/ICMP Checksum Engine

The TCP/UDP/ICMP Checksum Engine processes the IPv4 or IPv6 header (including extension headers) and determines whether the encapsulated payload is TCP, UDP, or ICMP. The checksum is calculated for the TCP, UDP, or ICMP payload and inserted into its corresponding field in the header. The Tx COE can work in the following two modes:

- The TCP, UDP, or ICMPv6 pseudo-header is not included in the checksum calculation and is assumed to be present in the Checksum field of the input packet. This engine includes the Checksum field in the checksum calculation, and then replaces the Checksum field with the final calculated checksum.
- The engine ignores the Checksum field, includes the TCP, UDP, or ICMPv6 pseudo-header data into the checksum calculation, and overwrites the checksum field with the final calculated value.

Note For ICMP-over-IPv4 packets, the Checksum field in the ICMP packet must always be 16'h0000 in both modes, because pseudo-headers are not defined for such packets. If it does not equal 16’h0000, an incorrect checksum may be inserted into the packet.

The result of this operation is indicated by the Payload Checksum Error status bit in the Transmit Status vector (Bit 12 in Table 19-3 on page 1324). This engine sets the Payload Checksum Error status bit when it detects that the packet has been forwarded to the MAC Transmitter engine in the store-and-forward mode without the end of packet (EOP) being written to the FIFO, or when the packet ends before the number of bytes indicated by the Payload Length field in the IP Header is received. When the packet is longer than the indicated payload length, the COE ignores them as stuff bytes, and no error is reported. When this engine detects the first type of error, it does not modify the TCP, UDP, or ICMP header. For the second error type, it still inserts the calculated checksum into the corresponding header field.

Table 9-1 describes the functions supported by Transmit Checksum Offload engine based on the packet type. When the MAC does not insert the checksum, it is indicated as "No" in the table.

Note You should not enable checksum insertion for IPv4 or IPv6 packets that are greater than the frame size constraint specified in Description of Transmit Checksum Offload Engine because it may result in incorrect checksum insertion or unexpected behavior.

Table 9-1 Transmit Checksum Offload Engine Functions for Different Packet Types

Packet Type	Hardware IP headerchecksum insertion	Hardware TCP/UDPchecksum insertion
Non-IPv4 or IPv6 packet	No	No
IPv4 packet is greater than 1,522 bytes (2,000 bytes when IEEE 802.3ad Support for 2K Packets is enabled in MAC) but less than or equal to the frame size constraint specified in "Description of Transmit Checksum Offload Engine" on page 340.	Yes	Yes
IPv6 packet is greater than 1,522 bytes (2,000 bytes when IEEE 802.3ad Support for 2K Packets is enabled in MAC) but less than or equal to the frame size constraint specified in "Description of Transmit Checksum Offload Engine" on page 340.	Not Applicable	Yes
IPv4 with TCP, UDP, or ICMP	Yes	Yes
IPv4 packet has IP options (IP header is longer than 20 bytes)	Yes	Yes
Packet is an IPv4 fragment	Yes	No
IPv6 packet with the following next header fields in main or extension headers: ■ Hop-by-hop options (in IPv6 main header) ■ Hop-by-hop options (in IPv6 extension header) ■ Destinations options ■ Routing (with segment left 0) ■ Routing (with segment left > 0) ■ TCP, UDP, or ICMP ■ Authentication ■ Any other next header field in main or extension headers	■ Not Applicable ■ Not Applicable ■ Not Applicable ■ Not Applicable ■ Not Applicable ■ Not Applicable ■ Not Applicable ■ Not Applicable	■ Yes ■ No ■ Yes ■ No ■ No ■ Yes ■ No ■ No
IPv4 packet has TCP header with Options fields	Yes	Yes

Packet Type	Hardware IP headerchecksum insertion	Hardware TCP/UDPchecksum insertion
IPv4 Tunnels: ■ IPv4 packet in an IPv4 tunnel ■ IPv6 packet in an IPv4 tunnel	■ Yes (IPv4 tunnel header) ■ Yes (IPv4 tunnel header)	■ No ■ No
IPv6 Tunnels: ■ IPv4 packet in an IPv6 tunnel ■ IPv6 packet in an IPv6 tunnel	■ Not applicable ■ Not applicable	■ No ■ No
IPv4 packet has 802.3ac tag (with C-VLAN Tag or S-VLAN Tag when enabled).	Yes	Yes
IPv6 packet has 802.3ac tag (with C-VLAN Tag or S-VLAN Tag when enabled).	Not applicable	Yes
IPv4 frames with security features (such as encapsulated security payload)	Yes	No
IPv6 frames with security features (such as encapsulated security payload)	Not applicable	No

9.1.3 Enabling the Transmit Checksum Offload Engine

To enable this feature while configuring the controller in the coreConsultant GUI:

1. Select the **Enable Transmit TCP/IP Checksum Offload** option under the TCP/IP Offloading Features section during the **Specify Configuration** activity in coreConsultant.
2. Select the **Enable Support for Tx COE in Tx Queue 0** to **Enable Support for Tx COE in Tx Queue 7** options if you want to enable separate COE for each Tx queue.

For details about these option, see TCP/IP Offloading Features Parameters in the "Parameter Descriptions" on page 419.

Note See the "Enabled" field of the parameter descriptions to understand the dependencies.

9.2 Receive Checksum Offload Engine

9.2.1 Overview of Receive Checksum Offload Engine

DWC_ether_qos provides the Checksum Offload Engine that is used to detect any error in an IPv4 or IPv6 packet in the receive path. The MAC verifies the checksum field of the received packet with the internally calculated checksum and provides the status.

9.2.2 Description of Receive Checksum Offload Engine

The Receive Checksum Offload engine is used to detect errors in IP packets by calculating the header checksum and further matching it with the received header checksum. This engine also identifies a TCP, UDP, or ICMP payload in received IP packets and calculates the checksum of such payloads appropriately.

Here, both IPv4 and IPv6 packet in the received Ethernet packets are detected and processed for data integrity. The MAC receiver identifies IPv4 or IPv6 packets by checking for value 0x0800 or 0x86DD, respectively, in the Type field of the received Ethernet packet. This identification is applicable to single VLAN-tagged packets. It is also applicable to double VLAN-tagged packets when the Enable Double VLAN Processing option is selected and the EDVLP bit of the MAC_VLAN_Tag register is set.

The Rx COE calculates the IPv4 header checksums and checks that they match the received IPv4 header checksums. The result of this operation (pass or fail) is given to the RFC module for insertion into the receive status word. The IP Header Error bit is set for any mismatch between the indicated payload type (Ethernet Type field) and the IP header version, or when the received packet does not have enough bytes, as indicated by the Length field of the IPv4 header (or when fewer than 20 bytes are available in an IPv4 or IPv6 header).

This engine also identifies a TCP, UDP, or ICMP payload in the received IP datagrams (IPv4 or IPv6) and calculates the checksum of such payloads properly, as defined in the TCP, UDP, or ICMP specifications. This engine includes the TCP, UDP, or ICMPv6 pseudo-header bytes for checksum calculation and checks whether the received checksum field matches the calculated value. The result of this operation is given as a Payload Checksum Error bit in the receive status word. This status bit is also set if the length of the TCP, UDP, or ICMP payload does not match the expected payload length given in the IP header.

Table 9-2 describes the functions supported by the Rx COE based on the packet type. When the payload of an IP packet is not processed (indicated as "No" in the table), the information (whether the checksum engine is bypassed or not) is given in the receive status.

Note The MAC does not append any payload checksum bytes to the received Ethernet packets.

Table 9-2 Receive Checksum Offload Engine Functions for Different Packet Types

Packet type	Hardware IP header checksum checking	Hardware TCP/UDP/ICMP checksum checking
Non-IPv4 or IPv6	No	No
IPv4 packet is greater than 1,522 bytes (2,000 bytes when IEEE 802.3ad Support for 2K Packets is enabled in the MAC)	Yes	Yes

Packet type	Hardware IP header checksum checking	Hardware TCP/UDP/ICMP checksum checking
IPv6 packet is greater than 1,522 bytes (2,000 bytes when IEEE 802.3ad Support for 2K Packets is enabled in the MAC)	Not Applicable	Yes
IPv4 with TCP, UDP, or ICMP	Yes	Yes
IPv4 header's protocol field contains a protocol other than TCP, UDP, or ICMP	Yes	No
IPv4 packet has IP options (IP header is longer than 20 bytes)	Yes	Yes
Packet is an IPv4 fragment	Yes	No
IPv6 packet with the following next header fields in main or extension headers: ■ Hop-by-hop options (in IPv6 main header) ■ Hop-by-hop options (in IPv6 extension header) ■ Destinations options ■ Routing (with segment left 0) ■ Routing (with segment left > 0) ■ TCP, UDP, or ICMP ■ Any other next header field in main or extension headers	■ Not Applicable ■ Not Applicable ■ Not Applicable ■ Not Applicable ■ Not Applicable ■ Not Applicable ■ Not Applicable	■ Yes ■ No ■ Yes ■ Yes ■ No ■ Yes ■ No
IPv4 packet has TCP header with Options fields	Yes	Yes
IPv4 Tunnels: ■ IPv4 packet in an IPv4 tunnel ■ IPv6 packet in an IPv4 tunnel	■ Yes (IPv4 tunnel header) ■ Yes (IPv4 tunnel header)	■ No ■ No
IPv4 Tunnels: ■ IPv4 packet in an IPv6 tunnel ■ IPv6 packet in an IPv6 tunnel	■ Not Applicable ■ Not Applicable	■ No ■ No
IPv4 packet has 802.3ac tag (with C-VLAN Tag or S-VLAN Tag when enabled)	Yes	Yes
IPv6 packet has 802.3ac tag (with C-VLAN Tag or S-VLAN Tag when enabled)	Not Applicable	Yes

Packet type	Hardware IP header checksum checking	Hardware TCP/UDP/ICMP checksum checking
IPv4 frames with security features (such as encapsulated security payload)	Yes	No
IPv6 frames with security features (such as encapsulated security payload)	Not Applicable	No

9.2.3 Enabling the Receive Checksum Offload Engine

You can enable the Receive Checksum Offload Engine feature in DWC_ether_qos using the coreConsultant GUI.

1. Select the **Enable Receive TCP/IP Checksum Offload** option under the TCP/IP Offloading Features section during the Specify Configuration activity in coreConsultant.

Note Select the Enable Receive TCP/IP Checksum Offload option under the TCP/IP Offloading Features section during the Specify Configuration activity in coreConsultant.

For details about this option, see TCP/IP Offloading Features Parameters in "Parameter Descriptions" on page 419.

Note See the "Enabled" field of the parameter descriptions to understand the dependencies.

2. Set the IPC bit of the MAC_Configuration register.

9.3 Header-Payload Split

In configurations with DMA, the MAC is capable of identifying the boundary between header and payload of the received packet and store them into separate buffers. This feature is useful for multiple reasons:

- Header buffers can be located in faster memory/cache while Payload buffers are located in slower DRAM memory. This enables the software stack to process the headers faster
- Buffers used for payload can be directly forwarded to the application layer without the need to copy the payload into application buffer
- In case Large Receive Offload (LRO) function is implemented in software/driver layer, it is easy to link the buffers of multiple and contiguous payload data and form a bigger packet with a new header. This reduces the number of packets forwarded to the upper layer and thus improves system software performance

The DWC_ether_qos supports multiple methods and packet types for header-payload splitting. This is controlled by the setting the SPLM field in MAC_Ext1_cfg register and described in Table 9-3.

Table 9-3 Split Header Support Depending on the Packet Type

Packet Type	SPLM	Description
TCP or UDP Packet	00 (L3/L4 Split)	The DMA writes the Ethernet header + IP header + TCP or UDP header into the header buffer.
IP packet (not TCP/UDP)		The DMA writes the Ethernet header + IP header into the header buffer.
Non-IP packet		The DMA does not split the header and payload
Any packet	01 (L2 Split)	The DMA writes the Ethernet header based on split offset (SPLOFST) into the header buffer.
IP packet	10 (Combination of L2 or L3/L4 Split)	L3/L4 split
Non-IP packet		L2 split
NA	11	Reserved

Note L3/L4 split is applicable for IP packets that are either untagged or VLAN stripped. If VLAN tag is retained in the packet forwarded to the DMA, L3-L4 split is not performed.

The IP header includes IPv4 options in case of a IPv4 packet, and IPv6 extension headers in case of IPv6 frames. The points at which the L3/L4 header is split are shown in Figure 9-1.

Figure 9-1 L3/L4 Header Split Points

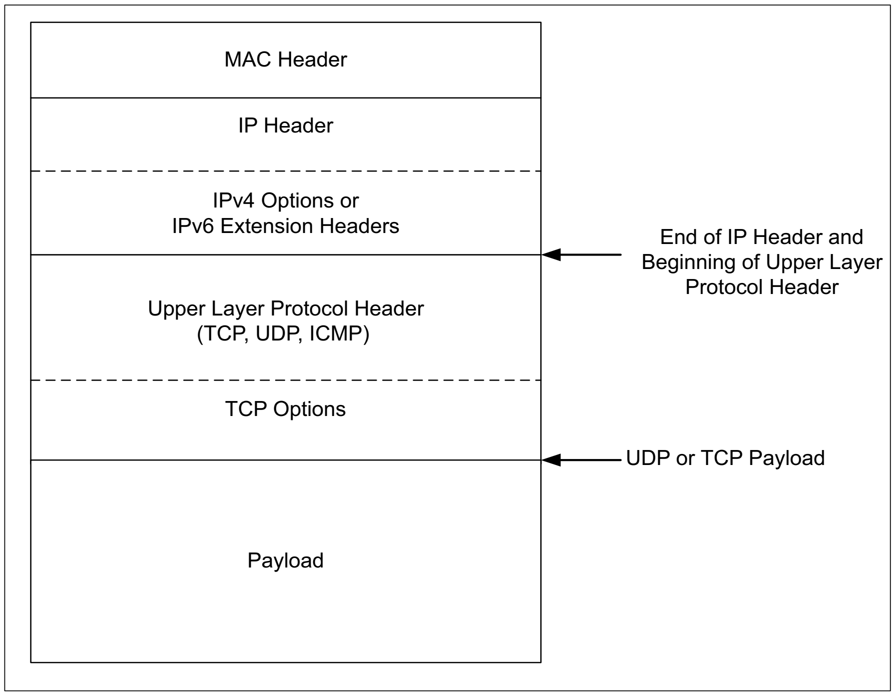
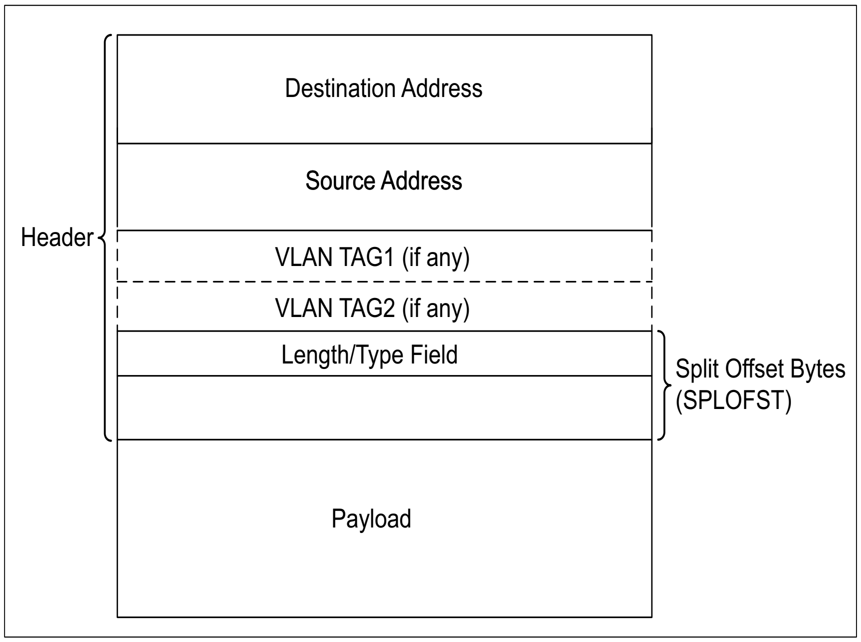

MAC Header
IP Header
IPv4 Options or IPv6 Extension Headers
Upper Layer Protocol Header (TCP, UDP, ICMP)
TCP Options
Payload
End of IP Header and Beginning of Upper Layer Protocol Header
UDP or TCP Payload

Figure 9-2 shows the L2 Split boundary. In this mode, the header payload boundary is determined for received packets at a fixed length offset starting from the Length/Type field, as indicated by the SPLOFST field of the MAC_Ext_Cfg1 register. This mode is applicable for all received packets when SPLM=01 and only for non-IP packets when SPLM=10. For more details, see the "Register Descriptions" on page 605.

For more details about the descriptor structure for the split header feature, see "Descriptor Structure for Split Header Support" on page 1319.

Figure 9-2 L2 Header Split



9.3.1 Enabling Splitting the Header on Receive

1. Select the **Enable Split Header Feature** option under the TCP/IP Offloading Features section during the Specify Configuration activity in coreConsultant.

   For details about this option, see TCP/IP Offloading Features in "Parameter Descriptions" on page 419.

Note See the "Enabled" field of the parameter descriptions to understand the dependencies.

9.3.2 Programming Guidelines for Splitting the Header on Receive

See "Programming Guidelines for Header Payload Split Receive" on page 1378.

9.4 TCP/IP Segmentation Offload (TSO) Engine

9.4.1 Overview of TCP/IP Segmentation Offload Engine

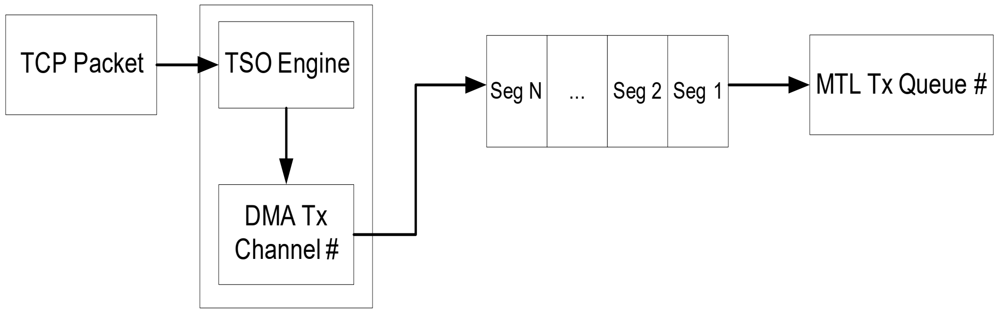
The TCP Segmentation Offload (TSO) engine is useful in offloading the TCP segmentation functions to the hardware.

It also supports UDP Segmentation Offload (USO) in which the UDP payload is segmented in the hardware. There are no sequencing/ordering controls available or updated in the segments, so it can be used in point to point applications in which out of order packets are not expected by the receiver. The description and flow of TSO mentioned in this section is same as USO, any deviation is provided as notes.

9.4.2 Description of TCP Segmentation Offload Engine

In the TCP segmentation offload (TSO) feature, the DMA splits a large TCP packet into multiple small packets and passes these packets to the MTL as shown in Figure 9-3.

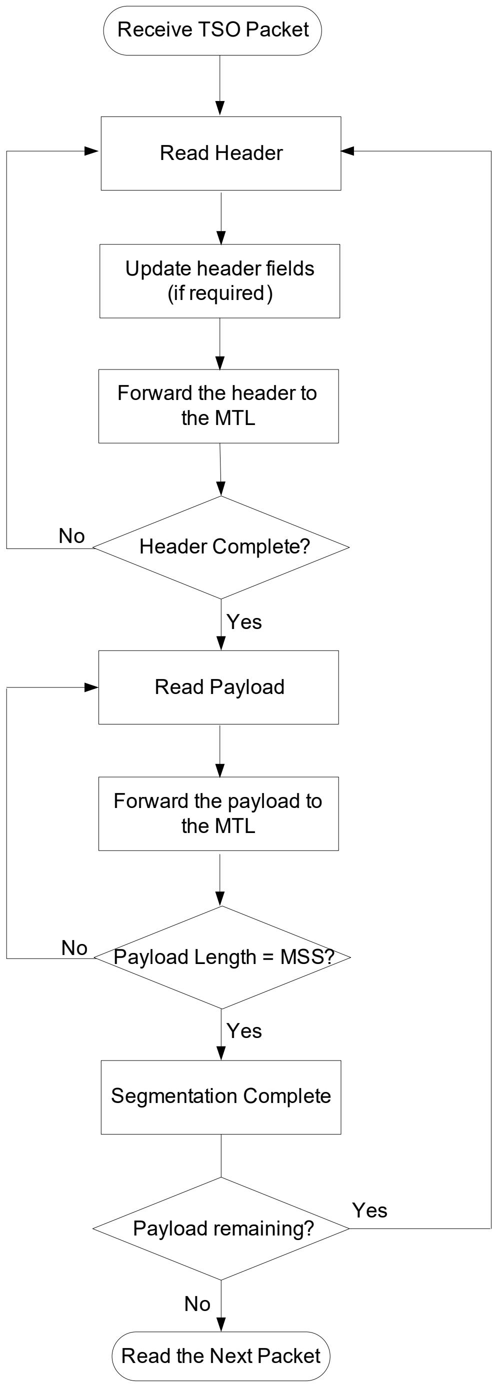
Figure 9-3 TCP Segmentation Offload Overview



9.4.2.1 DMA Operation with TSO Feature

The Figure 9-4 shows the TSO flow.

Figure 9-4 TCP Segmentation Offload Flow



For the TSO feature, the Tx DMA operation is as follows:

1. The application sets up the Transmit descriptor (TDES0-TDES3) and sets the Own bit (TDES3[31]) after setting up the corresponding data buffer(s) with Ethernet Packet data.
2. The application increases the offset value of the Descriptor Tail pointer of the DMA Tx channel.
3. While in the Run state, the DMA fetches the next available descriptor and does one of the following:

- If the descriptor is a context descriptor and the context is not between the first and last descriptors of a packet, the DMA stores the context values.
- If the descriptor is a context descriptor and the context is between the first and last descriptors of a packet, the DMA closes the context descriptor indicating a Context Descriptor Error (TDES3[23]) and fetches the next descriptor.
- If the descriptor is a normal descriptor, the DMA checks for the TSE bit. If TSE bit is not set, the DMA continues with the default mode of operation or OSF operation (if enabled).

4. The DMA calculates the number of segments from the TCP payload length(TDES3[17:0]) and the MSS value.
5. The DMA goes through channel arbitration to fetch the data buffers. The DMA fetches the header and payload separately.
6. For the first segment, the DMA fetches the header from the system memory and stores it in the TSO memory (if present and when the length of header is not greater than the TSO memory size).

   If the current segmented packet is not the first segment, the DMA fetches the header from the local TSO memory if available. Otherwise, it will fetch the header buffer in system memory again, as done for the first segment. In such cases (header not available is TSO memory), the DMA will not close the first descriptor containing the header buffer until the header for last segment is fetched.
7. The required fields in the header bytes are modified/updated as per the segmentation requirements and written into the corresponding MTL TxQ.
8. The DMA then takes the payload buffer pointer, fetches the MSS number of payload bytes from the system memory and directly pushes it into the MTL TxQ. In case the buffer(s) in the descriptor do not have enough data for the MSS payload (except for the last segment), the DMA closes this descriptor.
9. The DMA jumps to Step 3 and repeats the process until the last segment is written into the TxQ.
10. The DMA closes the last descriptor and also the first descriptor (containing the header buffer when it is not stored in TSO memory) and then moves on to the next packet transfer.

The DMA repeats all these steps if more descriptors are available. When no descriptor is available, the DMA enters the suspend state.

Note The TSO engine determines whether to perform TSO or USO operation based on the THL field (TCP Header Length) in TDES3 of first Normal Tx descriptor for the packet. The value of 2 indicates USO and any value greater than or equal to 5 indicates TSO.

The DMA repeats all these steps if more descriptors are available. When no descriptor is available, the DMA enters the suspend state.

9.4.2.2 TCP/IP Header Fields

While segmenting a TCP packet, the DMA automatically updates the TCP/IP header fields. Table 9-4 describes how the TCP and IP headers are updated.

Table 9-4 TSO: TCP and IP Header Fields

Packet Sequence	TCP Header	IP Header
First packet	1. The sequence number is not updated. The value provided in the header is used. 2. The TCP checksum is calculated again. 3. If set, the FIN and PSH flags are cleared.	■ IPv4 Header ❑ Total Length = MSS + TCP Header Length + IP Header Length ❑ Identification field is not modified. It is sent as per the header provided by the software. ❑ IPv4 Header Checksum is recalculated. ■ IPv6 Header Payload Length = MSS + TCP Header Length + IPExtension Header Length
Subsequent packets	1. The sequence number is updated. The MSS value is added to the sequence number value of previous segment. 2. If set, the FIN and PSH flags are cleared. 3. The TCP checksum is calculated again.	■ IPv4 Header ❑ Total Length = MSS + TCP Header Length + IP Header Length ❑ Identification field = Previous Identification Field + 1 ❑ IPv4 Header Checksum is recalculated ■ IPv6 Header Payload Length = MSS + TCP Header Length + IPExtension Header Length
Last packet	1. The sequence number is updated. The MSS value is added to the sequence number value of previous segment. 2. If FIN and PSH flags were set in original header, these flags are set. 3. The TCP checksum is calculated again.	■ IPv4 Header ❑ Total Length = Remaining Payload + TCP Header Length + IP Header Length ❑ Identification Field = Previous Identification Field + 1 ❑ IPv4 header Checksum is recalculated ■ IPv6 Header Payload Length = Remaining Payload Length + TCP Header Length + IP Extension Header Length

Note In case of USO, the engine updates only the following fields in the IP-UDP headers:

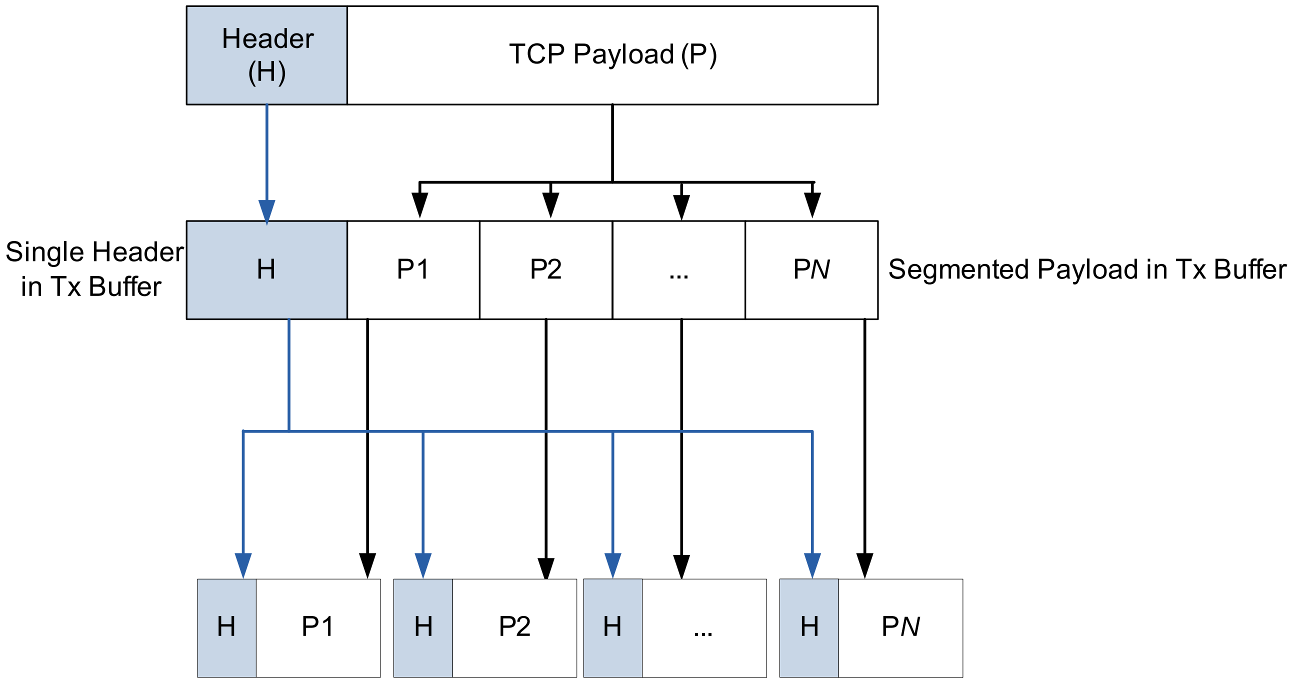
- Total Length, Identification Field, and Header Checksum field in IPv4 header
- Payload Length field in IPv6 header
- Length and Checksum field in UDP header

9.4.2.3 Header and Payload Fields of Segmented Packets

After segmentation, the split packets use the same header as the parent TCP packet for header fields other than the ones described in Table 9-4. Figure 9-5 shows how same header is used for the header fields of segmented packets.

The application must create the header in Buffer 1 of the first descriptor of the packet to be segmented and provide the header length in TDES2 of the first descriptor (FD = 1). When the FD bit is set, the DMA reads the header from the header buffer to which the TDES0 is pointing. Buffer 2 of the first descriptor can be used for payload and TDES0 and TDES1 of subsequent descriptors. For subsequent descriptors (FD = 0), the address to which the TDES0 and TDES1 are pointing is treated as payload buffer address of the same packet.

Figure 9-5 Header and Payload Fields of Segmented Packets



9.4.2.4 Context Descriptor Sequence

You can use the context descriptor to provide the MSS value for segmentation. The application must provide the context descriptor before the normal descriptor to be used for the corresponding TCP packet. If the application wants to provide a new MSS, it must create the context descriptor in the descriptor list before the first normal descriptor of the packet to be segmented with the new MSS value. The MSS value in the context descriptor is valid only if the TCMSSV bit of TDES3 in context descriptor is set and the OSTC bit is reset.

When the application provides a context descriptor with a valid MSS value, the DMA internally stores the MSS value and uses this MSS value for all subsequent packets that have the TSO Enabled through the TSE bit of TDES3 normal descriptor.

If the application places a context descriptor in the middle of a packet (between the first and last descriptors of a packet), the DMA does the following:

1. The DMA ignores the context and closes the descriptor.
2. The DMA indicates the error in descriptor status
3. The DMA generates an interrupt if the CDEE bit is set in the Interrupt_Enable register corresponding to a DMA channel.

The application can read the interrupt status through CDE bit of Status register corresponding to a DMA channel.

9.4.3 Building the Descriptor and the Packet for the TSO Feature

Set the TSE bit of TDES3 of first normal descriptor. If the TSE bit is set in TDES3 for a non TCP/IP packet, the DMA behavior is unpredictable.

The application must program the length of the TCP packet payload in TDES3[17:0] and the TCP header in TDES3[22:19]. The maximum length of TCP packet payload that can be segmented is 256 KB.

Note The TDES3[22:19] field is used as slot number in case of AV configurations. If both TSO and AV are enabled for the same DMA Tx channel (TSE bit is set in DMA_Tx_Control_Register), then you must not enable slot checking for that DMA Tx Channel (do not set ESC bit in DMA_CH[n]_Slot_Function_Control_Status).

The header of the packet including Ethernet header, L3 header and L4 header should be provided in Buffer1 of the first normal descriptor of the TSO packet. Only buffer 1 of the first normal descriptor of a packet enabled for TSO is taken as the buffer containing the header.

The TCP payload can begin from buffer 2 of the first normal descriptor and continue to buffer1 and buffer 2 of second normal descriptor and subsequent descriptors.

The TCP payload may span across multiple buffers and multiple descriptors. The size of buffers containing the TCP payload should add up to be equal to the TCP payload length provided in TDES3[17:0] of the first normal descriptor.

The MAC always calculates and appends CRC and inserts Padding (if required) for all packets segmented by the DMA. If the TSE bit of TDES3 is enabled, the CRC PAD Control (CPC) field of TDES3 is reserved. To determine the size of a TCP packet after segmentation, the DMA uses the Maximum Segment Size (MSS) provided by the application through context descriptor. The DMA segments only those packets which have payload size greater than MSS. The application must provide the MSS by either programming the MSS value in the DMA_CH#_Control register or by providing a context descriptor. The DMA uses the last programmed value of MSS or the last MSS value provided through context (whichever is provided later).

The header length plus the MSS size (which is equal to the size of each TCP segment) should not exceed 16383 bytes otherwise the MAC transmitter truncates the packet after 16383 bytes causing a CRC error.

The header length plus MSS size plus programmed PBL value in DMA_CH#_Tx_Control register should be lesser than the Tx Queue size programmed in TQS field of MTL_TxQ#_Operation_mode register. Synopsys recommends a MSS+header equal to half the programmed Tx Queue size.

The DMA also supports segmentation of VLAN Tagged TCP/IP frames, so if the TCP packet has a VLAN Tag, then the same tag is used for all the segments irrespective of the VLAN tag type provided (C-VLAN or S-VLAN). The VLAN Tag insert/replace control bits are used for all segments.

If the Double VLAN Feature is selected, then the DMA passes both the tags for all segments irrespective of the VLAN tag types provided (C-VLAN or S-VLAN). The VLAN Tag Insert/Replace control bits for both tags is applicable for all segments.

If Double VLAN feature is not selected, then the application must not set the TSE bit in TDES3 for a TCP/IP packet with two tags. The DMA behavior in this scenario is unpredictable.

If the TSE bit is set in TDES3 for the packet and TCP header length provided is less than 5 (meaning an invalid TCP header because it is less than 20 bytes), the DMA does not perform segmentation, instead it transmits the entire packet as a single packet. In this scenario, the CRC pad control bits are forced by DMA to 2'b00 (MAC does CRC and padding) and checksum insertion control bits are forced to 2'b11 (Hardware does the checksum calculation for both header and payload).

9.4.4 Enabling the TSO Engine

1. Select the **Enable TCP Segmentation Offloading for TCP/IP Packets** option under the TCP/IP Offloading Features section during the **Specify Configuration** activity in coreConsultant.
2. Select the required number in the **Number of Tx DMA Channels Enabled for TSO** option.

   For details about these options, see TCP/IP Offloading Features Parameters in "Parameter Descriptions" on page 419.

Note See the "Enabled" field of the parameter descriptions to understand the dependencies.

3. Perform the relevant steps of "Programming Guidelines for TSO" on page 1376.

Note
- The TCP Segmentation Offload (TSO) feature is not supported for TxPBL values programmed to less than 8 in the DMA_CH#_Tx_Control register.
- The TSO feature is supported only when the MAC is operating in Full Duplex mode.

9.4.5 Programming Guidelines for TSO

See "Programming Guidelines for TSO" on page 1376.

9.4.6 External TSO Memory Interface

You can add memory interface for storing the TSO headers into a memory attached to the DWC_ether_qos and save application bus cycles. The TSO Memory Interface (TMI) is used to read and write the header into an external single port RAM (SPRAM) interface.

When the TSO memory is not enabled, the DMA reads the header buffer from the system memory for each segmented packet. To reduce the buffer reads, you can use the TSO memory in which the DMA can store the header data. The DMA reads the header buffer from the system memory for the first segment and writes it into the TSO memory. For the subsequent segments, the DMA reads the header from the TSO memory instead of fetching the data again from the system memory. If the header does not fit into the TSO memory, the DMA fetches the header from system memory for all segments.

When you enable the TMI, the read and write data width is same as the configured data width and the memory is synchronous to the application clock (clk_app_i). Minimum 64 bytes external memory is required for the TMI. The maximum size is 1K bytes. You can increase the memory size by using the Per-Channel TSO Memory Size in Bytes option. The memory address width is given by the following:

```
M = log2 ((DWC_EQOS_TSO_MEM_SIZE * DWC_EQOS_NUM_DMA_TSO_CH) / DWC_EQOS_DATAWIDTH)
```

For more details, see "TCP/IP Header Buffer" on page 161.

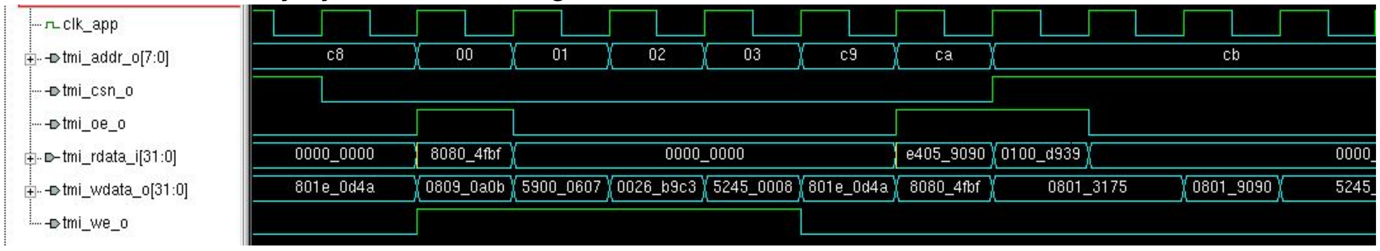
9.4.6.1 External TSO Memory Interface Signals

For a description of signals related to the External TSO Memory Interface, See TSO Memory Interface (TMI) Signals in the "Signal Descriptions" on page 471.

9.4.6.2 TSO Memory Synchronous Timing

Figure 9-6 shows the synchronous timing for the TSO memory.

Figure 9-6 TSO Memory Synchronous Timing



9.4.6.3 Enabling the External TSO Memory Interface

To enable this feature while configuring the controller in the coreConsultant GUI:

1. Select the **Enable Separate Memory for TCP/IP Headers** option under the **TCP/IP Offloading Features** section during the **Specify Configuration** activity in coreConsultant.
2. Select a size in the **Per-Channel TSO Memory Size in Bytes** option.

For details about this option, see TCP/IP Offloading Features Parameters in the "Parameter Descriptions" on page 419.

Note See the "Enabled" field of the parameter descriptions to understand the dependencies.

9.5 UDP/IPv4 Fragment Offload (UFO) Engine

DWC_ether_qos supports breaking a large UDP packet into smaller and multiple packets for transmission, using IPv4 Fragmentation. Each IPv4 fragment contains the following header or payload information:

- Ethernet Header
- IP Header
- UDP Header (only in the first fragment)
- UDP Payload

Note

- Fragmentation is supported only for UDP over IP. Fragmentation is not supported for TCP packets.
- Fragmentation of UDP is supported only over IPv4. This support is not available over IPv6.

The UDP Fragment Offload (UFO) Engine is an add-on function of the TSO engine. Therefore, both UFO and TSO functions can be enabled on the same Transmit DMA channel. In this mode, the UDP packets are fragmented and TCP packets are segmented. If the UFO function is not enabled, then both TCP packets and UDP packets are segmented into smaller packets as described in "TCP/IP Segmentation Offload (TSO) Engine" on page 350.

9.5.1 Description of UFO Engine

The UFO breaks down a large UDP packet into smaller packets using IPv4 fragmentation. In this mode, the MSS field in the context Descriptor (TDES2[17:0]) and DMA_CH(#)_Control register, are interpreted as Maximum Fragmentation Size (MFS). This is the maximum size of the payload after the IPv4 header in each fragment. When UFO is enabled, the MSS field must be programmed in multiples of 8 bytes, because, the IP fragments offset is defined in terms of 8 bytes.

UFO functions in two modes:

- UFO with valid UDP checksum
- UFO with no UDP checksum

9.5.1.1 UFO with valid UDP checksum

In this mode, the UFO engine calculates the UDP payload checksum and inserts it in the respective UDP Header field. However, as the checksum must be calculated over the complete payload ending in the last fragment and then inserted in the UDP Header encapsulated in the first IPv4 fragment, the fragments are not transmitted in order. The first IPv4 fragment containing the UDP Header is sent last. Figure 9-7 shows this sequence.

Figure 9-7 Fragmentation Flow With Checksum

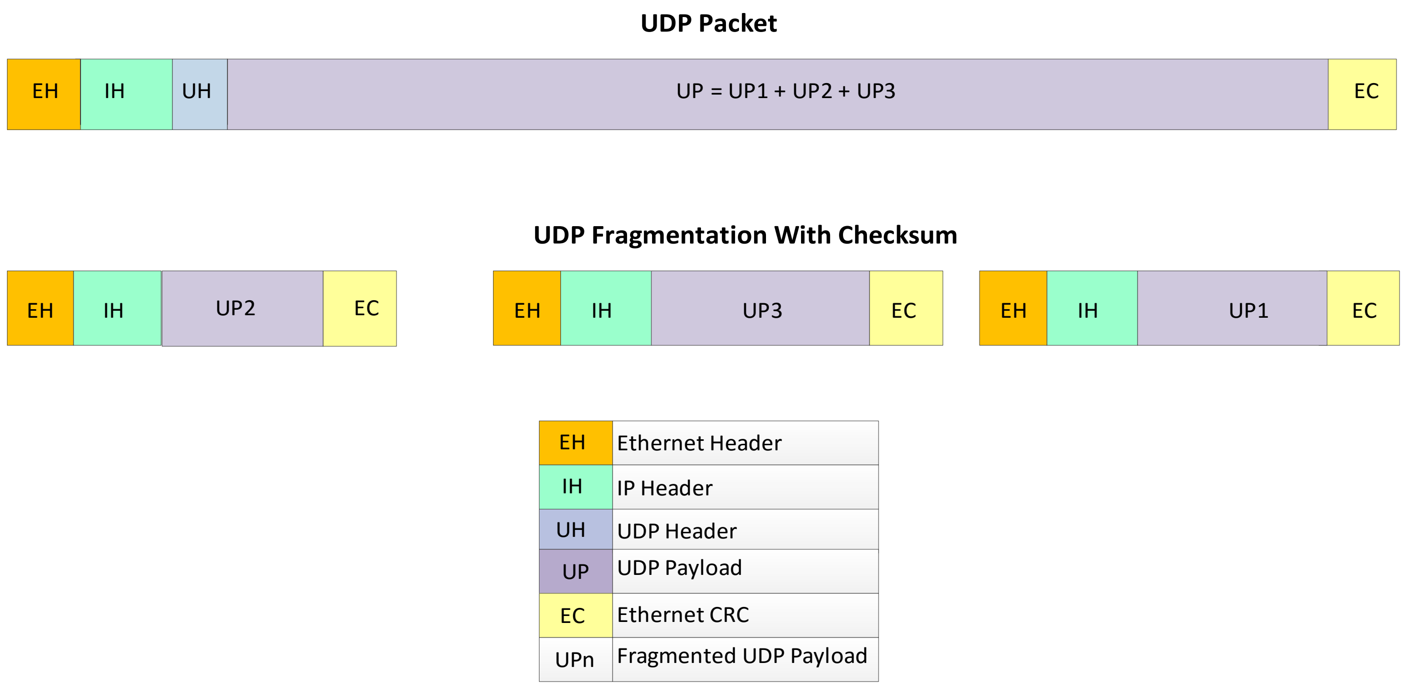


Figure 9-5 shows the details of the IP/UDP Header updates performed in each fragment by the UFO Engine in this mode.

Table 9-5 Details of IPv4 Fragmentation with UDP Checksum

Packet Sequence	UDP Header	IPv4 Header	Comments
First Fragment (This segment is sent last)	Calculate the UDP header checksum.	1. Total Length = 32 Bytes + IPv4 Header Length 2. DF = 0 (Do not Fragment) 3. MF = 1 4. Fragment offset = 0 5. Update the IPv4 header checksum	The header (and 32 Bytes IPv4 fragment payload) data is stored in TSO memory in this step.
Second fragment	NA (No UDP header)	1. Total Length = MFS + IPv4 Header Length 2. DF = 0 (Do not Fragment) 3. MF = 1 4. Fragment offset = Previous fragment offset + (32/8) 5. Update the IPv4 header checksum	The header data is read from the TSO memory. Only the Ethernet + IPv4 header is read.

Packet Sequence	UDP Header	IPv4 Header	Comments
Intermediate (non-first and non-last) fragment	NA (No UDP header)	1. Total Length = MFS + IPv4 Header Length 2. DF = 0 (Do not Fragment) 3. MF = 1 4. Fragment offset = Previous fragment offset + (MFS/8) 5. Update the IPv4 header checksum	The header data is read from the TSO memory. Only the Ethernet + IPv4 header is read.
Last fragment	NA (No UDP header)	1. Total Length = Remaining payload + IPv4 Header Length 2. DF = 0 (Do not Fragment) 3. MF = 0 4. Fragment offset = Previous fragment offset + (MFS/8) 5. Update the IPv4 header checksum	The header data is read from the TSO memory. Only the Ethernet + IPv4 header is read.

9.5.1.2 UFO with no UDP checksum

In this mode, the UDP payload checksum is not calculated and an all-zero value is inserted in the corresponding Checksum field of UDP Header. Hence all the IP Fragments packets are transmitted in sequence. Fig 9-7 shows the sequence of Fragmented packets created and transmitted by the UFO Engine.

Figure 9-8 Fragmentation Flow Without UDP Checksum

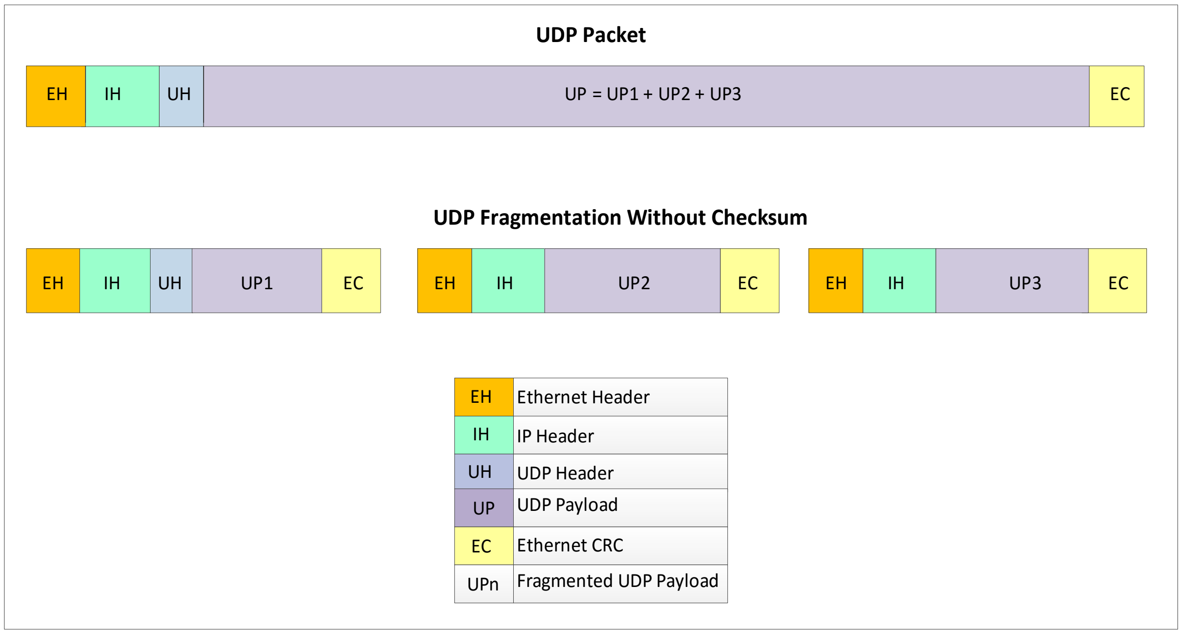


Table 9-6 shows the details of IPv4/UDP header updates performed in each fragment b the UFO engine, in this mode.

Table 9-6 Details of IPv4 Fragmentation without UDP Checksum

Fragment Sequence	UDP Header	IPv4 Header	Comments
First Fragment	Replace the UDP checksum field with all 0s.	1. Total Length = MFS (MSS field) + IPv4 Header Length 2. DF = 0 (Do not Fragment) 3. MF = 1 4. Fragment offset = 0 5. Update the IPv4 header checksum	The header data is stored in the TSO memory in this step.
Intermediate (non-first and non-last fragment)	NA (No UDP header)	1. Total Length = MFS (MSS field) + IPv4 Header Length 2. DF = 0 (Do not Fragment) 3. MF = 1 4. Fragment offset = Previous fragment offset + (MSS/8) 5. Update the IPv4 header checksum	The header data is read from TSO memory. Only Ethernet + IPv4 header is read.
Last fragment	NA (No UDP header)	1. Total Length = Remaining Payload + IPv4 Header Length 2. DF = 0 (Do not Fragment) 3. MF = 0 4. New fragment offset = Previous fragment offset + (MSS/8) 5. Update the IPv4 header checksum	The header data is read from TSO memory. Only Ethernet + IPv4 header is read.

9.5.2 Segmentation Versus Fragmentation

Table 9-7 shows the differences between UDP packet fragmentation performed by UFO and UDP segmentation performed by TSO engine.

Table 9-7 Segmentation Versus Fragmentation

Segmentation	Fragmentation
The UDP header is replicated for each segment. The length field in the UDP header is updated for each segment	The UDP header is not updated; however, checksum field in the UDP header is updated.
MSS field indicates the size of the maximum segment after the L4(TCP/UDP) header.	MFS field indicates the size of the fragment after the L3 (IPv4) header
The TCP packet is broken down into multiple chunks by keeping L2+L3+L4 header	The UDP packet is broken down into multiple chunks by keeping the L2 + L3 header

9.5.3 Enabling the UFO Engine

To enable the UFO engine,

1. Select all of the following under the TCP/IP Offloading Features section during the Specify Configuration activity in the coreConsultant GUI:
   - Enable TCP Segmentation Offloading for TCP/IP Packets

- Enable Separate Memory for TCP/IP Headers
- Per-Channel TSO Memory Size in Bytes should be set to a value 128 or more.

For details about these options, see TCP/IP Offloading Features Parameters in "Parameter Descriptions" on page 419.

2. Perform the relevant steps in the "Programming Guidelines for UFO" on page 1377.

9.5.4 Registers for the UFO Engine

The UFO functions are controlled by the following registers:

- DMA_CH(#)_TX_Control register, TSE_MODE field
- DMA_CH(#)_Control Register and Context Descriptor TDES2, MSS Field

9.5.5 Programming Guidelines for the UFO Engine

For details about the programming guidelines for the UFO engine, see "Programming Guidelines for UFO" on page 1377

9.6 Using the IPv4 ARP Offload Engine

9.6.1 Introduction to IPv4 ARP Offload Engine

DWC_ether_qos supports the Address Recognition Protocol (ARP) Offload for IPv4 packets. This feature allows the processing of the IPv4 ARP request packet in the receive path and generating corresponding ARP response packet in the transmit path. DWC_ether_qos generates the ARP reply packets for appropriate ARP request packets. The ARP packet for IPv4 is L2 layer packet with Length/Type of 0x0806

9.6.2 Description of IPv4 ARP Offload Engine

The ARP offloading process is as follows:

1. The MAC receiver gets an ARP request if the Target Protocol Address of request matches the IPv4 address programmed in the L3 register of the MAC.
2. The MAC generates an ARP reply packet.
3. The MAC copies the Sender Hardware Address field in the ARP request to the following fields:
   - DA field of the Ethernet packet header
   - Target Hardware Address field of the ARP reply packet
4. The MAC copies the Sender Protocol Address field in the ARP request to the Target Protocol Address field in the ARP reply packet.
5. The MAC places its MAC address in the following fields:
   - SA field of the Ethernet packet header
   - Sender Hardware Address field of the ARP reply packet
6. The MAC copies the Target Protocol Address field in the ARP request to the Sender Protocol Address field in the ARP reply packet.
7. The MAC sets the opcode field in ARP reply packet to 2 indicating ARP reply.
8. The MAC recalculates the CRC and performs padding for generated ARP reply packet.
9. The MAC transmitter sends the ARP reply.

The MAC processes only one ARP request at a time. It does not store the fields of multiple ARP requests. If the MAC receives an ARP request when it is already processing an earlier ARP request, the MAC does not generate the ARP reply for new ARP request. The MAC forwards the new ARP request packet to application with ARP Reply Not Generated (Bit 34) status bit set. However, in power-down mode, if the MAC receives an ARP request when it is already processing an earlier ARP request, the MAC drops the new ARP request.

If the Disable CRC check bit of the MAC Extension Configuration bit is set, then the MAC does not check for valid CRC of a ARP request Packet. It can generate an ARP response packet if the other conditions are valid.

If DWC_EQOS_PMT_EN is present, the ARP request Packet must always have a valid CRC.

Note When the received ARP request is less than 64 bytes packet length, DWC_ether_qos does not send a ARP response. It is treated as normal packet and forwarded to the application based on the DWC_ether_qos filter settings.

9.6.3 Enabling the IPv4 ARP Offload Engine

To enable this feature while configuring the controller in the coreConsultant GUI:

1. Select the **Enable IPv4 ARP Offload option** under the **TCP/IP Offloading Features** section during the **Specify Configuration** activity in coreConsultant.

   For details about this option, see TCP/IP Offloading Features Parameters, in "Parameter Descriptions" on page 419.

Note See the "Enabled" field of the parameter descriptions to understand the dependencies.

10 Power Management and Energy Efficient Ethernet

This chapter describes the power management and Energy Efficient Ethernet features supported by the DWC_ether_qos. It contains the following section:

10.1 Overview of Low-Power Management

The DWC_ether_qos supports the following techniques to save power.

- Magic Packet
- Remote Wakeup
- Energy Efficient Ethernet

The Magic Packet and Remove Wakeup techniques are used to save power in the host system when it is idle (Sleep mode) and has to be woken up only at the reception of specific packets from the Ethernet network. In the Sleep mode, the power to the host logic along with majority of the DWC_ether_qos (except the MAC receiver logic), can be shut down. On receiving the specific packets from the network, the MAC provides the trigger to restore the power to the host system and come back to normal state.

The Energy Efficient Ethernet (EEE) mode is compliant with the IEEE 802.3az-2010 standard. It is primarily targeted to save power in the Ethernet port when there is no traffic on the line. In this mode, the host indicates to the far-end that it does not have any packets to transmit for near future and puts the transmitter port (MAC Controller, PCS and PHY layers) into low-power mode. Similarly, the Receiver port can also be put into low-power mode when the far-end host indicates that it does not have any traffic to transfer. This allows significant saving of power in the Ethernet port (mainly in the PHY) with intermittent and bursty traffic profile. The triggering of entry and exit out of the EEE mode is controlled by the MAC and is supported within the DWC_ether_qos.

Simultaneous operation of the EEE mode along with any or both the other power saving modes is also supported in DWC_ether_qos.

10.2 Description of Magic Packet Mode

This section describes the power saving through magic packet detection.

Note

- The magic packet feature is based on the Magic Packet technology white paper from Advanced Micro Device (AMD).
- The watchdog timeout limit for a magic packet is 2,048 bytes irrespective of the value programmed in WD bit of MAC_Configuration register and PWE bit in MAC_Watchdog_Timeout register.
- The value programmed in DCRCC bit of MAC_Ext_Configuration register is applicable to a magic packet only when Enable IPv4 ARP Offload is selected in the configuration.

In the Magic Packet based power saving is a mode, the reception of valid magic packet by MAC receiver triggers an exit from low-power mode. The MAC enters power saving mode when PWRDWN bit of MAC_PMT_Control_Status register is programmed to 1. Exit from the magic packet based low-power mode is enabled by setting the MGKPKTEN bit of MAC_PMT_Control_Status register to 1.

The magic packet contains a unique pattern at any offset after the Destination address, Source address, and Length/Type fields. In addition to the unique pattern matching, the MAC receiver also checks for the following, to detect the received packet as a valid magic packet:

- The packet must be addressed to it (Destination Address of the received packet should perfect match the MAC_Address0_High and MAC_Address0_Low registers) or with multicast/broadcast address
- The packet must not have length error, FCS error, dribble bit error, GMII error, and collision
- The packet must not be runt (length including Ethernet header and FCS is at least 64 bytes)

10.2.1 Magic Packet Data Format

The content of the unique pattern in magic packet is described as

- 6 bytes of all-ones (48'hFF_FF_FF_FF_FF_FF) called the synchronization stream. There can be more than six bytes of 8'hFF, but last 6 are considered.
- The synchronization stream is immediately followed by 16 repetitions of Destination address field of the packet (MAC Address (MAC_Address0_High and MAC_Address0_Low registers) or multicast/broadcast address)
- No break or interruption between synchronization stream and first repetition of Destination address field or within its 16 repetitions

If the MAC address of a node is 48'h00_11_22_33_44_55, the MAC scans for the following data sequence:

```
Destination Address Source Address Length/Type.................. FF FF FF FF FF FF00 11 22
33 44 55 00 11 22 33 44 55 00 11 22 33 44 55 00 11 22 33 44 5500 11 22 33 44 55
00 11 22 33 44 55 00 11 22 33 44 55 00 11 22 33 44 5500 11 22 33 44 55 00 11 22
33 44 55 00 11 22 33 44 55 00 11 22 33 44 5500 11 22 33 44 55 00 11 22 33 44 55
00 11 22 33 44 55 00 11 22 33 44 55...CRC
```

10.3 Description of Remote Wakeup Packet Mode

This section describes the Remote Wakeup Packet based power saving mode

Note

- The remote wake-up packet feature is implemented based on the *Device Class Power Management Reference Specification* and various implementation-specific white papers.
- The watchdog timeout limit for a magic packet is 2,048 bytes irrespective of the value programmed in WD bit of MAC_Configuration register and PWE bit in MAC_Watchdog_Timeout register.
- The value programmed in DCRCC bit of MAC_Ext_Configuration register is applicable to a magic packet only when Enable IPv4 ARP Offload is selected in the configuration.

In the Remote Wakeup Magic Packet based power saving mode, the reception of expected remote wakeup packet by MAC receiver triggers the exit from low-power mode. The MAC enters power saving mode when PWRDWN bit of MAC_PMT_Control_Status register is programmed to 1. Exit from the remote wakeup based low-power mode is enabled by programming RWKPKTEN bit of MAC_PMT_Control_Status register to 1.

The MAC implements a filter lookup table (programmed through MAC_RWK_Packet_Filter register) in which CRC, offset, and byte mask of the pattern embedded in remote wakeup packet and the filter operation commands are programmed.

The pattern embedded in the remote wakeup packet is located at any offset after the Destination address and Source address fields. In addition to the CRC match for the pattern, the MAC receiver also checks the following, to detect the received packet as a valid remote wakeup packet:

- The packet must be addressed to it (Destination Address of the received packet should perfect match the MAC_Address0_High and MAC_Address0_Low registers) or with multicast/broadcast address
- The packet must not have length error, FCS error, dribble bit error, GMII error, and collision
- The packet must not be runt (length including Ethernet header and FCS is at least 64 bytes)

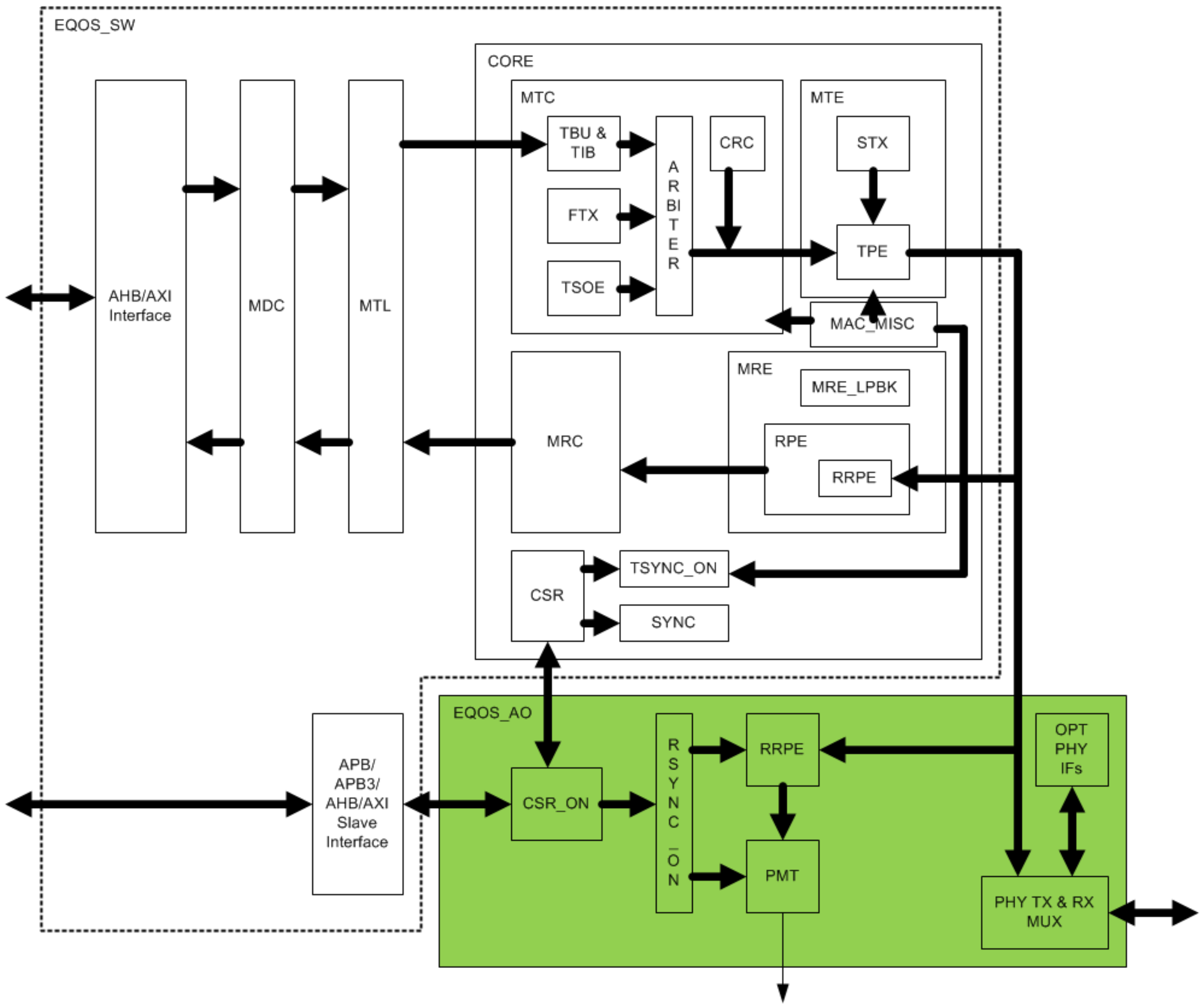
When a valid remote wakeup packet is received, the MAC receiver sets the RWKPRCVD bit in MAC_PMT_Control_Status register and triggers the interrupt on pmt_intr_o output port. The PMTIS bit in MAC_Interrupt_Status register is set when power-gating is not enabled in low-power mode. An interrupt is triggered to the application (on the mci_intr_o output port in EQOS-CORE and EQOS-MTL configurations or sbd_intr_o output port in DMA configurations) when interrupt is enabled (PMTIE bit in MAC_Interrupt_Enable register is set) and CSR clock is not gated off in low-power mode.

10.4 Design Partitioning for Low-Power Support

The design is physically partitioned into power switchable and power always-on domains to support power gating (compliant with Unified Power Format, UPF Standard) in Magic Packet and Remote Wakeup Packet based power saving modes.

Figure 10-9 shows the block diagram of Design hierarchy without DWC_EQOS_ARP_EN feature.

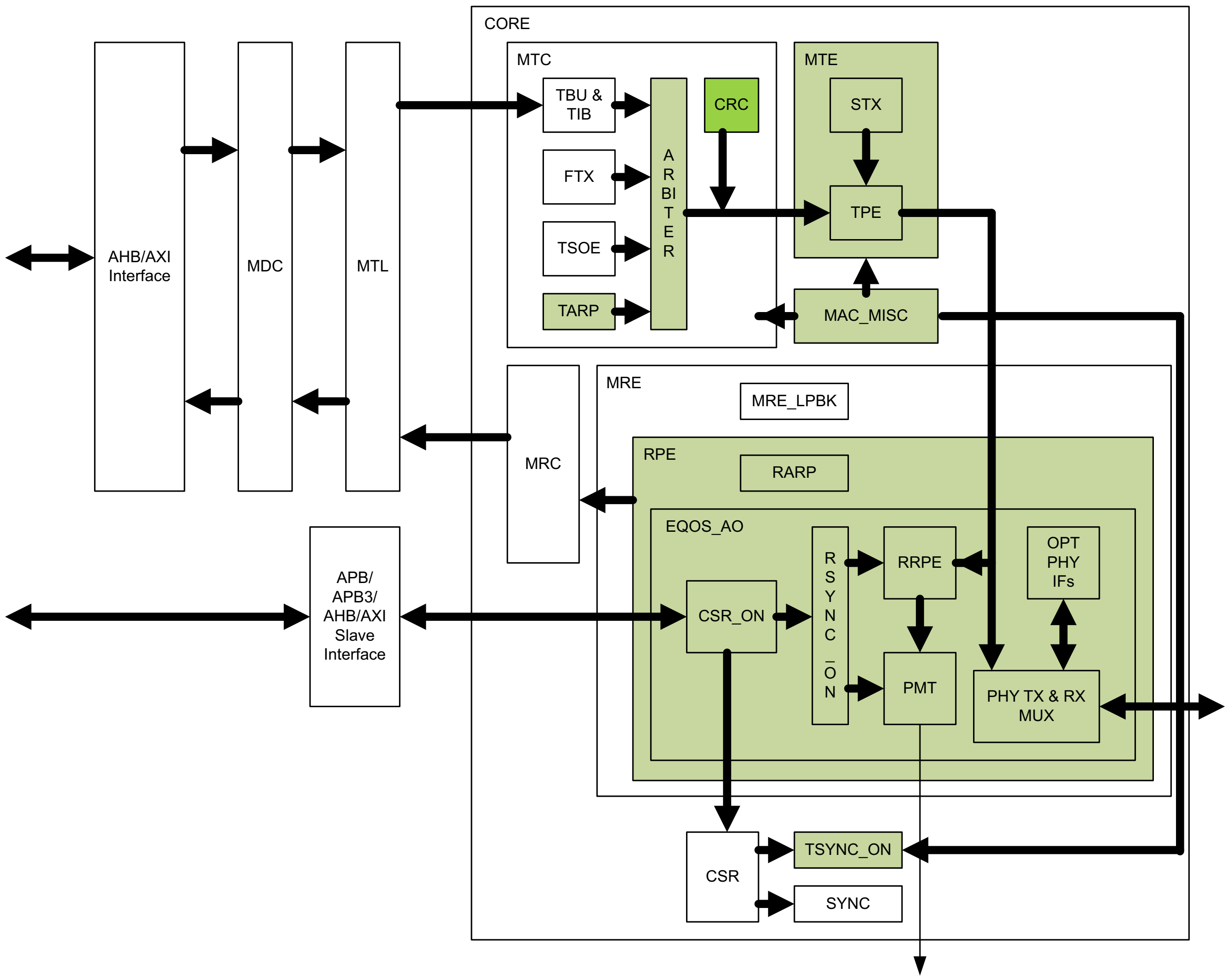
Figure 10-9 UPF Flow Block Diagram



When DWC_EQOS_ARP_EN feature is selected in the configuration, the design is logically partitioned (specified using power intent format compliant to Unified Power Format, UPF Standard). Some of the sub-blocks from the physically partitioned power switchable domain are associated with power always-on domain.

Figure 10-10 shows the block diagram of Design hierarchy with DWC_EQOS_ARP_EN feature.

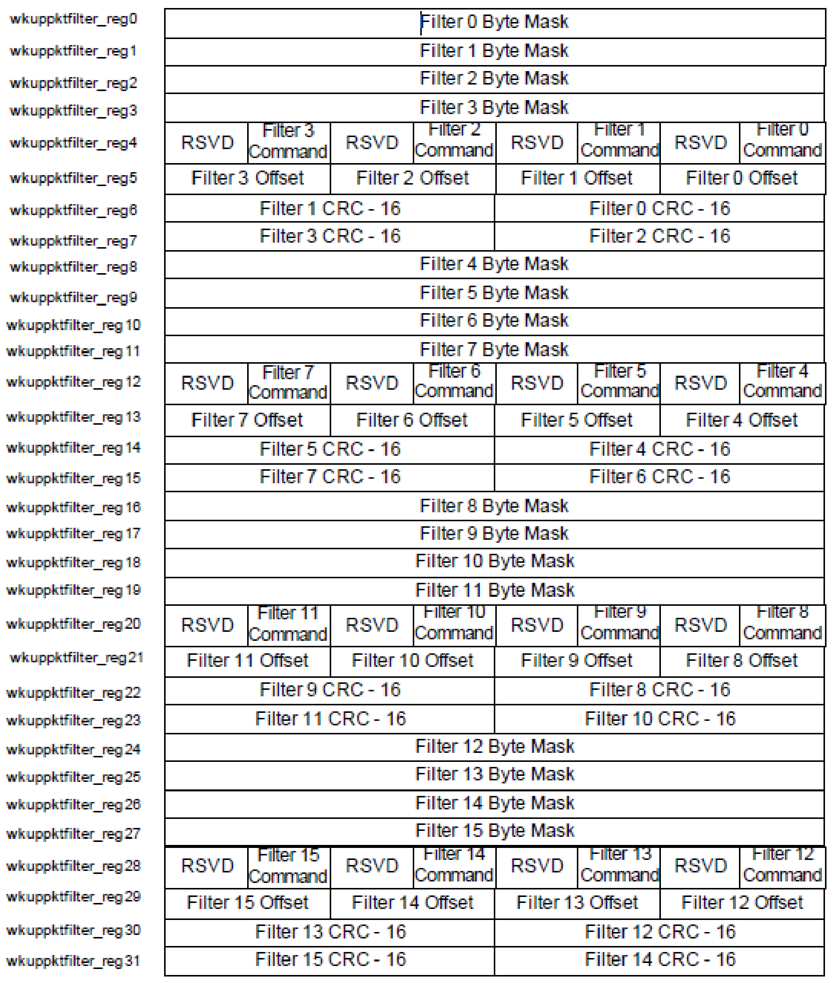
Figure 10-10 UPF Flow Block Diagram with ARP Feature



10.5 Remote Wake-Up Packet Filters

When Remote Wakeup based power saving mode is selected, it allows selection of 4, 8 or 16 Remote Wakeup Filters. The Remote Wakeup Filters structure is shown in Figure 10-11.

Figure 10-11 Remote Wake-Up Packet Filter Register



wkuppktfilter_reg0	Filter 0 Byte Mask							
wkuppktfilter_reg1	Filter 1 Byte Mask							
wkuppktfilter_reg2	Filter 2 Byte Mask							
wkuppktfilter_reg3	Filter 3 Byte Mask							
wkuppktfilter_reg4	RSVD	Filter 3 Command	RSVD	Filter 2 Command	RSVD	Filter 1 Command	RSVD	Filter 0 Command
wkuppktfilter_reg5	Filter 3 Offset		Filter 2 Offset		Filter 1 Offset		Filter 0 Offset	
wkuppktfilter_reg6	Filter 1 CRC - 16				Filter 0 CRC - 16			
wkuppktfilter_reg7	Filter 3 CRC - 16				Filter 2 CRC - 16			
wkuppktfilter_reg8	Filter 4 Byte Mask							
wkuppktfilter_reg9	Filter 5 Byte Mask							
wkuppktfilter_reg10	Filter 6 Byte Mask							
wkuppktfilter_reg11	Filter 7 Byte Mask							
wkuppktfilter_reg12	RSVD	Filter 7 Command	RSVD	Filter 6 Command	RSVD	Filter 5 Command	RSVD	Filter 4 Command
wkuppktfilter_reg13	Filter 7 Offset		Filter 6 Offset		Filter 5 Offset		Filter 4 Offset	
wkuppktfilter_reg14	Filter 5 CRC - 16				Filter 4 CRC - 16			
wkuppktfilter_reg15	Filter 7 CRC - 16				Filter 6 CRC - 16			
wkuppktfilter_reg16	Filter 8 Byte Mask							
wkuppktfilter_reg17	Filter 9 Byte Mask							
wkuppktfilter_reg18	Filter 10 Byte Mask							
wkuppktfilter_reg19	Filter 11 Byte Mask							
wkuppktfilter_reg20	RSVD	Filter 11 Command	RSVD	Filter 10 Command	RSVD	Filter 9 Command	RSVD	Filter 8 Command
wkuppktfilter_reg21	Filter 11 Offset		Filter 10 Offset		Filter 9 Offset		Filter 8 Offset	
wkuppktfilter_reg22	Filter 9 CRC - 16				Filter 8 CRC - 16			
wkuppktfilter_reg23	Filter 11 CRC - 16				Filter 10 CRC - 16			
wkuppktfilter_reg24	Filter 12 Byte Mask							
wkuppktfilter_reg25	Filter 13 Byte Mask							
wkuppktfilter_reg26	Filter 14 Byte Mask							
wkuppktfilter_reg27	Filter 15 Byte Mask							
wkuppktfilter_reg28	RSVD	Filter 15 Command	RSVD	Filter 14 Command	RSVD	Filter 13 Command	RSVD	Filter 12 Command
wkuppktfilter_reg29	Filter 15 Offset		Filter 14 Offset		Filter 13 Offset		Filter 12 Offset	
wkuppktfilter_reg30	Filter 13 CRC - 16				Filter 12 CRC - 16			
wkuppktfilter_reg31	Filter 15 CRC - 16				Filter 14 CRC - 16			

The Remote Wakeup Filters are arranged in blocks of 4 filters each and each such block have eight 32-bit wide registers, viz. wkuppktfilter_reg0-7, wkuppktfilter_reg8-15, wkuppktfilter_reg16-23 and wkuppktfilter_reg24-31. The fields of Remote Wakeup Filter are as follows:

Filter i Byte Mask

The filter i byte mask register defines the bytes of the packet that are examined by filter i (0, 1, 2, 3, .., 15) to determine whether or not a packet is a wake-up packet.

- The MSB (31st bit) must be zero.
- Bit j[30:0] is the byte mask.
- If Bit j (byte number) of the byte mask is set, the CRC block processes the Filter i Offset + j of the incoming packet; otherwise Filter i Offset + j is ignored

Filter i Command

The 4-bit filter i command controls the filter i operation.

- Bit 3 specifies the address type, defining the destination address type of the pattern. When the bit is set, the pattern applies to only multicast packets; when the bit is reset, the pattern applies only to unicast packet
- Bit 2 (Inverse Mode), when set, reverses the logic of the CRC16 hash function signal, to reject a packet with matching CRC-16 value. Bit 2, along with Bit 1, allows a MAC to reject a subset of remote wake-up packets by creating filter logic such as "Pattern 1 AND NOT Pattern 2".
- Bit 1 (And_Previous) implements the Boolean logic. When set, the result of the current entry is logically ANDed with the result of the previous filter. This AND logic allows a filter pattern longer than 32 bytes by splitting the mask among two, three, or four filters. This depends on the number of filters that have the And_Previous bit set. The details are as follows:
  - The And_Previous bit setting is applicable within a set of 4 filters.
  - Setting of And_Previous bit of filter that is not enabled has no effect, that is setting And_Previous bit of lowest number filter in the set of 4 filters has no effect. For example, setting of And_Previous bit of Filter 0 has no effect.
  - If And_Previous bit is set for filter to form AND chained filter, the AND chain breaks at the point any filter is not enabled. For example: If Filter 2 And_Previous bit is set (bit 1 in Filter 2 command is set) but Filter 1 is not enabled (bit 0 in Filter 1 command is reset), then only Filter 2 result is considered. If Filter 2 And_Previous bit is set (bit 1 in Filter 2 command is set), Filter 3 And_Previous bit is set (bit 1 in Filter 3 command is set), but Filter 1 is not enabled (bit 0 in Filter 1 command is reset), then only Filter 2 result ANDed with Filter 3 result is considered. If Filter 2 And_Previous bit is set (bit 1 in Filter 2 command is set), Filter 3 And_Previous bit is set (bit 1 in Filter 3 command is set), but Filter 2 is not enabled (bit 0 in Filter 2 command is reset), then since setting of Filter 2 And_Previous bit has no effect only Filter 1 result ORed with Filter 3 result is considered.
  - If filters chained by And_Previous bit setting have complementary programming, then a frame may never pass the AND chained filter. For example, if Filter 2 And_Previous bit is set (bit 1 in Filter 2 command is set), Filter 1 Address_Type bit is set (bit 3 in Filter 1 command is set) indicating multicast detection and Filter 2 Address_Type bit is reset (bit 3 in Filter 2 command is reset) indicating unicast detection or vice versa, a remote wakeup frame does not pass the AND chained filter as a remote wakeup frame cannot be of both unicast and multicast address type.

  Bit 0 is the enable for filter i. If Bit 0 is not set, filter i is disabled.

Filter i Offset

The filter i offset register defines the offset (within the packet) from which the filter i examines the packets.

- This 8-bit pattern-offset is the offset for the filter i first byte to be examined.
- The minimum allowed offset is 12, which refers to the 13th byte of the packet.
- The offset value 0 refers to the first byte of the packet.

Filter i CRC-16

The filter i CRC-16 register contains the CRC-16 value calculated from the pattern and the byte mask programmed in the Remote Wakeup filter register.

- The 16-bit CRC calculation uses the following polynomial:

  G(x) = x^16 + x^15 + x^2 + 1
- Each mask, used in the hash function calculation, is compared with a 16-bit value associated with that mask. Each filter has the following:

- 32-bit Mask: Each bit in this mask corresponds to one byte in the detected packet. If the bit is 1, the corresponding byte is taken into the CRC-16 calculation.
- 8-bit Offset Pointer: Specifies the byte to start the CRC-16 computation. The pointer and the mask are used together to locate the bytes to be used in the CRC-16 calculations.

The Remote Wakeup Filter registers are implemented as 8, 16, or 32 indirect access registers (wkuppktfilter_reg#i) based on whether 4, 8, or 16 Remote Wakeup Filters are selected in the configuration and accessed by application through MAC_RWK_Packet_Filter register. When the Remote Wakeup Filters are to be programmed, the entire set of wkuppktfilter_reg registers must be written. The wkuppktfilter_reg register is programmed by sequentially writing the eight, sixteen or thirty-two register values in MAC_RWK_Packet_Filter register for wkuppktfilter_reg0, wkuppktfilter_reg1, ..., wkuppktfilter_reg31 respectively. The wkuppktfilter_reg register is read in a similar way. The MAC updates the wkuppktfilter_reg register current pointer value in RWKPTR field of MAC_PMT_Control_Status register.

Note

- If the MAC_RWK_Packet_Filter register is accessed in byte or half-word mode, the internal counter to access the appropriate wkuppktfilter_reg is incremented when CPU accesses Lane 3 (or Lane 0 in big-endian mode).
- When MAC_RWK_Packet_Filter register is written, the content is transferred from CSR clock domain to PHY receive clock domain after the write operation, there should not be any further write to the MAC_RWK_Packet_Filter register until the first write is updated in PHY receive clock domain. Otherwise, the second write operation does not get updated to the PHY receive clock domain. Therefore, the delay between two writes to the MAC_RWK_Packet_Filter register should be at least 4 cycles of the PHY receive clock.

10.6 PMT Interrupt

The PMT interrupt signal is asserted when a valid remote wake-up packet is received. In addition to the mci_intr_o (sbd_intr_o in EQOS-DMA configurations), the pmt_intr_o (synchronous to Rx clock) signal is asserted. The pmt_intr_o signal, synchronous to the Rx clock domain, is provided so that you can stop the application clock when the MAC is in the power-down mode. As the pmt_intr_o signal is generated in the PHY Rx clock domain, it is not cleared immediately when the PMT Control and Status register is read. This is because the resultant clear signal has to cross to the PHY Rx clock domain, and then clear the interrupt source. This delay is at least 4 clock cycles of Rx clock and can be significant when the DWC_ether_qos is operating in the 10 Mbps mode. When the application clears the PWRDWN bit in Remote Wake-Up Packet Detection register, the MAC comes out of the power-down mode, but this event does not generate the PMT interrupt. For a description of signals related to PMT, see Power Management Interface Signals in "Signal Descriptions" on page 471.

10.7 Enabling PMT through Remote Wake-Up Packet Mode

To enable the remote wake-up packet detection feature:

1. From the coreConsultant GUI, to enable PMT block and UPF, select the **Enable Power Management** option under the **Low Power Management** section during the **Specify Configuration** activity in coreConsultant.
2. To enable Remote Wake-Up Packet Detection, select the **Enable Remote Wake-Up Packet Detection** option under the **Low Power Management** section during the **Specify Configuration** activity in coreConsultant.

   For details about these options, see Low Power Management Parameters in "Parameter Descriptions" on page 419.

Note See the "Enabled" field of the parameter descriptions to understand the dependencies.

When you enable the low-power mode in the PMT block, the MAC drops all received packets and does not forward any packet to the MTL Rx Queue or the application. The MAC comes out of the low-power mode when a remote wake-up packet is received and the corresponding detection is enabled.

10.8 Enabling PMT through Magic Packet Mode

To enable the remote magic packet mode feature:

1. From the coreConsultant GUI, to enable PMT block and UPF, select the **Enable Power Management** option under the **Low Power Management** section during the **Specify Configuration** activity in coreConsultant.
2. To enable **Magic Packet** mode, select the **Enable Magic Packet Detection** option under the **Low Power Management** section during the **Specify Configuration** activity in coreConsultant.

   For details about these options, see Low Power Management Parameters in "Parameter Descriptions" on page 419.

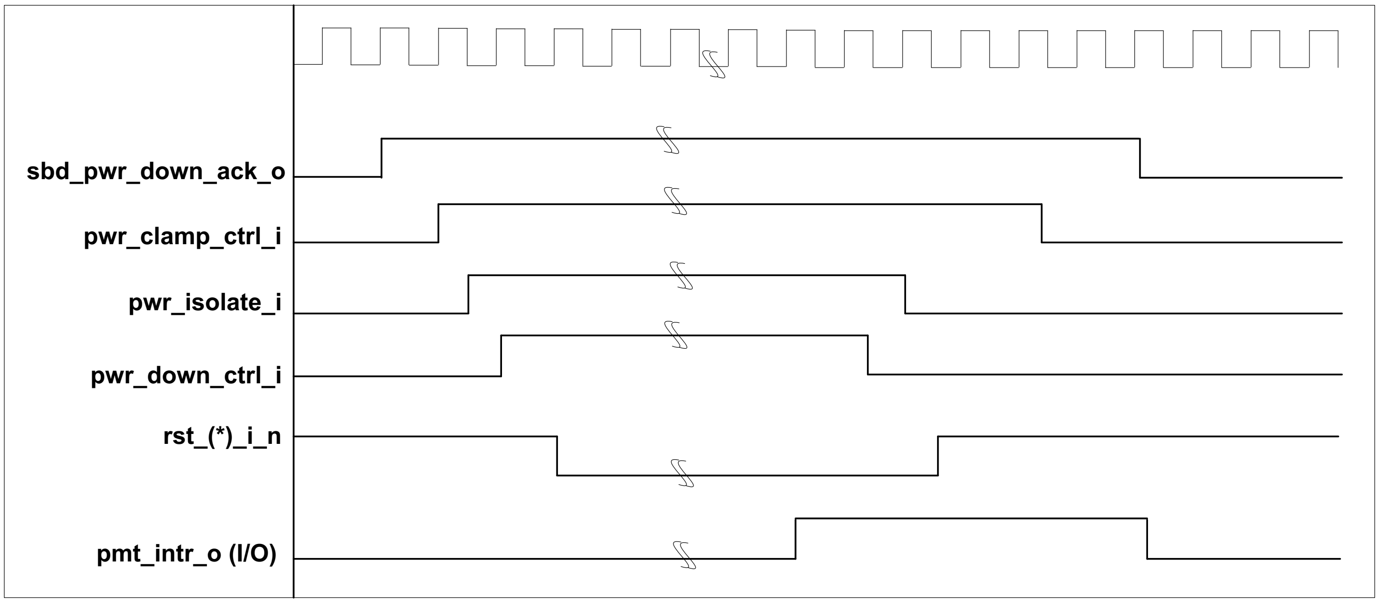
Note See the "Enabled" field of the parameter descriptions to understand the dependencies.

When you enable the low-power mode in the PMT block, the MAC drops all received packets and does not forward any packet to the MTL Rx Queue or the application. The MAC comes out of the low-power mode when a remote wake-up packet is received and the corresponding detection is enabled.

10.9 Power-Down and Power-Up Sequence

Figure 10-12 shows the power-down and power-up sequence.

Figure 10-12 Power-Down and Power-Up Sequence



10.9.1 Power-Down Sequence

The power-down sequence is described as follows:

1. The software performs the following tasks:
   a. Disable the Transmit DMA (if applicable) by clearing the ST bit of the DMA_CH0_Tx_Control register.
   b. Wait for any previous frame transmissions to complete. You can check this by reading Bits [18:16] of MAC_Debug register (Register 69) and Bit[4] of MTL_TxQ[n]_Debug register of all MTL Tx Queues.
   c. Disable the MAC transmitter and MAC receiver by clearing Bit 1 (TE) and Bit 0 (RE) in MAC_Configuration register.
   d. Wait till the Receive DMA empties all frames from the Rx FIFO (in EQOS-MTL and EQOS-DMA configurations).

      You can check this by reading Bits [29:16] of MTL_RxQ[n]_Debug Register of all Rx Queues. If these bits are zero, it indicates that the Rx FIFO is empty.
   e. Configure the magic packet (bit 2) and/or remote wake-up (bit 1) detection in the MAC_PMT_-Control_Status register (offset 0x00C0).
   f. If DWC_EQOS_ARP_EN is set to enable ARP Offload during Power Down, set Bit 31 (ARPEN) in the MAC Configuration Register.
   g. Enable the MAC Receiver by setting Bit 0 (RE), and then set Bit 0 (PWRDWN) in the PMT Control and Status register to initiate the power-down sequence in MAC.

   When only clock-gating is employed in low-power mode, after step 1.g above the sbd_pwr_down_ack_o signal can be used to gate-off the Transmit clock (only when ARP Offloading is not enabled in low-power mode), Application clock and CSR clock (when DWC_EQOS_CSR_SLV_CLK is selected in configuration).

   When power gating is employed in low-power mode, follow these steps to enter power-down mode:
2. The assertion of the sbd_pwr_down_ack_o signal indicates that the power always-on block is actively programmed and the power management controller can start switching off the power to power switchable block. The Power Management Controller should perform the following tasks:
   a. Assert the power clamp control (pwr_clamp_ctrl_i) to clamp the resets to the always-on blocks.
   b. Assert the power isolation control signal (pwr_isolate_i) to clamp the voltages of the isolation cells.
   c. Assert the pwr_down_ctrl_i signal to shutdown the power to the blocks in the power-OFFhierarchy.

Note

- If the Energy Efficient Ethernet feature is enabled and the MAC Transmitter is in the LPI mode when it is put into the power-down mode, then the GMII or MII interface gets clamped to assert the LPI pattern. If the MAC Transmitter is not in the LPI mode when it is put into the power-down mode, the GMII or MII interface gets clamped to all-zero.
- After the sbd_pwr_down_ack_o signal is asserted by MAC, the pwr_clamp_ctrl_i can be asserted after 1 CSR clock cycle. Wait for clamp circuit delay (which depends on the cell library) to assert pwr_isolate_i signal. Wait for isolation cell delay (which depends on the cell library) to assert pwr_down_ctrl_i signal.

10.9.2 Power-Up Sequence

The MAC wakes up on receiving the magic packet or remote wake-up frame. The power-up sequence is described as follows:

1. The MAC asserts pmt_intr_o.

   When only clock-gating is employed in low-power mode, after step 1 above the pmt_intr_o signal can be used to start the clocks that were gated-off after entering low-power mode.

   When power gating is employed in low-power mode, perform the following steps to exit:

2. The Power Management Controller performs the following tasks:
   a. De-assert the pwr_down_ctrl_i signal to enable power to the blocks in the power-OFF hierarchy.
   b. De-assert the pwr_isolate_i isolation cell control signal.
   c. Assert resets to ensure that all registers in the blocks that were shutdown are reset. To properly initialize all asynchronous logic, the reset should be active for at least 4-clocks duration of the slowest clock in DWC_ether_qos.
   d. De-assert the resets.
   e. De-assert the pwr_clamp_ctrl_i signal.

Note

- If the Energy Efficient Ethernet feature is enabled and the GMII or MII interface is clamped to the LPI pattern when the MAC is in the power-down mode, the GMII or MII transmitter starts driving all-zero (that is, the MAC transmitter comes out of the LPI mode) when the pwr_clamp_ctrl_i signal is de-asserted.
- In the EQOS-DMA, EQOS-AHB, and EQOS-AXI configurations, the application reset is synchronized to all clock domains. The application must wait for the de-assertion of resets (rst_clk_rx_n_o and rst_clk_app_n_o) before it de-asserts the pwr_clamp_ctrl_i signal. In the EQOS-CORE configuration the application must wait for the deassertion of reset input (rst_clk_rx_n) before it de-asserts the pwr_clamp_ctrl_i signal. In the EQOS-MTL configuration, the application must wait for deassertion of reset inputs (rst_clk_rx_n and rst_clk_app_n) before it de-asserts the pwr_clamp_ctrl_i signal.
- After getting the wakeup interrupt, pwr_down_ctrl_i signal can be de-asserted after 1 CSR clock cycle. Wait for stable power applied to de-assert pwr_isolate_i signal. Wait for isolation cell delay (which depends on the cell library) to de-assert the reset signal. Ensure at least 3 application clock cycles between de-assertion of pwr_down_ctrl_i and reset signals. After the reset signal is de-asserted, wait for 25 (~6 cycles for all resets to be asserted + ~7 cycles for reset stretching + ~6 cycles for all resets to be de-asserted; 25 to be on the safer side) clock cycles of slowest clock to de-assert the pwr_clamp_ctrl_i signal.

3. The software performs the following tasks:
   a. De-assert the pmt_intr_o by reading the PMT Control and Status register.
   b. Perform a write operation (with reset values) to the MAC_PMT_Control_Status and theMAC_RWK_Packet_Filter registers (if present) so that the corresponding values in the always-on block gets synchronized. Otherwise, the values of these registers are different.
   c. Perform write operations to the MAC_Configuration and MAC Address0 registers to synchronize the values in the CSR module and the respective bits in the always-on block. Otherwise, the MAC receiver will be ON even though the Receive Enable bit is set to 0.

After completing these steps, the software must initialize all registers, enable the transmitter, and program the DMA (in DMA configurations) to resume the normal operation.

10.10 Description of Energy Efficient Ethernet

EEE is an operational mode that enables the IEEE 802.3 Media Access Control (MAC) sub layer along with a family of physical layers to operate in the Low-Power Idle (LPI) mode. The EEE operational mode supports the IEEE 802.3 MAC operation at 100 Mbps, 1000 Mbps, and 10 Gbps. The DWC_ether_qos supports the IEEE 802.3az-2010 for EEE.

The LPI mode allows power saving by switching off the parts of the communication device functionality when there is no data to be transmitted and received. The systems on both sides of the link can disable some functionalities to save power during the periods of low-link utilization. The MAC controls whether the system should enter or exit the LPI mode and communicates this to the PHY.

The EEE specifies the capabilities negotiation methods that the link partners can use to determine whether EEE is supported, and then select the set of parameters that are common to both devices.

10.10.1 Transmit Path Functions

The transmit path functions include tasks that the MAC must perform to make the PHY to enter the LPI state.

In the transmit path, the software must set the LPIEN bit of the MAC_LPI_Control_Status register to indicate to the MAC to stop transmission and initiate the LPI protocol. The MAC completes the transmission in progress, generates its transmission status, and starts transmitting the LPI pattern instead of the IDLE pattern if the link status has been up continuously for a period specified in the LPI LS TIMER field of MAC_LPI_Control_Status register. The PHY Link Status bit of the LPI Control and Status Register indicates the link status of the PHY.

Note

- The EEE feature is not supported when the MAC is configured to use the TBI, RTBI, SMII, RMII, or SGMII single PHY interface. Even if the MAC supports multiple PHY interfaces, you should activate the EEE mode only when the MAC is operating with GMII, MII, or RGMII interface.
- According to the Energy Efficient Ethernet standard (IEEE 802.3az-2010), the PHY must not stop the TxCLK clock during the LPI state in the MII (10 or 100) mode. However, the MAC can stop the GTX_CLK clock during the LPI state in the GMII (1000) code.

To make the PHY enter the LPI state, the MAC performs the following tasks:

1. De-asserts TX_EN
2. Asserts TX_ER
3. Sets TXD[3:0] to 0x1 (for 100 Mbps) or TXD[7:0] to 0x01 (for 1,000 Mbps)

Note The MAC maintains the same state of the TX_EN, TX_ER, and TXD signals for the entire duration during which the PHY remains in the LPI state.

4. Updates the status (TLPIEN bit of MAC_LPI_Control_Status register) and generates an interrupt

To bring the PHY out of the LPI state, that is, when the software resets the LPIEN bit, the MAC performs the following tasks:

1. Stops transmitting the LPI pattern and starts transmitting the IDLE pattern
2. Starts the LPI TW TIMER

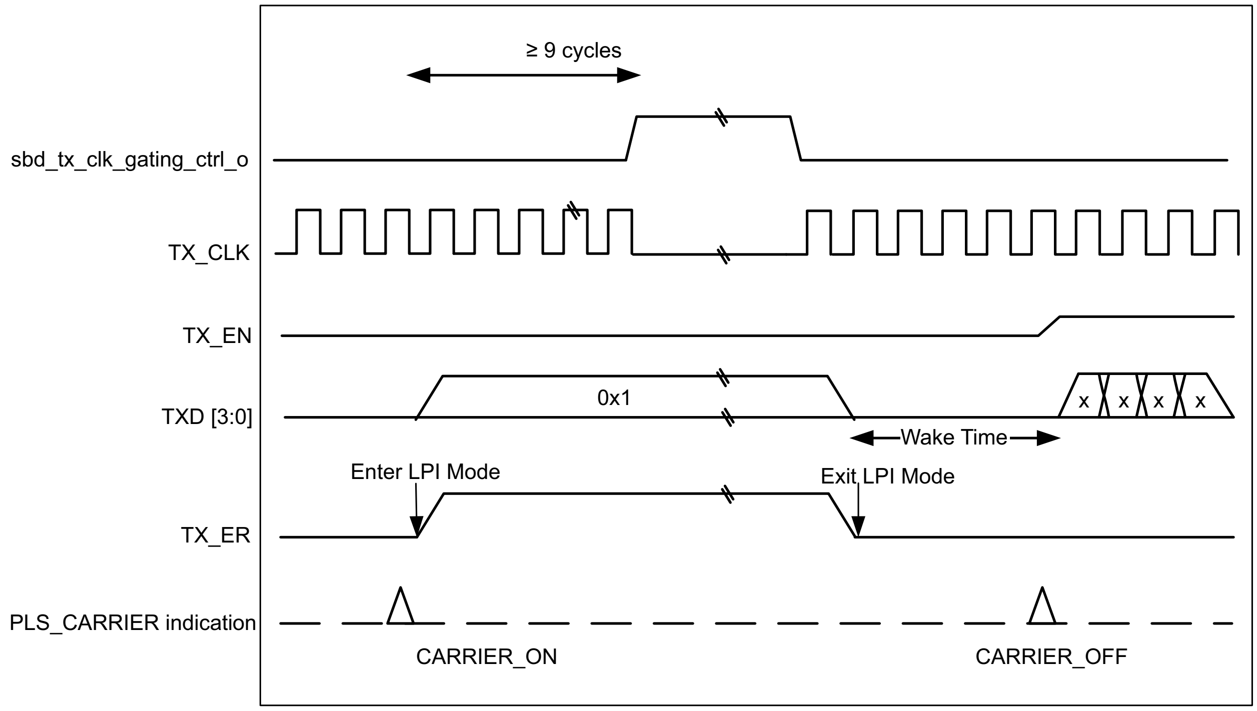
   The MAC cannot start the transmission until the wake-up time specified for the PHY expires. The auto-negotiated wake-up interval is programmed in the TWT field of the MAC_LPI_Timers_Control register.
3. Updates the LPI exit status (TLPIEX bit of the MAC_LPI_Control_Status register) and generates an interrupt

Figure 10-13 shows the behavior of TX_EN, TX_ER, and TXD[3:0] signals during the LPI mode transitions.

Note

- The MAC does not stop the TX_CLK clock. You can stop this clock (as shown in Figure 10-13) if your PHY supports it and when the MAC sets the sbd_tx_clk_gating_ctrl_o signal to 1. The sbd_tx_clk_gating_ctrl_o signal is asserted after nine Tx Clock Cycles, one Pulse Synchronizer delay (BCM22), and one CSR clock cycle. The assertion of the sbd_tx_clk_gating_ctrl_o signal is dependent on the LPITCSE bit of the MAC_LPI_Control_Status register.
- If RGMII Interface is selected, the Tx clock is required for transmitting the LPI pattern and so the Tx Clock cannot be gated.
- In the EQOS-CORE configuration, when the Tx clock gating is done during the LPI mode, you cannot use the LPITXA bit of the MAC_LPI_Control_Status register.
- If the MAC is in the Tx LPI mode and the Tx clock is stopped, the application should not write to CSR registers that are synchronized to Tx clock domain.
- If the MAC is in the LPI mode and the application issues a soft reset or hard reset, the MAC transmitter comes out of the LPI mode.

Figure 10-13 LPI Transitions (Transmit)



10.10.2 Automated Entry/Exit of LPI mode in Transmit Path

The MAC transmitter can be programmed to enter and exit LPI IDLE mode automatically based on whether it is IDLE for a specific period of time or has a packet to transfer. These modes are enabled and controlled by MAC_LPI_Control_Status register.

When LPITXA (Bit[19]) and LPITXEN (Bit[16]) of MAC_LPI_Control_Status register are set, the MAC transmitter enters LPI IDLE state when the MAC transmit path (including the MTL layers and DMA layers) are idle. The MAC transmitter will exit the LPI IDLE state and clear the LPITXEN bit as soon as any of functions in the TX path (DMA, MTL or MAC) becomes non-idle due to initiation of a packet transfer.

In addition to the above, when Bit[20] (LPIATE) is also set, the MAC transmitter will enter LPI IDLE state only if the Transmit path remains in idle state (no activity) for the time period indicated by the value in MAC_LPI_Entry_Timer. In this mode also, the MAC transmitter will exit the LPI IDLE state as soon as any of the functions becomes non-idle. However, the LPITXEN bit is not cleared but remains active so that reentry to LPI IDLE state is possible without any software intervention when the MAC becomes idle again.

When both LPIATE and LPITXA bits are cleared, you can directly control the entry and exit of LPI IDLE state by programming the LPITXEN bit.

10.10.3 Receive Path Functions

The receive path functions include the tasks that the PHY and MAC must perform when the PHY receives signals from the link partner to exit the LPI state.

In the receive path, when the PHY receives the signals from the link partner to enter into the LPI state, the PHY and MAC perform the following tasks:

1. The PHY asserts RX_ER
2. The PHY sets RXD[7:0] to 0x01
3. The PHY de-asserts RX_DV

Note The PHY maintains the same state of the RX_ER, RXD, and RX_DV signals for the entire duration during which it remains in the LPI state.

4. The MAC updates the RLPIEN bit of the MAC_LPI_Control_Status register and immediately generates an interrupt

Note

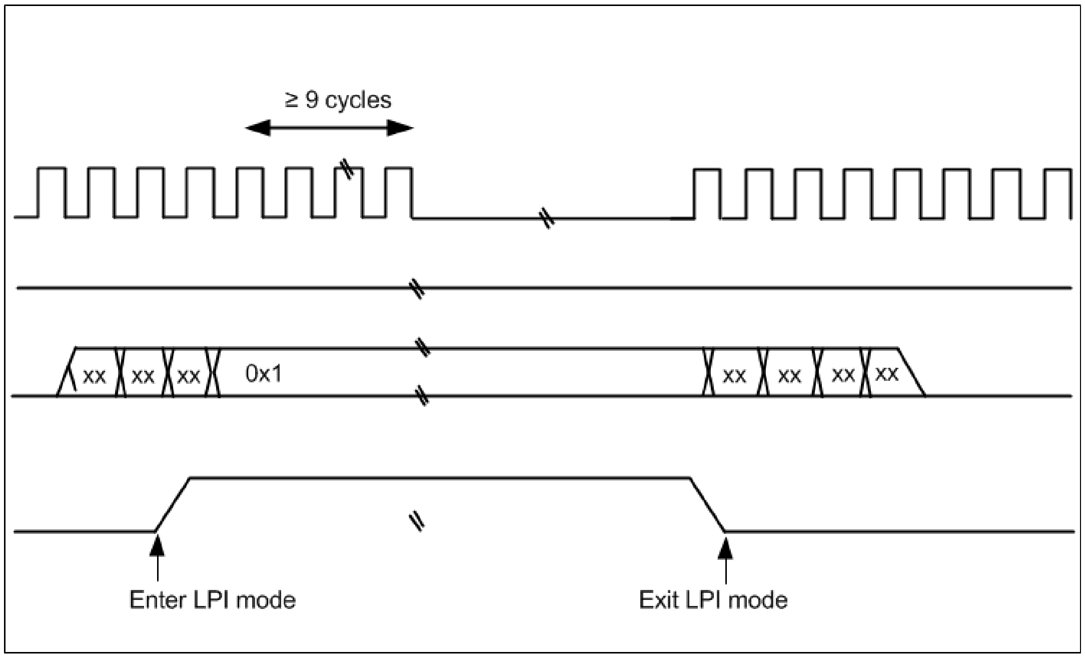
- If the LPI pattern is detected for a very short duration (that is, less than two cycles of Rx clock), the MAC does not enter the Rx LPI mode.
- If the duration between end of the current Rx LPI pattern and start of the next Rx LPI pattern, is very short (that is, less than two cycles of Rx clock), then the MAC exits and again enters the Rx LPI mode. The MAC does not give the Rx LPI Exit and Entry interrupts

When the PHY receives signals from the link partner to exit the LPI state, the PHY and MAC perform the following tasks:

1. The PHY de-asserts RX_ER and returns to a normal inter-packet state.
2. The MAC updates the RLPIEX bit of the MAC_LPI_Control_Status register and generates an interrupt immediately. The sideband signal lpi_intr_o (synchronous to Rx clock) is also asserted.

Figure 10-14 shows the behavior of RX_ER, RX_DV, and RXD[3:0] signals during the LPI mode transitions.

Figure 10-14 LPI Transitions (Receive)



Note

- If the RX_CLK_stoppable bit (in the PHY register written through MDIO) is asserted when the PHY is indicating LPI to the MAC, the PHY may halt the RX_CLK at any time more than nine clock cycles after the start of the LPI state as shown in Figure 10-14.
- If the MAC is in the LPI mode and the application issues a soft reset or hard reset, the MAC receiver comes out of the LPI mode during reset. If the LPI pattern is still received after the reset is de-asserted, the MAC receiver again enters the LPI state.
- If the RX clock is stopped in the RX LPI mode, the application should not write to the CSR registers that are being synchronized to the RX clock domain.
- When the PHY sends the LPI pattern, if EEE feature is enabled, the MAC automatically enters the LPI state. There is no software control to prevent the MAC from entering the LPI state.

10.10.4 EEE Registers

The following are the registers related to Energy Efficient Ethernet

- MAC_LPI_Control_Status
- MAC_LPI_Timers_Control
- MAC_LPI_Entry_Timer

For detailed description of these registers, see “Register Descriptions” on page 605.

10.10.5 EEE Signals

For a description of signals related to Energy Efficient Ethernet Interface Signals, see Energy Efficient Ethernet Interface Signals in “Signal Descriptions” on page 471.

10.10.6 LPI Timers

The transmitter maintains the LPI LS TIMER, LPI TW TIMER, and LPI AUTO ENTRY TIMER timers.

Following are the LPI timers that are loaded with the respective values from the MAC_LPI_Timers_Control and MAC_LPI_Entry_Timer registers:

- LPI LS TIMER

  The LPI LS TIMER counts, in milliseconds, the time expired since the link status is up. You can enable the monitoring of the LUD bit of the MAC_PHYIF_Control_Status register by setting the PLSEN bit of MAC_LPI_Control_Status register.

  The link status is indicated to the MAC by the LNKSTS bit of the MAC_PHYIF_Control_Status register or the value programmed by the software in the PLS bit of the MAC_LPI_Control_Status register. If the link status is not available in the MAC_PHYIF_Control_Status, the software should get the PHY link status by reading the PHY register and accordingly update the PLS bit.

  This timer is cleared every time the link goes down. It starts to increment when the link is up again and continues to increment until the value of the timer becomes equal to the terminal count. Once the terminal count is reached, the timer remains at the same value as long as the link is up. The terminal count is the value programmed in Bits[25:16] of the MAC_LPI_Timers_Control register. The GMII interface does not assert the LPI pattern unless the terminal count is reached. This ensures a minimum time for which no LPI pattern is asserted after a link is established with the remote station. This period is defined as 1 second in the IEEE 802.3-az-2010. The LPI LS TIMER is 10-bit wide. Therefore, the software can program up to 1023 milliseconds.

- LPI TW TIMER

  The LPI TW TIMER counts, in microseconds, the time expired since the de-assertion of LPI. The terminal count should be programmed in Bit[15:0] of MAC Register 53 (LPI Timers Control Register). The terminal count of the timer is the value of resolved Transmit TW that is the auto-negotiated time after which the MAC can resume the normal transmit operation. After exiting the LPI mode, the MAC resumes its normal operation after the TW timer reaches the terminal count.

  The MAC supports the LPI TW TIMER in units of microsecond. The LPI TW TIMER is 16-bit wide. Therefore, the software can program up to 65535 ?s.

- LPI AUTO ENTRY TIMER

  This timer counts in steps of eight microseconds, the time for which the MAC transmit path has to remain in idle state (no activity), before the MAC Transmitter enters the LPI IDLE state and starts transmitting the LPI pattern. This timer is enabled when LPITE bit in MAC_LPI_Control_Status register is set.

Note Program the PLS bit of MAC_LPI_Control_Status to 1'b0 before switching between the GMII and MII modes. This resets the internal timers. If the mode is changed after the LPI LS TIMER or LPI TW TIMER starts, the change in the Tx clock frequency can result in incorrect timeout.

10.10.7 LPI Interrupt

The MAC generates the LPI interrupt when the Tx or Rx side enters or exits the LPI state. The interrupt mci_intr_o (sbd_intr_o in EQOS-DMA configurations) is asserted when the LPI interrupt status is set. The LPI interrupt can be cleared by reading the MAC_LPI_Control_Status register.

When the MAC exits the Rx LPI state, then in addition to the mci_intr_o (sbd_intr_o in EQOS-DMA configurations), the sideband signal lpi_intr_o (synchronous to Rx clock) is asserted. You can use the lpi_intr_o signal to trigger the external clock-gating circuitry to restore the application clock to the MAC. The lpi_intr_o signal, synchronous to the Rx clock domain, is provided so that you can stop the application clock when the MAC is in the LPI state. If you do not want to gate-off the application clock during the Rx LPI state, you can leave the lpi_intr_o signal unconnected and use the mci_intr_o (sbd_intr_o in EQOSDMA configurations) signal to detect Rx LPI exit.

The lpi_intr_o signal is generated in the Rx clock domain. It may not be cleared immediately afterthe MAC_LPI_Control_Status register is read. This is because the clear signal, generated in CSRclock domain, has to cross the Rx clock domain, and then clear the interrupt source. This delay is atleast four clock cycles of Rx clock and can be significant when the DWC_ether_qos is operating in the 10Mbps mode.

10.10.8 Enabling Energy Efficient Ethernet (EEE)

To enable the energy efficient ethernet feature:

1. In the coreConsultant GUI, select the **Enable Energy Efficient Ethernet (EEE)** option under the **Low Power Management** section during the Specify Configuration activity in coreConsultant.

   For details about this option, see Low Power Management Parameters in "Parameter Descriptions" on page 419

   Note See the "Enabled" field of the parameter descriptions to understand the dependencies.

2. Perform the relevant steps in "Entering and Exiting the Tx LPI Mode" on page 1372

Note

- The EEE feature is not supported when the MAC is configured to use the TBI, RTBI, SMII, or RMII single PHY interface. Even if the MAC supports multiple PHY interfaces, you should activate the EEE mode only when the MAC is operating with the GMII, MII, or RGMII interface.
- According to the Energy Efficient Ethernet standard (IEEE 802.3az-2010), the LPI mode is supported only in the full-duplex mode. Therefore, you must not enable the LPI mode when the MAC Transmitter is configured for the half-duplex mode.

10.10.9 Programming Guidelines for Energy Efficient Ethernet

For detailed guidelines on the programming guidelines, see

- "Entering and Exiting the Tx LPI Mode" on page 1372
- "Gating Off the CSR Clock in the Rx LPI Mode" on page 1373

11 MAC Management Counters

DWC_ether_qos supports a set of registers known as MAC Management Counters that store statistics on the received and transmitted packets. These counters can be enabled in coreConsultant.

This topic describes the MAC Management Counters.

11.1 MAC Management Counters

The DWC_ether_qos supports storing the statistics about the received and transmitted packets in registers that are accessible through the application.

The counters in the MAC Management Counters (MMC) module can be viewed as an extension of the register address space of the CSR module. The MMC module maintains a set of registers for gathering statistics on the received and transmitted packets. The register set includes a control register for controlling the behavior of the registers, two status registers containing interrupts generated (receive and transmit), and Interrupt Enable registers (receive and transmit). These registers are accessible from the Application through the MAC Control Interface (MCI). Each register is 32-bits wide. The write data is qualified with the corresponding mci_be_i signals. Therefore, non-32-bit accesses are allowed as long as the address is word-aligned. The MMCs are accessed using transactions, in the same way the CSR address space is accessed.

The MMC counters are free running. There is no separate enable for the counters to start. If a particular MMC counter is present in the RTL, it starts counting when corresponding packet is received or transmitted. The Receive MMC counters are updated for packets that are passed by the Address Filter (AFM) block. The statistics of packets, dropped by the AFM module, are not updated unless they are runt packets of less than 6 bytes (DA bytes are not received fully). To get statistics of all packets, set Bit 0 in the “MAC_Packet_Filter” register.

The MMC module gathers statistics on encapsulated IPv4, IPv6, TCP, UDP, or ICMP payloads in received Ethernet packets. This gathering is only enabled when you select the Receive Checksum Offload Engine in coreConsultant.

You can select the MMC counters individually during configuration. The addresses for counters that are not selected become reserved. The width of each MMC counter is 32-bit, by default. You can individually change the width to 16-bit for each selected MMC counter during configuration.

For MMC register details, see registers “MMC_Control” to “MMC_Rx_FPE_Fragment_Cntr” in “Register Descriptions” on page 605

11.2 Address Assignments

The MMC registers follow a certain naming convention and their descriptions use certain terminologies that you should know.

The MMC register naming conventions are as follows.

- "tx" as a prefix or suffix indicates counters associated with transmission
- "rx" as a prefix or suffix indicates counters associated with reception
- "_g" as a suffix indicates registers that count only good packets
- "_gb" as a suffix indicates registers that count packets regardless of whether they are good or bad.

The following definitions define the terminology used in MMC register descriptions.

- Transmitted packets are considered "good" if transmitted successfully. In other words, a transmitted packet is good if the packets transmission is not aborted because of any of the following errors:
  - Jabber Timeout
  - No Carrier or Loss of Carrier
  - Late Collision
  - Packet Underflow
  - Excessive Deferral
  - Excessive Collision
- Received packets are considered "good" if none of the following errors exists:
  - CRC error
  - Runt packet (shorter than 64 bytes)
  - Alignment error (in 10/100 Mbps only)
  - Length error (non-Type packet only)
  - Out of Range (non-Type packet only, longer than 1518 bytes)
  - GMII_RXER Input error
- The maximum transmit frame size depends on the frame type, as follows:
  - Untagged frame maxsize = 1,518
  - VLAN Frame maxsize = 1,522
  - Jumbo Frame maxsize = 9,018
  - JumboVLAN Frame maxsize = 9,022
- The maximum receive packet size depends on the packet type and control bits (JE, S2KP, GPSLCE and EDVLP), as shown in the Table 11-1 table.

Table 11-1 Size of the Maximum Receive Packet

JE	S2KP	GPSLCE	EDVLP	Untagged Frame maxsize in bytes	Single VLAN Frame maxsize in bytes	Double VLAN Frame maxsize in bytes
1	X	X	1	9018	9022	9026
0	1	X	X	2000	2000	2000

JE	S2KP	GPSLCE	EDVLP	Untagged Frame maxsize in bytes	Single VLAN Frame maxsize in bytes	Double VLAN Frame maxsize in bytes
0	0	1	1	GPSL	GPSL+4	GPSL+8
0	0	0	1	1518	1522	1526
1	X	X	0	9018	9022	9022
0	0	1	0	GPSL	GPSL+4	GPSL+4
0	0	0	0	1518	1522	1522

Note

- The MMC counters registers at the following offset addresses are of type Read-Only and have the default value of 0:
  - 0x0714 to 0x0778
  - 0x0780 to 0x07E4
  - 0x0810 to 0x0844
  - 0x0850 to 0x0884
- The MMC counters registers at the following offset addresses are of type Read-Only and have the default value of 0:

11.3 Enabling MAC Management Counters

To enable this feature while configuring the controller in the coreConsultant GUI:

1. Select the **Enable MAC Management Counters** option under the RMON Counters section during the **Specify Configuration** activity in coreConsultant.
2. Select all or required MMC Transmit counters under the RMON Transmit Counters section.
3. Select all or required MMC Receive counters under the RMON Receive Counters section.
4. Select all or required MMC IPC Receive counters under the RMON IPC Receive Counters section.

For details about these options, see "Parameter Descriptions" on page 419.

Note See the "Enabled" field of the parameter descriptions to understand the dependencies.

12 Flow Control

This chapter describes the flow control for Transmit and Receive paths.

12.1 Transmit Flow Control

The Transmit Flow Control involves transmitting Pause packets in full-duplex mode and backpressure in half-duplex mode to control the flow of packets from the remote end.

12.1.1 Flow Control in Full-Duplex Mode

DWC_ether_qos supports full-duplex flow control operations.

In full-duplex mode, the DWC_ether_qos uses one of the following packet types for flow control:

- IEEE 802.3x Pause packets
- Priority flow control (PFC) packets

The PFC packets are used only when the Enable Data Center Bridging option is selected. The PFCE bit of MAC_Rx_Flow_Ctrl register determines whether PFC packets or IEEE 802.3x Pause Control packets are used for flow control. If the PFCE bit is set, the MAC sends the PFC packets.

12.1.1.1 Pause Packet Structure

Table 12-1 describes the fields of a Pause packet.

Table 12-1 Pause Packet Fields

Field	Description
DA	Contains the special multicast address
SA	Contains the MAC address 0
Type	Contains 8808
MAC Control opcode	Contains 0001 for IEEE 802.3x Pause Control packets; 0101 for PFC packets
PT	Contains Pause time specified in the PT field of the MAC_Q#_Tx_Flow_Ctrl register

12.1.1.2 Pause Packet Control

When the FCB bit is set, the MAC generates and transmits a single Pause packet. If the FCB bit is set again after the Pause packet transmission is complete, the MAC sends another Pause packet irrespective of whether the pause time is complete or not. To extend the pause or terminate the pause prior to the time specified in the previously-transmitted Pause packet, the application should program the Pause Time register with appropriate value and then again set the FCB bit.

Similarly, when the mti_flowctrl_i signal is asserted, the MAC generates and transmits a single Pause packet. If the mti_flowctrl_i signal remains asserted at a configurable number of slot times before the Pause time runs out, the MAC transmits a second Pause packet. This process is repeated as long as the mti_flowctrl_i signal remains active. If the mti_flowctrl_i signal goes inactive prior to the sampling time, the MAC transmits a Pause packet with zero Pause time (if the DZPQ bit in MAC_Q#_Tx_Flow_Ctrl register is set to 0) to indicate to the remote end that the Receive buffer is ready to receive new data packets.

For PFC packets, you can specify the priority, pause time, and other controls for a queue in MAC_Q#_Tx_Flow_Ctrl register. The MAC supports independent flow control for each Rx queue. There is one trigger input for each Rx queue. Therefore, when multiple triggers come simultaneously, the MAC sends one PFC packet for each trigger. Based on the hardware trigger or software trigger through respective queues, the MAC sends a PFC packet with programmed Pause Time and priority. If multiple priorities are programmed in same Rx queue, multiple priorities are set for the PFC packet with same Pause Time values. A separate pause timer is available for each Rx queue.

12.1.2 Flow Control in Half-Duplex Mode

In half-duplex mode, the MAC uses the deferral mechanism for the flow control (backpressure). When the application requests to stop receiving packets, the MAC sends a JAM pattern of 32 bytes when it senses a packet reception, provided the transmit flow control is enabled. This results in a collision and the remote station backs off. If the application requests a packet to be transmitted, it is scheduled and transmitted even when the backpressure is activated. If the backpressure is kept activated for a long time (and more than 16 consecutive collision events occur), the remote stations abort the transmission because of excessive collisions.

Table 12-2 describes the flow control in the Tx path for Queue 0 based on the setting of the following bits:

- EHFC bit of MTL_RxQ0_Operation_Mode register
- TFE bit of MAC_Q0_Tx_Flow_Ctrl register
- DM bit of MAC_Configuration register

Flow control is similar for all queues.

Table 12-2 Tx MAC Flow Control

EHFC	TFE	DM	Description
x	0	x	The MAC transmitter does not perform the flow control or backpressure operation.
0	1	0	The MAC transmitter performs back-pressure when Bit 0 of MAC_Q0_Tx_Flow_Ctrl register is set or the sideband signal sbd_flowctrl_i is 1.
1	1	0	The MAC transmitter performs back-pressure when Bit 0 of MAC_Q0_Tx_Flow_Ctrl register is set or the sideband signal sbd_flowctrl_i is 1. In addition, the MAC Tx performs back-pressure when Rx Queue level crosses the threshold set by Bits[10:8] of MTL_RxQ0_Operation_Mode register.
0	1	1	The MAC transmitter sends the Pause packet when Bit 0 of MAC_Q0_Tx_Flow_Ctrl register is set or the sideband signal sbd_flowctrl_i is 1.
1	1	1	The MAC transmitter sends the Pause packet when Bit 0 of MAC_Q0_Tx_Flow_Ctrl register is set or the sideband signal sbd_flowctrl_i is 1. In addition, the MAC Tx sends a Pause packet when Rx Queue level crosses the threshold set by Bits[10:8] of MTL_RxQ0_Operation_Mode register.

12.1.3 Enabling Transmit Flow Control

To independently enable Transmit Flow Control for each Tx queue, set the TFE bit in the MAC_Q#_Tx_Flow_Ctrl register.

12.1.4 Triggering Transmit Flow Control

The Transmit Flow Control involves transmitting Pause packets in full-duplex mode and backpressure in half-duplex mode to control the flow of packets from the remote end. The application can request the MAC to send a Pause packet or initiate back-pressure by using either of the following methods:

- **Software Trigger**: In this method, the application sets the FCB_BPA bit in the corresponding MAC_Q#_Tx_Flow_Ctrl register.
- **Hardware Trigger**: In this method, the application triggers the flow control by asserting the mti_-flowctrl_i signal. In EQOS-CORE configurations, the mti_flowctrl_i signal is an input signal to the MAC. In EQOS-AHB, EQOS-AXI, EQOS-DMA, and EQOS-MTL configurations, tThe mti_flowctrl_i signal is generated as follows:

  ```
  mti_flowctrl_i = sbd_flowctrl_i | hw_flowctrl
  ```

The flow control is triggered based on the following:

- **Rx Queue Threshold**: The flow control operation of the MAC is enabled when the EHFC bit of corresponding MTL_RxQ#_Operation_Mode register is set. The flow control signal to the MAC is asserted when the fill level of the Rx queue crosses the threshold configured in RFA field of MTL_RxQ#_Operation_Mode register. This flow control signal is de-asserted when the fill-level of the queue falls below the threshold configured in the RFD field.

  The hardware flow control generated based on the Rx Queue threshold crossing condition is applicable only when the Rx queue size is 4,096 bytes or more.

Note

The RFS sets the threshold at which the flow control is triggered and the MAC schedules to send a PAUSE frame to be sent by the local transmitter.

The remote transmitter can continue to send packets until it receives this PAUSE packet. So the RXQ needs space to take in this data even after the flow-control is triggered, to avoid overflow and loss of packets.

The max-space required that is calculated theoretically can be 2 max-sized packets + a little more assuming that the read from the RXFIFO is not occurring during this whole time. Theoretically, this makes RFA=4 or more, which implies that RxQ must have at the least 4KB size.

Practically, the system works with RFA=2 (Full – 2KB, with very little probability of overflow).

If you have FIFO size of less than 4K, then with RFA=2, the flow control will be triggered with just 1000B in the RxQ which is even less than max packet size.

The RFA field MTL_RxQ[n]_Operation_Mode register is used for activating the Flow Control. These bits control the threshold (fill-level of Rx queue) at which the flow control is activated.

For Example: If your RXQ size is 2K and you set the RFA to 1K; as soon as you receive 1K, the Flow Control is activated. If you are sending a packet of 1500 bytes (which is allowed as per spec), you will be triggering flow-control even before you receive a complete packet. This will reduce the throughput considerably.

If your RXQ size is 4K, you can set the RFA to 2K, so that you are able to receive the whole packet

- **Sideband Flow Control Signal**: In EQOS-AHB, EQOS-AXI, EQOS-DMA, and EQOS-MTL configurations, an optional and separate sideband flow control signal sbd_flowctrl_i is provided at the top-level. You can select this signal by selecting the Enable Transmit Flow Control Input option while configuring the core. This signal can be used to enable the hardware flow control for any Rx queue size.

  For configurations with 4,096 bytes or larger Rx queues, if sideband flow control signal is also enabled, the flow control signal based on the Rx Queue (hw_flowctrl) and the sideband signal (sbd_flowctrl_i) are logically OR'ed and provided to the MAC. Therefore, flow control is initiated when either of these signals is asserted. The flow control is disabled when both these signals are de-asserted.

12.2 Receive Flow Control

In the Receive path, the Flow Control is functional only in the full-duplex mode. If any Pause packet is received in the half-duplex mode, the packet is considered as a normal control packet.

Note Receive pause packets should have a frame size of 64 bytes.

12.2.1 Description of Receive Flow Control

The Receive Flow Control is implemented by the MAC based on the bit value of the respective register, and the destination address and different fields of the received packet.

Table 12-3 describes the flow control in the Rx path based on the setting of the following bits:

- RFE bit of MAC_Rx_Flow_Ctrl register
- DM bit of MAC_Configuration register

Table 12-3 Rx MAC Flow Control

RFE	DM	Description
0	x	The MAC receiver does not detect the received Pause packets.
1	0	The MAC receiver does not detect the received Pause packets but recognizes such packets as Control packets.
1	1	The MAC receiver detects or processes the Pause packets and responds to such packets by stopping the MAC transmitter.

In configurations with multiple queues, you can enable the PFC packet detection by setting the PFCE bit in the MAC_Rx_Flow_Ctrl register. If PFC packet detection is enabled, the MTL Tx queue corresponding to the received priority is blocked when the PFC packet is received. If PFC packet detection is not enabled in configurations with multiple queues and RFE bit is enabled, the MAC transmitter is blocked when the 802.3x Pause packet is received.

The following list describes the Rx flow control:

1. The MAC checks the destination address of the received Pause packet for either of the following:
   - Multicast destination address: The DA matches the unique multicast address specified for the control packet (48'h0180C2000001).
   - Unicast destination address: The DA matches the content of the MAC Address Register 0 and the UP bit of MAC_Rx_Flow_Ctrl register is set.

     If the UP bit is set and the MAC processes Pause packets with unicast destination address in addition to the unique multicast address.
2. The MAC decodes the following fields of the received packet:
   - Type field: This field is checked for 16'h8808.
   - Opcode field: This field is checked for 16'h0001 (Pause packet) or 16'h0101 (PFC packet).
   - Pause Time or Pause Time Vector field: The Pause time (for Pause packet) is captured to determine the time for which transmitter needs to be blocked. The Pause Time Vector field (for PFC packet) is

captured to determine the time for which the MTL Tx queue corresponding to the received priority needs to be blocked.

- Priority Enable Vector field: This field is valid only for PFC packets. It is captured to determine the MTL Tx queue corresponding to the received priority.

3. If the byte count of the status indicates 64 bytes and there is no CRC error, the MAC transmitter does one of the following:
   - For 802.3x Pause packets, the MAC pauses the transmission of any data packet for the duration of the decoded Pause Time value multiplied by the slot time (64 byte times).
   - For PFC packets, the MAC blocks the Tx queues corresponding to the priority. Based on the priorities assigned to the Tx queues in MAC_TxQ_Prty_Map0 and MAC_TxQ_Prty_Map1 registers, the MAC asserts the respective bit in the mti_disable_txq_o signal and loads the Pause timer corresponding to the priority. The Tx queue corresponding to the priority is blocked till the Pause timer expires.

     In addition to the PFC-based flow control operation, when PFCE bit is set, all bits of the mti_disable_txq_o signal are asserted while in LPI mode so that the ETS or CBS scheduler start over again after exiting the LPI mode.
4. The MAC transfers the received control packet to the application based on the setting of the PCF field in MAC_Packet_Filter register.

If subsequent Pause or PFC packets are received before the earlier Pause Time expires, the MAC updates the Pause Timer with new value.

12.2.2 Enabling Receive Flow Control

To enable Pause Flow Control, set the RFE bit in the MAC_Rx_Flow_Ctrl register.

13 Loopback Mode

DWC_ether_qos supports loopback of transmitted packets to its receiver.

This chapter has the following section:

13.1 Loopback Mode

The MAC supports Loopback of transmitted packets to its receiver.

13.1.1 Guidelines for Using Loopback Mode

The following are some guidelines for using the loopback mode:

- Enable loopback only with the full-duplex mode. In half-duplex mode, the carrier sense signal (crs) or collision (col) signal inputs get sampled which may result into issues such as packet dropping.
- If the loopback mode is enabled without connecting a PHY chip (for example, in FPGA setup), you should externally generate the Tx and Rx clocks and provide these clocks to the MAC.
- Do not loop back big packets. Big packets may get corrupted in the loopback FIFO.

The Transmit and Receive clocks can have an asynchronous timing relationship. Therefore, an asynchronous FIFO is used to make the loopback path of the phy_txd_o data to the Receive path. The asynchronous FIFO is 10-bits (6-bits in 10/100 Mbps mode) wide to accommodate phy_txd_o, phy_txen_o, and phy_txer_o. The depth of FIFO is five in 1000 Mbps mode and nine in 10/100 Mbps mode. The FIFO is free-running to write on the write clock (clk_tx_i) and read on every read clock (clk_rx_i).

At the start of each packet read out of the FIFO, the Write and Read pointers get re-initialized to have an offset of 2 (4 in 10/100 Mbps mode). This avoids overflow or underflow during a packet transfer. This also ensures that the overflow or underflow occurs only during the IPG period between the packets. The FIFO depth of five or nine is sufficient to prevent data corruption for packet sizes up to 9,022 bytes with a difference of 200 ppm between (G)MII Transmit and Receive clock frequencies. Therefore, bigger packets should not be looped back because they may get corrupted in this loopback FIFO.

At the end of every received packet, the Receive Protocol Engine module generates received packet status and sends it to the Receive Packet Controller module. The control, missed packet, and filter fail status are added to the Receive status in the Receive Packet Controller module.

The MAC does not process ARP or PMT packets that are looped back.

13.1.2 Enabling Loopback Mode

The MAC supports Loopback of transmitted packets to its receiver.

To enable this feature, program the LM bit of the MAC_Configuration register.

You can enable loopback for all PHY interfaces. The data is always looped back through internal asynchronous FIFO on to the internal Receive MII or GMII interface, irrespective of which PHY interface is selected. The loopback data is also passed through the corresponding transmit PHY interface block, on to the Ethernet line.

14 Automotive Safety Features

This chapter describes the automotive safety features supported by DWC_ether_qos and enabled by the DWC_EQOS_ASP_ALL parameter when you have the corresponding license.

This chapter has the following sections:

14.1 Error Correction Code (ECC) Protection for Memories

14.1.1 Overview

The Error Correction Code (ECC) block can correct single-bit error and detect double-bit error.

14.1.2 Description of ECC Protection for Memories

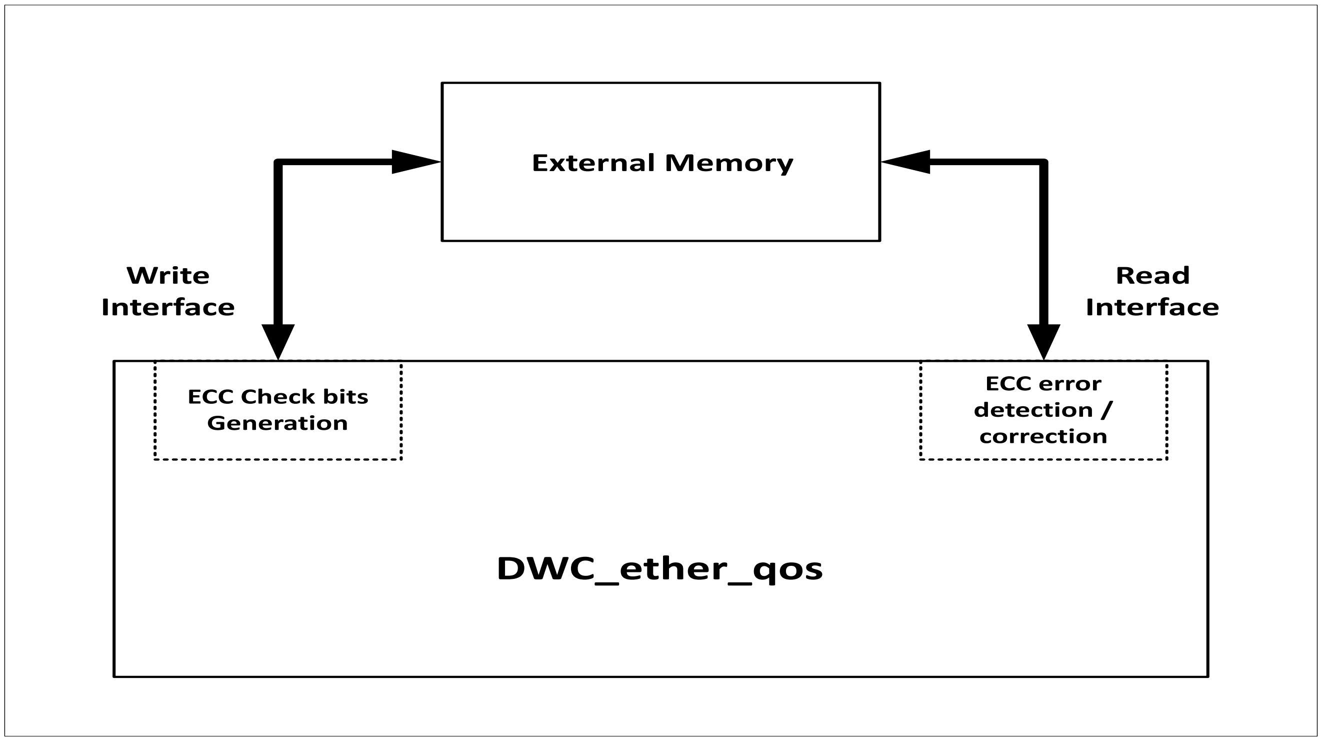
The Error Correction Code (ECC) block can correct single-bit error and detect double-bit error.

At the write interface, ECC checkbits are generated by computing ECC on the contents of the data bus and respective address is appended with the data that is written to the memory.

At the read interface, ECC checkbits are recomputed on the content of the read data and the respective address is compared with the received checkbits in the memory.

Figure 14-1shows the block diagram of ECC protection for memories

Figure 14-1 Block Diagram of ECC Protection for Memories



The number of ECC-checkbits generated for each data block depends on the size of the data bus and address bus. Figure 14-1 describes the memory bus widths, ECC code size and the generated checkbits size.

Table 14-1 Memory Bus Width, ECC Code Size, and Checkbits Size

Memory Block	Configured Data Width	Additional Control Bits	Total Data Width (without ECC)	Address Width (Depends on Memory Size)	ECC Code Reference Range	Required Checkbits	Total Data Width (with ECC)
MTL Tx Memory (2 memories in case of SPRAM and 1 memory in case of DPRAM configuration	32	3	35 bits	5 to 14	Total Data width + Address width <= 57 bits	7 bits	42 bits
	64	4	68 bits	5 to 14	Total Data width + Address width <= 120 bits	8 bits	76 bits
	128	5	133 bits	5 to 14	Total Data width + Address width <= 247 bits	9 bits	142 bits
MTL Rx Memory (2 memories in case of SPRAM and 1 memory in case of DPRAM configuration	32	3	35 bits	5 to 14	Total Data width + Address width <= 57 bits	7 bits	42 bits
	64	4	68 bits	5 to 14	Total Data width + Address width <= 120 bits	8 bits	76 bits
	128	5	133 bits	5 to 14	Total Data width + Address width <= 247 bits	9 bits	142 bits
MTL EST Memory	DWC_EQOS_NUM_TXQ (1 to 8) + DWC_EQOS_EST_WID (16, 20, 24)		DWC_EQOS_NUM_TXQ (1 to 8) + DWC_EQOS_EST_WID (16, 20, 24)	6 to 10	Total Data width + Address width <= 26 bits Total Data width + Address width <= 57 bits	5 bits or 7 bits	DWC_EQOS_NUM_TXQ (1 to 8)+ DWC_EQOS_EST_WID (16, 20, 24)+ (6 or 7 bits)
MTL Rx Parser Memory	96		96 bits	8 bits	96 + 8 <= 120 bits	8 bits	104 bits

Memory Block	Configured Data Width	Additional Control Bits	Total Data Width (without ECC)	Address Width (Depends on Memory Size)	ECC Code Reference Range	Required Checkbits	Total Data Width (with ECC)
DMA TSO Memory	32		32 bits	2 to 11	Total Data width + Address width <= 57 bits	7 bits	39
	64		64 bits	2 to 11	Data width + Address width <= 120 bits	8 bits	72
	128		128 bits	2 to 11	Data width + Address width <= 247 bits	9 bits	137

14.1.2.1 Handling the Address Mismatch

The ECC is calculated over the sum of memory data and memory location address. However only data is written in the memory along with the ECC. (address is excluded).

On the read interface, the ECC is checked over the sum of memory DATA, memory ECC, and Internally generated address. When an ECC correctable error is detected over the range of the address bit position, it is treated as an uncorrectable error. This is because data might have been read from a different location.

14.1.2.2 Diagnostic Support for the Error Management

Following statistics are provided for monitoring the error behavior on each of the memory blocks.

- Error status provided to management
  - A separate status for correctable, uncorrectable, and address mismatch is specified in the MTL_ECC_Interrupt_Status register.
  - Memory locations at which correctable and uncorrectable errors are detected is specified in the MTL_ECC_Err_Addr_Status register. In addition, a control bit (MEEAO field of MTL_ECC_Control register) decides, whether the first errored memory address is reported or the latest errored memory address is reported.
  - MTL_ECC_Err_Cntr_Status register has separate counters to count the number of correctable and uncorrectable errors
- Interrupts provided to management
  - Separate interrupts are generated for correctable and uncorrectable errors.
  - sbd_sfty_ue_intr_o interrupt is generated when uncorrectable errors are detected and these errors cannot be masked.
  - sbd_sfty_ce_intr_o interrupt is generated when correctable errors are detected. DWC_ether_qos sets the respective interrupt enable bits (in the DMA/MTL_ECC_Interrupt_enable register).

- To find the root cause of the error, read the DMA/MTL_Safety_Interrupt_Status and DMA/MTL_ECC_Interrupt_Status registers.
- To clear the interrupt, write 1 in the respective interrupt status bit in DMA/MTL_ECC_Interrupt_Status registers.

For more details, see the "Register Descriptions" on page 605.

Note
- The status/counters/interrupts are generated per memory.
- MTL Tx and Rx memory is logically divided into multiple queues. But, ECC diagnostics (status/counter/interrupts) are common to all queues.

14.1.2.3 Handling the Uncorrectable Errors

Recovery from an uncorrectable error is based on the memory block and the operating mode of DWC_ether_qos.

- **MTL Tx memory**

  If an uncorrectable error is detected, the packet is terminated with a forcible EOF. A CRC error is introduced by MAC for the transmitted packet, and the remaining packets in the FIFO are flushed and flush status (Bit 13) is indicated to the application in the TDES3 write-back descriptor belonging to that queue, (in DMA configuration) or ati_txstatus_o (in MTL configuration). For more details, see "TDES3 Normal Descriptor (Write-Back Format)" on page 1329.

  When an uncorrectable error is detected and reported by asserting the safety uncorrectable interrupt, reset DWC_ether_qos to ensure proper functionality.

  DWC_ether_qos generates an interrupt to indicate to the application that a correctable, uncorrectable or address mismatch error has occurred and the status is indicated in MTL_ECC_Interrupt_Status register.

- **MTL Rx memory**

  When an uncorrectable error is detected, reset DWC_ether_qos to ensure proper functionality.

  DWC_ether_qos generates an interrupt to indicate to the application that correctable, uncorrectable or address mismatch errors are detected and the status is indicated in MTL_ECC_Interrupt_Status register.

Note In DMA configuration, issue soft reset and in MTL configuration issue hard reset (soft reset is not supported in MTL configuration).

- **MTL EST memory**

  When an uncorrectable error is detected in the MTL EST memory, DWC_ether_qos follows these sequence of actions for recovery

  a. DWC_ether_qos disables the EST feature by resetting the EEST bit of MTL_EST_Control register. It sets all the gates to open state, generates an interrupt, and sets the status in the MTL_ECC_Interrupt_status register. This allows the software to reprogram the correct value in the EST memory.
  b. The location of the uncorrected error in the EST memory is available in the ESTUEA/ESTCEA field of the MTL_EST_ECC_Err_Addr_Status register.

- **MTL Rx Parser memory**

  When an uncorrectable error is detected in the MTL Rx Parser memory, an error indication is provided to the Rx Parser block. The parser halts the parsing of the corresponding packet and all subsequent packets are accepted and marked as 'Parsing Incomplete (RXPI)' in the packet status. For more details, see "Receive Normal Descriptor (Write-Back Format)" on page 1340.
- **DMA TSO memory**

  When an uncorrectable error is detected in the DMA TSO memory for any channel, DWC_ether_qos transmits all the related TSO packets with CRC error. It also updates the status in the EUE bit[16] of the transmit descriptor TDES3 (write- back format) to indicate the ECC Error. For more details, see "TDES3 Normal Descriptor (Write-Back Format)" on page 1329.

14.1.2.4 ECC Error Injection Capabilities

DWC_ether_qos supports error injection capabilities for each memory as a static configuration. The position where the errors are injected in the data word is random. For each memory, 3 control bits are provided to inject errors. The 3 bits are:

- A single bit to enable error injection
- Two bits to indicate the type of error to be injected
  - 00: 1 bit error
  - 01: 2 bit error
  - 10: 3 bit error
  - 11: 1 bit error in address field

The control bit descriptions for

- MTL Tx/Rx and DMA TSO memory are specified in the MTL_DBG_CTL register, for and for R.
- MTL EST memory are specified in the MTL_EST_GCL_Control register
- Rx parser memory are specified in the MTL_RXP_Indirect_Acc_Control_Status register

Note

While using debug mode for ECC error injection,

- There should be no traffic in DWC_ether_qos
- When multiple CSR writes are required for writing single data word into the memory, the application should ensure that all the CSR writes corresponding to one memory write maintains the same value for the error injection control word.

14.1.3 Enabling ECC Protection for Memories

Set any one of the following parameters to 1, to enable ECC protection for memories feature. After setting the parameter, set the appropriate bits of the MTL_ECC_Control register to enable ECC on specific features as per your configuration.

- DWC_EQOS_ASP_ECC
- DWC_EQOS_ASP_ALL

For details about these options, see **Automotive Safety Features Parameters** in "Parameter Descriptions" on page 419.

Note See the “Enabled” field of the parameter descriptions to understand the dependencies.

14.1.4 Signals Related to ECC Protection for Memories

- tmi_wdata_ecc_o
- tmi_rdata_ecc_i
- est_wdata_ecc_o
- est_rdata_ecc_o
- tx_odd_wdata_ecc_o
- tx_odd_rdata_ecc_o
- tx_even_wdata_ecc_o
- tx_even_rdata_ecc_o
- rx_odd_wdata_ecc_o
- rx_odd_rdata_ecc_o
- rx_even_wdata_ecc_o
- rx_even_rdata_ecc_o
- twc_wr_data_ecc_o
- trc_rd_data_ecc_o
- rwc_wr_data_ecc_o
- rrc_rd_data_ecc_o
- ari_ecc_err_o
- sbd_sfty_ue_intr_o
- sbd_sfty_ce_intr_o

For a description of signals related to ECC Protection for Memories, see “Signal Descriptions” on page 471.

14.1.5 Registers Related to ECC Protection for Memories

- MTL_ECC_Control
- MTL_DBG_Control
- MTL_ECC_Interrupt_enable
- MTL_ECC_Interrupt_Status
- MTL_ECC_Err_Addr_Status
- MTL_ECC_Err_Sts_Rctl
- DMA_ECC_Interrupt_enable
- DMA_ECC_Interrupt_Status
- MTL_Safety_Interrupt_Status
- DMA_Safety_Interrupt_Status

For more details, see the “Register Descriptions” on page 605.

14.1.6 Programming Guidelines for ECC Protection for Memories

For more details about programming guidelines for ECC protection for memories, see “Programming Guidelines for ECC Protection for Memories” on page 1384.

14.2 On-Chip Data Path Parity Protection

14.2.1 Overview

This feature provides parity protection for Transmit and Receive data paths, where feasible. For every byte of data, one bit parity is generated.

At the transmit side, the data from the host memory is protected with parity until it is written into the MTL Tx FIFO.

At the receive side, data from the MTL Rx FIFO read side is protected with parity until it is sent to the host.

DWC_ether_qos does not support parity protection for data paths in MAC and MTI/MRI interface side in MTL. This is because data manipulation and offloading is performed on the data.

Note

- Byte level granularity is not supported for generating or checking of data path parity. However, the data word is parity protected.
- In case of invalid bytes during EoF, parity is generated and checked for the invalid bytes, if any.

14.2.2 Description

14.2.2.1 Transmit Datapath Parity Protection

The transmit DMA fetches the data packet and descriptor from the host memory through the AHB/AXI/Native Master interface. The descriptors are consumed in the transmit DMA.

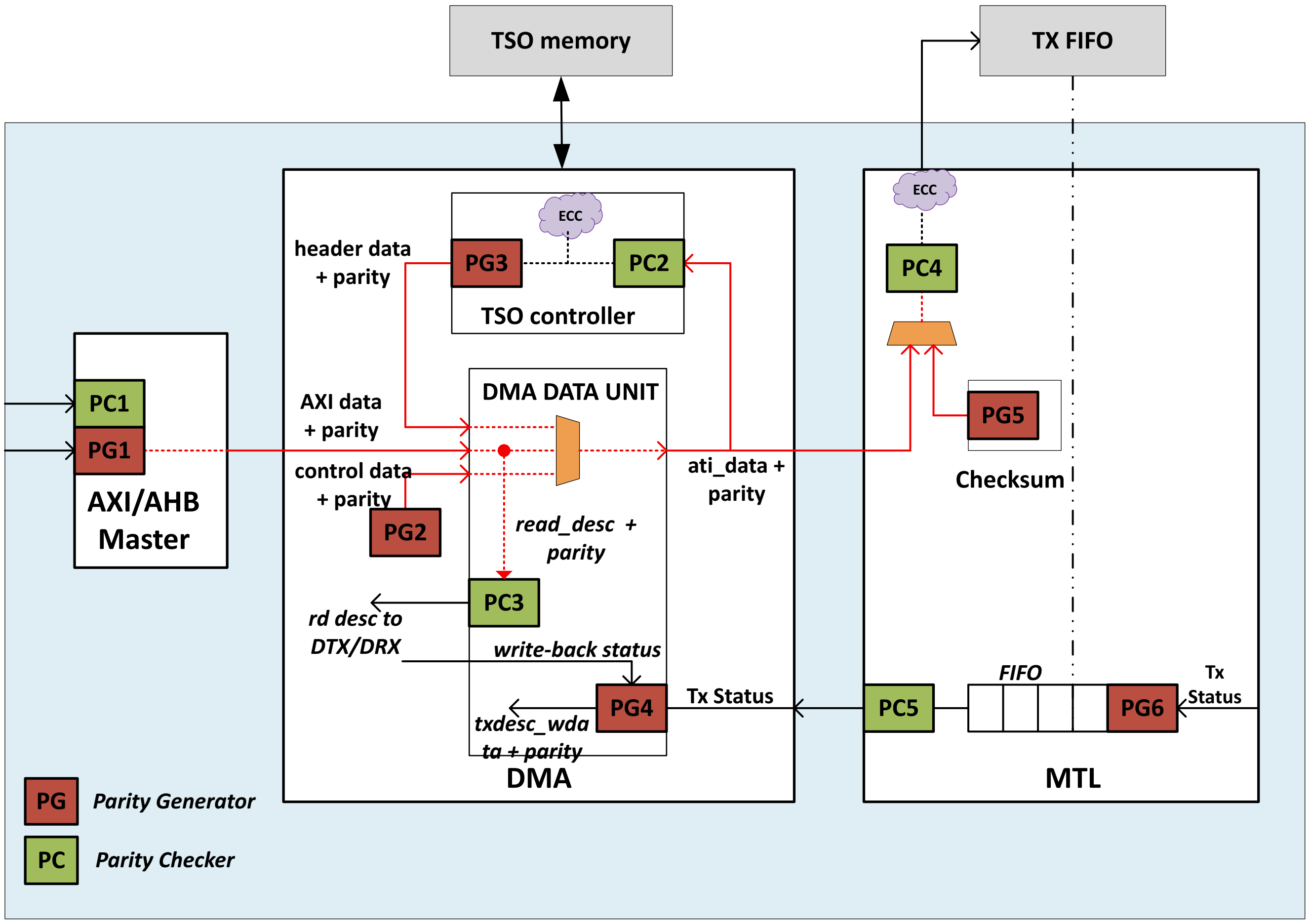
When the TSO feature is enabled, the transmit DMA modifies some of the header fields of the packet for segmentation. So, the header data from TSO memory and the control data from the data transmit module are multiplexed with data from host memory and sent on the ATI interface, to MTL.

When the TSO feature is not enabled, the fetched packet is transferred by the transmit DMA to the MTL TXFIFO after aligning/concatenating the fetched data (when necessary) into different burst transfers. The transmit DMA also generates and pushes the "control words" for each packet in the same data path before the start of a packet.

In the MTL, when TX Checksum Offload Engine is enabled, the data packet is stored/written into the TXFIFO memory. The calculated checksums and the generated control signals are also written into the TXFIFO memory. Therefore, the input data from the ATI interface along with the internally generated checksum control word are multiplexed and then written into the MTL TX FIFO. Transmit status received from MAC is written into a FIFO and sent to the DMA where it is updated along with the descriptor write-back status and then written to the host memory.

Figure 14-2 describes the locations of parity generators and checkers in the transmit data path:

Figure 14-2 Block Diagram of Transmit Data Path Parity Protection



Following are the locations of the parity generators and checkers in the transmit data path:

- Parity is generated on the input data port, PG1 in Figure 14-2. If "parity on external ports" (DWC_EQOS_ASP_PPE) is selected, the input parity provided by the application is propagated; otherwise parity is generated internally. See "Signal Descriptions" on page 471 for details.

Note Parity generator PG1 exists only if "parity on external ports" is selected (DWC_EQOS_ASP_PPE). Otherwise PG1 will not be present.

- A parity check is performed on the input parity data received from the application interface, PC1 in Figure 14-2, and the respective error status is specified in the ATPES field of MAC_DPP_FSM_Interrupt_status register. This check identifies if the fault is internal or external to DWC_ether_qos. The PC1 parity checker is present only if "parity on external ports" (DWC_EQOS_ASP_PPE) is selected during configuration.
- A parity is generated on the control data received from the data transmit module, PG2 in Figure 14-2.

- When TCP segmentation (DWC_EQOS_TSO_EN) is selected during the configuration, parity is generated on the modified header data provided by the DMA TSO controller, PG3 in Figure 14-2).
- When TCP segmentation (DWC_EQOS_TSO_EN) is selected during the configuration, parity check is performed on the feedback header data (PC2 in Figure 14-2) which is originally generated from the DMA TSO controller. The respective parity error is reported in the TPES field of MAC_DPP_FSM_Interrupt_status register.
- As the read descriptor data is consumed in both data tranmit and data receive modules, parity check is performed on this data (PC3 in Figure 14-2). The respective parity error is reported in RDPES field of MAC_DPP_FSM_Interrupt_status register.
- Parity is generated on the modified Tx write-back status which is generated from DMA data unit (PG4 in Figure 14-2).
- When the checksum offloading feature (DWC_EQOS_TX_COE) is selected during the configuration, parity is generated on the checksum data generated in MTL (PG5 in Figure 14-2).
- Parity check is performed at the transmit write controller in MTL (PC4 in Figure 14-2) before writing the data to TX FIFO. The parity generation is terminated at this checker because the ECC protection for memories feature protects the external memory paths. The respective parity error status is reported in MPES field of MAC_DPP_FSM_Interrupt_status register.
- If transmit Status drop (DWC_EQOS_TX_STS_DROP) is disabled in MTL, parity is generated on the status received from MAC (PG6 in Figure 14-2).
- If transmit Status drop (DWC_EQOS_TX_STS_DROP) is disabled in MTL, parity check is performed on the Tx Status which is transferred on ATI interface (PC5 in Figure 14-2). The respective parity error status is reported in MTSPES bit of MAC_DPP_FSM_Interrupt_Status register.

Note When a parity error is detected at the transmit ATI interface, the ati_err is asserted which eventually inserts CRC error for the packet.

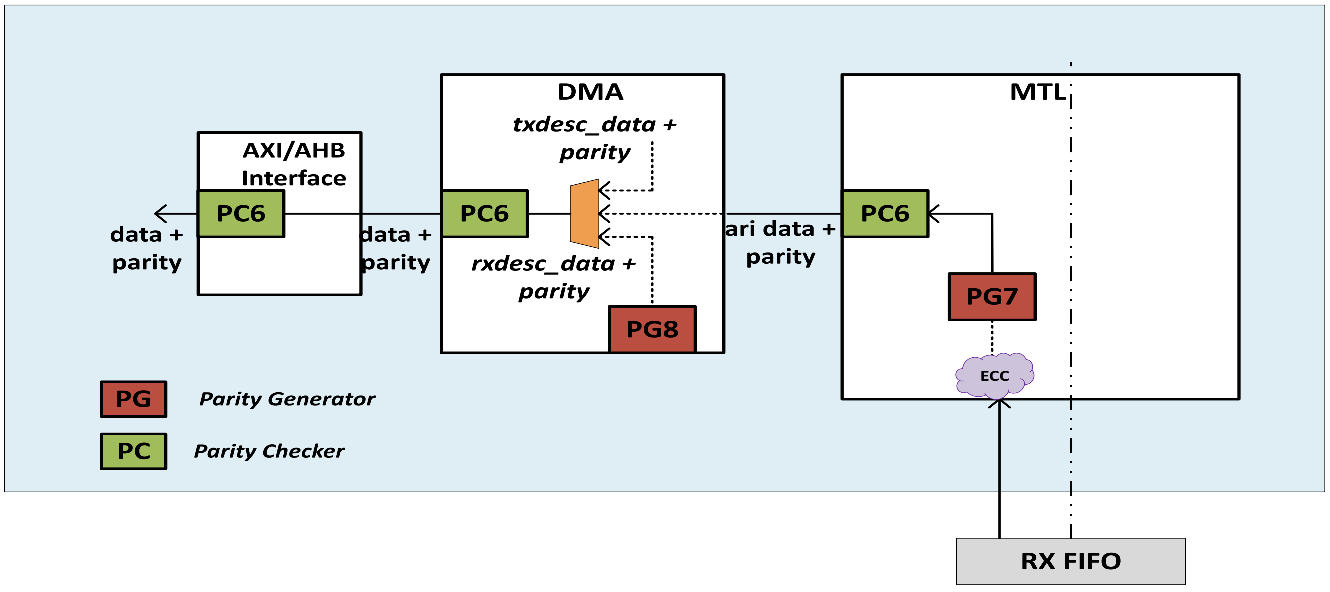
14.2.2.2 Receive Data Path Parity Protection

On the receive side, the data packet and status from the MTL Rx FIFO are sent to DMA where it is multiplexed with the descriptor write-back status from the data transmit/receive module, before transferring to the host. If "parity on external ports" (DWC_EQOS_ASP_PPE) is selected during the configuration, a parity output port is provided to the application. See "Signal Descriptions" on page 471 for more details.

Following are the locations of parity generators and checkers in the receive data path:

- Parity is generated on the MTL Rx FIFO read data (PG7 in Figure 14-3) after ECC correction.
- Parity is generated on the DMA Rx write-back descriptor status after the update by the DMA data unit (PG8 in Figure 14-3).
- Parity check is performed on the output data of AXI Master interface or MCI interface or ARI interface (PC6 in Figure 14-3, based on configuration) and the respective parity error status is reported in the ARPES field of MAC_DPP_FSM_Interrupt_Status register.

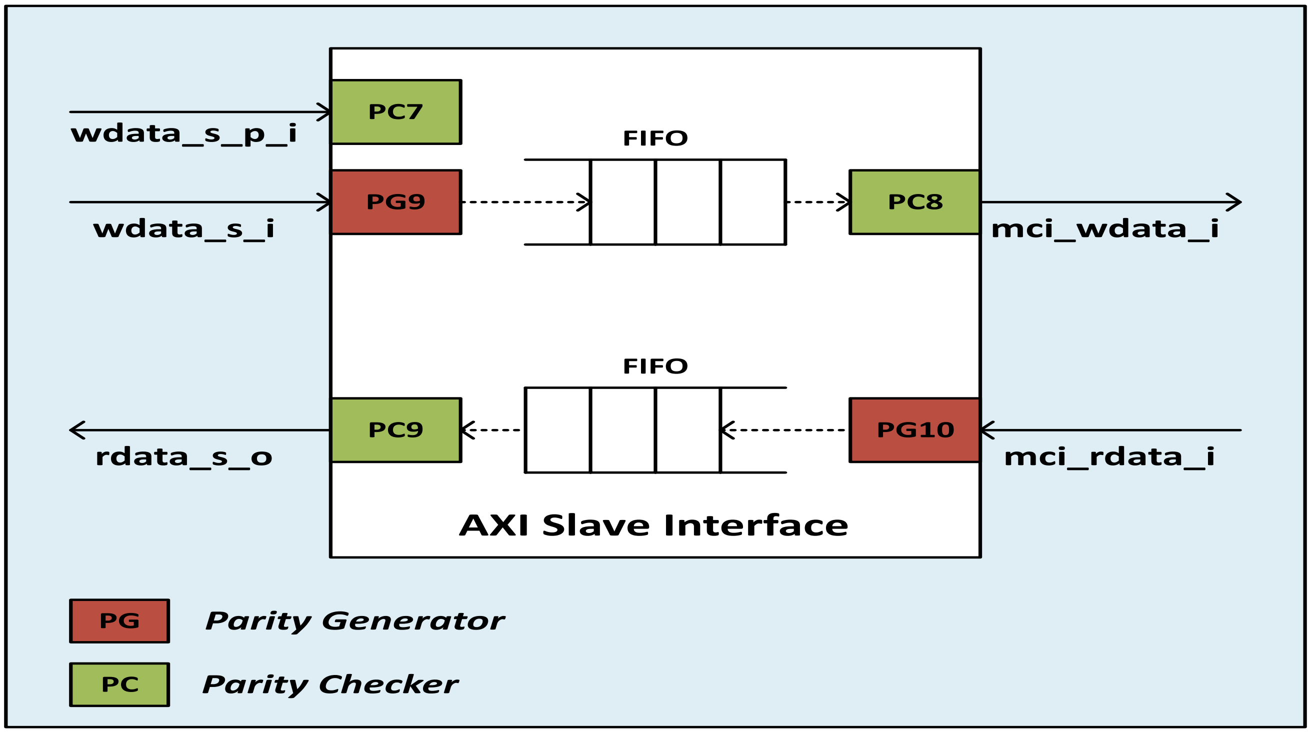
Figure 14-3 Receive Data Path Parity Protection



14.2.2.3 AXI Slave Data Path Parity Protection

The AXI slave interfaces with the application and provides the 32bit, 64-bit or 128-bit CSR access and also handles the Narrow burst and FIXED or INCR burst. As the data is not modified in this block, the parity generators and checkers are provided at the AXI and CSR interface. If "parity on external ports" feature (DWC_EQOS_ASP_PPE) is selected during the configuration, input/output parity ports are driven by/to the application. See "Signal Descriptions" on page 471 for more details.

Figure 14-4 AXI Slave Data Path Parity Protection



Following are the locations of parity generators and checkers in the AXI salve module:

- Parity is generated on the AXI slave write data (wdata_s_i, at PG9 in Figure 14-3). PG9 is always present, irrespective of the selection of "parity on external ports" feature(DWC_EQOS_ASP_PPE).

  When "parity on external ports" feature is selected, either externally provided parity or the internal generated parity can be selected by setting the EPSI field of MTL_DPP_Control register.
- Parity is generated on the CSR read data (mci_rdata_o, PG10 in Figure 14-3).
- When "parity on external ports" feature is selected, (DWC_EQOS_ASP_PPE is enabled) and if the parity check on the external slave ports is enabled (by setting the EPSI bit of MTL_DPP_Control register), parity check is performed on the AXI salve write data (wdata_s_p_i, PC7 in Figure 14-3). The respective parity error status is reported in the CWPES field of the MAC_DPP_FSM_Interrupt_Status register.
- Parity check is performed on the CSR write data (mci_wdata_i, PC8 in Figure 14-3. The respective parity error status is reported in the CWPES field of the MAC_DPP_FSM_Interrupt_Status register.

  If a parity error detected on the CSR write data, the write to the respective address location is stopped and the address for which parity error is detected is specified in the MAC_AXI_SLV_DPE_Addr_Status register.
- Parity check is performed on the AXI Slave read data (rdata_s_o, PC9 in Figure 14-3) and the respective parity error status is reported in the ASRPES field of MAC_DPP_FSM_Interrupt_Status register.

14.2.2.4 Parity Error Injection

Each parity generator supports a control to inject parity error. When enabled, the parity of the first valid data at the generators are flipped. Application must set the appropriate bit in the MTL_DPP_Control register; after the respective parity bit is flipped the hardware clears the same.

After the generator inserts error in the generated parity data, the subsequent parity checker should detect the mismatch and report the error status in the MAC_DPP_FSM_Interrupt_Status register.

For example, if DWC_ether_qos needs to insert an error in the input data (at parity generator PG1 Figure 14-2), the sequence of operations are as follows:

1. Application sets the IPEID bit of MTL_DPP_Control register.
2. The parity generator (PG1 Figure 14-2) inserts an error in the parity bits generated for the first valid data.
3. After the error is inserted, DWC_ether_qos resets the IPEID field of the MTL_DPP_Control register.
4. As the data traverses through multiple paths, the subsequent checkers (PC3 or PC4 as Figure 14-2 in this example) detects the parity mismatch and reports the error status in MAC_DPP_FSM_Interrupt_Status register and generates the sbd_sfty_ue_intr_o interrupt.

14.2.2.5 Interrupt for Parity Error

DWC_ether_qos generates an interrup, sbd_sfty_ue_intr_o to indicate to the application if any parity error is detected. Appropriate status is indicated in DMA/MTL_Safety_Interrupt_Status and MAC_DPP_FSM_Interrupt_Status registers. All the parity errors reported by the different checkers are logged in the MAC_DPP_FSM_Interrupt_Status register.

14.2.3 Enabling On-Chip Data Path Parity Protection

To enable this feature, set the DWC_EQOS_ASP_ALL parameter during configuration. When this feature is enabled, by default, even parity generation/detection is enabled. You can program to select odd parity generation/detection (OPE bit of MTL_DPP_Control register).

For details about these options, see **Automotive Safety Features Parameters** in "Parameter Descriptions" on page 419.

Note See the "Enabled" field of the parameter descriptions to understand the dependencies.

14.2.4 Signals Related to On-Chip Data Path Parity Protection

- sbd_sfty_ue_intr_o
- mci_wdata_p_i
- pwdata_p_i
- prdata_p_o
- wdata_m_p_o
- rdata_m_p_i

- wdata_s_p_i
- rdata_s_p_o
- hrdata_p_i
- hwdata_p_o
- hwdata_p_i
- hrdata_p_o
- mdc_rdata_p_i
- mdc_wdata_p_o

For a description of signals related to On-Chip Data Path Protection, see "Signal Descriptions" on page 471.

14.2.5 Registers Related to On-Chip Data Path Parity Protection

- MAC_DPP_FSM_Interrupt_Status
- MAC_AXI_SLV_DPE_Addr_Status
- MTL_DPP_Control

For more details, see the "Register Descriptions" on page 605.

14.2.6 Programming Guidelines for On-Chip Data Path Parity Protection

For details about programming guidelines for on-chip data path parity protection, see "Programming Guidelines for On-Chip Datapath Parity Protection" on page 1385.

14.3 FSM Parity and Timeout Protection

14.3.1 Overview

The FSM protection feature supports FSM parity and time out protection.

14.3.2 Description

14.3.2.1 FSM State Parity Protection

DWC_ether_qos implements one-hot encoding scheme for holding the state register when the DWC_EQOS_ASP_ALL parameter is set to 1 during configuration. Odd parity is implemented on all the FSM state register bits. Parity is monitored for every clock, after the reset is de-asserted. When a bit flips due to transient errors or permanent faults, the erroneous FSM is set to its default state and an uncorrectable error is indicated.

When the FSM state parity protection is enabled, by setting PRTYEN field of MAC_FSM_Control Register, the FSM state error is indicated by setting the FSMPES field of the MAC_DPP_FSM_Interrupt_Status register. Also, the Safety Interrupt (sbd_sfty_ue_intr_o) is asserted when an FSM Error is detected.

Error Injection mode is also supported for FSM parity error check. Program the Error Injection enable for the respective clock domain denoted by [23:16] bits of MAC_FSM_Control Register.

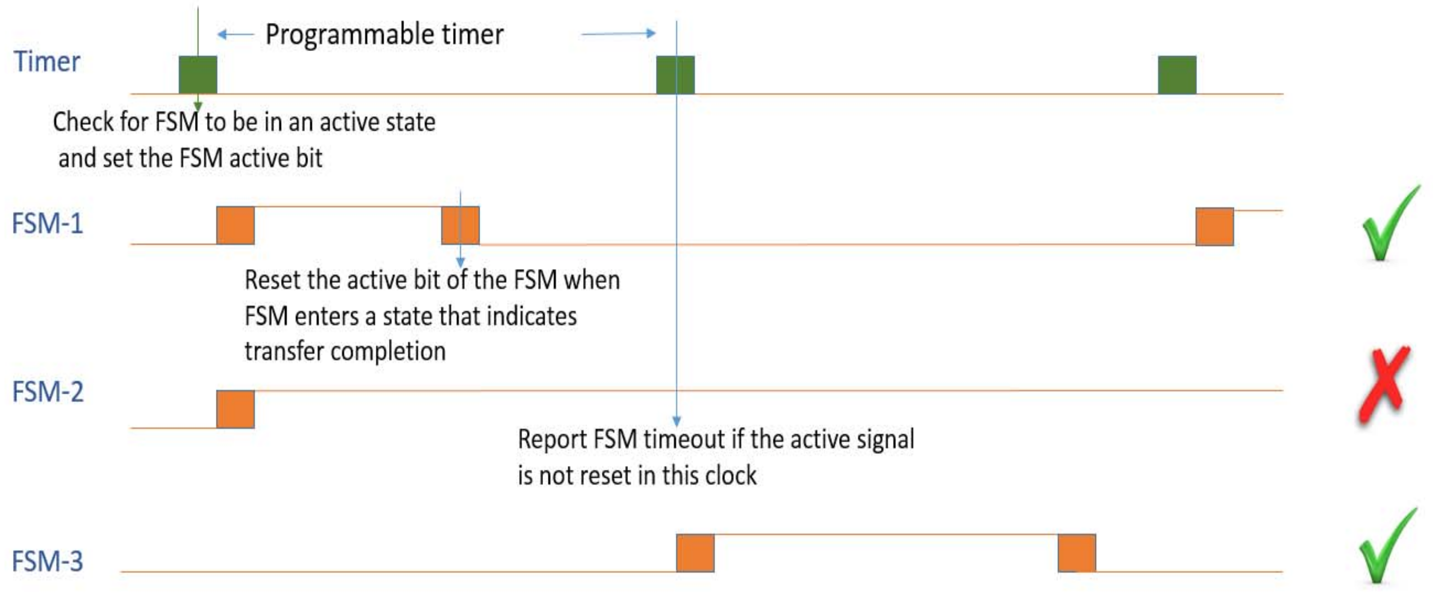
Note The FSM Parity error is classified as non-recoverable error and software intervention (IP reset) is required to recover from the error state.

14.3.2.2 FSM Timeout Protection

The FSM Timeout feature provides a mechanism to ensure that all FSMs in DWC_ether_qos can complete a transaction and return to a known completion state (IDLE or any other state that indicates completion of a transaction/transfer) within the programmed timeout value.

Program the TMR field of MAC_FSM_ACT_Timer register with a value that indicates the number of CSR cycles required to generate a 1 microsecond tic. This microsecond tic is internally used to generate the programmed timeout duration. Two timeout tics (normal mode timeout and large mode timeouts) are generated to provide flexibility to choose one per clock domain, by using the bits [31:24] of MAC_FSM_Control Register. Supported values for Timeout duration are 1us, 1.024ms, 16.384ms, 65.5536ms, 262.144ms, 1048.576ms (~1sec), 4194.304ms (~4s), 8.388s, 16.777s, and 33.554s which is based on the programmable bits NTMRMD, LTMRMD fields of MAC_FSM_ACT_Timer register. Interface timeouts are based on the normal mode tic generation; only the FSM timeouts depend on either normal or large tick selection.

Figure 14-5 FSM Timeout Timing Diagram


Timer
Programmable timer
Check for FSM to be in an active state
and set the FSM active bit
FSM-1
Reset the active bit of the FSM when
FSM enters a state that indicates
transfer completion
FSM-2
Report FSM timeout if the active signal
is not reset in this clock
FSM-3

InFigure 14-5, the timer ticks are generated based on the programmable timeout value. The timer ticks are synchronized and made available in all the clock domains that have the FSMs.

At every timer tick, all the FSMs are monitored for being in active state (non-IDLE state, which indicates the FSM is actively processing transactions or hand-shakes) and an FSM active flag bit is set. The active flag bit is implemented for every FSM that is monitored for timeout. When the FSM reaches an IDLE or transaction completion state, the active flag bit of that FSM is reset. At the subsequent timer tick, all the FSMs' active flag bits are monitored and any flag that is set indicates a timeout for that FSM. This process of setting the flag and checking at subsequent timer tick is repeated at every tick.

Timeout Error Injection per clock domain is enabled by programming [15:8] fields of MAC_FSM_Control register. When Timeout error injection enable is set for a clock domain, the FSMs in that clock domain automatically timeout and generate an interrupt even without traffic. As error injection is a debug mode, TMR value need not indicate 1us, but can be programmed to a smaller value at which debug mode ticks are needed. FSM Timeout and Parity Error Injection modes can be used for testing at key-on/key-off.

The FSM timeout status is specified in the [15:8] field of MAC_DPP_FSM_Interrupt_Status register as per the respective clock domains. One FSM timeout status bit is made available per clock domain. The safety interrupt (sbd_sfty_ue_intr_o) is asserted when the timeout error is set for any of the clock domains. DWC_ether_qos does not attempt to recover from a FSM timeout condition and relies on the application to take the corrective action (resetting DWC_ether_qos).

Note

- Based on the time at which the FSM enters the active state relative to the timer ticks the timeout period could be any value between the programmed timeout and 2x programmed timeout.
- FSM related to PPS (when configured) will not be protected by timeouts as the delays involved in this FSM state transitions can be huge.

14.3.3 Enabling FSM Parity and Timeout Protection

To enable FSM protection, set the DWC_EQOS_ASP_ALL parameter to 1 during the configuration.

FSM timeout feature is enabled by setting the TMOUTEN field of MAC_FSM_Control register.

For details about these options, see **Automotive Safety Features Parameters** in "Parameter Descriptions" on page 419.

Note See the "Enabled" field of the parameter descriptions to understand the dependencies.

14.3.4 Signals Related to FSM Parity and Timeout Protection

- sbd_stfy_ce_intr_o
- sbd_stfy_ue_intr_o

For a description of signals related to FSM Parity and Timeout Protection, see "Signal Descriptions" on page 471.

14.3.5 Registers Related to FSM Parity and Timeout Protection

- MAC_FSM_Control
- MAC_FSM_ACT_Timer
- MAC_DPP_FSM_Interrupt_Status

For more details, see the "Register Descriptions" on page 605.

14.3.6 Programming Guidelines for FSM Parity and Timeout Protection

For details about programming guidelines for FSM protection, see "Programming Guidelines for FSM Parity and Timeout" on page 1386.

14.4 Application/CSR Interface Timeout Protection

14.4.1 Overview

This feature provides timeout protection to the application/CSR Interface.

14.4.2 Description

All the interfaces which has the handshake mechanism in DWCare monitored for potential hangs due to the external agent (Master/Slave/Interconnect/Application client) not responding to the requests/transfers initiated by the QoS. After the request is initiated the response arrival interval is monitored. If the response does not arrive within a programmed time (TMR field of MAC_FSM_ACT_Timer) the timeout is triggered, using similar trigger generation as explained in the Section FSM protection.

The hardware does not attempt to recover from Application/CSR Interface hangs and relies on software to take appropriate corrective action.

The Application Interface timeout status is provided in the [23:16] field of MAC_DPP_FSM_Interrupt_Status register. The Safety interrupt (sbd_sfty_ue_intr_o) is asserted when application timeout status is set.

14.4.2.1 AHB Master Interface

The timeout is triggered when DWC_ether_qos initiates SEQ, NON-SEQ and BUSY transfers on HTRANS[1:0]; and HREADY is not asserted by Slave within the programmed time.

This feature is available in the EQOS-AHB configuration.

The Timeout status for the AHB master interface is set in the MSTTES field of the MAC_DPP_FSM_Interrupt_Status register.

14.4.2.2 ATI and ARI Interface

The timeout is triggered under the following scenarios:

- The DWC_ether_qos requests for the Transmit packet status transfer by asserting the ati_txsta-tus_val_o and the application does not respond with assertion of ati_txstatus_ack_i, within the programmed time.
- The DWC_ether_qos request for a data/status of receive packet by asserting ari_rx_status_val_o and the application does not respond with assertion of ari_ack_i within the programmed time

This feature is available in the EQOS-MTL configuration.

The Timeout status for the ATI and ARI interface is set in the MSTTES field of MAC_DPP_FSM_Interrupt_Status register.

14.4.2.3 AXI Master Interface

The timeout is triggered under the following scenarios:

- The DWC_ether_qos initiates Write or Read ADDRESS transfer (by asserting AWVALID or ARVALID) and corresponding ready (AWREADY or ARREADY) is not asserted within the programmed time.

- The DWC_ether_qos initiates Write DATA transfer (by asserting WVALID) and corresponding ready (WREADY) is not asserted within a programmed time

This feature is available in EQOS-AXI configuration.

The Timeout status for the AXI master interface is set in the MTTES field of MAC_DPP_FSM_Interrupt_Status register.

14.4.2.4 AXI Slave Interface

The timeout is triggered under the following scenarios:

- The DWC_ether_qos responds to a Read request (by asserting RVALID) and external Master does not accept the response data, by asserting (RREADY) within a programmed time.
- The DWC_ether_qos responds to a Write request (by asserting BVALID) and external Master does not accept the response by asserting (BREADY) within a programmed time.

This feature is available in AXI-SLAVE configuration

The Timeout status for the AXI slave interface is set in "SLVTES" field of MAC_DPP_FSM_Interrupt_Status register.

Note

- Protection is not needed for other Slave interfaces (MCI/APB/AHB) because potential hangs can be only when DWC_ether_qos does not respond to the requests. In such cases the DWC_ether_qos internal protection logic (FSM timeout) detects such defects.
- Based on the time at which the transfer is initiated relative to the timer ticks, the timeout period could be any value between the programmed timeout and 2x times the programmed timeout.
- When an uncorrectable safety interrupt is issued and the read of the Interrupt status register returns all zeros, it implies that the CSR read is not functional. As soft reset of DWC_ether_qos is not possible without the CSR access, and therefore, a hard reset of DWC_ether_qos is recommended.

14.4.3 Enabling Application/CSR Interface Timeout Protection

Based on your configuration, this feature is available as follows:

- AHB Master Interface timeout protection is available in EQOS-AHB configuration.
- ATI and ARI Interface timeout protection is available in EQOS-MTL configuration.
- AXI Master Interface timeout protection is available in EQOS-AXI configuration.
- AXI Slave Interface timeout protection is available in AXI_SLAVE configuration.

14.4.4 Signals Related to Application/CSR Interface Timeout Protection

Following is the main signal related to Application/CSR interface timeout protection feature:

- sbd_sfty_ue_intr_o

For a description of signals related to Application/CSR Interface Timeout Protection, see “Signal Descriptions” on page 471.

14.4.5 Programming Guidelines for Application/CSR Interface Timeout Protection

See "Programming Guidelines for FSM Parity and Timeout" on page 1386 for details.

For a description of signals related to Application/CSR Interface Timeout Protection, see "Signal Descriptions" on page 471.

15

Parameter Descriptions

This chapter details all the configuration parameters. **You can use the coreConsultant GUI configuration reports to determine the complete configuration state of the core.** Some expressions might refer to TCL functions or procedures (sometimes identified as **<functionof>**) that coreConsultant uses to make calculations. The exact formula used by these TCL functions is not provided in this chapter. However, when you configure the core in coreConsultant, all TCL functions and parameters are evaluated completely; and the resulting values are displayed where appropriate in the coreConsultant GUI reports.

These tables define all of the user configuration options for this component.

15.1 Features / Application Interface Parameters

Table 15-1 Features / Application Interface Parameters

Label	Description
Application Interface Configuration	Specifies the top-level Application Interface and configuration. **Values:** ■ EQOS-CORE (0) ■ EQOS-MTL (1) ■ EQOS-DMA (2) ■ EQOS-AHB (3) ■ EQOS-AXI (4) ■ EQOS-AXI4 (5) **Default Value:** EQOS-AHB **Enabled:** Always **Parameter Name:** DWC_EQOS_SYS
Data Width	Configures the width of the data bus on the Application interface. **Values:** 32, 64, 128 **Default Value:** 32 **Enabled:** Always **Parameter Name:** DWC_EQOS_DATAWIDTH
Endian Mode	Configures the DWC_ether_qos to operate in either little-endian or big-endian mode, or to have an input pin (sbd_data_endianess_i) to configure both modes. **Values:** ■ LITTLE_ENDIAN (0) ■ BIG_ENDIAN (1) ■ BOTH_OPTIONS (2) **Default Value:** LITTLE_ENDIAN **Enabled:** Always **Parameter Name:** DWC_EQOS_ENDIAN_NESS
Address Width	Configures the width of the address bus on the System Interface **Values:** 32, 40, 48 **Default Value:** 32 **Enabled:** DWC_EQOS_SYS>3 **Parameter Name:** DWC_EQOS_ADDRWIDTH

Label	Description
CSR Interface	Provides Slave port for CSR access (MCI/APB/APB3/APB4 for EQOS-CORE/EQOS-MTL/EQOS-DMA configurations, APB/AHB/APB3/APB4 for EQOS-AHB configurations and APB/AHB/AXI/APB3/APB4/AXI4-Lite for EQOS-AXI/EQOS-AXI4 configurations). **Values:** ■ MCI Interface (0) ■ APB Interface (1) ■ AHB Interface (2) ■ AXI Interface (3) ■ APB3 Interface (4) ■ AXI4-Lite Interface (5) ■ APB4 Interface (6) **Default Value:** (DWC_EQOS_SYS==3) ? 2 : (DWC_EQOS_SYS==4||DWC_EQOS_SYS==5) ? 3 : 0 **Enabled:** Always **Parameter Name:** DWC_EQOS_CSR_PORT
Use Different Clock for CSR	Uses a separate clock port instead of the default application clock for the CSR interface. **Values:** 0, 1 **Default Value:** (DWC_EQOS_SYS==4||DWC_EQOS_SYS==5)&&DWC_EQOS_AHB_SLAVE **Enabled:** DWC_EQOS_SYS!=0&&!((DWC_EQOS_SYS==4||DWC_EQOS_SYS==5)&&DWC_EQOS_AHB_SLAVE) **Parameter Name:** DWC_EQOS_CSR_SLV_CLK
CSR Port Data Width	Configures the width of the data bus on the AHB/AXI Slave port. **Values:** 32, 64, 128 **Default Value:** (DWC_EQOS_CSR_PORT==3) ? DWC_EQOS_DATAWIDTH : (DWC_EQOS_CSR_PORT!=2) ? 32 : [<functionof> "%item"] **Enabled:** DWC_EQOS_CSR_PORT==2||DWC_EQOS_CSR_PORT==3 **Parameter Name:** DWC_EQOS_CSR_DATAWIDTH
Enable Internal DMA Signals as Sideband Ports	Enables DMA internal Interface Control Signals as SideBand I/O ports. **Values:** 0, 1 **Default Value:** 0 **Enabled:** DWC_EQOS_SYS==3 **Parameter Name:** DWC_EQOS_MDC_INTF_SBD_IO_EN
Enable Sideband Inputs for DMA Start and Stop	Enables DMA Starting and Stopping through SideBand Input ports. **Values:** 0, 1 **Default Value:** 0 **Enabled:** !DWC_EQOS_CORE&&!DWC_EQOS_MTL_SUBSYS **Parameter Name:** DWC_EQOS_DMA_STRTSTP

Label	Description
Enable Transmit Flow Control Input	Adds the sideband sbd_flowctrl_i as input port for triggering flow control (backpressure or Pause packets). **Values:** 0, 1 **Default Value:** 0 **Enabled:** DWC_EQOS_SYS!=0 **Parameter Name:** DWC_EQOS_SBD_FC_EN
Add MAC Tx/Rx Disable as Primary Input	Adds the sideband signals sbd_dis_transmit_i and sbd_dis_receive_i as input ports for Disable control of MAC transmitter and receiver. **Values:** 0, 1 **Default Value:** 0 **Enabled:** (DWC_EQOS_SYS==1)||(DWC_EQOS_SYS==0) **Parameter Name:** DWC_EQOS_ADD_TXRX_DIS_IO
Add General Purpose IO	Adds the sideband general purpose IO signals. **Values:** 0, 1 **Default Value:** 0 **Enabled:** Always **Parameter Name:** DWC_EQOS_GPIO_EN
GP Input Signal Width	Sets the width of the general purpose input bus. **Values:** 0, 4, 8, 12, 16 **Default Value:** DWC_EQOS_GPIO_EN ? 4 : 0 **Enabled:** DWC_EQOS_GPIO_EN **Parameter Name:** DWC_EQOS_GIW
GP Output Signal Width	Sets the width of the general purpose output bus. **Values:** 0, 4, 8, 12, 16 **Default Value:** DWC_EQOS_GPIO_EN ? 4 : 0 **Enabled:** DWC_EQOS_GPIO_EN **Parameter Name:** DWC_EQOS_GOW
AXI Maximum Burst Length	Sets the maximum burst length allowed on the AXI bus. **Values:** 16, 32, 64, 128, 256 **Default Value:** DWC_EQOS_SYS==5 ? 256 : 16 **Enabled:** DWC_EQOS_SYS==4 **Parameter Name:** DWC_EQOS_AXI_BL
AXI Master Bus ID Width	Sets the AXI Master bus ID width. **Values:** 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12 **Default Value:** ([<functionof> (DWC_EQOS_NUM_DMA_TX_CH > DWC_EQOS_NUM_DMA_RX_CH ? DWC_EQOS_NUM_DMA_TX_CH : DWC_EQOS_NUM_DMA_RX_CH)])>8 ? 5 : 4 **Enabled:** DWC_EQOS_SYS==4||DWC_EQOS_SYS==5 **Parameter Name:** DWC_EQOS_AXI_GM_AXI_ID_WIDTH

Label	Description
AXI Slave Bus ID Width	Sets the AXI Slave bus ID width. **Values:** 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16 **Default Value:** 8 **Enabled:** DWC_EQOS_CSR_PORT == 3 **Parameter Name:** DWC_EQOS_GS_ID
AXI Maximum Outstanding Read Requests	Indicates maximum Outstanding Read requests on the AXI interface. **Values:** 4, 8, 16, 32 **Default Value:** DWC_EQOS_NUM_DMA_CSR_CH<=2 ? 4 : DWC_EQOS_NUM_DMA_CSR_CH<=4 ? 8 : DWC_EQOS_NUM_DMA_CSR_CH<=8 ? 16 : 32 **Enabled:** DWC_EQOS_SYS==4||DWC_EQOS_SYS==5 **Parameter Name:** DWC_EQOS_AXI_GM_MAX_RD_REQUESTS
AXI Maximum Outstanding Write Requests	Indicates maximum Outstanding Write Requests on AXI interface. **Values:** 4, 8, 16, 32 **Default Value:** DWC_EQOS_NUM_DMA_CSR_CH<=2 ? 4 : DWC_EQOS_NUM_DMA_CSR_CH<=4 ? 8 : DWC_EQOS_NUM_DMA_CSR_CH<=8 ? 16 : 32 **Enabled:** DWC_EQOS_SYS==4||DWC_EQOS_SYS==5 **Parameter Name:** DWC_EQOS_AXI_GM_MAX_WR_REQUESTS

15.2 Features / General Features Parameters

Table 15-2 Features / General Features Parameters

Label	Description
	Operating Mode
Mode of Operation	Configures the DWC_ether_qos to work in the 10/100/1000 Mbps mode. Select 10/100/1000 Mbps for enabling both Fast Ethernet and Gigabit operations. Select 10/100 Mbps for Fast Ethernet-only operations. Select 1000 Mbps for Gigabit-only operations. **Values:** ■ 10/100/1000 Mbps (0) ■ 10/100 Mbps (1) ■ 1000 Mbps (2) **Default Value:** 10/100/1000 Mbps **Enabled:** Always **Parameter Name:** DWC_EQOS_OP_MODE
	Reset Mode
Enable Asynchronous or Synchronous Resets for All Flops	Enables the asynchronous or synchronous reset for all flip-flops. **Values:** ■ Synchronous (0) ■ Asynchronous (1) **Default Value:** Asynchronous **Enabled:** Always **Parameter Name:** DWC_EQOS_ASYNC_RST
Enable Full-Duplex Only Configuration	Configures the DWC_ether_qos to work only in the full-duplex mode. When disabled, both half-duplex and full-duplex operations are supported. **Values:** 0, 1 **Default Value:** 0 **Enabled:** Always **Parameter Name:** DWC_EQOS_FDUPLX_ONLY
User-Defined Version of Core	Configurable 8-bit value that is hard-coded into bits[15:8] of the MAC Version Register (for example, 8'h10 for version 1.0). **Values:** 0x0, ..., 0xff **Default Value:** 0x10 **Enabled:** Always **Parameter Name:** DWC_EQOS_USER_VER
Enable Double VLAN Processing	Enables Double VLAN Processing **Values:** 0, 1 **Default Value:** 0 **Enabled:** Always **Parameter Name:** DWC_EQOS_DOUBLE_VLAN_EN

Label	Description
Enable Queue/Channel based VLAN tag insertion on Tx	Enables Queue/Channel based VLAN tag insertion **Values:** 0, 1 **Default Value:** 0 **Enabled:** DWC_EQOS_NUM_TXQ>1 **Parameter Name:** DWC_EQOS_CBTI_EN
Enable SA and VLAN Insertion on Tx	Enables Source Address Insert/Replace and VLAN Insert/Replace/Delete logic **Values:** 0, 1 **Default Value:** DWC_EQOS_CBTI_EN **Enabled:** !DWC_EQOS_CBTI_EN **Parameter Name:** DWC_EQOS_SA_VLAN_INS_CTRL_EN
Disable Transmit Status in MTL	Enables Transmit Status Dropping in the MTL. The Transmit Status FIFO is removed from the configuration. **Values:** 0, 1 **Default Value:** 0 **Enabled:** DWC_EQOS_SYS!=0 **Parameter Name:** DWC_EQOS_TX_STS_DROP

15.3 Features / General Features / CDC Synchronization Parameters

Table 15-3 Features / General Features / CDC Synchronization Parameters

Label	Description
	CDC Synchronization
Synchronize CSR MAC Address to Tx/Rx Clock Domain	Enables the synchronization of MAC address registers to the transmit or receive clock domain. Otherwise, the CSR MAC Address Register value, written with the application clock, is directly used in the transmit or receive clock domain. Default is direct usage. **Values:** 0, 1 **Default Value:** 0 **Enabled:** Always **Parameter Name:** DWC_EQOS_SYNC_MACADDR
Synchronize Pause Time to Tx Clock Domain	Enables the synchronization of the Pause Timer register to the transmit clock domain.Default is direct usage. **Values:** 0, 1 **Default Value:** 0 **Enabled:** Always **Parameter Name:** DWC_EQOS_SYNC_PAUSETIME
Synchronize VLAN Tag Register to Rx Clock Domain	Enables the synchronization of the VLAN Tag register to the receive clock domain. Default is direct usage. **Values:** 0, 1 **Default Value:** 0 **Enabled:** Always **Parameter Name:** DWC_EQOS_SYNC_VLANTAG
Synchronize Hash Table to Rx Clock Domain	Enables the synchronization of the Hash Table register to the receive clock domain.Default is direct usage. **Values:** 0, 1 **Default Value:** 0 **Enabled:** DWC_EQOS_HASH_EN==1 **Parameter Name:** DWC_EQOS_SYNC_HASHTABLE
Synchronize Layer 3 and Layer 4 Address Registers to Rx Clock Domain	Enables the synchronization of the Layer 3 and Layer 4 Address Registers to the receive clock domain. Default is direct usage. **Values:** 0, 1 **Default Value:** 0 **Enabled:** DWC_EQOS_L3_L4_FILTER_EN==1 **Parameter Name:** DWC_EQOS_SYNC_L3_L4_ADDR
Synchronize ARP Protocol Address Register to Rx and Tx Clock Domains	Enables the synchronization of the ARP Protocol Address Register to the receive and transmit clock domain. Default is direct usage. **Values:** 0, 1 **Default Value:** 0 **Enabled:** DWC_EQOS_ARP_EN **Parameter Name:** DWC_EQOS_SYNC_ARP_PROT_ADDR

Label	Description
Enable support for back to back register writes	Allows back to back register writes for some of the MAC and MTL registers **Values:** 0, 1 **Default Value:** 0 **Enabled:** Always **Parameter Name:** DWC_EQOS_CSR_B2B_WR_SUPP

15.4 Features / Buffer Management Parameters

Table 15-4 Features / Buffer Management Parameters

Label	Description
MTL Receive FIFO Size (in Bytes)	Size (Depth = Size/(Data-bus width in bytes)) in data-bytes (excludes overhead bits) of MTL Receive FIFO **Values:** 256, 512, 1024, 2048, 4096, 8192, 16384, 32768, 65536, 131072, 262144 **Default Value:** 2048 **Enabled:** DWC_EQOS_SYS != 0 **Parameter Name:** DWC_EQOS_RXFIFO_SIZE
MTL Rx FIFO Address Bus Width	Specifies the address bus width for selected Receive FIFO depth. For SPRAM, the address bus width for each memory is one less than this value **Values:** 3, ..., 16 **Default Value:** [<functionof> (log(DWC_EQOS_RXFIFO_SIZE)/log(2)) - ((DWC_EQOS_DATAWIDTH/64)+2)] **Enabled:** DWC_EQOS_SYS!=0 **Parameter Name:** DWC_EQOS_RX_FIFO_PTR_WIDTH
MTL FIFO Data Width	Specifies the data bus width to DPRAM including overhead bits. **Values:** 35, ..., 133 **Default Value:** [<functionof> (DWC_EQOS_DATAWIDTH/64)+3+DWC_EQOS_DATAWIDTH] **Enabled:** DWC_EQOS_SYS != 0 **Parameter Name:** DWC_EQOS_FIFO_DATA_WIDTH
MTL Transmit FIFO Size (in Bytes)	Size (Depth = Size/(Data-bus width in bytes)) in data-bytes (excludes overhead bits) of MTL Transmit FIFO **Values:** 256, 512, 1024, 2048, 4096, 8192, 16384, 32768, 65536, 131072 **Default Value:** 2048 **Enabled:** DWC_EQOS_SYS != 0 **Parameter Name:** DWC_EQOS_TXFIFO_SIZE
Maximum Number of Packets in the Tx FIFO	Specifies the maximum number of packets that can be pushed into the Tx FIFO at a time without any transmit status being read out on the ATI interface. **Values:** 2, 4, 8 **Default Value:** 2 **Enabled:** DWC_EQOS_SYS == 1 && !DWC_EQOS_TX_STS_DROP **Parameter Name:** DWC_EQOS_MAX_FRAME_IN_TXFIFO
MTL Tx FIFO Address Bus Width	Specifies the address bus width for selected Transmit FIFO depth. For SPRAM, the address bus width for each memory is one less than this value **Values:** 4, ..., 15 **Default Value:** [<functionof> (log(DWC_EQOS_TXFIFO_SIZE)/log(2)) - ((DWC_EQOS_DATAWIDTH/64)+2)] **Enabled:** DWC_EQOS_SYS != 0 **Parameter Name:** DWC_EQOS_TX_FIFO_PTR_WIDTH

Label	Description
Use single port RAM	Provides a single port RAM interface for MTL transmit and receive FIFOs, when enabled. When disabled, provides a dual port RAM interface. For single port RAM use two memory blocks that is half the required memory size, for each FIFO. **Values:** 0, 1 **Default Value:** 0 **Enabled:** DWC_EQOS_SYS>0 **Parameter Name:** DWC_EQOS_SPRAM
Enable Debug Memory Access	Enables Debug Memory Access **Values:** 0, 1 **Default Value:** DWC_EQOS_ASP_ECC == 1 **Enabled:** DWC_EQOS_SYS>0&&!DWC_EQOS_ASP_ECC **Parameter Name:** DWC_EQOS_DBG_FIFO_ACCESS

15.5 Features / PHY Interface Parameters

Table 15-5 Features / PHY Interface Parameters

Label	Description
	Line Interface
Enable RGMII	Enables Reduced GMI Interface as Line Interface. **Values:** 0, 1 **Default Value:** 0 **Enabled:** DWC_EQOS_OP_MODE == 0||DWC_EQOS_OP_MODE == 2 **Parameter Name:** DWC_EQOS_RGMII_EN
Enable RMII	Enables Reduced Media Independent Interface as Line Interface. **Values:** 0, 1 **Default Value:** 0 **Enabled:** DWC_EQOS_OP_MODE == 1||DWC_EQOS_OP_MODE == 0 **Parameter Name:** DWC_EQOS_RMII_EN
Enable SMII	Enables Serial Media Independent Interface as Line Interface. **Values:** 0, 1 **Default Value:** 0 **Enabled:** DWC_EQOS_OP_MODE == 1||DWC_EQOS_OP_MODE == 0 **Parameter Name:** DWC_EQOS_SMII_EN
Enable REVMII	Enables Reverse MII Interface as Line Interface. **Values:** 0, 1 **Default Value:** 0 **Enabled:** Always **Parameter Name:** DWC_EQOS_REVMII_EN
Enable SGMII	Enables SGMI Interface as Line Interface. **Values:** 0, 1 **Default Value:** 0 **Enabled:** DWC_EQOS_OP_MODE == 0||DWC_EQOS_OP_MODE == 2 **Parameter Name:** DWC_EQOS_SGMII_EN
Enable TBI	Enables Ten Bit Interface as Line Interface. **Values:** 0, 1 **Default Value:** 0 **Enabled:** DWC_EQOS_OP_MODE == 0||DWC_EQOS_OP_MODE == 2 **Parameter Name:** DWC_EQOS_TBI_EN
Enable RTBI	Enables Reduced Ten Bit Interface as Line Interface. **Values:** 0, 1 **Default Value:** 0 **Enabled:** DWC_EQOS_OP_MODE == 0||DWC_EQOS_OP_MODE == 2 **Parameter Name:** DWC_EQOS_RTBI_EN

Label	Description
Enable GMII/MII (Default)	Enables GMI/MI Interface as Line Interface. **Values:** 0, 1 **Default Value:** DWC_EQOS_GMII_INTF_ONLY||!DWC_EQOS_SINGLE_PHY_INTF **Enabled:** 0 **Parameter Name:** DWC_EQOS_GMII_EN
Enable Single PHY Interface	Enables a single PHY interface without any MUX logic at the input or output. When this option is enabled, the phy_intf_sel_i input port is removed. **Values:** 0, 1 **Default Value:** !DWC_EQOS_SGMII_EN&&!DWC_EQOS_RMII_EN&&!DWC_EQOS_RGMII_EN&&!DWC_EQOS_TBI_EN&&!DWC_EQOS_RTBI_EN&&!DWC_EQOS_SMII_EN&&!DWC_EQOS_REVMII_EN **Enabled:** (!DWC_EQOS_SGMII_EN&&!DWC_EQOS_RMII_EN&&!DWC_EQOS_RGMII_EN&&!DWC_EQOS_TBI_EN&&!DWC_EQOS_RTBI_EN&&!DWC_EQOS_SMII_EN&&!DWC_EQOS_REVMII_EN) ? 0 : ((DWC_EQOS_SGMII_EN&&!DWC_EQOS_RMII_EN&&!DWC_EQOS_RGMII_EN&&!DWC_EQOS_TBI_EN&&!DWC_EQOS_RTBI_EN&&!DWC_EQOS_SMII_EN&&!DWC_EQOS_REVMII_EN)||(DWC_EQOS_RGMII_EN&&!DWC_EQOS_RMII_EN&&!DWC_EQOS_SGMII_EN&&!DWC_EQOS_TBI_EN&&!DWC_EQOS_RTBI_EN&&!DWC_EQOS_SMII_EN&&!DWC_EQOS_REVMII_EN)||(DWC_EQOS_RMII_EN&&!DWC_EQOS_RGMII_EN&&!DWC_EQOS_TBI_EN&&!DWC_EQOS_RTBI_EN&&!DWC_EQOS_SGMII_EN&&!DWC_EQOS_SMII_EN&&DWC_EQOS_M110_ONLY&&!DWC_EQOS_REVMII_EN)||(DWC_EQOS_TBI_EN&&DWC_EQOS_M1000_ONLY&&!DWC_EQOS_RMII_EN&&!DWC_EQOS_RGMII_EN&&!DWC_EQOS_RTBI_EN&&!DWC_EQOS_SGMII_EN&&!DWC_EQOS_SMII_EN&&!DWC_EQOS_REVMII_EN)||(DWC_EQOS_RTBI_EN&&DWC_EQOS_M1000_ONLY&&!DWC_EQOS_RMII_EN&&!DWC_EQOS_RGMII_EN&&!DWC_EQOS_TBI_EN&&!DWC_EQOS_SGMII_EN&&!DWC_EQOS_SMII_EN&&!DWC_EQOS_REVMII_EN)||(!DWC_EQOS_RTBI_EN&&!DWC_EQOS_RMII_EN&&!DWC_EQOS_RGMII_EN&&!DWC_EQOS_TBI_EN&&!DWC_EQOS_SGMII_EN&&!DWC_EQOS_SMII_EN&&!DWC_EQOS_REVMII_EN)||(DWC_EQOS_SMII_EN&&!DWC_EQOS_RMII_EN&&!DWC_EQOS_RGMII_EN&&!DWC_EQOS_TBI_EN&&!DWC_EQOS_RTBI_EN&&!DWC_EQOS_SGMII_EN&&DWC_EQOS_M110_ONLY&&!DWC_EQOS_REVMII_EN)||(DWC_EQOS_REVMII_EN&&!DWC_EQOS_RGMII_EN&&!DWC_EQOS_TBI_EN&&!DWC_EQOS_RTBI_EN&&!DWC_EQOS_SGMII_EN&&!DWC_EQOS_SMII_EN&&!DWC_EQOS_RMII_EN)) **Parameter Name:** DWC_EQOS_SINGLE_PHY_INTF
Enable SMII Source Synchronous Mode	Enables the Source Synchronous mode for Serial Media Independent Interface (SSSMII). **Values:** 0, 1 **Default Value:** 0 **Enabled:** DWC_EQOS_SMII_EN **Parameter Name:** DWC_EQOS_SSSMII_EN

Label	Description
Enable TXSYNC as Input in SMII Source Synchronous Mode	Converts the TXSYNC as input signal in the Source Synchronous mode for Serial Media Independent Interface (SSSMII). **Values:** 0, 1 **Default Value:** 0 **Enabled:** DWC_EQOS_SMII_EN&&DWC_EQOS_SSSMII_EN **Parameter Name:** DWC_EQOS_SSSMII_TXSYNC_IN
Enable Station Management (MDIO Interface)	Enables the Station Management block (MDIO Master Interface). **Values:** 0, 1 **Default Value:** (DWC_EQOS_REVMII_EN&&DWC_EQOS_SINGLE_PHY_INTF) == 0 **Enabled:** (DWC_EQOS_REVMII_EN&&DWC_EQOS_SINGLE_PHY_INTF) == 0 **Parameter Name:** DWC_EQOS_SMA_EN

15.6 Features / Filtering Parameters

Table 15-6 Features / Filtering Parameters

Label	Description
Enable Additional 1-31 MAC Address Registers	Enables number of MAC Address registers from 1 to 31 for the Address Filter block. **Values:** 0, ..., 31 **Default Value:** 0 **Enabled:** Always **Parameter Name:** DWC_EQOS_ADD_MAC_ADDR_REG
Enable Additional 32 MAC Address Registers (32-63)	Enables MAC Address registers 32-63 for the Address Filter block. **Values:** 0, 1 **Default Value:** 0 **Enabled:** Always **Parameter Name:** DWC_EQOS_ADD32_MAC_ADDR_REG
Enable Additional 64 MAC Address Registers (64-127)	Enables MAC Address registers 64-127 for the Address Filter block. **Values:** 0, 1 **Default Value:** 0 **Enabled:** Always **Parameter Name:** DWC_EQOS_ADD64_MAC_ADDR_REG
Enable Address Filter Hash Table	Enables the Hash Table registers and related address filter logic. **Values:** 0, 1 **Default Value:** 0 **Enabled:** Always **Parameter Name:** DWC_EQOS_HASH_EN
Hash Table Size	Specifies the size (width) of the Hash Table filter. **Values:** 0, 64, 128, 256 **Default Value:** (DWC_EQOS_HASH_EN) ? 64 : 0 **Enabled:** DWC_EQOS_HASH_EN **Parameter Name:** DWC_EQOS_HASH_TABLE
Enable Flexible Programmable Receive Parser	This selects the Flexible Rx Parser logic for filtering or forwarding received packets to specific Rx DMA channel. **Values:** 0, 1 **Default Value:** 0 **Enabled:** DWC_EQOS_SYS>1&&DWC_EQOS_SPRAM&&(DWC_EQOS_SPLIT_HDR_EN||DWC_EQOS_NUM_DMA_RX_CH>1)&&<DWC-ETHERNET-QOS-TSN2 feature authorized> **Parameter Name:** DWC_EQOS_FRP_EN

Label	Description
Maximum Packet Header Size for Parsing	This parameter decides the maximum number of bytes from the Start of Packet that will be processed by the Parser. This has significant impact on area because ping-pong buffers are instantiated for storing the packet headers during parsing. **Values:** 64, 128, 256 **Default Value:** 256 **Enabled:** DWC_EQOS_FRP_EN **Parameter Name:** DWC_EQOS_FRP_BUF_SIZE
Maximum Entries of Parser Lookup Table	Entries in the lookup table used by the Parser. This lookup table is implemented in a SRAM of width 96 bits and depth equal to this parameter. **Values:** 64, 128, 256 **Default Value:** 256 **Enabled:** DWC_EQOS_FRP_EN **Parameter Name:** DWC_EQOS_FRP_ENTRIES
Enable VLAN Hash Table Based Filtering	Enables the VLAN Hash Table register and related filter logic. **Values:** 0, 1 **Default Value:** 0 **Enabled:** Always **Parameter Name:** DWC_EQOS_VLAN_HASH_EN
Extended Rx VLAN Filter Enable	Enables the VLAN Tags based Filtering and Routing logic. **Values:** 0, 1 **Default Value:** 0 **Enabled:** Always **Parameter Name:** DWC_EQOS_ERVFE
Number of VLAN Tag Filters	Specifies the number of VLAN Tag based filters. **Values:** 1, 4, 8, 16, 24, 32 **Default Value:** (DWC_EQOS_ERVFE==1) ? 4 : 1 **Enabled:** DWC_EQOS_ERVFE==1 **Parameter Name:** DWC_EQOS_NRVF
Enable Layer 3 and Layer 4 Packet Filter	Enables Layer 3 and Layer 4 based frame filtering for receive packets. **Values:** 0, 1 **Default Value:** 0 **Enabled:** Always **Parameter Name:** DWC_EQOS_L3_L4_FILTER_EN
Number of Layer 3 and Layer 4 Packet Filters	Specifies the number of additional Layer 3 and Layer 4 packet filters. **Values:** 0, 1, 2, 3, 4, 5, 6, 7, 8 **Default Value:** (DWC_EQOS_L3_L4_FILTER_EN==1) ? 1 : 0 **Enabled:** DWC_EQOS_L3_L4_FILTER_EN==1 **Parameter Name:** DWC_EQOS_L3_L4_FILTER_NUM

Label	Description
Enable Packet Duplication Support	Adds Packet Duplication Support **Values:** 0, 1 **Default Value:** 0 **Enabled:** (DWC_EQOS_SYS>1)&&(DWC_EQOS_NUM_DMA_RX_CH>1)&&SNPS_RSVDPARAM_0 **Parameter Name:** DWC_EQOS_PDUP

15.7 Features / Multiple Queues Support Parameters

Table 15-7 Features / Multiple Queues Support Parameters

Label	Description
Number of Transmit Queues	Selects the number of transmit queues **Values:** 1, 2, 3, 4, 5, 6, 7, 8 **Default Value:** DWC_EQOS_AV_ENABLE ? 2 : 1 **Enabled:** Always **Parameter Name:** DWC_EQOS_NUM_TXQ
Number of Receive Queues	Selects the number of receive queues **Values:** 1, 2, 3, 4, 5, 6, 7, 8 **Default Value:** DWC_EQOS_DCB_EN ? 2 : 1 **Enabled:** Always **Parameter Name:** DWC_EQOS_NUM_RXQ
Enable Data Center Bridging	Enables Data Center Bridging Feature. **Values:** 0, 1 **Default Value:** 0 **Enabled:** Always **Parameter Name:** DWC_EQOS_DCB_EN
Enable Audio Video Bridging	Adds logic for supporting the Audio Video functionality and provides the option to select additional channels for queuing the time-sensitive traffic. **Values:** 0, 1 **Default Value:** 0 **Enabled:** Always **Parameter Name:** DWC_EQOS_AV_ENABLE
Enable Support for AV in Tx Queue 1	Enables AV Support in Tx Queue 1 **Values:** 0, 1 **Default Value:** DWC_EQOS_SYS>0&&DWC_EQOS_AV_ENABLE&&DWC_EQOS_NUM_TXQ==2 **Enabled:** DWC_EQOS_SYS>0&&DWC_EQOS_AV_ENABLE&&DWC_EQOS_NUM_TXQ>=2 **Parameter Name:** DWC_EQOS_TX_Q1_AV_EN
Enable Support for AV in Tx Queue 2	Enables AV Support in Tx Queue 2 **Values:** 0, 1 **Default Value:** DWC_EQOS_SYS>0&&DWC_EQOS_AV_ENABLE&&((DWC_EQOS_NUM_TXQ>2&&(DWC_EQOS_TX_Q1_AV_EN))||DWC_EQOS_NUM_TXQ==3) **Enabled:** DWC_EQOS_SYS>0&&DWC_EQOS_AV_ENABLE&&DWC_EQOS_NUM_TXQ>=3&&!DWC_EQOS_TX_Q1_AV_EN **Parameter Name:** DWC_EQOS_TX_Q2_AV_EN

Label	Description
Enable Support for AV in Tx Queue 3	Enables AV Support in Tx Queue 3 **Values:** 0, 1 **Default Value:** DWC_EQOS_SYS>0&&DWC_EQOS_AV_ENABLE&&((DWC_EQOS_NUM_TXQ>3&&(DWC_EQOS_TX_Q2_AV_EN))||DWC_EQOS_NUM_TXQ==4) **Enabled:** DWC_EQOS_SYS>0&&DWC_EQOS_AV_ENABLE&&DWC_EQOS_NUM_TXQ>=4&&!DWC_EQOS_TX_Q2_AV_EN **Parameter Name:** DWC_EQOS_TX_Q3_AV_EN
Enable Support for AV in Tx Queue 4	Enables AV Support in Tx Queue 4 **Values:** 0, 1 **Default Value:** DWC_EQOS_SYS>0&&DWC_EQOS_AV_ENABLE&&((DWC_EQOS_NUM_TXQ>4&&(DWC_EQOS_TX_Q3_AV_EN))||DWC_EQOS_NUM_TXQ==5) **Enabled:** DWC_EQOS_SYS>0&&DWC_EQOS_AV_ENABLE&&DWC_EQOS_NUM_TXQ>=5&&!DWC_EQOS_TX_Q3_AV_EN **Parameter Name:** DWC_EQOS_TX_Q4_AV_EN
Enable Support for AV in Tx Queue 5	Enables AV Support in Tx Queue 5 **Values:** 0, 1 **Default Value:** DWC_EQOS_SYS>0&&DWC_EQOS_AV_ENABLE&&((DWC_EQOS_NUM_TXQ>5&&(DWC_EQOS_TX_Q4_AV_EN))||DWC_EQOS_NUM_TXQ==6) **Enabled:** DWC_EQOS_SYS>0&&DWC_EQOS_AV_ENABLE&&DWC_EQOS_NUM_TXQ>=6&&!DWC_EQOS_TX_Q4_AV_EN **Parameter Name:** DWC_EQOS_TX_Q5_AV_EN
Enable Support for AV in Tx Queue 6	Enables AV Support in Tx Queue 6 **Values:** 0, 1 **Default Value:** DWC_EQOS_SYS>0&&DWC_EQOS_AV_ENABLE&&((DWC_EQOS_NUM_TXQ>6&&(DWC_EQOS_TX_Q5_AV_EN))||DWC_EQOS_NUM_TXQ==7) **Enabled:** DWC_EQOS_SYS>0&&DWC_EQOS_AV_ENABLE&&DWC_EQOS_NUM_TXQ>=7&&!DWC_EQOS_TX_Q5_AV_EN **Parameter Name:** DWC_EQOS_TX_Q6_AV_EN

Label	Description
Enable Support for AV in Tx Queue 7	Enables AV Support in Tx Queue 7 **Values:** 0, 1 **Default Value:** DWC_EQOS_SYS>0&&DWC_EQOS_AV_ENABLE&&((DWC_EQOS_NUM_TXQ>7&&(DWC_EQOS_TX_Q6_AV_EN))||DWC_EQOS_NUM_TXQ==8) **Enabled:** DWC_EQOS_SYS>0&&DWC_EQOS_AV_ENABLE&&DWC_EQOS_NUM_TXQ==8&&!DWC_EQOS_TX_Q6_AV_EN **Parameter Name:** DWC_EQOS_TX_Q7_AV_EN
Number of DMA Channels on the Transmit Side	Selects the number of DMA Channels on the transmit side. **Values:** 1, 2, 3, 4, 5, 6, 7, 8 **Default Value:** (DWC_EQOS_SYS>1) ? DWC_EQOS_NUM_TXQ : 1 **Enabled:** DWC_EQOS_SYS>1 **Parameter Name:** DWC_EQOS_NUM_DMA_TX_CH
Number of DMA Channels on the Receive Side	Selects the number of DMA Channels on the receive side. **Values:** 1, 2, 3, 4, 5, 6, 7, 8 **Default Value:** 1 **Enabled:** DWC_EQOS_SYS>1 **Parameter Name:** DWC_EQOS_NUM_DMA_RX_CH

15.8 Features / IEEE 1588 Timestamp Parameters

Table 15-8 Features / IEEE 1588 Timestamp Parameters

Label	Description
	IEEE 1588 Timestamp
Enable IEEE 1588 Timestamp Support	Adds logic for supporting IEEE1588 timestamping protocol. Snoops receive packets and captures Timestamp for packets marked as PTP packets. The Timestamp is also captured for transmit packets marked as PTP packets. **Values:** 0, 1 **Default Value:** DWC_EQOS_AV_ENABLE==1 **Enabled:** DWC_EQOS_AV_ENABLE==0 **Parameter Name:** DWC_EQOS_TIME_STAMPING
IEEE 1588 System Time Source	Specifies the source of 64-bit reference System Time to take the timestamps that are provided with the status. **Values:** ■ Internal (0) ■ External (1) ■ Both Options (2) **Default Value:** Internal **Enabled:** DWC_EQOS_TIME_STAMPING==1 **Parameter Name:** DWC_EQOS_SYSTIME_SOURCE
Add IEEE 1588 Higher Word Register	Adds a register to store the most significant 16 bits of the 48-bit seconds time register for IEEE 1588-2008 implementation. **Values:** 0, 1 **Default Value:** 0 **Enabled:** DWC_EQOS_TIME_STAMPING&&DWC_EQOS_SYSTIME_SOURCE!=1 **Parameter Name:** DWC_EQOS_ADV_TIME_HIGH_WORD
Enable IEEE 1588 Sub Nanoseconds timestamp support	Adds sub nanoseconds resolution to the system timestamp and timestamp correction **Values:** 0, 1 **Default Value:** 0 **Enabled:** DWC_EQOS_OST_EN&&DWC_EQOS_SYSTIME_SOURCE!=1 **Parameter Name:** DWC_EQOS_ADV_SUBNSEC_SUPPORT
Add IEEE 1588 Auxiliary Snapshot	Adds the capability of storing the timestamp snapshots, taken with an external trigger, into a 4-deep FIFO. **Values:** 0, 1 **Default Value:** 0 **Enabled:** DWC_EQOS_TIME_STAMPING==1 **Parameter Name:** DWC_EQOS_ADV_TIME_AUX_SNAP
Number of IEEE 1588 Auxiliary Snapshot Inputs	Specifies the number of external trigger inputs for capturing the timestamps. **Values:** 0, 1, 2, 3, 4 **Default Value:** (DWC_EQOS_ADV_TIME_AUX_SNAP==1) ? 1:0 **Enabled:** DWC_EQOS_ADV_TIME_AUX_SNAP==1 **Parameter Name:** DWC_EQOS_AUX_SNAP_IN_NUM

Label	Description
FIFO Depth for IEEE 1588 Auxiliary Snapshots	Specifies the depth of FIFO to store the IEEE 1588 Auxiliary snapshots. **Values:** 4, 8, 16 **Default Value:** 4 **Enabled:** DWC_EQOS_ADV_TIME_AUX_SNAP==1 **Parameter Name:** DWC_EQOS_AUX_FIFO_DEPTH
Enable Flexible Pulse-Per-Second Output	Enables the feature to program the start or stop time, width, and interval of the pulse generated on the ptp_pps_o output. **Values:** 0, 1 **Default Value:** 0 **Enabled:** DWC_EQOS_TIME_STAMPING==1&&DWC_EQOS_SYSTIME_SOURCE!=1 **Parameter Name:** DWC_EQOS_FLEXI_PPS_OUT_EN
Number of Pulse-Per-Second Outputs	Specifies the number of Pulse-Per-Second (PPS) Outputs. **Values:** 0, 1, 2, 3, 4 **Default Value:** (DWC_EQOS_FLEXI_PPS_OUT_EN==0) ? 0 : 1 **Enabled:** DWC_EQOS_FLEXI_PPS_OUT_EN **Parameter Name:** DWC_EQOS_PPS_OUT_NUM
Enable PTP Timestamp Offload Feature	Enables PTP Offload (Automatic generation of SYNC, Delay_Resp, Pdelay_Req, and Pdelay_Resp PTP packets) in the Transmit path. **Values:** 0, 1 **Default Value:** 0 **Enabled:** DWC_EQOS_TIME_STAMPING==1 **Parameter Name:** DWC_EQOS_PTO_EN
Enable One step timestamp for PTP over UDP/IP feature	Enables One step timestamp operation for PTP message sent over UDP/IP **Values:** 0, 1 **Default Value:** 0 **Enabled:** DWC_EQOS_TIME_STAMPING==1 **Parameter Name:** DWC_EQOS_POU_OST_EN
Enable One-Step Timestamp Feature	Enables One-Step Timestamp in the transmit path. **Values:** 0, 1 **Default Value:** DWC_EQOS_PTO_EN||DWC_EQOS_POU_OST_EN **Enabled:** DWC_EQOS_TIME_STAMPING&&!DWC_EQOS_PTO_EN&&!DWC_EQOS_POU_OST_EN **Parameter Name:** DWC_EQOS_OST_EN

15.9 Features / Time Sensitive Networking Parameters

Table 15-9 Features / Time Sensitive Networking Parameters

Label	Description
Enable Enhancements to Scheduling Traffic (EST)	Adds logic for supporting the TSN- EST (Enhancements to Scheduling Traffic). **Values:** 0, 1 **Default Value:** 0 **Enabled:** DWC_EQOS_TX_AV_EN&&(DWC_EQOS_SYS>0)&&<DWC-Ether-QOS-TSN1 feature authorized> **Parameter Name:** DWC_EQOS_AV_EST
Width of the Time Interval field in the Gate Control List	Width of the Time Interval field in the Gate Control List. **Values:** 16, 20, 24 **Default Value:** 24 **Enabled:** DWC_EQOS_AV_EST **Parameter Name:** DWC_EQOS_EST_WID
Depth of one instance of Gate Control List	Depth of one instance of Gate Control List. **Values:** 64, 128, 256, 512, 1024 **Default Value:** 256 **Enabled:** DWC_EQOS_AV_EST **Parameter Name:** DWC_EQOS_EST_DEP
Gate Control List (GCL) Memory depth	Gate Control List (GCL) Memory depth. **Values:** 128, 256, 512, 1024, 2048 **Default Value:** [<functionof> ((DWC_EQOS_EST_DEP)*2)] **Enabled:** DWC_EQOS_AV_EST **Parameter Name:** DWC_EQOS_EST_TDEP
GCL Memory Data Width	GCL Memory Data Width. **Values:** 17, ..., 32 **Default Value:** [<functionof> ((DWC_EQOS_EST_WID)+(DWC_EQOS_NUM_TXQ))] **Enabled:** DWC_EQOS_AV_EST **Parameter Name:** DWC_EQOS_EMW
GCL Memory Address Width	GCL Memory Address Width. **Values:** 7, ..., 11 **Default Value:** [<functionof> (log(DWC_EQOS_EST_TDEP)/log(2))] **Enabled:** DWC_EQOS_AV_EST **Parameter Name:** DWC_EQOS_GMAW
Enable Frame Preemption Support	Adds logic for Frame Preemption Support **Values:** 0, 1 **Default Value:** 0 **Enabled:** (DWC_EQOS_SYS>0&&DWC_EQOS_NUM_TXQ>1&&DWC_EQOS_NUM_RXQ>1) &&<DWC-Ether-QOS-TSN1 feature authorized> **Parameter Name:** DWC_EQOS_FPE

15.10 Features / Time Based Scheduling Parameters

Table 15-10 Features / Time Based Scheduling Parameters

Label	Description
Enable Time Based Scheduling	Adds logic for Time Based Scheduling **Values:** 0, 1 **Default Value:** 0 **Enabled:** (DWC_EQOS_SYS>0)&&DWC_EQOS_TIME_STAMPING&&(DWC_EQOS_NUM_TXQ>1)&&<DWC-Ether-QOS-TSN1 feature authorized> **Parameter Name:** DWC_EQOS_TBS

15.11 Features / TCP/IP Offloading Features Parameters

Table 15-11 Features / TCP/IP Offloading Features Parameters

Label	Description
	Checksum Offload Engine
Enable Receive TCP/IP Checksum Check	Enables the Checksum Offload engine in the receive path. In this mode, the Checksum Offload engine checks and detects the IPv4 header checksum errors and TCP, UDP, or ICMP checksum errors encapsulated in IPv4 or IPv6 datagrams payload of the received frames. **Values:** 0, 1 **Default Value:** DWC_EQOS_TIME_STAMPING||DWC_EQOS_L3_L4_FILTER_EN||DWC_EQOS_SPLIT_HDR_EN **Enabled:** !DWC_EQOS_TIME_STAMPING&&!DWC_EQOS_L3_L4_FILTER_EN&&!DWC_EQOS_SPLIT_HDR_EN **Parameter Name:** DWC_EQOS_RX_COE
Enable Transmit TCP/IP Checksum Offload	Enables the Checksum Offload Engine in the transmit path. In this mode, the DWC_ether_qos can calculate and insert the checksums of TCP, UDP, or ICMP segments encapsulated in IPv4 or IPv6 datagrams. The DWC_ether_qos can also insert IPv4 header checksums in the transmitted frames. **Values:** 0, 1 **Default Value:** DWC_EQOS_TSO_EN||DWC_EQOS_POU_OST_EN **Enabled:** DWC_EQOS_SYS!=0&&!DWC_EQOS_TSO_EN&&!DWC_EQOS_POU_OST_EN **Parameter Name:** DWC_EQOS_TX_COE
Enable Support for Tx COE in Tx Queue 0	Enables Tx COE in TX Queue 0 **Values:** 0, 1 **Default Value:** DWC_EQOS_TX_COE **Enabled:** DWC_EQOS_TX_COE&&DWC_EQOS_NUM_TXQ>1&&!DWC_EQOS_TSO_EN **Parameter Name:** DWC_EQOS_TX_Q0_TX_COE
Enable Support for Tx COE in Tx Queue 1	Enables Tx COE in TX Queue 1 **Values:** 0, 1 **Default Value:** DWC_EQOS_NUM_DMA_TSO_CH>1 **Enabled:** (DWC_EQOS_TX_COE&&DWC_EQOS_NUM_TXQ>1&&!(DWC_EQOS_TSO_EN&&DWC_EQOS_NUM_DMA_TSO_CH>1)) **Parameter Name:** DWC_EQOS_TX_Q1_TX_COE

Label	Description
Enable Support for Tx COE in Tx Queue 2	Enables Tx COE in TX Queue 2 **Values:** 0, 1 **Default Value:** DWC_EQOS_NUM_DMA_TSO_CH>2 **Enabled:** (DWC_EQOS_TX_COE&&DWC_EQOS_NUM_TXQ>2&&!(DWC_EQOS_TSO_EN&&DWC_EQOS_NUM_DMA_TSO_CH>2)) **Parameter Name:** DWC_EQOS_TX_Q2_TX_COE
Enable Support for Tx COE in Tx Queue 3	Enables Tx COE in TX Queue 3 **Values:** 0, 1 **Default Value:** DWC_EQOS_NUM_DMA_TSO_CH>3 **Enabled:** (DWC_EQOS_TX_COE&&DWC_EQOS_NUM_TXQ>3&&!(DWC_EQOS_TSO_EN&&DWC_EQOS_NUM_DMA_TSO_CH>3)) **Parameter Name:** DWC_EQOS_TX_Q3_TX_COE
Enable Support for Tx COE in Tx Queue 4	Enables Tx COE in TX Queue 4 **Values:** 0, 1 **Default Value:** DWC_EQOS_NUM_DMA_TSO_CH>4 **Enabled:** (DWC_EQOS_TX_COE&&DWC_EQOS_NUM_TXQ>4&&!(DWC_EQOS_TSO_EN&&DWC_EQOS_NUM_DMA_TSO_CH>4)) **Parameter Name:** DWC_EQOS_TX_Q4_TX_COE
Enable Support for Tx COE in Tx Queue 5	Enables Tx COE in TX Queue 5 **Values:** 0, 1 **Default Value:** DWC_EQOS_NUM_DMA_TSO_CH>5 **Enabled:** (DWC_EQOS_TX_COE&&DWC_EQOS_NUM_TXQ>5&&!(DWC_EQOS_TSO_EN&&DWC_EQOS_NUM_DMA_TSO_CH>5)) **Parameter Name:** DWC_EQOS_TX_Q5_TX_COE
Enable Support for Tx COE in Tx Queue 6	Enables Tx COE in TX Queue 6 **Values:** 0, 1 **Default Value:** DWC_EQOS_NUM_DMA_TSO_CH>6 **Enabled:** (DWC_EQOS_TX_COE&&DWC_EQOS_NUM_TXQ>6&&!(DWC_EQOS_TSO_EN&&DWC_EQOS_NUM_DMA_TSO_CH>6)) **Parameter Name:** DWC_EQOS_TX_Q6_TX_COE
Enable Support for Tx COE in Tx Queue 7	Enables Tx COE in TX Queue 7 **Values:** 0, 1 **Default Value:** DWC_EQOS_NUM_DMA_TSO_CH>7 **Enabled:** (DWC_EQOS_TX_COE&&DWC_EQOS_NUM_TXQ>7&&!(DWC_EQOS_TSO_EN&&DWC_EQOS_NUM_DMA_TSO_CH>7)) **Parameter Name:** DWC_EQOS_TX_Q7_TX_COE

Label	Description
	TCP Segmentation Offload
Enable TCP Segmentation Offloading for TCP/IP Packets	Enables TCP Segmentation on the transmit side **Values:** 0, 1 **Default Value:** 0 **Enabled:** DWC_EQOS_SYS>=2&&DWC_EQOS_SYS<=5 **Parameter Name:** DWC_EQOS_TSO_EN
Number of Tx DMA Channels Enabled for TSO	Selects the number of Tx DMA Channels Enabled for TSO. **Values:** 1, 2, 3, 4, 5, 6, 7, 8 **Default Value:** 1 **Enabled:** DWC_EQOS_TSO_EN **Parameter Name:** DWC_EQOS_NUM_DMA_TSO_CH
Enable Separate Memory for TCP/IP Headers	Enables use of a separate Memory for TCP/IP headers used in TCP Segmentation Offloading **Values:** 0, 1 **Default Value:** 0 **Enabled:** DWC_EQOS_TSO_EN **Parameter Name:** DWC_EQOS_TSO_MEM_EN
Per-Channel TSO Memory Size in Bytes	Specifies the size allocated (in bytes) for each TSO channel. **Values:** 64, 128, 256, 512, 1024 **Default Value:** 256 **Enabled:** DWC_EQOS_TSO_MEM_EN **Parameter Name:** DWC_EQOS_TSO_MEM_SIZE
Enable Split Header Feature	Enables the logic to split the packet header (Layer 2, Layer 3, and Layer 4) in a different buffer and the payload in a different buffer. **Values:** 0, 1 **Default Value:** 0 **Enabled:** DWC_EQOS_SYS>=2&&DWC_EQOS_SYS<=5 **Parameter Name:** DWC_EQOS_SPLIT_HDR_EN
Enable IPv4 ARP Offload	Enables the IPv4 ARP Offload feature. In this mode, the DWC_ether_qos can process the ARP request packet in the receive path and generate corresponding ARP response packet in the transmit path. **Values:** 0, 1 **Default Value:** 0 **Enabled:** Always **Parameter Name:** DWC_EQOS_ARP_EN

15.12 Features / Low Power Management Parameters

Table 15-12 Features / Low Power Management Parameters

Label	Description
	Low Power Management
Enable Energy Efficient Ethernet (EEE)	Adds Logic for supporting the Energy Efficient Ethernet (EEE). **Values:** 0, 1 **Default Value:** 0 **Enabled:** !DWC_EQOS_RTBI_INTF_ONLY&&!DWC_EQOS_TBI_INTF_ONLY&&!DWC_EQOS_SGMII_INTF_ONLY&&!DWC_EQOS_SMII_INTF_ONLY&&!DWC_EQOS_RMII_INTF_ONLY&&!DWC_EQOS_REVMII_INTF_ONLY **Parameter Name:** DWC_EQOS_EEE_EN
Enable Power Management	Enables the Power Management (PMT) block in the DWC_ether_qos core. **Values:** 0, 1 **Default Value:** 0 **Enabled:** Always **Parameter Name:** DWC_EQOS_PMT_EN
Enable Magic Packet Detection	Enables the detection of a magic packet. **Values:** 0, 1 **Default Value:** 0 **Enabled:** DWC_EQOS_PMT_EN **Parameter Name:** DWC_EQOS_PMT_MGK_EN
Enable Remote Wake-Up Packet Detection	Enables the detection of a remote wake-up packet. **Values:** 0, 1 **Default Value:** 0 **Enabled:** DWC_EQOS_PMT_EN **Parameter Name:** DWC_EQOS_PMT_RWK_EN
Number of Remote Wake-Up Packet Filters	Selects the number of remote wake-up packet filters. **Values:** 0, 4, 8, 16 **Default Value:** DWC_EQOS_PMT_RWK_EN ? 4 : 0 **Enabled:** DWC_EQOS_PMT_RWK_EN **Parameter Name:** DWC_EQOS_NUM_PMT_RWK_FILT

15.13 Features / RMON Counters Parameters

Table 15-13 Features / RMON Counters Parameters

Label	Description
	RMON Counters
Enable MAC Management Counters (MMC)	Enables the MAC Management (RMON) counters. **Values:** 0, 1 **Default Value:** 0 **Enabled:** Always **Parameter Name:** DWC_EQOS_MMC_EN
Enable the MAC Management Counters for Receive TCP/IP Checksum Offload	Enables the MAC Management (RMON) counters for the Checksum Offload module. **Values:** 0, 1 **Default Value:** 0 **Enabled:** DWC_EQOS_MMC_EN&&DWC_EQOS_RX_COE **Parameter Name:** DWC_EQOS_MMC_IPC_EN

15.14 Features / RMON Counters / RMON Transmit Counters Parameters

Table 15-14 Features / RMON Counters / RMON Transmit Counters Parameters

Label	Description
	RMON Transmit Counters
Enable All MMC Tx Counters	Enables all Tx packet counters. **Values:** 0, 1 **Default Value:** 0 **Enabled:** DWC_EQOS_MMC_EN **Parameter Name:** DWC_EQOS_MMC_EN_TX_ALL
Enable MMC Tx Octet Counters for Various Packet Sizes	Enables the octet counters for all transmitted packets of different sizes (=64, 65-127, 128-255, 256-511, 512-1023, 1024-maxsize). **Values:** 0, 1 **Default Value:** DWC_EQOS_MMC_EN_TX_ALL **Enabled:** DWC_EQOS_MMC_EN&&!DWC_EQOS_MMC_EN_TX_ALL **Parameter Name:** DWC_EQOS_MMC_TXADDR_RNG_CNT_EN
Enable 16-bit Support	Enables the 16-bit octet counters for all transmitted packets of different sizes (=64, 65-127, 128-255, 256-511, 512-1023, 1024-maxsize). **Values:** 0, 1 **Default Value:** 0 **Enabled:** DWC_EQOS_MMC_TXADDR_RNG_CNT_EN **Parameter Name:** DWC_EQOS_MMC_TXADDR_RNG_CNT_16BIT_EN
Enable MMC Tx Octet Counter for Good and Bad Packets	Enables Octet counter for all good and bad transmitted packets. **Values:** 0, 1 **Default Value:** DWC_EQOS_MMC_EN_TX_ALL **Enabled:** DWC_EQOS_MMC_EN&&!DWC_EQOS_MMC_EN_TX_ALL **Parameter Name:** DWC_EQOS_MMC_TXOCTGB_CNT_EN
Enable MMC Tx Counter for Good and Bad Packets	Enables Packet counter for all good and bad transmitted packets. **Values:** 0, 1 **Default Value:** DWC_EQOS_MMC_EN_TX_ALL **Enabled:** DWC_EQOS_MMC_EN&&!DWC_EQOS_MMC_EN_TX_ALL **Parameter Name:** DWC_EQOS_MMC_TXPKTGB_CNT_EN
Enable 16-bit Support	Enables 16-bit support for Frame counter for all good and bad transmitted packets. **Values:** 0, 1 **Default Value:** 0 **Enabled:** DWC_EQOS_MMC_TXPKTGB_CNT_EN **Parameter Name:** DWC_EQOS_MMC_TXPKTGB_CNT_16BIT_EN
Enable MMC Tx Counter for Good Broadcast Packets	Enables counter for transmitted good broadcast packets. **Values:** 0, 1 **Default Value:** DWC_EQOS_MMC_EN_TX_ALL **Enabled:** DWC_EQOS_MMC_EN&&!DWC_EQOS_MMC_EN_TX_ALL **Parameter Name:** DWC_EQOS_MMC_TXBCASTG_CNT_EN

Label	Description
Enable 16-bit Support	Enables 16-bit counter support for transmitted good broadcast packets. **Values:** 0, 1 **Default Value:** 0 **Enabled:** DWC_EQOS_MMC_TXBCASTG_CNT_EN **Parameter Name:** DWC_EQOS_MMC_TXBCASTG_CNT_16BIT_EN
Enable MMC Tx Counter for Good Multicast Packets	Enables counter for transmitted good multicast packets. **Values:** 0, 1 **Default Value:** DWC_EQOS_MMC_EN_TX_ALL **Enabled:** DWC_EQOS_MMC_EN&&!DWC_EQOS_MMC_EN_TX_ALL **Parameter Name:** DWC_EQOS_MMC_TXMCASTG_CNT_EN
Enable 16-bit Support	Enables 16-bit counter support for transmitted good multicast packets. **Values:** 0, 1 **Default Value:** 0 **Enabled:** DWC_EQOS_MMC_TXMCASTG_CNT_EN **Parameter Name:** DWC_EQOS_MMC_TXMCASTG_CNT_16BIT_EN
Enable MMC Tx Counter for Good and Bad Unicast Packets	Enables counter for transmitted good or bad unicast packets. **Values:** 0, 1 **Default Value:** DWC_EQOS_MMC_EN_TX_ALL **Enabled:** DWC_EQOS_MMC_EN&&!DWC_EQOS_MMC_EN_TX_ALL **Parameter Name:** DWC_EQOS_MMC_TXUCASTGB_CNT_EN
Enable 16-bit Support	Enables 16-bit counter support for transmitted good or bad unicast packets. **Values:** 0, 1 **Default Value:** 0 **Enabled:** DWC_EQOS_MMC_TXUCASTGB_CNT_EN **Parameter Name:** DWC_EQOS_MMC_TXUCASTGB_CNT_16BIT_EN
Enable MMC Tx Counter for Good and Bad Multicast Packets	Enables counter for transmitted good or bad multicast packets. **Values:** 0, 1 **Default Value:** DWC_EQOS_MMC_EN_TX_ALL **Enabled:** DWC_EQOS_MMC_EN&&!DWC_EQOS_MMC_EN_TX_ALL **Parameter Name:** DWC_EQOS_MMC_TXMCASTGB_CNT_EN
Enable 16-bit Support	Enables 16-bit counter support for transmitted good or bad multicast packets. **Values:** 0, 1 **Default Value:** 0 **Enabled:** DWC_EQOS_MMC_TXMCASTGB_CNT_EN **Parameter Name:** DWC_EQOS_MMC_TXMCASTGB_CNT_16BIT_EN
Enable MMC Tx Counter for Good and Bad Broadcast Packets	Enables counter for transmitted good or bad broadcast packets. **Values:** 0, 1 **Default Value:** DWC_EQOS_MMC_EN_TX_ALL **Enabled:** DWC_EQOS_MMC_EN&&!DWC_EQOS_MMC_EN_TX_ALL **Parameter Name:** DWC_EQOS_MMC_TXBCASTGB_CNT_EN

Label	Description
Enable 16-bit Support	Enables 16-bit counter support for transmitted good or bad broadcast packets. **Values:** 0, 1 **Default Value:** 0 **Enabled:** DWC_EQOS_MMC_TXBCASTGB_CNT_EN **Parameter Name:** DWC_EQOS_MMC_TXBCASTGB_CNT_16BIT_EN
Enable MMC Tx Counter for Packets with Underflow Error	Enables counter for packets transmitted with underflow error bit set. **Values:** 0, 1 **Default Value:** DWC_EQOS_MMC_EN_TX_ALL **Enabled:** DWC_EQOS_MMC_EN&&!DWC_EQOS_MMC_EN_TX_ALL **Parameter Name:** DWC_EQOS_MMC_TXUNDRFLW_CNT_EN
16-bit Support	Enables 16-bit counter support for packets transmitted with underflow error bit set. **Values:** 0, 1 **Default Value:** 0 **Enabled:** DWC_EQOS_MMC_TXUNDRFLW_CNT_EN **Parameter Name:** DWC_EQOS_MMC_TXUNDRFLW_CNT_16BIT_EN
Enable MMC Tx Counter for Good Packets with Single Collision	Enables counter for packets transmitted with single collision in the half-duplex mode . **Values:** 0, 1 **Default Value:** DWC_EQOS_MMC_EN_TX_ALL&&!DWC_EQOS_FDUPLX_ONLY **Enabled:** DWC_EQOS_MMC_EN&&!DWC_EQOS_MMC_EN_TX_ALL&&!DWC_EQOS_FDUPLX_ONLY **Parameter Name:** DWC_EQOS_MMC_TXSNGLCOLG_CNT_EN
Enable 16-bit Support	Enables 16-bit counter support for packets transmitted with single collision in the half-duplex mode. **Values:** 0, 1 **Default Value:** 0 **Enabled:** DWC_EQOS_MMC_TXSNGLCOLG_CNT_EN **Parameter Name:** DWC_EQOS_MMC_TXSNGLCOLG_CNT_16BIT_EN
Enable MMC Tx Counter for Good Packets with Multiple Collisions	Enables counter for packets transmitted with multiple collision in the half-duplex mode. **Values:** 0, 1 **Default Value:** DWC_EQOS_MMC_EN_TX_ALL&&!DWC_EQOS_FDUPLX_ONLY **Enabled:** DWC_EQOS_MMC_EN&&!DWC_EQOS_MMC_EN_TX_ALL&&!DWC_EQOS_FDUPLX_ONLY **Parameter Name:** DWC_EQOS_MMC_TXMULTCOLG_CNT_EN
Enable 16-bit Support	Enables 16-bit counter support for packets transmitted with multiple collision in the half-duplex mode. **Values:** 0, 1 **Default Value:** 0 **Enabled:** DWC_EQOS_MMC_TXMULTCOLG_CNT_EN **Parameter Name:** DWC_EQOS_MMC_TXMULTCOLG_CNT_16BIT_EN

Label	Description
Enable MMC Tx Counter for Deferred Packets	Enables counter for packets deferred before transmission in the half-duplex mode. **Values:** 0, 1 **Default Value:** DWC_EQOS_MMC_EN_TX_ALL&&!DWC_EQOS_FDUPLX_ONLY **Enabled:** DWC_EQOS_MMC_EN&&!DWC_EQOS_MMC_EN_TX_ALL&&!DWC_EQOS_FDUPLX_ONLY **Parameter Name:** DWC_EQOS_MMC_TXDEFRD_CNT_EN
Enable 16-bit Support	Enables 16-bit counter support for packets deferred before transmission in the half-duplex mode. **Values:** 0, 1 **Default Value:** 0 **Enabled:** DWC_EQOS_MMC_TXDEFRD_CNT_EN **Parameter Name:** DWC_EQOS_MMC_TXDEFRD_CNT_16BIT_EN
Enable MMC Tx Counter for Packets with Late Collision	Enables counter for packets aborted because of late collision in the half-duplex mode. **Values:** 0, 1 **Default Value:** DWC_EQOS_MMC_EN_TX_ALL&&!DWC_EQOS_FDUPLX_ONLY **Enabled:** DWC_EQOS_MMC_EN&&!DWC_EQOS_MMC_EN_TX_ALL&&!DWC_EQOS_FDUPLX_ONLY **Parameter Name:** DWC_EQOS_MMC_TXLATECOL_CNT_EN
Enable 16-bit Support	Enables 16-bit counter support for packets aborted because of late collision in the half-duplex mode. **Values:** 0, 1 **Default Value:** 0 **Enabled:** DWC_EQOS_MMC_TXLATECOL_CNT_EN **Parameter Name:** DWC_EQOS_MMC_TXLATECOL_CNT_16BIT_EN
Enable MMC Tx Counter for Packets with Excessive Collision	Enables counter for packets aborted because of excessive collision in the half-duplex mode. **Values:** 0, 1 **Default Value:** DWC_EQOS_MMC_EN_TX_ALL&&!DWC_EQOS_FDUPLX_ONLY **Enabled:** DWC_EQOS_MMC_EN&&!DWC_EQOS_MMC_EN_TX_ALL&&!DWC_EQOS_FDUPLX_ONLY **Parameter Name:** DWC_EQOS_MMC_TXEXSCOL_CNT_EN
Enable 16-bit Support	Enables 16-bit counter support for packets aborted because of excessive collision in the half-duplex mode. **Values:** 0, 1 **Default Value:** 0 **Enabled:** DWC_EQOS_MMC_TXEXSCOL_CNT_EN **Parameter Name:** DWC_EQOS_MMC_TXEXSCOL_CNT_16BIT_EN

Label	Description
Enable MMC Tx Counter for Packets with Carrier Error (Loss of Carrier/No Carrier)	Enables counter for packets aborted because of carrier errors in the half-duplex mode. **Values:** 0, 1 **Default Value:** DWC_EQOS_MMC_EN_TX_ALL&&!DWC_EQOS_FDUPLX_ONLY **Enabled:** DWC_EQOS_MMC_EN&&!DWC_EQOS_MMC_EN_TX_ALL&&!DWC_EQOS_FDUPLX_ONLY **Parameter Name:** DWC_EQOS_MMC_TXCARR_CNT_EN
Enable 16-bit Support	Enables 16-bit counter support for packets aborted because of carrier errors in the half-duplex mode. **Values:** 0, 1 **Default Value:** 0 **Enabled:** DWC_EQOS_MMC_TXCARR_CNT_EN **Parameter Name:** DWC_EQOS_MMC_TXCARR_CNT_16BIT_EN
Enable MMC Tx Octet Counter for Good Packets	Enables Octet counter for only good transmitted packets. **Values:** 0, 1 **Default Value:** DWC_EQOS_MMC_EN_TX_ALL **Enabled:** DWC_EQOS_MMC_EN&&!DWC_EQOS_MMC_EN_TX_ALL **Parameter Name:** DWC_EQOS_MMC_TXOCTG_CNT_EN
Enable MMC Tx Counter for Good Packets	Enables Frame counter for only good transmitted packets. **Values:** 0, 1 **Default Value:** DWC_EQOS_MMC_EN_TX_ALL **Enabled:** DWC_EQOS_MMC_EN&&!DWC_EQOS_MMC_EN_TX_ALL **Parameter Name:** DWC_EQOS_MMC_TXPKTG_CNT_EN
Enable 16-bit Support	Enables 16-bit Frame counter support for only good transmitted packets. **Values:** 0, 1 **Default Value:** 0 **Enabled:** DWC_EQOS_MMC_TXPKTG_CNT_EN **Parameter Name:** DWC_EQOS_MMC_TXPKTG_CNT_16BIT_EN
Enable MMC Tx Counter for Excessive Deferral Packets	Enables counter for packets aborted because of excessive deferral in the half-duplex mode. **Values:** 0, 1 **Default Value:** DWC_EQOS_MMC_EN_TX_ALL&&!DWC_EQOS_FDUPLX_ONLY **Enabled:** DWC_EQOS_MMC_EN&&!DWC_EQOS_MMC_EN_TX_ALL&&!DWC_EQOS_FDUPLX_ONLY **Parameter Name:** DWC_EQOS_MMC_TXEXSDEF_CNT_EN

Label	Description
Enable 16-bit Support	Enables 16-bit counter support for packets aborted because of excessive deferral in the half-duplex mode. **Values:** 0, 1 **Default Value:** 0 **Enabled:** DWC_EQOS_MMC_TXEXSDEF_CNT_EN **Parameter Name:** DWC_EQOS_MMC_TXEXSDEF_CNT_16BIT_EN
Enable MMC Tx Counter for Pause Packets	Enables packet counter for transmitted PAUSE packets. **Values:** 0, 1 **Default Value:** DWC_EQOS_MMC_EN_TX_ALL **Enabled:** DWC_EQOS_MMC_EN&&!DWC_EQOS_MMC_EN_TX_ALL **Parameter Name:** DWC_EQOS_MMC_TXPAUSE_CNT_EN
Enable 16-bit Support	Enables 16-bit packet counter support for transmitted PAUSE packets. **Values:** 0, 1 **Default Value:** 0 **Enabled:** DWC_EQOS_MMC_TXPAUSE_CNT_EN **Parameter Name:** DWC_EQOS_MMC_TXPAUSE_CNT_16BIT_EN
Enable MMC Tx Counter for VLAN Packets	Enables packet counter for transmitted VLAN-tagged packets. **Values:** 0, 1 **Default Value:** DWC_EQOS_MMC_EN_TX_ALL **Enabled:** DWC_EQOS_MMC_EN&&!DWC_EQOS_MMC_EN_TX_ALL **Parameter Name:** DWC_EQOS_MMC_TXVLAN_CNT_EN
Enable 16-bit Support	Enables 16-bit packet counter support for transmitted VLAN-tagged packets. **Values:** 0, 1 **Default Value:** 0 **Enabled:** DWC_EQOS_MMC_TXVLAN_CNT_EN **Parameter Name:** DWC_EQOS_MMC_TXVLAN_CNT_16BIT_EN
Enable MMC Tx Counter for Good Oversize Packets	Enables the counter for transmitted Oversize (size > 1518/1522/2000 bytes) packets. **Values:** 0, 1 **Default Value:** DWC_EQOS_MMC_EN_TX_ALL **Enabled:** DWC_EQOS_MMC_EN&&!DWC_EQOS_MMC_EN_TX_ALL **Parameter Name:** DWC_EQOS_MMC_TXOVERSZG_CNT_EN
Enable 16-bit Support	Enables the 16-bit counter support for transmitted Oversize (size > 1518/1522/2000 bytes) packets. **Values:** 0, 1 **Default Value:** 0 **Enabled:** DWC_EQOS_MMC_TXOVERSZG_CNT_EN **Parameter Name:** DWC_EQOS_MMC_TXOVERSZG_CNT_16BIT_EN

Label	Description
Enable MicroSecond Timer for TX LPI Assert	Enables the Micro Second Timer for Tx LPI Asserted **Values:** 0, 1 **Default Value:** DWC_EQOS_EEE_EN&&DWC_EQOS_MMC_EN_TX_ALL **Enabled:** DWC_EQOS_MMC_EN&&DWC_EQOS_EEE_EN&&!DWC_EQOS_MMC_EN_TX_ALL **Parameter Name:** DWC_EQOS_MMC_TX_LPI_MICROSEC_TIMER_EN
Enable 16-bit Support	Enables the 16 bit support for Micro Second Timer for Tx LPI Asserted **Values:** 0, 1 **Default Value:** 0 **Enabled:** DWC_EQOS_MMC_TX_LPI_MICROSEC_TIMER_EN **Parameter Name:** DWC_EQOS_MMC_TX_LPI_MICROSEC_TIMER_16BIT_EN
Enable Counter for counting the no. of Tx LPI Assertions	Enables the counter for Tx LPI assertion **Values:** 0, 1 **Default Value:** DWC_EQOS_EEE_EN&&DWC_EQOS_MMC_EN_TX_ALL **Enabled:** DWC_EQOS_MMC_EN&&DWC_EQOS_EEE_EN&&!DWC_EQOS_MMC_EN_TX_ALL **Parameter Name:** DWC_EQOS_MMC_TX_LPI_TRANSITION_CNT_EN
Enable 16-bit Support	Enables the 16 bit support for Counter for Tx LPI Assertions **Values:** 0, 1 **Default Value:** 0 **Enabled:** DWC_EQOS_MMC_TX_LPI_TRANSITION_CNT_EN **Parameter Name:** DWC_EQOS_MMC_TX_LPI_TRANSITION_CNT_16BIT_EN

15.15 Features / RMON Counters / RMON Receive Counters Parameters

Table 15-15 Features / RMON Counters / RMON Receive Counters Parameters

Label	Description
	RMON Receive Counters
Enable All MMC Rx Counters	Enables all receive packet counters. **Values:** 0, 1 **Default Value:** 0 **Enabled:** DWC_EQOS_MMC_EN **Parameter Name:** DWC_EQOS_MMC_EN_RX_ALL
Enable MMC Rx Counter for Good and Bad Packets	Enables Packet counter for received good and bad packets. **Values:** 0, 1 **Default Value:** DWC_EQOS_MMC_EN_RX_ALL **Enabled:** DWC_EQOS_MMC_EN&&!DWC_EQOS_MMC_EN_RX_ALL **Parameter Name:** DWC_EQOS_MMC_RXPKTGB_CNT_EN
Enable 16-bit Support	Enables 16-bit Packet counter support for received good and bad packets. **Values:** 0, 1 **Default Value:** 0 **Enabled:** DWC_EQOS_MMC_RXPKTGB_CNT_EN **Parameter Name:** DWC_EQOS_MMC_RXPKTGB_CNT_16BIT_EN
Enable MMC Rx Octet Counter for Good Packets	Enables Octet counter for received good packets. **Values:** 0, 1 **Default Value:** DWC_EQOS_MMC_EN_RX_ALL **Enabled:** DWC_EQOS_MMC_EN&&!DWC_EQOS_MMC_EN_RX_ALL **Parameter Name:** DWC_EQOS_MMC_RXOCTG_CNT_EN
Enable MMC Rx Octet Counter for Good and Bad Packets	Enables Octet counter for all received packets. **Values:** 0, 1 **Default Value:** DWC_EQOS_MMC_EN_RX_ALL **Enabled:** DWC_EQOS_MMC_EN&&!DWC_EQOS_MMC_EN_RX_ALL **Parameter Name:** DWC_EQOS_MMC_RXOCTGB_CNT_EN
Enable MMC Rx Counter for Good Broadcast Packets	Enables counter for received good broadcast packets. **Values:** 0, 1 **Default Value:** DWC_EQOS_MMC_EN_RX_ALL **Enabled:** DWC_EQOS_MMC_EN&&!DWC_EQOS_MMC_EN_RX_ALL **Parameter Name:** DWC_EQOS_MMC_RXBCASTG_CNT_EN
Enable 16-bit Support	Enables 16-bit counter support for received good broadcast packets. **Values:** 0, 1 **Default Value:** 0 **Enabled:** DWC_EQOS_MMC_RXBCASTG_CNT_EN **Parameter Name:** DWC_EQOS_MMC_RXBCASTG_CNT_16BIT_EN

Label	Description
Enable MMC Rx Counter for Good Multicast Packets	Enables counter for received good multicast packets. **Values:** 0, 1 **Default Value:** DWC_EQOS_MMC_EN_RX_ALL **Enabled:** DWC_EQOS_MMC_EN&&!DWC_EQOS_MMC_EN_RX_ALL **Parameter Name:** DWC_EQOS_MMC_RXMCASTG_CNT_EN
Enable 16-bit Support	Enables 16-bit counter support for received good multicast packets. **Values:** 0, 1 **Default Value:** 0 **Enabled:** DWC_EQOS_MMC_RXMCASTG_CNT_EN **Parameter Name:** DWC_EQOS_MMC_RXMCASTG_CNT_16BIT_EN
Enable MMC Rx Counter for Packets with CRC Error	Enables counter for packets received with CRC error. **Values:** 0, 1 **Default Value:** DWC_EQOS_MMC_EN_RX_ALL **Enabled:** DWC_EQOS_MMC_EN&&!DWC_EQOS_MMC_EN_RX_ALL **Parameter Name:** DWC_EQOS_MMC_RXCRCERR_CNT_EN
Enable 16-bit Support	Enables 16-bit counter support for packets received with CRC error. **Values:** 0, 1 **Default Value:** 0 **Enabled:** DWC_EQOS_MMC_RXCRCERR_CNT_EN **Parameter Name:** DWC_EQOS_MMC_RXCRCERR_CNT_16BIT_EN
Enable MMC Rx Counter for Packets with Alignment Error	Enables counter for packets received with Alignment Error (dribble) in the 100 Mbps mode. **Values:** 0, 1 **Default Value:** DWC_EQOS_MMC_EN_RX_ALL&&!DWC_EQOS_M1000_ONLY **Enabled:** DWC_EQOS_MMC_EN&&!DWC_EQOS_MMC_EN_RX_ALL&&!DWC_EQOS_M1000_ONLY **Parameter Name:** DWC_EQOS_MMC_RXALGNERR_CNT_EN
Enable 16-bit Support	Enables 16-bit counter support for packets received with Alignment Error (dribble) in the 100 Mbps mode. **Values:** 0, 1 **Default Value:** 0 **Enabled:** DWC_EQOS_MMC_RXALGNERR_CNT_EN **Parameter Name:** DWC_EQOS_MMC_RXALGNERR_CNT_16BIT_EN
Enable MMC Rx Counter for Runt Packets	Enables counter for received runt (size less than 64 bytes and CRC error) packets. **Values:** 0, 1 **Default Value:** DWC_EQOS_MMC_EN_RX_ALL **Enabled:** DWC_EQOS_MMC_EN&&!DWC_EQOS_MMC_EN_RX_ALL **Parameter Name:** DWC_EQOS_MMC_RXRUNTERR_CNT_EN

Label	Description
Enable 16-bit Support	Enables 16-bit counter support for received runt (size less than 64 bytes and CRC error) packets. **Values:** 0, 1 **Default Value:** 0 **Enabled:** DWC_EQOS_MMC_RXRUNTERR_CNT_EN **Parameter Name:** DWC_EQOS_MMC_RXRUNTERR_CNT_16BIT_EN
Enable MMC Rx Counter for Packets with Jabber Error	Enables counter for packets received with Jabber error. **Values:** 0, 1 **Default Value:** DWC_EQOS_MMC_EN_RX_ALL **Enabled:** DWC_EQOS_MMC_EN&&!DWC_EQOS_MMC_EN_RX_ALL **Parameter Name:** DWC_EQOS_MMC_RXJABERR_CNT_EN
Enable 16-bit Support	Enables 16-bit counter support for packets received with Jabber error. **Values:** 0, 1 **Default Value:** 0 **Enabled:** DWC_EQOS_MMC_RXJABERR_CNT_EN **Parameter Name:** DWC_EQOS_MMC_RXJABERR_CNT_16BIT_EN
Enable MMC Rx Counter for Good Undersize Packets	Enables counter for received under-size (size less than 64 bytes and no errors) packets. **Values:** 0, 1 **Default Value:** DWC_EQOS_MMC_EN_RX_ALL **Enabled:** DWC_EQOS_MMC_EN&&!DWC_EQOS_MMC_EN_RX_ALL **Parameter Name:** DWC_EQOS_MMC_RXUNDERSZG_CNT_EN
Enable 16-bit Support	Enables 16-bit counter support for received under-size (size less than 64 bytes and no errors) packets. **Values:** 0, 1 **Default Value:** 0 **Enabled:** DWC_EQOS_MMC_RXUNDERSZG_CNT_EN **Parameter Name:** DWC_EQOS_MMC_RXUNDERSZG_CNT_16BIT_EN
Enable MMC Rx Counter for Good Oversize Packets	Enables the Rx packet counter for oversized packets received without errors. Oversized packets have length greater than maxsize (1,518 or 1,522 bytes for VLAN tagged packets; 2000 bytes if enabled in bit 27 of MAC Configuration Register). **Values:** 0, 1 **Default Value:** DWC_EQOS_MMC_EN_RX_ALL **Enabled:** DWC_EQOS_MMC_EN&&!DWC_EQOS_MMC_EN_RX_ALL **Parameter Name:** DWC_EQOS_MMC_RXOVERSZG_CNT_EN
Enable 16-bit Support	Enables the 16-bit counter support for received Oversize (size > 1518/1522/2000 bytes) packets. **Values:** 0, 1 **Default Value:** 0 **Enabled:** DWC_EQOS_MMC_RXOVERSZG_CNT_EN **Parameter Name:** DWC_EQOS_MMC_RXOVERSZG_CNT_16BIT_EN

Label	Description
Enable MMC Rx Octet Counters for Various Packet Sizes	Enables the octet counters for all received packets of different sizes (64, 65-127, 128-255, 256-511, 512-1023, 1024-maxsize). **Values:** 0, 1 **Default Value:** DWC_EQOS_MMC_EN_RX_ALL **Enabled:** DWC_EQOS_MMC_EN&&!DWC_EQOS_MMC_EN_RX_ALL **Parameter Name:** DWC_EQOS_MMC_RXADDR_RNG_CNT_EN
Enable 16-bit Support	Enables the 16-bit octet counters for all received packets of different sizes (64, 65-127, 128-255, 256-511, 512-1023, 1024-maxsize). **Values:** 0, 1 **Default Value:** 0 **Enabled:** DWC_EQOS_MMC_RXADDR_RNG_CNT_EN **Parameter Name:** DWC_EQOS_MMC_RXADDR_RNG_CNT_16BIT_EN
Enable MMC Rx Counter for Good Unicast Packets	Enables counter for received good Unicast packets. **Values:** 0, 1 **Default Value:** DWC_EQOS_MMC_EN_RX_ALL **Enabled:** DWC_EQOS_MMC_EN&&!DWC_EQOS_MMC_EN_RX_ALL **Parameter Name:** DWC_EQOS_MMC_RXUCASTG_CNT_EN
Enable 16-bit Support	Enables 16-bit counter support for received good Unicast packets. **Values:** 0, 1 **Default Value:** 0 **Enabled:** DWC_EQOS_MMC_RXUCASTG_CNT_EN **Parameter Name:** DWC_EQOS_MMC_RXUCASTG_CNT_16BIT_EN
Enable MMC Rx Counter for Packets with Length Error	Enables counter for packets received with Length error. **Values:** 0, 1 **Default Value:** DWC_EQOS_MMC_EN_RX_ALL **Enabled:** DWC_EQOS_MMC_EN&&!DWC_EQOS_MMC_EN_RX_ALL **Parameter Name:** DWC_EQOS_MMC_RXLENERR_CNT_EN
Enable 16-bit Support	Enables 16-bit counter support for packets received with Length error. **Values:** 0, 1 **Default Value:** 0 **Enabled:** DWC_EQOS_MMC_RXLENERR_CNT_EN **Parameter Name:** DWC_EQOS_MMC_RXLENERR_CNT_16BIT_EN
Enable MMC Rx Counter for Out-of-Range Packets	Enables Out of Range (Length Field between 1500 & 1536 decimal) packet counter. **Values:** 0, 1 **Default Value:** DWC_EQOS_MMC_EN_RX_ALL **Enabled:** DWC_EQOS_MMC_EN&&!DWC_EQOS_MMC_EN_RX_ALL **Parameter Name:** DWC_EQOS_MMC_RXOUTOFRNG_TYP_CNT_EN

Label	Description
Enable 16-bit Support	Enables 16-bit Out of Range (Length Field between 1500 & 1536 decimal) packet counter. **Values:** 0, 1 **Default Value:** 0 **Enabled:** DWC_EQOS_MMC_RXOUTOFRNG_TYP_CNT_EN **Parameter Name:** DWC_EQOS_MMC_RXOUTOFRNG_TYP_CNT_16BIT_EN
Enable MMC Rx Counter for Pause Packets	Enables counter for received Pause packets. **Values:** 0, 1 **Default Value:** DWC_EQOS_MMC_EN_RX_ALL **Enabled:** DWC_EQOS_MMC_EN&&!DWC_EQOS_MMC_EN_RX_ALL **Parameter Name:** DWC_EQOS_MMC_RXPAUSEPKT_CNT_EN
Enable 16-bit Support	Enables 16-bit counter support for received Pause packets. **Values:** 0, 1 **Default Value:** 0 **Enabled:** DWC_EQOS_MMC_RXPAUSEPKT_CNT_EN **Parameter Name:** DWC_EQOS_MMC_RXPAUSEPKT_CNT_16BIT_EN
Enable MMC Rx Counter for Packets with FIFO Overflow Error	Enables counter to count the number of times Rx MTL FIFO overflowed. **Values:** 0, 1 **Default Value:** DWC_EQOS_MMC_EN_RX_ALL&&!DWC_EQOS_CORE **Enabled:** DWC_EQOS_MMC_EN&&!DWC_EQOS_MMC_EN_RX_ALL&&!DWC_EQOS_COR E **Parameter Name:** DWC_EQOS_MMC_RXFIFOOVFL_CNT_EN
Enable 16-bit Support	Enables 16-bit counter support to count the number of times Rx MTL FIFO overflowed. **Values:** 0, 1 **Default Value:** 0 **Enabled:** DWC_EQOS_MMC_RXFIFOOVFL_CNT_EN **Parameter Name:** DWC_EQOS_MMC_RXFIFOOVFL_CNT_16BIT_EN
Enable MMC Rx Counter for VLAN Packets	Enables counter for received good or bad VLAN packets. **Values:** 0, 1 **Default Value:** DWC_EQOS_MMC_EN_RX_ALL **Enabled:** DWC_EQOS_MMC_EN&&!DWC_EQOS_MMC_EN_RX_ALL **Parameter Name:** DWC_EQOS_MMC_RXVLANPKTGB_CNT_EN
Enable 16-bit Support	Enables 16-bit counter support for received good or bad VLAN packets. **Values:** 0, 1 **Default Value:** 0 **Enabled:** DWC_EQOS_MMC_RXVLANPKTGB_CNT_EN **Parameter Name:** DWC_EQOS_MMC_RXVLANPKTGB_CNT_16BIT_EN

Label	Description
Enable MMC Rx Counter for Packets with Watchdog Error	Enables counter for packets with Watchdog timeout error. **Values:** 0, 1 **Default Value:** DWC_EQOS_MMC_EN_RX_ALL **Enabled:** DWC_EQOS_MMC_EN&&!DWC_EQOS_MMC_EN_RX_ALL **Parameter Name:** DWC_EQOS_MMC_RXWDGERR_CNT_EN
Enable 16-bit Support	Enables 16-bit counter support for packets with Watchdog timeout error. **Values:** 0, 1 **Default Value:** 0 **Enabled:** DWC_EQOS_MMC_RXWDGERR_CNT_EN **Parameter Name:** DWC_EQOS_MMC_RXWDGERR_CNT_16BIT_EN
Enable MMC Rx Counter for Packets with Receive Error	Enables counter for packets with Receive error. **Values:** 0, 1 **Default Value:** DWC_EQOS_MMC_EN_RX_ALL **Enabled:** DWC_EQOS_MMC_EN&&!DWC_EQOS_MMC_EN_RX_ALL **Parameter Name:** DWC_EQOS_MMC_RXRCVERR_CNT_EN
Enable 16-bit Support	Enables 16-bit counter support for packets with Receive error. **Values:** 0, 1 **Default Value:** 0 **Enabled:** DWC_EQOS_MMC_RXRCVERR_CNT_EN **Parameter Name:** DWC_EQOS_MMC_RXRCVERR_CNT_16BIT_EN
Enable MMC Rx Counter for all good control Packets	Enables counter for all types of good control packets. **Values:** 0, 1 **Default Value:** DWC_EQOS_MMC_EN_RX_ALL **Enabled:** DWC_EQOS_MMC_EN&&!DWC_EQOS_MMC_EN_RX_ALL **Parameter Name:** DWC_EQOS_MMC_RXCTRLPKT_CNT_EN
Enable 16-bit Support	Enables 16-bit counter support for types of good control packets. **Values:** 0, 1 **Default Value:** 0 **Enabled:** DWC_EQOS_MMC_RXCTRLPKT_CNT_EN **Parameter Name:** DWC_EQOS_MMC_RXCTRLPKT_CNT_16BIT_EN
Enable MicroSecond Timer for RX LPI Assert	Enables the Micro Second Timer for Rx LPI Asserted **Values:** 0, 1 **Default Value:** DWC_EQOS_EEE_EN&&DWC_EQOS_MMC_EN_RX_ALL **Enabled:** DWC_EQOS_MMC_EN&&DWC_EQOS_EEE_EN&&!DWC_EQOS_MMC_EN_RX_ALL **Parameter Name:** DWC_EQOS_MMC_RX_LPI_MICROSEC_TIMER_EN

Label	Description
Enable 16-bit Support	Enables the 16 bit support for Micro Second Timer for Rx LPI Asserted **Values:** 0, 1 **Default Value:** 0 **Enabled:** DWC_EQOS_MMC_RX_LPI_MICROSEC_TIMER_EN **Parameter Name:** DWC_EQOS_MMC_RX_LPI_MICROSEC_TIMER_16BIT_EN
Enable Counter for counting the no. of Rx LPI Assertions	Enables the counter for Rx LPI assertion **Values:** 0, 1 **Default Value:** DWC_EQOS_EEE_EN&&DWC_EQOS_MMC_EN_RX_ALL **Enabled:** DWC_EQOS_MMC_EN&&DWC_EQOS_EEE_EN&&!DWC_EQOS_MMC_EN_RX_ALL **Parameter Name:** DWC_EQOS_MMC_RX_LPI_TRANSITION_CNT_EN
Enable 16-bit Support	Enables the 16 bit support for Counter for Rx LPI Assertions **Values:** 0, 1 **Default Value:** 0 **Enabled:** DWC_EQOS_MMC_RX_LPI_TRANSITION_CNT_EN **Parameter Name:** DWC_EQOS_MMC_RX_LPI_TRANSITION_CNT_16BIT_EN

15.16 Features / RMON Counters / RMON IPC Receive Counters Parameters

Table 15-16 Features / RMON Counters / RMON IPC Receive Counters Parameters

Label	Description
	RMON IPC Receive Counters
Enable All MMC IPC Rx Counters	Enables all Rx IPC counters. **Values:** 0, 1 **Default Value:** 0 **Enabled:** DWC_EQOS_MMC_IPC_EN **Parameter Name:** DWC_EQOS_MMC_IPC_EN_RX_ALL
Enable MMC Counter for Good IPv4 Packets	Enables MMC Counter for IPv4 packets with no checksum errors. **Values:** 0, 1 **Default Value:** DWC_EQOS_MMC_IPC_EN_RX_ALL **Enabled:** DWC_EQOS_MMC_IPC_EN&&!DWC_EQOS_MMC_IPC_EN_RX_ALL **Parameter Name:** DWC_EQOS_MMC_RX_IPV4_GD_PKTS_CNT_EN
Enable 16-bit Support	Enables 16-bit MMC Counter for IPv4 packets with no checksum errors. **Values:** 0, 1 **Default Value:** 0 **Enabled:** DWC_EQOS_MMC_RX_IPV4_GD_PKTS_CNT_EN **Parameter Name:** DWC_EQOS_MMC_RX_IPV4_GD_PKTS_CNT_16BIT_EN
Enable MMC Counter for IPv4 Packets with Header Error	Enables MMC Counter for IPv4 packets with checksum errors. **Values:** 0, 1 **Default Value:** DWC_EQOS_MMC_IPC_EN_RX_ALL **Enabled:** DWC_EQOS_MMC_IPC_EN&&!DWC_EQOS_MMC_IPC_EN_RX_ALL **Parameter Name:** DWC_EQOS_MMC_RX_IPV4_HDRERR_PKTS_CNT_EN
Enable 16-bit Support	Enables 16-bit MMC Counter for IPv4 packets with checksum errors. **Values:** 0, 1 **Default Value:** 0 **Enabled:** DWC_EQOS_MMC_RX_IPV4_HDRERR_PKTS_CNT_EN **Parameter Name:** DWC_EQOS_MMC_RX_IPV4_HDRERR_PKTS_CNT_16BIT_EN
Enable MMC Counter for IPv4 Packets without TCP, UDP, and ICMP	Enables MMC Counter for IPv4 packets which do not have TCP, UDP, or ICMP. **Values:** 0, 1 **Default Value:** DWC_EQOS_MMC_IPC_EN_RX_ALL **Enabled:** DWC_EQOS_MMC_IPC_EN&&!DWC_EQOS_MMC_IPC_EN_RX_ALL **Parameter Name:** DWC_EQOS_MMC_RX_IPV4_NOPAY_PKTS_CNT_EN
Enable 16-bit Support	Enables 16-bit MMC Counter for IPv4 packets which do not have TCP, UDP, or ICMP. **Values:** 0, 1 **Default Value:** 0 **Enabled:** DWC_EQOS_MMC_RX_IPV4_NOPAY_PKTS_CNT_EN **Parameter Name:** DWC_EQOS_MMC_RX_IPV4_NOPAY_PKTS_CNT_16BIT_EN

Label	Description
Enable MMC Counter for Fragmented IPv4 Packets	Enables Rx frame counter for all IPv4 packets received with fragments. **Values:** 0, 1 **Default Value:** DWC_EQOS_MMC_IPC_EN_RX_ALL **Enabled:** DWC_EQOS_MMC_IPC_EN&&!DWC_EQOS_MMC_IPC_EN_RX_ALL **Parameter Name:** DWC_EQOS_MMC_RX_IPV4_FRAG_PKTS_CNT_EN
Enable 16-bit Support	Enables 16-bit MMC Counter for IPv4 packets received with fragments. **Values:** 0, 1 **Default Value:** 0 **Enabled:** DWC_EQOS_MMC_RX_IPV4_FRAG_PKTS_CNT_EN **Parameter Name:** DWC_EQOS_MMC_RX_IPV4_FRAG_PKTS_CNT_16BIT_EN
Enable MMC Counter for IPv4 UDP Packets with Checksum Disabled	Enables MMC Counter for IPv4 UDP packets with checksum disabled. **Values:** 0, 1 **Default Value:** DWC_EQOS_MMC_IPC_EN_RX_ALL **Enabled:** DWC_EQOS_MMC_IPC_EN&&!DWC_EQOS_MMC_IPC_EN_RX_ALL **Parameter Name:** DWC_EQOS_MMC_RX_IPV4_UDSBL_PKTS_CNT_EN
Enable 16-bit Support	Enables 16-bit MMC Counter for IPv4 UDP packets with checksum disabled. **Values:** 0, 1 **Default Value:** 0 **Enabled:** DWC_EQOS_MMC_RX_IPV4_UDSBL_PKTS_CNT_EN **Parameter Name:** DWC_EQOS_MMC_RX_IPV4_UDSBL_PKTS_CNT_16BIT_EN
Enable MMC Counter for Good IPv6 Packets	Enables MMC Counter for IPv6 packets that do not have checksum-related errors. **Values:** 0, 1 **Default Value:** DWC_EQOS_MMC_IPC_EN_RX_ALL **Enabled:** DWC_EQOS_MMC_IPC_EN&&!DWC_EQOS_MMC_IPC_EN_RX_ALL **Parameter Name:** DWC_EQOS_MMC_RX_IPV6_GD_PKTS_CNT_EN
Enable 16-bit Support	Enables 16-bit MMC Counter for IPv6 packets that do not have checksum-related errors. **Values:** 0, 1 **Default Value:** 0 **Enabled:** DWC_EQOS_MMC_RX_IPV6_GD_PKTS_CNT_EN **Parameter Name:** DWC_EQOS_MMC_RX_IPV6_GD_PKTS_CNT_16BIT_EN
Enable MMC Counter for IPv6 Packets with Header Error	Enables MMC Counter for IPv6 packets received with header error. **Values:** 0, 1 **Default Value:** DWC_EQOS_MMC_IPC_EN_RX_ALL **Enabled:** DWC_EQOS_MMC_IPC_EN&&!DWC_EQOS_MMC_IPC_EN_RX_ALL **Parameter Name:** DWC_EQOS_MMC_RX_IPV6_HDRERR_PKTS_CNT_EN

Label	Description
Enable 16-bit Support	Enables 16-bit MMC Counter for IPv6 packets received with header error. **Values:** 0, 1 **Default Value:** 0 **Enabled:** DWC_EQOS_MMC_RX_IPV6_HDRERR_PKTS_CNT_EN **Parameter Name:** DWC_EQOS_MMC_RX_IPV6_HDRERR_PKTS_CNT_16BIT_EN
Enable MMC Counter for IPv6 Packets without TCP, UDP, and ICMP	Enables MMC Counter for IPv6 packets not having TCP, UDP, or ICMP. **Values:** 0, 1 **Default Value:** DWC_EQOS_MMC_IPC_EN_RX_ALL **Enabled:** DWC_EQOS_MMC_IPC_EN&&!DWC_EQOS_MMC_IPC_EN_RX_ALL **Parameter Name:** DWC_EQOS_MMC_RX_IPV6_NOPAY_PKTS_CNT_EN
Enable 16-bit Support	Enables 16-bit MMC Counter for IPv6 packets not having TCP, UDP, or ICMP. **Values:** 0, 1 **Default Value:** 0 **Enabled:** DWC_EQOS_MMC_RX_IPV6_NOPAY_PKTS_CNT_EN **Parameter Name:** DWC_EQOS_MMC_RX_IPV6_NOPAY_PKTS_CNT_16BIT_EN
Enable MMC Counter for UDP Packets without Error	Enables MMC Counter for packets containing UDP with no errors. **Values:** 0, 1 **Default Value:** DWC_EQOS_MMC_IPC_EN_RX_ALL **Enabled:** DWC_EQOS_MMC_IPC_EN&&!DWC_EQOS_MMC_IPC_EN_RX_ALL **Parameter Name:** DWC_EQOS_MMC_RX_UDP_GD_PKTS_CNT_EN
Enable 16-bit Support	Enables 16-bit MMC Counter for packets containing UDP with no errors. **Values:** 0, 1 **Default Value:** 0 **Enabled:** DWC_EQOS_MMC_RX_UDP_GD_PKTS_CNT_EN **Parameter Name:** DWC_EQOS_MMC_RX_UDP_GD_PKTS_CNT_16BIT_EN
Enable MMC Counter for UDP Packets with Error	Enables MMC Counter for packets containing UDP with errors. **Values:** 0, 1 **Default Value:** DWC_EQOS_MMC_IPC_EN_RX_ALL **Enabled:** DWC_EQOS_MMC_IPC_EN&&!DWC_EQOS_MMC_IPC_EN_RX_ALL **Parameter Name:** DWC_EQOS_MMC_RX_UDP_ERR_PKTS_CNT_EN
Enable 16-bit Support	Enables 16-bit MMC Counter for packets containing UDP with errors. **Values:** 0, 1 **Default Value:** 0 **Enabled:** DWC_EQOS_MMC_RX_UDP_ERR_PKTS_CNT_EN **Parameter Name:** DWC_EQOS_MMC_RX_UDP_ERR_PKTS_CNT_16BIT_EN
Enable MMC Counter for TCP Packets without Error	Enables MMC Counter for packets containing TCP with no errors. **Values:** 0, 1 **Default Value:** DWC_EQOS_MMC_IPC_EN_RX_ALL **Enabled:** DWC_EQOS_MMC_IPC_EN&&!DWC_EQOS_MMC_IPC_EN_RX_ALL **Parameter Name:** DWC_EQOS_MMC_RX_TCP_GD_PKTS_CNT_EN

Label	Description
Enable 16-bit Support	Enables 16-bit MMC Counter for packets containing TCP with no errors. **Values:** 0, 1 **Default Value:** 0 **Enabled:** DWC_EQOS_MMC_RX_TCP_GD_PKTS_CNT_EN **Parameter Name:** DWC_EQOS_MMC_RX_TCP_GD_PKTS_CNT_16BIT_EN
Enable MMC Counter for TCP Packets with Error	Enables MMC Counter for packets containing TCP with errors. **Values:** 0, 1 **Default Value:** DWC_EQOS_MMC_IPC_EN_RX_ALL **Enabled:** DWC_EQOS_MMC_IPC_EN&&!DWC_EQOS_MMC_IPC_EN_RX_ALL **Parameter Name:** DWC_EQOS_MMC_RX_TCP_ERR_PKTS_CNT_EN
Enable 16-bit Support	Enables 16-bit MMC Counter for packets containing TCP with errors. **Values:** 0, 1 **Default Value:** 0 **Enabled:** DWC_EQOS_MMC_RX_TCP_ERR_PKTS_CNT_EN **Parameter Name:** DWC_EQOS_MMC_RX_TCP_ERR_PKTS_CNT_16BIT_EN
Enable MMC Counter for ICMP Packets without Error	Enables MMC Counter for packets containing ICMP with no errors. **Values:** 0, 1 **Default Value:** DWC_EQOS_MMC_IPC_EN_RX_ALL **Enabled:** DWC_EQOS_MMC_IPC_EN&&!DWC_EQOS_MMC_IPC_EN_RX_ALL **Parameter Name:** DWC_EQOS_MMC_RX_ICMP_GD_PKTS_CNT_EN
Enable 16-bit Support	Enables 16-bit MMC Counter for packets containing ICMP with no errors. **Values:** 0, 1 **Default Value:** 0 **Enabled:** DWC_EQOS_MMC_RX_ICMP_GD_PKTS_CNT_EN **Parameter Name:** DWC_EQOS_MMC_RX_ICMP_GD_PKTS_CNT_16BIT_EN
Enable MMC Counter for ICMP Packets with Error	Enables MMC Counter for packets containing ICMP with errors. **Values:** 0, 1 **Default Value:** DWC_EQOS_MMC_IPC_EN_RX_ALL **Enabled:** DWC_EQOS_MMC_IPC_EN&&!DWC_EQOS_MMC_IPC_EN_RX_ALL **Parameter Name:** DWC_EQOS_MMC_RX_ICMP_ERR_PKTS_CNT_EN
Enable 16-bit Support	Enables 16-bit MMC Counter for packets containing ICMP with errors. **Values:** 0, 1 **Default Value:** 0 **Enabled:** DWC_EQOS_MMC_RX_ICMP_ERR_PKTS_CNT_EN **Parameter Name:** DWC_EQOS_MMC_RX_ICMP_ERR_PKTS_CNT_16BIT_EN

Label	Description
Enable MMC Octet Counter for IPv4 Packets without Checksum Error	Enables MMC Counter counting IPv4 datagrams bytes when there are no checksum errors. **Values:** 0, 1 **Default Value:** DWC_EQOS_MMC_IPC_EN_RX_ALL **Enabled:** DWC_EQOS_MMC_IPC_EN&&!DWC_EQOS_MMC_IPC_EN_RX_ALL **Parameter Name:** DWC_EQOS_MMC_RX_IPV4_GD_OCTET_CNT_EN
Enable MMC Octet Counter for IPv4 Packets with Header Checksum Error	Enables an Rx octet counter for all IPv4 datagrams received with a header error. **Values:** 0, 1 **Default Value:** DWC_EQOS_MMC_IPC_EN_RX_ALL **Enabled:** DWC_EQOS_MMC_IPC_EN&&!DWC_EQOS_MMC_IPC_EN_RX_ALL **Parameter Name:** DWC_EQOS_MMC_RX_IPV4_HDRERR_OCTET_CNT_EN
Enable MMC Octet Counter for IPv4 Packets without UDP, TCP, and ICMP	Enables MMC Counter counting bytes of IPv4 datagrams with no UDP, TCP, or ICMP. **Values:** 0, 1 **Default Value:** DWC_EQOS_MMC_IPC_EN_RX_ALL **Enabled:** DWC_EQOS_MMC_IPC_EN&&!DWC_EQOS_MMC_IPC_EN_RX_ALL **Parameter Name:** DWC_EQOS_MMC_RX_IPV4_NOPAY_OCTET_CNT_EN
Enable MMC Octet Counter for Fragmented IPv4 Packets	Enables an Rx octet counter for all IPv4 datagrams received with fragments. **Values:** 0, 1 **Default Value:** DWC_EQOS_MMC_IPC_EN_RX_ALL **Enabled:** DWC_EQOS_MMC_IPC_EN&&!DWC_EQOS_MMC_IPC_EN_RX_ALL **Parameter Name:** DWC_EQOS_MMC_RX_IPV4_FRAG_OCTET_CNT_EN
Enable MMC Octet Counter for IPv4 UDP Packets with Checksum Disabled	Enables MMC Counter counting bytes in IPv4 UDP datagram which has checksum disabled. This includes only UDP bytes and not the IPv4 header. **Values:** 0, 1 **Default Value:** DWC_EQOS_MMC_IPC_EN_RX_ALL **Enabled:** DWC_EQOS_MMC_IPC_EN&&!DWC_EQOS_MMC_IPC_EN_RX_ALL **Parameter Name:** DWC_EQOS_MMC_RX_IPV4_UDSBL_OCTET_CNT_EN
Enable MMC Octet Counter for IPv6 Packets without Checksum Error	Enables MMC Counter counting bytes in IPv6 datagram having no checksum errors. **Values:** 0, 1 **Default Value:** DWC_EQOS_MMC_IPC_EN_RX_ALL **Enabled:** DWC_EQOS_MMC_IPC_EN&&!DWC_EQOS_MMC_IPC_EN_RX_ALL **Parameter Name:** DWC_EQOS_MMC_RX_IPV6_GD_OCTET_CNT_EN
Enable MMC Octet Counter for IPv6 Packets with Header Error	Enables MMC Counter counting bytes in IPv6 datagram having header errors. **Values:** 0, 1 **Default Value:** DWC_EQOS_MMC_IPC_EN_RX_ALL **Enabled:** DWC_EQOS_MMC_IPC_EN&&!DWC_EQOS_MMC_IPC_EN_RX_ALL **Parameter Name:** DWC_EQOS_MMC_RX_IPV6_HDRERR_OCTET_CNT_EN

Label	Description
Enable MMC Octet Counter for IPv6 Packets without TCP, UDP, and ICMP	Enables MMC Counter counting bytes in IPv6 datagram not having TCP, UDP, or ICMP. **Values:** 0, 1 **Default Value:** DWC_EQOS_MMC_IPC_EN_RX_ALL **Enabled:** DWC_EQOS_MMC_IPC_EN&&!DWC_EQOS_MMC_IPC_EN_RX_ALL **Parameter Name:** DWC_EQOS_MMC_RX_IPV6_NOPAY_OCTET_CNT_EN
Enable MMC Octet Counter for UDP Packets without Checksum Error	Enables MMC Counter counting bytes in UDP which do not have checksum errors. **Values:** 0, 1 **Default Value:** DWC_EQOS_MMC_IPC_EN_RX_ALL **Enabled:** DWC_EQOS_MMC_IPC_EN&&!DWC_EQOS_MMC_IPC_EN_RX_ALL **Parameter Name:** DWC_EQOS_MMC_RX_UDP_GD_OCTET_CNT_EN
Enable MMC Octet Counter for UDP Packets with Checksum Error	Enables MMC Counter counting bytes in UDP which have checksum errors. **Values:** 0, 1 **Default Value:** DWC_EQOS_MMC_IPC_EN_RX_ALL **Enabled:** DWC_EQOS_MMC_IPC_EN&&!DWC_EQOS_MMC_IPC_EN_RX_ALL **Parameter Name:** DWC_EQOS_MMC_RX_UDP_ERR_OCTET_CNT_EN
Enable MMC Octet Counter for TCP Packets without Checksum Error	Enables MMC Counter counting bytes in TCP which do not have checksum errors. **Values:** 0, 1 **Default Value:** DWC_EQOS_MMC_IPC_EN_RX_ALL **Enabled:** DWC_EQOS_MMC_IPC_EN&&!DWC_EQOS_MMC_IPC_EN_RX_ALL **Parameter Name:** DWC_EQOS_MMC_RX_TCP_GD_OCTET_CNT_EN
Enable MMC Octet Counter for TCP Packets with Checksum Error	Enables MMC Counter counting bytes in TCP which have checksum errors. **Values:** 0, 1 **Default Value:** DWC_EQOS_MMC_IPC_EN_RX_ALL **Enabled:** DWC_EQOS_MMC_IPC_EN&&!DWC_EQOS_MMC_IPC_EN_RX_ALL **Parameter Name:** DWC_EQOS_MMC_RX_TCP_ERR_OCTET_CNT_EN
Enable MMC Octet Counter for ICMP Packets without Checksum Error	Enables MMC Counter counting bytes in ICMP which do not have checksum errors. **Values:** 0, 1 **Default Value:** DWC_EQOS_MMC_IPC_EN_RX_ALL **Enabled:** DWC_EQOS_MMC_IPC_EN&&!DWC_EQOS_MMC_IPC_EN_RX_ALL **Parameter Name:** DWC_EQOS_MMC_RX_ICMP_GD_OCTET_CNT_EN
Enable MMC Octet Counter for ICMP Packets with Checksum Error	Enables MMC Counter counting bytes in ICMP which have checksum errors. **Values:** 0, 1 **Default Value:** DWC_EQOS_MMC_IPC_EN_RX_ALL **Enabled:** DWC_EQOS_MMC_IPC_EN&&!DWC_EQOS_MMC_IPC_EN_RX_ALL **Parameter Name:** DWC_EQOS_MMC_RX_ICMP_ERR_OCTET_CNT_EN

15.17 Features / Automotive Safety features Parameters

Table 15-17 Features / Automotive Safety features Parameters

Label	Description
Enable All Automotive Safety Features	This Selects All the Automotive Safety Features viz. SECDED Protection for External Memories, On-chip Data Path Parity Protection, FSM State Parity and FSM Timeout Protection, Application and CSR Interface Timeout Protection. **Values:** 0, 1 **Default Value:** 0 **Enabled:** (DWC_EQOS_SYS>0)&&<DWC-Ethernet-QOS-ASP feature authorized> **Parameter Name:** DWC_EQOS_ASP_ALL
Enable SECDED ECC protection for external memories	Adds Error Correcting code(SECDED) for external memory interface **Values:** 0, 1 **Default Value:** DWC_EQOS_ASP_ALL **Enabled:** (DWC_EQOS_SYS>0)&&(!DWC_EQOS_ASP_ALL)&&<DWC-Ethernet-QOS-ASP feature authorized> **Parameter Name:** DWC_EQOS_ASP_ECC
ECC Code Width for Tx FIFO	This is the ECC Code width for Tx FIFO memory, this along with MTL FIFO Data Width determine Data width of the MTL Transmit FIFO when SECDED ECC protection for external memories feature is selected. **Values:** -2147483648, ..., 2147483647 **Default Value:** DWC_EQOS_ASP_ECC==1 ? (DWC_EQOS_DATAWIDTH==32 ? ((DWC_EQOS_TX_FIFO_PTR_WIDTH+35) <= 57 ? 7 : (DWC_EQOS_TX_FIFO_PTR_WIDTH+35) <= 120 ? 8 : 9) : DWC_EQOS_DATAWIDTH==64 ? ((DWC_EQOS_TX_FIFO_PTR_WIDTH+68) <= 120 ? 8 : 9) : (DWC_EQOS_TX_FIFO_PTR_WIDTH+133) <= 247 ? 9 : 10) : 0 **Enabled:** DWC_EQOS_SYS>0 **Parameter Name:** DWC_EQOS_TX_ECW
ECC Code Width for Rx FIFO	This is the ECC Code width for Rx FIFO memory, this along with MTL FIFO Data Width determine Data width of the MTL Receive FIFO when SECDED ECC protection for external memories feature is selected. **Values:** -2147483648, ..., 2147483647 **Default Value:** DWC_EQOS_ASP_ECC==1 ? (DWC_EQOS_DATAWIDTH==32 ? ((DWC_EQOS_RX_FIFO_PTR_WIDTH+35) <= 57 ? 7 : (DWC_EQOS_RX_FIFO_PTR_WIDTH+35) <= 120 ? 8 : 9) : DWC_EQOS_DATAWIDTH==64 ? ((DWC_EQOS_RX_FIFO_PTR_WIDTH+68) <= 120 ? 8 : 9) : (DWC_EQOS_RX_FIFO_PTR_WIDTH+133) <= 247 ? 9 : 10) : 0 **Enabled:** DWC_EQOS_SYS>0 **Parameter Name:** DWC_EQOS_RX_ECW
ECC Code Width for Rx Parser Memory	ECC Code width for Rx Parser memory **Values:** -2147483648, ..., 2147483647 **Default Value:** (DWC_EQOS_FRP_EN&&DWC_EQOS_ASP_ECC)==1 ? 8 : 0 **Enabled:** DWC_EQOS_FRP_EN **Parameter Name:** DWC_EQOS_RXP_ECW

Label	Description
ECC Code Width for GCL Memory	ECC Code width for GCL memory **Values:** -2147483648, ..., 2147483647 **Default Value:** (DWC_EQOS_AV_EST&&DWC_EQOS_ASP_ECC)==1 ? (((DWC_EQOS_EMW+DWC_EQOS_EMAW+1) <= 26) ? 6 : 7) : 0 **Enabled:** DWC_EQOS_AV_EST **Parameter Name:** DWC_EQOS_EST_ECW
ECC Code Width for TSO Memory	ECC Code width for TSO memory **Values:** -2147483648, ..., 2147483647 **Default Value:** (DWC_EQOS_TSO_MEM_EN&&DWC_EQOS_ASP_ECC)==1 ? ((((DWC_EQOS_DATAWIDTH+DWC_EQOS_TSO_MEM_ADDR_WIDTH) <= 57) ? 7 : ((DWC_EQOS_DATAWIDTH+DWC_EQOS_TSO_MEM_ADDR_WIDTH) <= 120) ? 8 : 9)) : 0 **Enabled:** DWC_EQOS_TSO_MEM_EN **Parameter Name:** DWC_EQOS_TSO_ECW
Enable Parity ports on External Application interface	This selects the Parity ports on External Application Interface enabling Interface Parity Checking. **Values:** 0, 1 **Default Value:** 0 **Enabled:** (DWC_EQOS_ASP_ALL)&&<DWC-Ethernet-QOS-ASP feature authorized> **Parameter Name:** DWC_EQOS_ASP_PPE

16 Signal Descriptions

This chapter details all possible I/O signals in the core. For configurable IP titles, your actual configuration might not contain all of these signals.

Inputs are on the left of the signal diagrams; outputs are on the right.

Attention: For configurable IP titles, do not use this document to determine the exact I/O footprint of the core. It is for reference purposes only.

When you configure the core in coreConsultant, you must access the I/O signals for your actual configuration at workspace/report/IO.html or workspace/report/IO.xml after you have completed the report creation activity. That report comes from the exact same source as this chapter but removes all the I/O signals that are not in your actual configuration. This does not apply to non-configurable IP titles. In addition, all parameter expressions are evaluated to actual values. Therefore, the widths might change depending on your actual configuration.

Some expressions might refer to TCL functions or procedures (sometimes identified as **<functionof>**) that coreConsultant uses to make calculations. The exact formula used by these TCL functions is not provided in this chapter. However, when you configure the core in coreConsultant, all TCL functions and parameters are evaluated completely; and the resulting values are displayed where appropriate in the coreConsultant GUI reports.

In addition to describing the function of each signal, the signal descriptions in this chapter include the following information:

- **Active State:** Indicates whether the signal is active high or active low. When a signal is not intended to be used in a particular application, then this signal needs to be tied or driven to the in-active state (opposite of the active state).
- **Registered:** Indicates whether or not the signal is registered directly inside the IP boundary without intervening logic (excluding simple buffers). A value of *No* does not imply that the signal is not synchronous, only that there is some combinatorial logic between the signal's origin or destination register and the boundary of the controller. A value of N/A indicates that this information is not provided for this IP title.
- **Synchronous to:** Indicates which clock(s) in the IP sample this input (drive for an output) when considering all possible configurations. A particular configuration might not have all of the clocks listed. This clock might not be the same as the clock that your application logic should use to clock (sample/drive) this pin. For more details, consult the clock section in the databook.
- **Exists:** Name of configuration parameter that populates this signal in your configuration.

- **Validated by:** Assertion or de-assertion of signal(s) that validates the signal being described.

The I/O signals are grouped as follows:

16.1 MII Clocks and Reset Signals

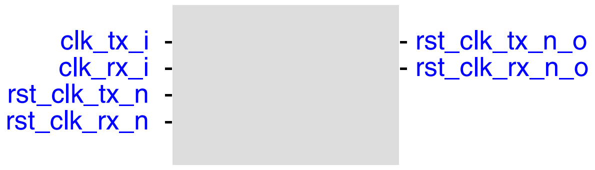


Table 16-1 MII Clocks and Reset Signals

Port Name	I/O	Description
clk_tx_i	I	MAC Transmitter Clock. The external PHY or oscillator provides this transmission clock. This is 312.5 MHz in 2.5 Gbps mode, 125 MHz in 1 Gbps mode, 25 MHz in 100 Mbps mode, 2.5 MHz in 10 Mbps mode. All transmission signals generated by the MAC are synchronous to this clock. This clock is required for all PHY interfaces. **Exists:** Always **Power Domain:** MAC_ON **Active State:** N/A **Synchronous to:** N/A **Registered:** N/A
clk_rx_i	I	MAC Receiver Clock. The external PHY provides this receive clock for RGMII, GMII, MII, and RMII interfaces. This clock is 312.5 MHz in 2.5 Gbps mode, 125 MHz in 1 Gbps mode, 25 MHz in 100 Mbps mode, 2.5 MHz in 10 Mbps mode. All RGMII, GMII, and MII receive signals received by the MAC are synchronous to this clock. This clock input is required for all PHY interfaces. **Exists:** Always **Power Domain:** MAC_ON **Active State:** N/A **Synchronous to:** N/A **Registered:** N/A
rst_clk_tx_n	I	Transmit Clock Reset. This is an active-low reset signal. When this signal goes high, the logic inside the Transmit Clock domain goes to the default state. **Exists:** DWC_EQOS_MTL_SUBSYS||DWC_EQOS_CORE **Power Domain:** MAC_ON **Active State:** Low **Synchronous to:** clk_tx_i **Registered:** No

Port Name	I/O	Description
rst_clk_rx_n	I	Receive Clock Reset. This is an active-low reset signal. When this signal goes high, the logic inside the Receive Clock domain goes to the default state. **Exists:** DWC_EQOS_MTL_SUBSYS||DWC_EQOS_CORE **Power Domain:** MAC_ON **Active State:** Low **Synchronous to:** clk_rx_i **Registered:** No
rst_clk_tx_n_o	O	Transmit Clock Reset Output. This signal is generated synchronous to the clk_tx_i clock whenever a hardware reset or a soft-reset is given to DWC_ether_qos. **Exists:** DWC_EQOS_AHB_SUBSYS||DWC_EQOS_DMA_SUBSYS||DWC_EQOS_AXI_SUBSYS **Power Domain:** MAC_OFF **Active State:** Low **Synchronous to:** clk_tx_i **Registered:** No
rst_clk_rx_n_o	O	Receive Clock Reset Output. This signal is generated synchronous to the clk_rx_i clock whenever a hardware reset or a soft-reset is given to DWC_ether_qos. **Exists:** DWC_EQOS_AHB_SUBSYS||DWC_EQOS_DMA_SUBSYS||DWC_EQOS_AXI_SUBSYS **Power Domain:** MAC_OFF **Active State:** Low **Synchronous to:** clk_rx_i **Registered:** No

16.2 MAC Control Interface Signals

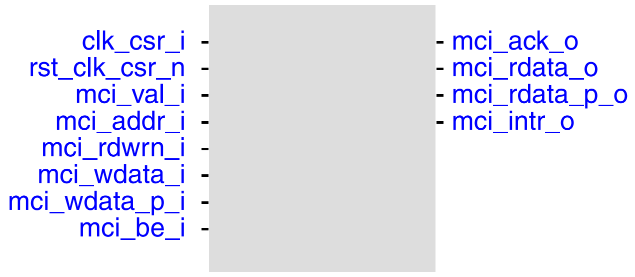


Table 16-2 MAC Control Interface Signals

Port Name	I/O	Description
clk_csr_i	I	CSR Clock. This is a free-running clock input provided by the Application. The MAC Control Interface, CSR, and Station Management Agent (SMA) of the MAC run on this clock. All signals generated by the CSR module are synchronous to the clk_csr_i clock. The minimum valid frequency of this clock is 20 MHz (minimum frequency is 25 MHz when the MMC module is active in 1000 Mbps mode). The maximum valid frequency depends on frequency at which logic timing is met. The frequency below the valid minimum frequency may result in incorrect operation. **Exists:** (DWC_EQOS_CSR_SLV_CLK&&!(DWC_EQOS_AXI_SUBSYS&&DWC_EQOS_AHB_SLAVE))||DWC_EQOS_CORE **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** N/A **Registered:** N/A
rst_clk_csr_n	I	CSR Clock Reset. This is an active-low reset signal. When this signal goes high, all registers and counters inside the CSR go to their default state. **Exists:** (DWC_EQOS_CSR_SLV_CLK&&DWC_EQOS_MTL_SUBSYS)||DWC_EQOS_CORE **Power Domain:** MAC_OFF **Active State:** Low **Synchronous to:** CSR clock **Registered:** No

Port Name	I/O	Description
mci_val_i	I	MCI Valid. The application drives this signal high to access the MAC CSR module, indicating to the MAC that the following are valid: ■ Address ■ Read or Write control ■ Data (for Write operation) **Exists:** !DWC_EQOS_AHB_SUBSYS&&!DWC_EQOS_APB_SLAVE&&!DWC_EQOS_APB3_SLAVE&&!DWC_EQOS_AXI_SUBSYS **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** CSR clock **Registered:** No
mci_addr_i[13:0]	I	MCI Address. This signal carries the register address of the CSR module from the Application. The address is valid only when mci_val_i is high. **Exists:** !DWC_EQOS_AHB_SUBSYS&&!DWC_EQOS_APB_SLAVE&&!DWC_EQOS_APB3_SLAVE&&!DWC_EQOS_AXI_SUBSYS **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** CSR clock **Registered:** No
mci_rdwrn_i	I	MCI Read or Write. This signal provides the Read or Write control. The application writes the data into the MAC Control registers by driving this signal low. When this signal is high, it indicates a Read operation. **Exists:** !DWC_EQOS_AHB_SUBSYS&&!DWC_EQOS_APB_SLAVE&&!DWC_EQOS_APB3_SLAVE&&!DWC_EQOS_AXI_SUBSYS **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** CSR clock **Registered:** No

Port Name	I/O	Description
mci_wdata_i[31:0]	I	MCI Write Data. This signal carries the Write data from the application to the Control registers of the CSR module. The validity of each byte in this signal is qualified with the corresponding byte enables in mci_be_i signal. **Exists:** !DWC_EQOS_AHB_SUBSYS&&!DWC_EQOS_APB_SLAVE&&!DWC_EQOS_APB3_SLAVE&&!DWC_EQOS_AXI_SUBSYS **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** CSR clock **Registered:** Yes
mci_wdata_p_i[3:0]	I	MCI Write Data Parity. This signal carries the Write Parity data from the application. The validity of each bit in this signal is qualified with the corresponding byte enables in mci_be_i signal. **Exists:** !DWC_EQOS_AHB_SUBSYS&&!DWC_EQOS_APB_SLAVE&&!DWC_EQOS_APB3_SLAVE&&!DWC_EQOS_AXI_SUBSYS&&DWC_EQOS_ASP_PPE **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** CSR clock **Registered:** Yes
mci_be_i[3:0]	I	MCI Byte Enable. This signal indicates the bytes that are valid on the mci_wdata_i signal. This signal is valid when the mci_val_i signal is high. **Exists:** !DWC_EQOS_AHB_SUBSYS&&!DWC_EQOS_APB_SLAVE&&!DWC_EQOS_APB3_SLAVE&&!DWC_EQOS_AXI_SUBSYS **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** CSR clock **Registered:** No
mci_ack_o	O	MCI Transfer Acknowledge. This is an acknowledgment from the MAC Control Interface indicating that the Write data on the mci_wdata_i bus has been accepted during CSR write operation and valid read data is available on the mci_rdata_o bus for the CSR read operation. **Exists:** !DWC_EQOS_AHB_SUBSYS&&!DWC_EQOS_APB_SLAVE&&!DWC_EQOS_APB3_SLAVE&&!DWC_EQOS_AXI_SUBSYS **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** CSR clock **Registered:** Yes

Port Name	I/O	Description
mci_rdata_o[31:0]	O	MCI Read Data. This signal carries the Read data from the MAC CSR to the application. The MAC supplies all bytes irrespective of the byte enable signal. **Exists:** !DWC_EQOS_AHB_SUBSYS&&!DWC_EQOS_APB_SLAVE&&!DWC_EQOS_APB3_SLAVE&&!DWC_EQOS_AXI_SUBSYS **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** CSR clock **Registered:** Yes
mci_rdata_p_o[3:0]	O	MCI Parity Read Data. This signal carries the Parity Read data from the MAC CSR to the application. **Exists:** !DWC_EQOS_AHB_SUBSYS&&!DWC_EQOS_APB_SLAVE&&!DWC_EQOS_APB3_SLAVE&&!DWC_EQOS_AXI_SUBSYS&&DWC_EQOS_ASP_PPE **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** CSR clock **Registered:** Yes
mci_intr_o	O	MCI Interrupt. When this signal is high, it indicates an interrupt event in: ■ The optional MMC, PCS, RGMII, Timestamp, GPI, LPI, or PMT module of the MAC ■ The Transmit or Receive Controller of the MAC. **Exists:** DWC_EQOS_CORE||DWC_EQOS_MTL_SUBSYS **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** CSR clock **Registered:** No

16.3 MAC Application Clock and Reset Interface Signals

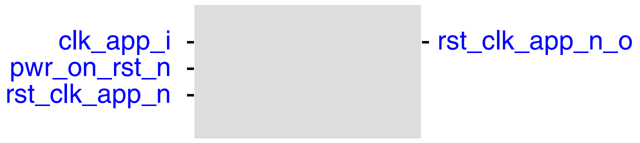


Table 16-3 **MAC Application Clock and Reset Interface Signals**

Port Name	I/O	Description
clk_app_i	I	Application Clock. This is a free-running input clock from the application. In the EQOS-DMA configuration, the DMA interface signals are synchronous to this clock. In the EQOS-MTL configuration, the MTL application interface operates synchronous to this clock. The minimum valid frequency is 25 MHz (it is ~1.5 times MAC transmitter/receiver in SPRAM configurations) and the maximum valid frequency depends on frequency at which logic timing is met. **Exists:** (DWC_EQOS_DMA_SUBSYS||DWC_EQOS_MTL_SUBSYS) **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** N/A **Registered:** N/A
pwr_on_rst_n	I	Application Clock Reset. This is a reset signal synchronous to the application clock domain in the EQOS-DMA configuration. When you select the Enable Power Management feature in the EQOS-CORE and EQOS-MTL configurations, this is an asynchronous reset signal. **Exists:** DWC_EQOS_DMA_SUBSYS||(DWC_EQOS_LP_MODE_EN&&(DWC_EQOS_CORE||DWC_EQOS_MTL_SUBSYS)) **Power Domain:** MAC_OFF **Active State:** Low **Synchronous to:** clk_app_i for the EQOS-DMA configuration. Asynchronous for the EQOS-CORE and EQOS-MTL configurations. **Registered:** N/A
rst_clk_app_n	I	Application Clock Reset. This is a reset signal synchronous to the application clock domain in the EQOS-MTL configuration. **Exists:** DWC_EQOS_SYS == 1 **Power Domain:** MAC_OFF **Active State:** Low **Synchronous to:** clk_app_i **Registered:** N/A

Port Name	I/O	Description
rst_clk_app_n_o	O	Application Clock Reset Output. This signal is generated synchronous to the application clock domain whenever a hardware reset or a soft-reset is given to DWC_ether_qos. **Exists:** DWC_EQOS_AHB_SUBSYS||DWC_EQOS_DMA_SUBSYS||DWC_EQOS_AXI_SUBSYS **Power Domain:** MAC_OFF **Active State:** Low **Synchronous to:** Application clock **Registered:** N/A

16.4 SGMII/TBI/RTBI/SMII Clocks and Reset Signals

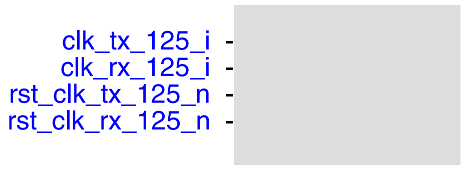


Table 16-4 SGMII/TBI/RTBI/SMII Clocks and Reset Signals

Port Name	I/O	Description
clk_tx_125_i	I	125 MHz Transmit Clock. This is the 125 MHz clock input for the SGMII, TBI, RTBI, or SMII Transmit interface. The clk_tx_i clock (312.5, 125, 25, or 2.5 MHz) should be synchronous with this clock. **Exists:** DWC_EQOS_PCS_EN||DWC_EQOS_SMII_EN **Power Domain:** MAC_ON **Active State:** N/A **Synchronous to:** N/A **Registered:** N/A
clk_rx_125_i	I	125 MHz Receive Clock. This is the 125 MHz clock input for the SGMII, TBI, RTBI, or SMII (SSSMII - Source Synchronous Mode) Receive interface. The clk_rx_i clock (312.5, 125, 25, or 2.5 MHz) must be synchronous with this clock. **Exists:** DWC_EQOS_PCS_EN||DWC_EQOS_SSSMII_EN **Power Domain:** MAC_ON **Active State:** N/A **Synchronous to:** N/A **Registered:** N/A
rst_clk_tx_125_n	I	125 MHz Transmit Clock Reset. This is a reset signal synchronous to the clk_tx_125_i. **Exists:** (DWC_EQOS_PCS_EN||DWC_EQOS_SMII_EN)&&(DWC_EQOS_MTL_SUBSYS||DWC_EQOS_CORE) **Power Domain:** MAC_ON **Active State:** Low **Synchronous to:** clk_tx_125_i **Registered:** No

Port Name	I/O	Description
rst_clk_rx_125_n	I	125 MHz Receive Clock Reset. This is a reset signal synchronous to the clk_rx_125_i. **Exists:** (DWC_EQOS_PCS_EN||DWC_EQOS_SSSMII_EN)&&(DWC_EQOS_MTL_SUBSYS||DWC_EQOS_CORE) **Power Domain:** MAC_ON **Active State:** Low **Synchronous to:** clk_rx_125_i **Registered:** No

16.5 RMII Clocks and Reset Signals

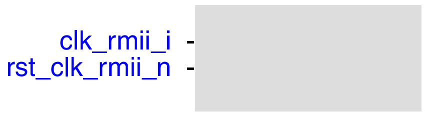


Table 16-5 RMII Clocks and Reset Signals

Port Name	I/O	Description
clk_rmii_i	I	RMII Clock. This is the 50-MHz clock used by the RMII interface. When the RMII interface is selected, the clk_rx_i clock should be 25 or 2.5 MHz and it should be derived from this clock. **Exists:** DWC_EQOS_RMII_EN **Power Domain:** MAC_ON **Active State:** N/A **Synchronous to:** N/A **Registered:** N/A
rst_clk_rmii_n	I	RMII Clock Reset. This is a reset signal synchronous to the clk_rmii_i. **Exists:** (DWC_EQOS_RMII_EN)&&(DWC_EQOS_MTL_SUBSYS||DWC_EQOS_CORE) **Power Domain:** MAC_ON **Active State:** Low **Synchronous to:** clk_rmii_i **Registered:** No

16.6 RGMII/RTBI Clocks and Reset Signals

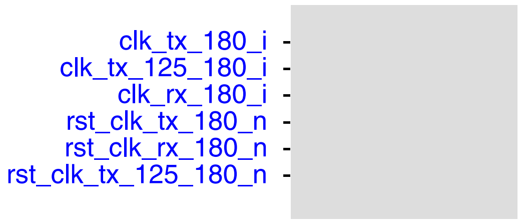


Table 16-6 RGMII/RTBI Clocks and Reset Signals

Port Name	I/O	Description
clk_tx_180_i	I	Inverted RGMII Transmit Clock. The RGMII transmit interface uses the rising edge of clk_tx_180_i to represent the falling edge of clk_tx_i. **Exists:** DWC_EQOS_RGMII_EN **Power Domain:** MAC_ON **Active State:** N/A **Synchronous to:** clk_tx_i **Registered:** N/A
clk_tx_125_180_i	I	Inverted RTBI Transmit Clock. The RTBI transmit interface uses the rising edge of clk_tx_125_180_i to represent the falling edge of clk_tx_125_i. **Exists:** DWC_EQOS_RTBI_EN **Power Domain:** MAC_ON **Active State:** N/A **Synchronous to:** clk_tx_125_i **Registered:** N/A
clk_rx_180_i	I	Inverted RGMII/RTBI Receive Clock. The RGMII receive interface uses the positive edge of clk_rx_180_i to represent the falling edge of clk_rx_i. The RTBI receive interface uses the positive edge of clk_rx_180_i to represent the falling edge of clk_rx_125_i. **Exists:** DWC_EQOS_RGMII_EN||DWC_EQOS_RTBI_EN **Power Domain:** MAC_ON **Active State:** N/A **Synchronous to:** clk_rx_i **Registered:** N/A

Port Name	I/O	Description
rst_clk_tx_180_n	I	Transmit Clock Reset. This is a reset signal synchronous to the clk_tx_180_i. **Exists:** (DWC_EQOS_RGMII_EN)&&(DWC_EQOS_MTL_SUBSYS||DWC_EQOS_CORE) **Power Domain:** MAC_ON **Active State:** Low **Synchronous to:** clk_tx_180_i **Registered:** No
rst_clk_rx_180_n	I	Receive Clock Reset. This is a reset signal synchronous to the clk_rx_180_i. **Exists:** (DWC_EQOS_RGMII_OR_RTBI)&&(DWC_EQOS_MTL_SUBSYS||DWC_EQOS_CORE) **Power Domain:** MAC_ON **Active State:** Low **Synchronous to:** clk_rx_180_i **Registered:** No
rst_clk_tx_125_180_n	I	125-MHz Transmit Clock (180) Reset. This is a reset signal synchronous to the clk_tx_125_180_i. **Exists:** (DWC_EQOS_RTBI_EN)&&(DWC_EQOS_MTL_SUBSYS||DWC_EQOS_CORE) **Power Domain:** MAC_ON **Active State:** Low **Synchronous to:** clk_tx_125_180_i **Registered:** No

16.7 TBI Interface Signals

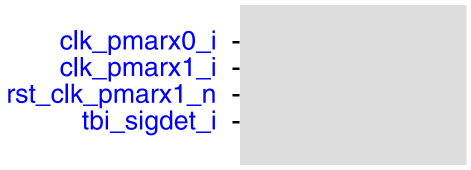


Table 16-7 TBI Interface Signals

Port Name	I/O	Description
clk_pmarx0_i	I	TBI Receive Clock. The MAC uses this 62.5 MHz recovered Receive clock to register the odd code groups of received data. **Exists:** DWC_EQOS_TBI_EN **Power Domain:** MAC_ON **Active State:** N/A **Synchronous to:** N/A **Registered:** N/A
clk_pmarx1_i	I	Inverted TBI Recovered Receive Clock. The MAC uses this 62.5 MHz recovered Receive clock to register the even code groups of received data. This clock is 180 degrees out of phase with clk_pmarx0_i. **Exists:** DWC_EQOS_TBI_EN **Power Domain:** MAC_ON **Active State:** N/A **Synchronous to:** clk_pmarx0_i **Registered:** N/A
rst_clk_pmarx1_n	I	Inverted TBI Receive Clock Reset. This is a reset signal synchronous to the clk_pmarx1_i. **Exists:** (DWC_EQOS_TBI_EN)&&(DWC_EQOS_MTL_SUBSYS||DWC_EQOS_CORE) **Power Domain:** MAC_ON **Active State:** Low **Synchronous to:** clk_pmarx0_i **Registered:** No

Port Name	I/O	Description
tbi_sigdet_i	I	TBI Signal Detect. This signal informs the MAC that a signal has been detected at the PMD device. The TBI interface is generally used with the Fiber media. Therefore, this signal indicates detection of light. **Exists:** DWC_EQOS_TBI_EN||DWC_EQOS_SGMII_EN||DWC_EQOS_RTBI_EN **Power Domain:** MAC_ON **Active State:** High **Synchronous to:** clk_rx_125_i **Registered:** No

16.8 RevMII Interface Signals

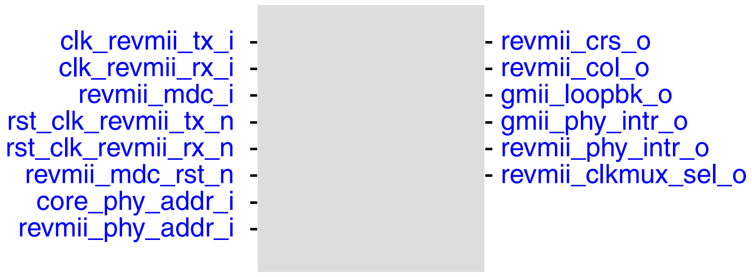


Table 16-8 RevMII Interface Signals

Port Name	I/O	Description
clk_revmii_tx_i	I	RevMII Transmit Clock. This clock signal is used to register the input (G)MII data/control coming from the Remote MAC Transmitter (phy_rxd_i, phy_rxdv_i, and phy_rxer_i). **Exists:** DWC_EQOS_REVMII_EN **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** N/A **Registered:** N/A
clk_revmii_rx_i	I	RevMII Receive Clock. This clock signal is used to drive the output (G)MII signals (phy_txd_o, phy_txen_o, and phy_txer_o) to the remote MAC Receiver. **Exists:** DWC_EQOS_REVMII_EN **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** N/A **Registered:** N/A
revmii_mdc_i	I	Remote MAC MDIO Clock. The remote MAC provides this clock signal. This clock signal is used to synchronously transfer data in and out of the RevMII using revmii_mdio pin. This clock operates at a maximum frequency of 2.5 MHz (as specified in IEEE 802.3u). **Exists:** DWC_EQOS_REVMII_EN **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** N/A **Registered:** N/A

Port Name	I/O	Description
rst_clk_revmii_tx_n	I	RevMII Transmit Clock Asynchronous Reset. This is a reset signal synchronous to the clk_revmii_tx_i. **Exists:** (DWC_EQOS_REVMII_EN)&&(DWC_EQOS_MTL_SUBSYS||DWC_EQOS_CORE) **Power Domain:** MAC_OFF **Active State:** Low **Synchronous to:** clk_revmii_tx_i **Registered:** No
rst_clk_revmii_rx_n	I	RevMII Receive Clock Asynchronous Reset. This is a reset signal synchronous to the clk_revmii_rx_i. **Exists:** (DWC_EQOS_REVMII_EN)&&(DWC_EQOS_MTL_SUBSYS||DWC_EQOS_CORE) **Power Domain:** MAC_OFF **Active State:** Low **Synchronous to:** clk_revmii_rx_i **Registered:** No
revmii_mdc_rst_n	I	MDC clock asynchronous reset from remote MAC. This is a reset signal synchronous to the revmii_mdc_i. **Exists:** (DWC_EQOS_REVMII_EN) **Power Domain:** MAC_OFF **Active State:** Low **Synchronous to:** revmii_mdc_i **Registered:** No
revmii_crs_o	O	RevMII Carrier Sense. The RevMII drives this signal high to indicate to the remote MAC that either the Transmit or Receive medium is not idle. The RevMII drives this signal low when both Transmit and Receive mediums are idle. **Exists:** DWC_EQOS_REVMII_EN **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** Asynchronous **Registered:** No
revmii_col_o	O	RevMII Collision Detect. The RevMII drives this signal high when a collision is detected on the medium. This signal remains high till the collision condition persists. **Exists:** DWC_EQOS_REVMII_EN **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** Asynchronous **Registered:** No

Port Name	I/O	Description
gmii_loopbk_o	O	Loopback mode for Local MAC. This signal is used to select the clk_rx_i source in the loopback mode. **Exists:** DWC_EQOS_REVMII_EN **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** CSR clock **Registered:** Yes
core_phy_addr_i[4:0]	I	Local PHY Address. This bus gives the PHY address of the Station Management registers that are connected to the Local MAC. The PHY address has a static value. **Exists:** DWC_EQOS_REVMII_EN **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** CSR clock **Registered:** No
revmii_phy_addr_i[4:0]	I	Remote PHY Address. This bus gives the PHY address of the Station Management registers connected to the Remote MAC. The PHY address has a static value. **Exists:** DWC_EQOS_REVMII_EN **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** revmii_mdc_i **Registered:** No
gmii_phy_intr_o	O	Interrupt to Local MAC. This signal gives the interrupt to the local host. This signal is generated when the Link Status Change interrupt is set in the MAC_RevMII_Interrupt_Status_Mask register of local MAC. **Exists:** DWC_EQOS_REVMII_EN **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** CSR clock **Registered:** No
revmii_phy_intr_o	O	Interrupt to Remote MAC. This signal gives the interrupt to the remote MAC. This signal is generated when the Link Status Change interrupt is set in the MAC_RevMII_Interrupt_Status_Mask register of remote MAC. **Exists:** DWC_EQOS_REVMII_EN **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** CSR clock **Registered:** No

Port Name	I/O	Description
revmii_clkmux_sel_o[2:0]	O	Remote MAC Control Register Bits for Clock Muxing. These bits are used for clock selection. Bit 2 is high during loopback operation. Bits[1:0] select the clocks according to the operation speed. Speeds are indicated as follows: ■ 2'b0x: 1000 Mbps (GMII) ■ 2'b10: 10 Mbps (MII) ■ 2'b11: 100 Mbps (MII) These bits reflect the values written in Bits 14, 6, and 13 of the RevMII PHY Control register of Remote MAC. **Exists:** DWC_EQOS_REVMII_EN **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** revmii_mdc_i **Registered:** Yes

16.9 IEEE 1588 Timestamp Interface Signals

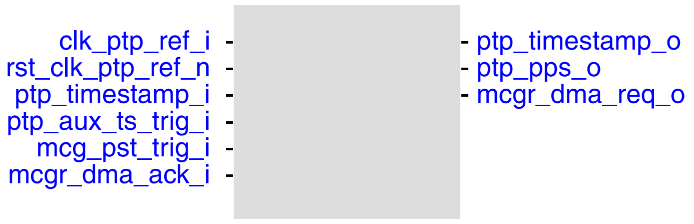


Table 16-9 IEEE 1588 Timestamp Interface Signals

Port Name	I/O	Description
clk_ptp_ref_i	I	PTP Reference Clock. This reference clock, provided by an external oscillator or a timing source, is used for the timestamp logic. **Exists:** DWC_EQOS_TIME_STAMPING **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** NA **Registered:** N/A
rst_clk_ptp_ref_n	I	PTP Reference Clock Reset. This is a reset signal synchronous to the clk_ptp_ref_i. **Exists:** DWC_EQOS_TIME_STAMPING&&(DWC_EQOS_MTL_SUBSYS||DWC_EQOS_CORE) **Power Domain:** MAC_OFF **Active State:** Low **Synchronous to:** clk_ptp_ref_i **Registered:** No
ptp_timestamp_i[63:0]	I	Reference Time Input. This bus provides the system time used to timestamp the specific Transmit packets or received packets. **Exists:** DWC_EQOS_TIME_STAMPING&&DWC_EQOS_SYSTIME_SOURCE!=0 **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** clk_ptp_ref_i **Registered:** Yes

Port Name	I/O	Description
ptp_timestamp_o[63:0]	O	Reference Time Output. This bus provides the System Time output when it is generated internally. **Exists:** DWC_EQOS_TIME_STAMPING&&DWC_EQOS_SYSTIME_SOURCE!=1 **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** clk_ptp_ref_i **Registered:** Yes
ptp_pps_o[0:0]	O	Pulse Per Second. This signal is high based on the PPS mode selected in the MAC_PPS_Control register. When PPS is programmed in Media Clock recovery Mode, this port indicates the recovered clock **Exists:** DWC_EQOS_TIME_STAMPING&&DWC_EQOS_SYSTIME_SOURCE!=1 **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** clk_ptp_ref_i **Registered:** Yes
ptp_aux_ts_trig_i[(DWC_EQOS_AUX_SNAP_IN_NUM-1):0]	I	Auxiliary Timestamp Trigger. When this signal goes high (rising edge), a snapshot of the System Time is taken and stored in the auxiliary timestamp FIFO. **Exists:** DWC_EQOS_ADV_TIME_AUX_SNAP **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** Asynchronous **Registered:** Yes
mcg_pst_trig_i[0:0]	I	Media Clock Generation Trigger Input Trigger Input to the DUT to capture presentation time. Based on presentation control value of MAC_PPS_Control register, this signal can be pulse or level signal. Width of this IO changes with the configured number of PPS instances. **Exists:** DWC_EQOS_FLEXI_PPS_OUT_EN **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** clk_ptp_ref_i **Registered:** Yes

Port Name	I/O	Description
mcgr_dma_req_o[0:0]	O	MCGR DMA Read Request Used to Request a Target Presentation time value for one of the comparator modules when the instance is running in recovery mode. In Media Clock generation mode, this I/O signal is held high indicating a valid Presentation time is captured and updated in MAC_PPS(#i)_Target_Time_Seconds register. One or more DMA read requests can simultaneously active, it is up to the DMA logic to maintain priority. Bits have a 1-1 correspondence to the comparator instances (i.e. Bit-0 belongs to comparator 0, bit-1 belongs to comparator-1 and so on). Once set, will hold high and will be reset the cycle after receiving the mcgr_dma_ack for that comparator. Width of this IO changes with the configured number of PPS instances. **Exists:** DWC_EQOS_FLEXI_PPS_OUT_EN **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** clk_ptp_ref_i **Registered:** Yes
mcgr_dma_ack_i[0:0]	I	MCGR DMA Ack This port indicate that the request indicated by mcgr_dma_req_o is accepted by the application . Should be an active high pulse for one cycle. Width of this IO changes with the configured number of PPS instances. **Exists:** DWC_EQOS_FLEXI_PPS_OUT_EN **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** clk_ptp_ref_i **Registered:** Yes

16.10 Sideband Interface Signals

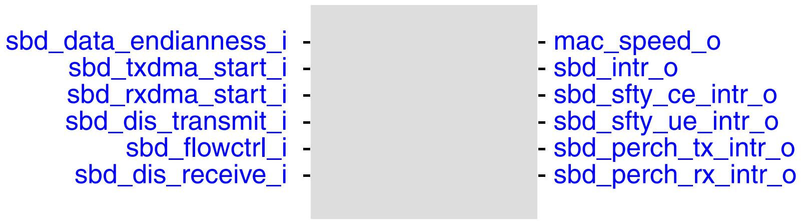


Table 16-10 Sideband Interface Signals

Port Name	I/O	Description
mac_speed_o[1:0]	O	Sideband MAC Speed Indication/Status. This signal indicates the 10Mbps, 100Mbps, 1000Mbps, or 2500Mbps operating speed and reflects the FES and PS bits of MAC_Configuration register. Speeds are indicated as follows: ■ 2'b00: 1000 Mbps (GMII) ■ 2'b01: 2500 Mbps (GMII) ■ 2'b10: 10 Mbps (MII) ■ 2'b11: 100 Mbps (MII) **Exists:** Always **Power Domain:** MAC_ON **Active State:** N/A **Synchronous to:** CSR clock **Registered:** Yes
sbd_data_endianness_i	I	Sideband Data Endianness Control. When this signal is high, it configures the DMA to Transfer on the data in big-endian format. When this signal is low (by default), the data is transferred in little-endian format. This signal is sampled during active reset (including soft-reset) only and ignored after reset is low. In EQOS-AXI configuration, sbd_data_endianess_i is also used for descriptor endianness. **Exists:** (DWC_EQOS_ENDIAN_NESS==2) **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** CSR clock **Registered:** Yes

Port Name	I/O	Description
sbd_txdma_start_i[(DWC_EQOS_NUM_DMA_TX_CH-1):0]	I	Sideband Tx DMA Start/Stop Control. When this signal goes high, rising edge indicates that Tx DMA needs to be started. When this signal goes low (default state), falling edge indicates that the Tx DMA needs to be stopped. This signal needs to be tied low when this function is not used. **Exists:** DWC_EQOS_DMA_STRTSTP **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** CSR clock **Registered:** Yes
sbd_rxdma_start_i[(DWC_EQOS_NUM_DMA_RX_CH-1):0]	I	Sideband Rx DMA Start/Stop Control. When this signal goes high, rising edge indicates that Rx DMA needs to be started. When this signal goes low (default state), falling edge indicates that the Rx DMA needs to be stopped. This signal needs to be tied low when this function is not used. **Exists:** DWC_EQOS_DMA_STRTSTP **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** CSR clock **Registered:** Yes
sbd_intr_o	O	Subsystem Interrupt. When this signal is high, it indicates an interrupt event to the application from one or multiple DMA Channels. **Exists:** DWC_EQOS_DMA_SUBSYS||DWC_EQOS_AHB_SUBSYS||DWC_EQOS_AXI_SUBSYS **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** CSR clock **Registered:** No
sbd_sfty_ce_intr_o	O	Subsystem Interrupt. When this signal is high, it indicates a correctable error interrupt event to the application from Automotive Safety features. This signal is available in non core and DWC_EQOS_ASP_ECC enabled configurations. **Exists:** DWC_EQOS_ASP_ECC **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** CSR clock **Registered:** Yes

Port Name	I/O	Description
sbd_sfty_ue_intr_o	O	Subsystem Interrupt. When this signal is high, it indicates an un-correctable error interrupt event to the application from Automotive Safety features. This signal is available in non core and "DWC_EQOS_ASP_ECC or DWC_EQOS_ASP_ALL" enabled configurations. **Exists:** DWC_EQOS_ASP **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** CSR clock **Registered:** Yes
sbd_perch_tx_intr_o[(DWC_EQOS_NUM_DMA_TX_CH-1):0]	O	Per Channel Transmit Interrupt signal to host system. These are per channel Transmit Completion Interrupt signals, it will be pulse or level based on the INTM control field programming in the DMA_Mode register. **Exists:** !DWC_EQOS_CORE&&!DWC_EQOS_MTL_SUBSYS **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** CSR clock **Registered:** No
sbd_perch_rx_intr_o[(DWC_EQOS_NUM_DMA_RX_CH-1):0]	O	Per Channel Receive Interrupt signal to host system. These are per channel Receive Completion Interrupt signals, it is pulse or level based on the INTM control field programming in the DMA_Mode register. **Exists:** !DWC_EQOS_CORE&&!DWC_EQOS_MTL_SUBSYS **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** CSR clock **Registered:** No
sbd_dis_transmit_i	I	Sideband MAC Transmitter Disable Control. This signal instructs the MAC transmitter to stop transmitting packets after completing current packet transfer. The MAC transmitter restarts transmission only if this signal is reset to low and the TE bit of MAC_Configuration register is set high. This signal is an optional port applicable only in EQOS-CORE and EQOS-MTL configurations. **Exists:** DWC_EQOS_ADD_TXRX_DIS_IO **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** Asynchronous **Registered:** Yes

Port Name	I/O	Description
sbd_flowctrl_i[(DWC_EQOS_NUM_RXQ-1):0]	I	Sideband Flow Control. When set high, instructs the MAC to transmit Pause frames in Full-duplex mode. In half-duplex mode, the MAC enables the backpressure function until this signal is made low again. This signal is an optional port applicable only in EQOS-AHB, EQOS-AXI, EQOS-DMA, and EQOS-MTL configurations. **Exists:** DWC_EQOS_SBD_FC_EN **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** Asynchronous **Registered:** Yes
sbd_dis_receive_i	I	Sideband MAC Receiver Disable Control. This signal instructs the MAC receiver to stop receiving packets after completing any current packet reception. The MAC receiver restarts reception only if this signal is reset to low and the RE bit of MAC_Configuration register is set high. This signal is an optional port applicable only in EQOS-CORE and EQOS-MTL configurations. **Exists:** DWC_EQOS_ADD_TXRX_DIS_IO **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** Asynchronous **Registered:** Yes

16.11 TBI/SGMII/RTBI/SMII PHY Interface Signals

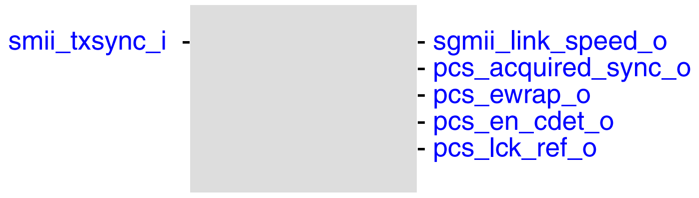


Table 16-11 TBI/SGMII/RTBI/SMII PHY Interface Signals

Port Name	I/O	Description
sgmii_link_speed_o[1:0]	O	SGMII Link Speed. This signal indicates the link speed: ■ 00: 10 Mbps ■ 01: 100 Mbps ■ 10: 1000 Mbps ■ 11: Reserved This signal should be used to change the frequency of clk_tx_i and clk_rx_i clocks. **Note:** If auto-negotiation is disabled in SGMII, this signal retains the value and does not change. The default value after reset is 10. **Exists:** DWC_EQOS_SGMII_EN **Power Domain:** MAC_ON **Active State:** N/A **Synchronous to:** CSR clock **Registered:** Yes
smii_txsync_i	I	SMII Transmit Synchronization. This is Transmit synchronization input signal for SMII in source synchronous (SSSMII) mode. **Exists:** DWC_EQOS_SSSMII_TXSYNC_IN **Power Domain:** MAC_ON **Active State:** High **Synchronous to:** clk_tx_125_i **Registered:** Yes
pcs_acquired_sync_o	O	PCS Acquired Synchronization. When this signal is high, it indicates that the PCS interface has synchronized to the running disparity of the receive code-groups. **Exists:** DWC_EQOS_TBI_EN||DWC_EQOS_SGMII_EN||DWC_EQOS_RTBI_EN **Power Domain:** MAC_ON **Active State:** High **Synchronous to:** clk_rx_125_i, clk_rx_i **Registered:** Yes

Port Name	I/O	Description
pcs_ewrap_o	O	PCS Enable Wrap. This signal enables the PHY loopback. When this signal is high, the PHY should loop back the Transmit serialized data to the Receive path. This signal reflects the ELE bit of MAC_AN_Control register. **Exists:** DWC_EQOS_TBI_EN||DWC_EQOS_SGMII_EN||DWC_EQOS_RTBI_EN **Power Domain:** MAC_ON **Active State:** High **Synchronous to:** CSR clock **Registered:** Yes
pcs_en_cdet_o	O	PCS Enable Detect. This signal reflects the ECD bit of MAC_AN_Control register. It is low after reset. **Exists:** DWC_EQOS_TBI_EN||DWC_EQOS_SGMII_EN||DWC_EQOS_RTBI_EN **Power Domain:** MAC_ON **Active State:** High **Synchronous to:** CSR clock **Registered:** Yes
pcs_lck_ref_o	O	PCS Lock Reference. This signal reflects the LR bit of MAC_AN_Control register. It is low after reset. **Exists:** DWC_EQOS_TBI_EN||DWC_EQOS_SGMII_EN||DWC_EQOS_RTBI_EN **Power Domain:** MAC_ON **Active State:** High **Synchronous to:** CSR clock **Registered:** Yes

16.12 General Purpose Interface Signals

Table 16-12 General Purpose Interface Signals

Port Name	I/O	Description
gpo_o[(DWC_EQOS_GOW-1):0]	O	General Purpose Output. The MAC drives this output based on the GPO field in the MAC_GPIO_Control register. **Exists:** DWC_EQOS_GPIO_EN **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** CSR clock **Registered:** Yes
gpi_i[(DWC_EQOS_GIW-1):0]	I	General Purpose Input. The MAC samples this input for change, updates status in MAC_GPIO_Status register and generates the interrupt based on GPIT field programming in MAC_GPIO_Control register. **Exists:** DWC_EQOS_GPIO_EN **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** Asynchronous **Registered:** Yes

16.13 Test Mode Interface Signals

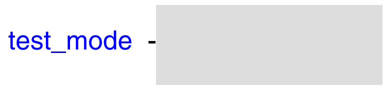


Table 16-13 Test Mode Interface Signals

Port Name	I/O	Description
test_mode	I	Test Mode. This input enables bypassing of internally-generated resets and directly connecting these resets to the external reset input (hreset_n, aresetn_i, or pwr_on_rst_n respectively). This input should normally be tied to zero. It should be set only during scan testing. **Exists:** DWC_EQOS_AHB_SUBSYS||DWC_EQOS_DMA_SUBSYS||DWC_EQOS_AXI_SUBSYS||DWC_EQOS_LP_MODE_EN **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** N/A **Registered:** N/A

16.14 Energy Efficient Ethernet Interface Signals

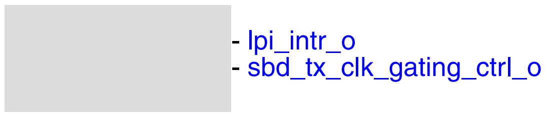


Table 16-14 Energy Efficient Ethernet Interface Signals

Port Name	I/O	Description
lpi_intr_o	O	LPI Interrupt. This signal is high when the MAC receiver exits the LPI state. **Exists:** DWC_EQOS_EEE_EN **Power Domain:** MAC_ON **Active State:** High **Synchronous to:** clk_rx_i **Registered:** Yes
sbd_tx_clk_gating_ctrl_o	O	LPI Tx Clock Gating Control. This signal is high after the MAC enters the Tx LPI mode. You can use this signal to control the Tx clock gating. **Active State:** High **Exists:** DWC_EQOS_EEE_EN **Power Domain:** MAC_ON **Active State:** High **Synchronous to:** CSR clock **Registered:** Yes

16.15 Power Management Interface Signals

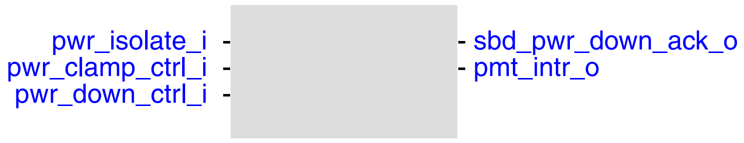


Table 16-15 Power Management Interface Signals

Port Name	I/O	Description
pwr_isolate_i	I	Power Isolate Control. This signal is the control input for clamping the isolation cells. **Exists:** DWC_EQOS_PMT_EN **Power Domain:** MAC_ON **Active State:** High **Synchronous to:** Asynchronous **Registered:** No
pwr_clamp_ctrl_i	I	Power Clamp Control. This signal is control input for clamping the reset signals to the always-on logic. **Exists:** DWC_EQOS_PMT_EN **Power Domain:** MAC_ON **Active State:** High **Synchronous to:** Asynchronous **Registered:** No
pwr_down_ctrl_i	I	Power Down Control. This signal controls the VDD power down. **Exists:** DWC_EQOS_PMT_EN **Power Domain:** MAC_ON **Active State:** High **Synchronous to:** Asynchronous **Registered:** No
sbd_pwr_down_ack_o	O	Power Down Acknowledge. This signal is acknowledgment to start the power-down sequence. **Exists:** DWC_EQOS_PMT_EN **Power Domain:** MAC_ON **Active State:** High **Synchronous to:** clk_rx_i **Registered:** No

Port Name	I/O	Description
pmt_intr_o	O	PMT Interrupt. This signal indicates an interrupt event in the optional PMT module of the DWC_ether_qos core. You can use it to wake the system from the Sleep mode. This interrupt has no masking control. **Exists:** DWC_EQOS_PMT_EN **Power Domain:** MAC_ON **Active State:** High **Synchronous to:** clk_rx_i **Registered:** No

16.16 PHY Interface Signals

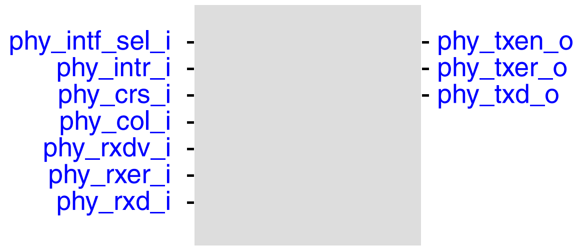


Table 16-16 PHY Interface Signals

Port Name	I/O	Description
phy_intf_sel_i[2:0]	I	PHY Interface Select. This signal selects the PHY interface of the MAC: ■ 000: GMII or MII ■ 001: RGMII ■ 010: SGMII ■ 011: TBI ■ 100: RMII ■ 101: RTBI ■ 110: SMII ■ 111: RevMII. This signal is sampled only during reset assertion, and it is ignored after that. In EQOS-CORE and EQOS-MTL configurations, the rst_clk_csr_n signal is used for sampling. In EQOS-DMA, EQOS-AHB, and EQOS-AXI configurations, the reset is generated internally in the core, and it gets activated when either the input reset (pwr_on_rst_n in EQOS-DMA, hreset_n in EQOS-AHB, or aresetn_i in EQOS-AXI) or the software reset is asserted. **Exists:** !DWC_EQOS_SINGLE_PHY_INTF **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** Asynchronous **Registered:** No
phy_intr_i	I	PHY Interrupt Input. When this signal is high, it indicates an interrupt event in the PHY. **Exists:** Always **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** Asynchronous **Registered:** Yes

Port Name	I/O	Description
phy_txen_o	O	PHY Transmit Data Enable. The MAC drives this signal. This signal has multiple functions depending on the selected PHY interface as described in the following list: ■ GMII/MII: When high, indicates that valid data is being transmitted on the phy_txd_o bus. Synchronous to: clk_tx_i ■ RMII: When high, indicates that valid data is being transmitted on the phy_txd_o bus. Synchronous to: clk_rmii_i ■ SMII: This is the Global synchronization signal for SMII. This signal is not present or valid when global sync signal is an input in SSSMII configuration. Synchronous to: clk_tx_125_i ■ RGMII: This signal is the control signal (rgmii_tctl) for the Transmit data, and it is driven on both edges of the clock. Synchronous to: clk_tx_i, clk_tx_180_i ■ SGMII or TBI: Contains Bit 8 of Transmit code group. Synchronous to: clk_tx_125_i ■ RTBI: This signal contains two bits (Bit 4 and Bit 9) of the 10-bit RTBI transmit code-group and it is driven on both edges of the clock. Synchronous to: clk_tx_125_i, clk_tx_125_180_i. ■ RevMII: When this signal is high, it indicates that valid data is being transmitted on the phy_txd_o bus. Synchronous to: clk_revmii_rx_i **Exists:** !(DWC_EQOS_SMII_INTF_ONLY&&DWC_EQOS_SSSMII_TXSYNC_IN) **Power Domain:** MAC_ON **Active State:** N/A **Synchronous to:** Depends the selected PHY interface. See description. **Registered:** This signal is a registered output only when a single PHY interface (except RGMII or RTBI) is selected. When MUX logic is present, it is unregistered.

Port Name	I/O	Description
phy_txer_o	O	PHY Transmit Error. The MAC drives this signal. It has multiple functions depending on the selected PHY interface as described in the following list: ■ GMII: When this signal is high, it indicates a Transmit error or carrier extension on the phy_txd_o bus. Synchronous to:clk_tx_i ■ MII, RMII, RGMII, SMII, or RTBI: Not used and tied to low. According to the IEEE standard, the TX_ER signal (phy_txer_o signal in the DWC_ether_qos core) is optional in the MII mode. However, if you have enabled the Energy Efficient Ethernet feature, the phy_txer_o signal is required. ■ SGMII or TBI: Contains Bit 9 of the Transmit code group. Synchronous to: clk_tx_125_i ■ RevMII: When this signal is high, it indicates a Transmit error or carrier extension on the phy_txd_o bus. Synchronous to: clk_revmii_rx_i **Note**: If you have configured the core for Single PHY interface, this output is present only for the required interface such as only in the GMII, SGMII, or TBI interface. **Exists:** DWC_EQOS_GMII_INTF_ONLY||(DWC_EQOS_PCS_EN&&!DWC_EQOS_RTBI_INTF_ONLY)||!DWC_EQOS_SINGLE_PHY_INTF||DWC_EQOS_REVMII_EN **Power Domain:** MAC_ON **Active State:** N/A **Synchronous to:** Depends the selected PHY interface. See description. **Registered:** This signal is a registered output only when a single PHY interface is selected. When MUX logic is present, it is unregistered.

Port Name	I/O	Description
phy_txd_o[(DWC_EQOS_TXD_IOW-1):0]	O	PHY Transmit Data. This is a group of transmit data signals driven by the MAC. It has multiple functions depending on the selected PHY interface as described in the following list: ■ GMII: All eight bits provide the GMII transmit data byte. The data is valid only when the phy_txen_o and phy_txer_o signals are high. Synchronous to: clk_tx_i ■ MII: Bits[3:0] provide the MII transmit data nibble. The data is valid only when the phy_txen_o and phy_txer_o signals are high. Synchronous to: clk_tx_i ■ RGMII: Bits[3:0] provide the RGMII transmit data. The data bus changes with both rising and falling edges of the transmit clock (clk_tx_i). The data is valid only when the phy_txen_o signal is high. Synchronous to: clk_tx_i and clk_tx_180_i ■ RMII: Bits[1:0] provide the RMII transmit data. The data is valid only when the phy_txen_o signal is high. Synchronous to: clk_rmii_i ■ SMII: Bit[0] provides the SMII transmit data. Synchronous to: clk_tx-_125_i ■ SGMII or TBI: The eight bits correspond to the transmit code group [7:0]. Synchronous to: clk_tx_125_i ■ RTBI: Bits[3:0] provide the eight bits (Bits[3:0] and Bits[8:5]) of the RTBI 10-bit Transmit code group. The data bus changes with both edges of the Transmit clock. Synchronous to: clk_tx_125_i and clk_tx_125_180_i ■ RevMII: All eight bits provide the RevMII transmit data byte. The data is valid only when the phy_txen_o and phy_txer_o signals are high. Synchronous to: clk_revmii_rx_i Unused bits in the RGMII, RTBI, RMII, and MII interface configurations are tied to low. **Exists:** Always **Power Domain:** MAC_ON **Active State:** N/A **Synchronous to:** Depends the selected PHY interface. See description. **Registered:** This signal is a registered output only when a single PHY interface (except RGMII or RTBI) is selected. When MUX logic is present, it is unregistered.
phy_crs_i	I	PHY CRS. This signal is valid only in the GMII or MII mode. The PHY drives this signal high when the Transmit or Receive medium is not idle. The PHY drives this signal low when both Transmit and Receive mediums are idle. This signal is not synchronous to any clock. **Exists:** DWC_EQOS_GMII_INTF_ONLY||!DWC_EQOS_SINGLE_PHY_INTF **Power Domain:** MAC_ON **Active State:** High **Synchronous to:** Asynchronous **Registered:** Yes

Port Name	I/O	Description
phy_col_i	I	PHY Collision. This signal is valid only in the GMII or MII mode. The PHY drives this signal high when a collision is detected on the medium. This signal is not synchronous to any clock. **Exists:** DWC_EQOS_GMII_INTF_ONLY||!DWC_EQOS_SINGLE_PHY_INTF **Power Domain:** MAC_ON **Active State:** High **Synchronous to:** Asynchronous **Registered:** Yes
phy_rxdv_i	I	PHY Receive Data Valid. The PHY drives this signal. It has multiple functions depending on the selected PHY interface as described in the following list: ■ GMII or MII: Indicates that the data on the phy_rxd_i bus is valid. It remains high continuously from the first recovered byte or nibble of the packet through the final recovered byte or nibble of the packet. Synchronous to: clk_rx_i ■ RGMII: This is the Receive control signal that is used to qualify the data received phy_rxd_i. This signal is sampled both edges of the clock. Synchronous to: clk_rx_i, clk_rx_180_i ■ RMII: Contains the crs and data valid information of the Receive interface. Synchronous to: clk_rmii_i ■ SMII: This is the Receive Global synchronization signal input for SSSMII. This signal is not present or valid when SMII is not configured for SSSMII. Synchronous to: clk_tx_125_i ■ SGMII: Contains Bit 8 of the Receive code group. Synchronous to: clk_rx_125_i ■ TBI: Contains Bit 8 of the Receive code group. Synchronous to: clk_pmarx0_i, clk_pmarx1_i ■ RTBI: This contains two bits (Bit 4 and Bit 9) of the 10-bit Receive code-group and is sampled at both edges of the clock. Synchronous to: clk_rx_125_i and clk_rx_180_i ■ RevMII: When high, indicates that the data on the phy_rxd bus is valid. It remains high continuously from the first recovered byte or nibble of the packet through the final recovered byte or nibble of the packet. Synchronous to: clk_revmii_tx_i **Exists:** !(DWC_EQOS_SMII_INTF_ONLY&&!DWC_EQOS_SSSMII_EN) **Power Domain:** MAC_ON **Active State:** N/A **Synchronous to:** Depends the selected PHY interface. See description. **Registered:** Yes, except when RTBI is selected along with TBI or SGMII

Port Name	I/O	Description
phy_rxer_i	I	PHY Receive Error. The PHY drives this signal. It has multiple functions depending on the selected PHY interface as described in the following list: ■ GMII or MII: Indicates an error or carrier extension (in GMII) in the received packet the phy_rxd_i bus. Synchronous to: clk_rx_i ■ RGMII, RMII, SMII, or RTBI: This signal is not used. ■ SGMII: Contains Bit 9 of the receive code group. Synchronous to: clk_rx_125_i ■ TBI: Contains Bit 9 of the receive code group. Synchronous to: clk_pmarx0_i, clk_pmarx1_i ■ RevMII: Indicates an error or carrier extension (in GMII) in the received packet the phy_rxd_i bus. Synchronous to: clk_revmii_tx_i **Note:** If you have configured the core for Single PHY interface, this input is present only for the required interface such as only in GMII, MII, SGMII, or TBI. **Exists:** DWC_EQOS_GMII_INTF_ONLY||(DWC_EQOS_PCS_EN&&!DWC_EQOS_RTBI_INTF_ONLY)||!DWC_EQOS_SINGLE_PHY_INTF||DWC_EQOS_REVMII_EN **Power Domain:** MAC_ON **Active State:** N/A **Synchronous to:** Depends the selected PHY interface. See description. **Registered:** Yes

Port Name	I/O	Description
phy_rxd_i[(DWC_EQOS_RXD_IOW-1):0]	I	PHY Receive Data. This is a group of data signals received from the PHY. It has multiple functions depending on the selected PHY interface as described in the following list: ■ GMII: All eight bits provide the GMII receive data byte. The data is valid only when the phy_rxdv_i and phy_rxer_i signals are high. Synchronous to: clk_rx_i ■ MII: Bits [3:0] provide the MII receive data nibble. The data is valid only when the phy_rxdv_i and phy_rxer_i signals are high. Synchronous to: clk_rx_i ■ RGMII: Bits [3:0] provide the RGMII receive data. The data bus is sampled with both rising and falling edges of the receive clock (clk_rx_i). The data is valid only when the phy_rxdv_i signal is high. Synchronous to: clk_rx_i, clk_rx_180_i ■ RMII: Bits [1:0] provide the RMII receive data. he data is valid only when the phy_rxdv_i signal is high. Synchronous to: clk_rmii_i ■ SMII: Bit[0] provides the SMII receive data. Synchronous to: clk_tx-_125_i ■ SGMII: The eight bits correspond to the receive code group [7:0]. Synchronous to: clk_rx_125_i ■ TBI: The eight bits correspond to the receive code group [7:0]. Synchronous to: clk_pmarx0_i, clk_pmarx1_i ■ RTBI: Bits [3:0] provides the eight bits (Bits[3:0] and Bits[8:5]) of the 10-bit receive code-group and is sampled at both edges of the clock. Synchronous to: clk_rx_125_i and clk_rx_180_i ■ RevMII: All eight bits provide the RevMII receive data byte. The data is valid only when the phy_rxdv_i and phy_rxer_i signals are high. Synchronous to: clk_revmii_tx_i **Exists:** Always **Power Domain:** MAC_ON **Active State:** N/A **Synchronous to:** Depends the selected PHY interface. See description. **Registered:** Yes, except when RTBI is selected along with TBI/SGMII.

16.17 SMA Interface Signals

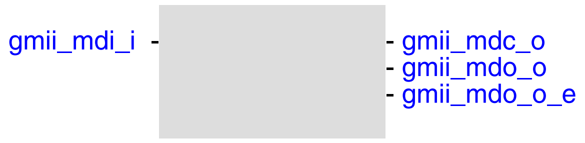


Table 16-17 SMA Interface Signals

Port Name	I/O	Description
gmii_mdc_o	O	Management Data Clock. In non-RevMII mode, the MAC provides timing reference for the gmii_mdi_i and gmii_mdo_o signals on GMII or MII through this aperiodic clock. This clock is generated from the application clock through a clock divider controlled by CR field of MAC_MDIO_Address register. **Exists:** DWC_EQOS_SMA_EN **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** CSR clock **Registered:** Yes
gmii_mdi_i	I	Management Data In. The PHY generates this signal to transfer register data during a read operation. This signal is driven synchronously with the gmii_mdc_o clock. In the RevMII mode, this signal transfers the control information and data from the remote MAC. **Exists:** DWC_EQOS_SMA_EN||DWC_EQOS_REVMII_EN **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** CSR clock or revmii_mdc_i **Registered:** Yes
gmii_mdo_o	O	Management Data Out. The MAC uses this signal to transfer control and data information to the PHY. In RevMII mode, this signal gives the data output during Read operations initiated by the remote MAC. **Exists:** DWC_EQOS_SMA_EN||DWC_EQOS_REVMII_EN **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** CSR clock or revmii_mdc_i **Registered:** Yes

Port Name	I/O	Description
gmii_mdo_o_e	O	Management Data Output Enable. This signal drives the gmii_mdo_o signal from an external three-state I/O buffer. This signal is high when valid data is driven on the gmii_mdo_o signal. **Exists:** DWC_EQOS_SMA_EN||DWC_EQOS_REVMII_EN **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** CSR clock or revmii_mdc_i **Registered:** Yes

16.18 APB Interface Signals

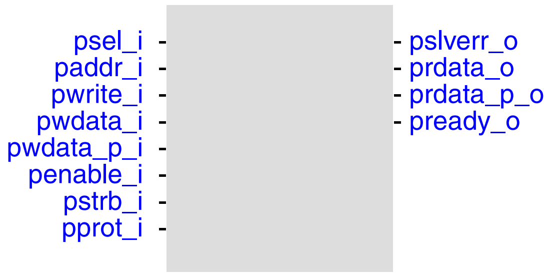


Table 16-18 APB Interface Signals

Port Name	I/O	Description
pslverr_o	O	APB Slave Error Response. When this signal is high, it indicates that the application has accessed reserved register offset address. This feature is enabled when SEEN bit in MAC_CSR_SW_Ctrl register is programmed to 1, otherwise the signal is always low. **Exists:** DWC_EQOS_SLVERR&&DWC_EQOS_APB3_SLAVE **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** CSR clock **Registered:** Yes
psel_i	I	APB Slave Select. This signal indicates the start of transaction and the validity of the control signals and the pwdata_i signal. **Exists:** DWC_EQOS_APB_SLAVE||DWC_EQOS_APB3_SLAVE **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** CSR clock **Registered:** No
paddr_i[13:0]	I	APB Address. This signal carries the register address of the CSR module. The address is valid only when signal psel_i is high. **Note:** paddr_i[13] should be connected to 0. **Exists:** DWC_EQOS_APB_SLAVE||DWC_EQOS_APB3_SLAVE **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** CSR clock **Registered:** No

Port Name	I/O	Description
pwrite_i	I	APB Read or Write. When this signal is high, it indicates a Write operation. When this signal is low, it indicates a Read operation. **Exists:** DWC_EQOS_APB_SLAVE||DWC_EQOS_APB3_SLAVE **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** CSR clock **Registered:** No
pwdata_i[31:0]	I	APB Write Data. This signal carries the Write data from the application for Control registers of the CSR module. **Exists:** DWC_EQOS_APB_SLAVE||DWC_EQOS_APB3_SLAVE **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** CSR clock **Registered:** Yes
pwdata_p_i[3:0]	I	APB Write Parity Data. This signal carries the Write Parity data from the application for Control registers of the CSR module. **Exists:** (DWC_EQOS_APB_SLAVE||DWC_EQOS_APB3_SLAVE)&&DWC_EQOS_ASP_PPE **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** CSR clock **Registered:** Yes
penable_i	I	APB Enable. When high, this signal completes the Read or Write transaction cycle. **Exists:** DWC_EQOS_APB_SLAVE||DWC_EQOS_APB3_SLAVE **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** CSR clock **Registered:** No

Port Name	I/O	Description
prdata_o[31:0]	O	APB Read Data. This signal outputs the Read data from the CSR module to the APB bus. When APB Slave is selected, valid data is output 1 clock cycle after the psel_i signal is high. If the psel_i signal continues to be high for burst transfers, valid data is output every 2 clock cycles during this period. When APB3/APB4 Slave is selected, valid data is output with the pready_o signal 2 clock cycles after the psel_i signal is high. **Exists:** DWC_EQOS_APB_SLAVE||DWC_EQOS_APB3_SLAVE **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** CSR clock **Registered:** Yes
prdata_p_o[3:0]	O	APB Parity Read Data. This signal outputs the Parity Read data from the CSR module to the APB bus. When APB Slave is selected, valid data is output 1 clock cycle after the psel_i signal is high. If the psel_i signal continues to be high for burst transfers, valid data is output every 2 clock cycles during this period. When APB3/APB4 Slave is selected, valid data is output with the pready_o signal 2 clock cycles after the psel_i signal is high. **Exists:** (DWC_EQOS_APB_SLAVE||DWC_EQOS_APB3_SLAVE)&&DWC_EQOS_ASP_PPE **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** CSR clock **Registered:** Yes
pready_o	O	APB Read Data Ready. This signal indicates that the data on the prdata_o signal is valid and can be accepted by the application. For writes, this signal indicates that the data on the pwdata_i signal is accepted by DWC_ether_qos. **Exists:** (DWC_EQOS_CSR_PORT==4||DWC_EQOS_CSR_PORT==6) **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** CSR clock **Registered:** Yes
pstrb_i[3:0]	I	APB4 Write strobes. This signal indicates which of the data bytes on the pwdata_i signal are valid. **Exists:** (DWC_EQOS_CSR_PORT==6) **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** CSR clock **Registered:** Yes

Port Name	I/O	Description
pprot_i[2:0]	I	APB4 Protection control. This signal indicates the normal, privileged, or secure protection level of the transaction and whether the transaction is a data access or an instruction access. The DWC_ether_qos ignores the content of this input signal. **Exists:** (DWC_EQOS_CSR_PORT==6) **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** CSR clock **Registered:** Yes

16.19 AXI Master Interface Signals

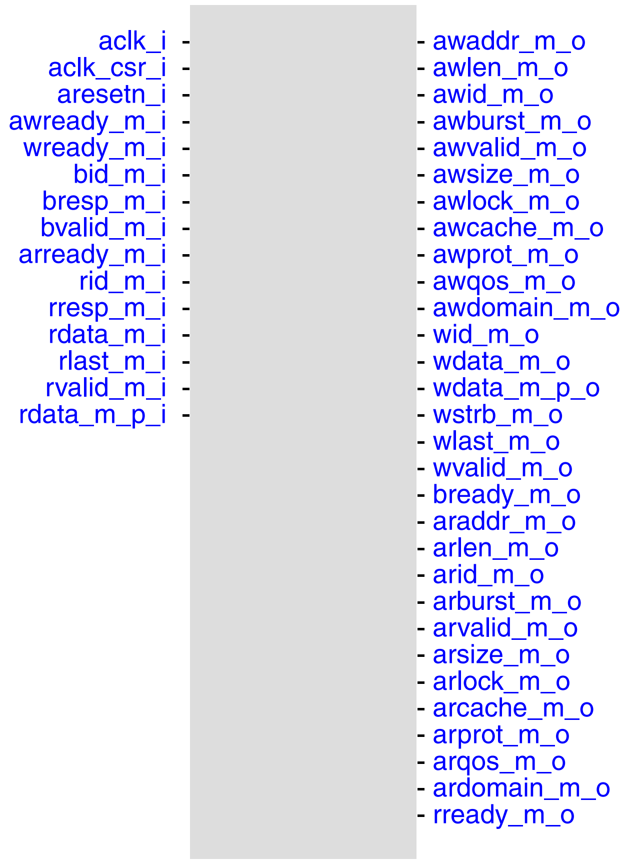


Table 16-19 AXI Master Interface Signals

Port Name	I/O	Description
aclk_i	I	AMBA AXI System Clock. This is the free-running AXI clock input provided by the application. The DMA and the MTL modules also operate with this clock. The minimum valid frequency is 25 MHz (it is ~1.5 times MAC transmitter/receiver in SPRAM configurations) and the maximum valid frequency depends on frequency at which logic timing is met. **Exists:** DWC_EQOS_SYS==4||DWC_EQOS_SYS==5 **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** N/A **Registered:** N/A

Port Name	I/O	Description
aclk_csr_i	I	Separate AXI Clock Port for Slave. This is separate AXI clock port for Slave when same Master-Slave clock is selected in configuration. This is provided so that the clock to application logic can be gated off while the clock to slave logic is active. **Exists:** DWC_EQOS_ASLV_CLK **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** aclk_i **Registered:** N/A
aresetn_i	I	AMBA AXI Power On System Reset. This signal is synchronous to aclk_i, the reset de-assertion must be synchronous to the aclk_i, the assertion can be synchronous or asynchronous based on the reset mode selected in configuration. **Exists:** DWC_EQOS_SYS==4||DWC_EQOS_SYS==5 **Power Domain:** MAC_OFF **Active State:** Low **Synchronous to:** aclk_i **Registered:** No
awaddr_m_o[(DWC_EQOS_AXI_GM_ADDR_WIDTH-1):0]	O	AXI Master Write Address. The Write address bus gives the address of the first transfer in a Write burst transaction. The associated control signals are used to determine the addresses of the remaining transfers in the burst. **Exists:** DWC_EQOS_SYS==4||DWC_EQOS_SYS==5 **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** aclk_i **Registered:** Yes
awlen_m_o[(DWC_EQOS_BLW-1):0]	O	AXI Master Write Burst Length. This signal gives the exact number of transfers in a burst. This information determines the number of data transfers associated with the address. The default width is 4 to support 16-beats transfer but it can be configured to up to 8-bit to support 256 beats transfers. The value of all-zeros indicates a single transfer. The minimum width of awlen_m_o signal is 4 (DWC_EQOS_AXI_BL = 16) and the maximum is 8 (DWC_EQOS_AXI_BL = 256). **Exists:** DWC_EQOS_SYS==4||DWC_EQOS_SYS==5 **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** aclk_i **Registered:** Yes

Port Name	I/O	Description
awid_m_o[(DWC_EQOS_AXI_GM_AXI_ID_WIDTH-1):0]	O	AXI Master Write Address ID. This signal is the identification tag for Write address group of signals. You can configure the width while configuring the core. **Exists:** DWC_EQOS_SYS==4||DWC_EQOS_SYS==5 **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** aclk_i **Registered:** Yes
awburst_m_o[1:0]	O	AXI Master Write Burst Type. The burst type, along with the size information, specifies how the address for each transfer within the burst is calculated. This is always hard wired to 2'b01 to indicate INCR burst type. **Exists:** DWC_EQOS_SYS==4||DWC_EQOS_SYS==5 **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** aclk_i **Registered:** N/A
awvalid_m_o	O	AXI Master Write Address Valid. This signal indicates that valid write address and control information are available: ■ 1: Address and control information available ■ 0: Address and control information not available The address and control information remain stable until the address acknowledge signal awready goes high. **Exists:** DWC_EQOS_SYS==4||DWC_EQOS_SYS==5 **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** aclk_i **Registered:** Yes
awready_m_i	I	AXI Master Write Address Ready. This signal indicates that the AXI slave is ready to accept an address and associated control signals: ■ 1: Slave ready ■ 0: Slave not ready **Exists:** DWC_EQOS_SYS==4||DWC_EQOS_SYS==5 **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** aclk_i **Registered:** No

Port Name	I/O	Description
awsize_m_o[2:0]	O	AXI Master Burst Size. This signal indicates the size of each transfer in the burst. Byte lane strobes indicate exactly which byte lanes to update. The value driven is always equal to the configured databus width (DWC_EQOS_DATAWIDTH): ■ 3'b010 for 32-bit interface ■ 3'b011 for 64-bit interface ■ 3'b100 for 128-bit interface ■ 3'b101 Not used **Exists:** DWC_EQOS_SYS==4||DWC_EQOS_SYS==5 **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** aclk_i **Registered:** N/A
awlock_m_o[1:0]	O	AXI Master Write Locked Access. This signal indicates the type of locked access to the Write address channel of the AXI Master. The width of this signal is 1 and 2 in AXI4 and AXI3 interfaces respectively. It is always driven to 0. **Exists:** DWC_EQOS_SYS==4||DWC_EQOS_SYS==5 **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** aclk_i **Registered:** No
awcache_m_o[3:0]	O	AXI Master Cache Type. This signal indicates the bufferable, cacheable, write-through, write-back, and allocate attributes of the transaction. When AXI3 Master is selected, this is always hard wired to 4'b0000 to indicate non-cacheable and non-bufferable data access. When AXI4 Master is selected, this is driven with value programmed in RDC/RHC/RPC/RDWC fields of AXI4_Rx_AW_ACE_Control registers based on the transfer type. **Exists:** DWC_EQOS_SYS==4||DWC_EQOS_SYS==5 **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** aclk_i **Registered:** N/A

Port Name	I/O	Description
awprot_m_o[2:0]	O	AXI Master Protection Type. This signal indicates the normal, privileged, or secure protection level of the transaction and whether the transaction is a data access or an instruction access. This is always hard wired to 3'b000 to indicate normal and secure data access. **Exists:** DWC_EQOS_SYS==4||DWC_EQOS_SYS==5 **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** aclk_i **Registered:** N/A
awqos_m_o[3:0]	O	AXI Master Write QOS Indicator. This is driven with TQOS field of per channel DMA_CH[n]_TX_Control register based on which Transmit DMA channel is accessing the AXI bus. **Exists:** DWC_EQOS_SYS == 5 **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** aclk_i **Registered:** N/A
awdomain_m_o[1:0]	O	AXI Master Write Domain Indicator. This signal indicates the shareability domain of a write transaction. This is driven with value programmed in RDD/RHD/RPD/RDWD fields of AXI4_Rx_AW_ACE_Control registers based on the transfer type. **Exists:** DWC_EQOS_SYS == 5 **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** aclk_i **Registered:** N/A
wid_m_o[(DWC_EQOS_AXI_GM_AXI_ID_WIDTH-1):0]	O	AXI Master Write ID Tag. This signal is the ID tag of the Write data transfer. The WID value must match the AWID value of the write transaction. This signal is not specified in AXI4 standard and can be left unconnected. **Exists:** DWC_EQOS_SYS==4||DWC_EQOS_SYS==5 **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** aclk_i **Registered:** Yes

Port Name	I/O	Description
wdata_m_o[(DWC_EQOS_AXI_GM_DATA_WIDTH-1):0]	O	AXI Master Write Data. The write data bus can be 32-bit, 64-bit, or 128-bit wide and takes the value of DWC_EQOS_DATAWIDTH parameter. **Exists:** DWC_EQOS_SYS==4||DWC_EQOS_SYS==5 **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** aclk_i **Registered:** Yes
wdata_m_p_o[(((DWC_EQOS_DATAWIDTH==32)?4:((DWC_EQOS_DATAWIDTH==64)?8:16))-1):0]	O	AXI Master Write Data Parity. The parity data bus width can be 4-bit, 8-bit, or 16-bit wide and takes the value of DWC_EQOS_DPW parameter. **Exists:** DWC_EQOS_AXI_SUBSYS&&DWC_EQOS_ASP_PPE **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** aclk_i **Registered:** Yes
wstrb_m_o[(DWC_EQOS_STRW-1):0]	O	AXI Master Write Strobes. This signal indicates which byte lanes to update in memory. There is one Write strobe for each eight bits of Write data bus. **Exists:** DWC_EQOS_SYS==4||DWC_EQOS_SYS==5 **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** aclk_i **Registered:** Yes
wlast_m_o	O	AXI Master Write Last. This signal indicates the last transfer in a Write burst. **Exists:** DWC_EQOS_SYS==4||DWC_EQOS_SYS==5 **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** aclk_i **Registered:** Yes
wvalid_m_o	O	AXI Master Write Valid. This signal indicates that valid Write data and strobes are available: ■ 1: Write data and strobes available ■ 0: Write data and strobes not available **Exists:** DWC_EQOS_SYS==4||DWC_EQOS_SYS==5 **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** aclk_i **Registered:** Yes

Port Name	I/O	Description
wready_m_i	I	AXI Master Write Ready. This signal indicates that the slave accepted the Write data: ■ 1: Slave accepted ■ 0: Slave has not accepted **Exists:** DWC_EQOS_SYS==4||DWC_EQOS_SYS==5 **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** aclk_i **Registered:** No
bid_m_i[(DWC_EQOS_AXI_GM_AXI_ID_WIDTH-1):0]	I	AXI Master Write Response ID. This signal is the identification tag of Write response. The BID value must match the AWID value of the write transaction to which the slave is responding. **Exists:** DWC_EQOS_SYS==4||DWC_EQOS_SYS==5 **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** aclk_i **Registered:** No
bresp_m_i[1:0]	I	AXI Master Write Response. This signal indicates the status of the Write transaction. The valid responses are OKAY (2'b00), EXOKAY (2'b01), SLVERR (2'b10), and DECERR (2'b11). **Exists:** DWC_EQOS_SYS==4||DWC_EQOS_SYS==5 **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** aclk_i **Registered:** No
bvalid_m_i	I	AXI Master Write Response Valid. This signal indicates that a valid Write response is available: ■ 1: Write response available ■ 0: Write response not available **Exists:** DWC_EQOS_SYS==4||DWC_EQOS_SYS==5 **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** aclk_i **Registered:** No

Port Name	I/O	Description
bready_m_o	O	AXI Master Write Response Ready. This signal indicates that the master can accept the response information. ■ 1: Master ready ■ 0: Master not ready This signal is always high. **Exists:** DWC_EQOS_SYS==4||DWC_EQOS_SYS==5 **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** aclk_i **Registered:** Yes
araddr_m_o[(DWC_EQOS_AXI_GM_ADDR_WIDTH-1):0]	O	AXI Master Read Address. The Read address bus gives the initial address of a Read burst transaction. It provides the start address of the burst and the control signals that are issued with the address. The control signals specify how the address is calculated for remaining transfers in the burst. **Exists:** DWC_EQOS_SYS==4||DWC_EQOS_SYS==5 **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** aclk_i **Registered:** Yes
arlen_m_o[(DWC_EQOS_BLW-1):0]	O	AXI Master Read Burst Length. The burst length gives the exact number of transfers in a burst. This information determines the number of data transfers associated with the address. The default width is 4 but width of this signal can be increased during RTL configuration to support bigger bursts. **Exists:** DWC_EQOS_SYS==4||DWC_EQOS_SYS==5 **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** aclk_i **Registered:** Yes
arid_m_o[(DWC_EQOS_AXI_GM_AXI_ID_WIDTH-1):0]	O	AXI Master Read Address ID. This signal is the identification tag for the read address group of signals. The width of this signal is configured during core configuration. **Exists:** DWC_EQOS_SYS==4||DWC_EQOS_SYS==5 **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** aclk_i **Registered:** Yes

Port Name	I/O	Description
arburst_m_o[1:0]	O	AXI Master Read Burst Type. The burst type, along with the size information, specifies how the address for each transfer within the burst is calculated. This is always hard wired to 2'b01 to indicate INCR burst type. **Exists:** DWC_EQOS_SYS==4||DWC_EQOS_SYS==5 **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** aclk_i **Registered:** N/A
arvalid_m_o	O	AXI Master Read Address Valid. When this signal is high, it indicates that the Read address and Control information is valid. This signal remains stable until the address acknowledge signal arready_m_i is high. ■ 1: Address and control information valid ■ 0: Address and control information not valid **Exists:** DWC_EQOS_SYS==4||DWC_EQOS_SYS==5 **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** aclk_i **Registered:** Yes
arready_m_i	I	AXI Master Read Ready. This signal indicates that the slave can accept the Read request: ■ 1: Slave ready ■ 0: Slave not ready **Exists:** DWC_EQOS_SYS==4||DWC_EQOS_SYS==5 **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** aclk_i **Registered:** No
arsize_m_o[2:0]	O	AXI Master Burst Length. This signal indicates the size of each transfer in the burst. This signal is always driven to have the System Data Width transfers: ■ 3'b010 for 32-bit ■ 3'b011 for 64-bit ■ 3'b100 for 128-bit interface **Exists:** DWC_EQOS_SYS==4||DWC_EQOS_SYS==5 **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** aclk_i **Registered:** N/A

Port Name	I/O	Description
arlock_m_o[1:0]	O	AXI Master Read Locked Access. This signal indicates the type of locked access to the Read address channel of AXI master. The width of this signal is 1 and 2 in AXI4 and AXI3 interfaces respectively. It is always driven to 0. **Exists:** DWC_EQOS_SYS==4||DWC_EQOS_SYS==5 **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** aclk_i **Registered:** No
arcache_m_o[3:0]	O	AXI Master Cache Type. This signal indicates the bufferable, cacheable, write-through, write-back, and allocate attributes of the transaction. When AXI3 Master is selected, this is always hard wired to 4'b0000 to indicate non-cacheable and non-bufferable data access. When AXI4 Master is selected, this is driven with value programmed in THC/TEC/TDRC fields of AXI4_Tx_AR_ACE_Control registers based on the transfer type. **Exists:** DWC_EQOS_SYS==4||DWC_EQOS_SYS==5 **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** aclk_i **Registered:** N/A
arprot_m_o[2:0]	O	AXI Master Protection Type. This signal provides protection unit information for the transaction. This is always hard wired to 3'b000 to indicate normal and secure data access. **Exists:** DWC_EQOS_SYS==4||DWC_EQOS_SYS==5 **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** aclk_i **Registered:** N/A
arqos_m_o[3:0]	O	AXI Master Read QOS Indicator. This is driven with RQOS field of per channel DMA_CH[n]_RX_Control register based on which Receive DMA channel is accessing the AXI bus. **Exists:** DWC_EQOS_SYS == 5 **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** aclk_i **Registered:** N/A

Port Name	I/O	Description
ardomain_m_o[1:0]	O	AXI Master Read Domain Indicator. This signal indicates the shareability domain of a read transaction. This is driven with value programmed in THD/TED/TDRD fields of AXI4_Tx_AR_ACE_Control registers based on the transfer type. **Exists:** DWC_EQOS_SYS == 5 **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** aclk_i **Registered:** N/A
rid_m_i[(DWC_EQOS_AXI_GM_AXI_ID_WIDTH-1):0]	I	AXI Master Read ID Tag. This signal is the ID tag of the read data group of signals. The rid value is generated by the slave and must match the arid value of Read transaction to which it is responding. **Exists:** DWC_EQOS_SYS==4||DWC_EQOS_SYS==5 **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** aclk_i **Registered:** No
rresp_m_i[1:0]	I	AXI Master Read Response. This signal indicates the status of the Read transfer. The valid responses are: ■ OKAY(2'b00) ■ EXOKAY(2'b01) ■ SLVERR(2'b10) ■ DECERR(2'b11). **Exists:** DWC_EQOS_SYS==4||DWC_EQOS_SYS==5 **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** aclk_i **Registered:** No
rdata_m_i[(DWC_EQOS_AXI_GM_DATA_WIDTH-1):0]	I	AXI Master Read Data. The Read data bus takes the value of DWC_EQOS_DATAWIDTH parameter. **Exists:** DWC_EQOS_SYS==4||DWC_EQOS_SYS==5 **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** aclk_i **Registered:** No

Port Name	I/O	Description
rlast_m_i	I	AXI Master Read Last. This signal indicates the last transfer in a Read burst. **Exists:** DWC_EQOS_SYS==4||DWC_EQOS_SYS==5 **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** aclk_i **Registered:** No
rvalid_m_i	I	AXI Master Read Valid. This signal indicates that the required Read data is available and the Read transfer can complete: ■ 1: Read data available ■ 0: Read data not available **Exists:** DWC_EQOS_SYS==4||DWC_EQOS_SYS==5 **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** aclk_i **Registered:** No
rready_m_o	O	AXI Master Read Ready. This signal indicates that the master can accept the read data and response information: ■ 1: Master ready ■ 0: Master not ready This signal is always high. However, it can go low for 1 or 2 cycles when checksum insertion is enabled for IP packets. It is specifically designed in this way to avoid temporary buffers in datapath while checksums are being written. **Exists:** DWC_EQOS_SYS==4||DWC_EQOS_SYS==5 **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** aclk_i **Registered:** Yes
rdata_m_p_i[(((DWC_EQOS_DATAWIDTH==32)?4:((DWC_EQOS_DATAWIDTH==64)?8:16))-1):0]	I	AXI Master Read Data Parity. The Read data bus takes the value of DWC_EQOS_DPW parameter. **Exists:** DWC_EQOS_AXI_SUBSYS&&DWC_EQOS_ASP_PPE **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** aclk_i **Registered:** No

16.20 AXI Slave Interface Signals

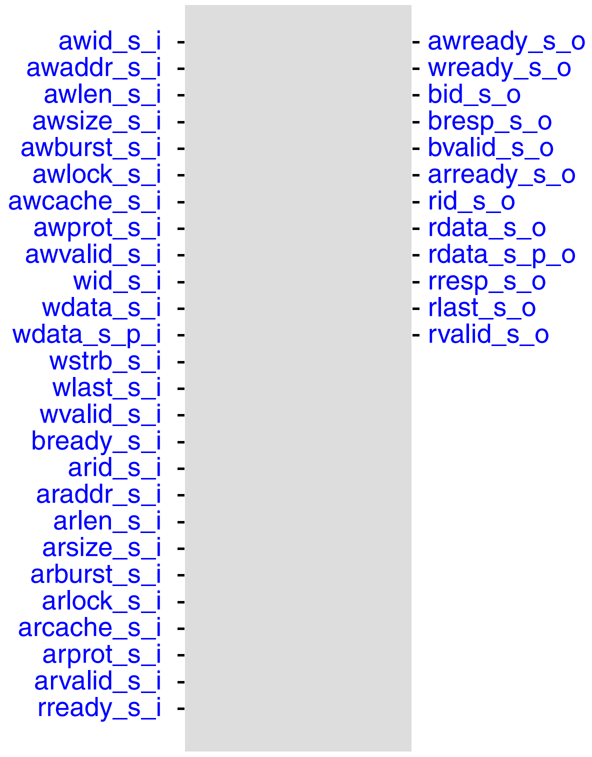


Table 16-20 AXI Slave Interface Signals

Port Name	I/O	Description
awid_s_i[(DWC_EQOS_GS_ID-1):0]	I	AXI Slave Write Address ID. This signal is the identification tag for Write address group of signals. **Exists:** (DWC_EQOS_CSR_PORT==3||DWC_EQOS_CSR_PORT==5) **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** aclk_i **Registered:** No
awaddr_s_i[(DWC_EQOS_AXI_GM_ADDR_WIDTH-1):0]	I	AXI Slave Write Address. The Write address bus gives the address of the first transfer in a Write burst transaction. The associated control signals are used to determine the addresses of the remaining transfers in the burst. **Exists:** (DWC_EQOS_CSR_PORT==3||DWC_EQOS_CSR_PORT==5) **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** aclk_i **Registered:** No

Port Name	I/O	Description
awlen_s_i[(DWC_EQOS_BLW-1):0]	I	AXI Slave Write Burst Length. The burst length gives the exact number of transfers in a burst. This information determines the number of data transfers associated with the address. **Exists:** DWC_EQOS_AXI_SLAVE&&!DWC_EQOS_AXI4_LITE_SLAVE **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** aclk_i **Registered:** No
awsize_s_i[2:0]	I	AXI Slave Burst Size. This signal indicates the size of each transfer in the burst. Byte lane strobes indicate exactly which byte lanes to update. **Exists:** DWC_EQOS_AXI_SLAVE&&!DWC_EQOS_AXI4_LITE_SLAVE **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** aclk_i **Registered:** No
awburst_s_i[1:0]	I	AXI Slave Burst Type. The burst type, along with the size information, specifies how the address for each transfer within the burst is calculated. **Exists:** DWC_EQOS_AXI_SLAVE&&!DWC_EQOS_AXI4_LITE_SLAVE **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** aclk_i **Registered:** No
awlock_s_i[1:0]	I	AXI Slave Write Locked Access. This signal indicates the type of locked access to the Write address channel of the AXI Slave. The DWC_ether_qos ignores the value of this signal. **Exists:** DWC_EQOS_AXI_SLAVE&&!DWC_EQOS_AXI4_LITE_SLAVE **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** aclk_i **Registered:** No

Port Name	I/O	Description
awcache_s_i[3:0]	I	AXI Slave Cache Type. This signal indicates the bufferable, cacheable, write-through, write-back, and allocate attributes of the transaction. The DWC_ether_qos core does not use this input because it supports only normal data transactions. **Exists:** DWC_EQOS_AXI_SLAVE&&!DWC_EQOS_AXI4_LITE_SLAVE **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** aclk_i **Registered:** N/A
awprot_s_i[2:0]	I	AXI Slave Protection Type. This signal indicates the normal, privileged, or secure protection level of the transaction and whether the transaction is a data access or an instruction access. The DWC_ether_qos core does not use this input because it supports only normal data transactions. **Exists:** (DWC_EQOS_CSR_PORT==3||DWC_EQOS_CSR_PORT==5) **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** aclk_i **Registered:** N/A
awvalid_s_i	I	AXI Slave Write Address Valid. This signal indicates that valid write address and control information are available: ■ 1: Address and control information available ■ 0: Address and control information not available The address and control information remain stable until the address acknowledge signal awready_s_o goes high. **Exists:** (DWC_EQOS_CSR_PORT==3||DWC_EQOS_CSR_PORT==5) **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** aclk_i **Registered:** No

Port Name	I/O	Description
awready_s_o	O	AXI Slave Write Address Ready. This signal indicates that the slave is ready to accept an address and associated control signals: ■ 1: Slave ready ■ 0: Slave not ready **Exists:** (DWC_EQOS_CSR_PORT==3||DWC_EQOS_CSR_PORT==5) **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** aclk_i **Registered:** No
wid_s_i[(DWC_EQOS_GS_ID-1):0]	I	AXI Slave Write ID Tag. This signal is the ID tag of the Write data transfer. The wid value must match the awid value of the Write transaction. This input is not used as the AXI slave does not accept more than one write request at a time. When AXI4 is selected drive this signal to 0. **Exists:** (DWC_EQOS_CSR_PORT==3||DWC_EQOS_CSR_PORT==5) **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** aclk_i **Registered:** N/A
wdata_s_i[(DWC_EQOS_CSR_DATAWIDTH-1):0]	I	AXI Slave Write Data. The Write data bus takes the value of the DWC_EQOS_DATAWIDTH parameter. **Exists:** (DWC_EQOS_CSR_PORT==3||DWC_EQOS_CSR_PORT==5) **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** aclk_i **Registered:** Yes
wdata_s_p_i[((DWC_EQOS_CSR_DATAWIDTH/8)-1):0]	I	AXI Slave Write Parity Data. The Write parity data bus takes the value of the DWC_EQOS_CSR_DATAWIDTH/8 parameter. **Exists:** DWC_EQOS_AXI_SLAVE&&DWC_EQOS_ASP_PPE **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** aclk_i **Registered:** Yes

Port Name	I/O	Description
wstrb_s_i[((DWC_EQOS_CSR_DATAWIDTH/8)-1):0]	I	AXI Slave Write Strobes This signal indicates the byte lanes that are updated in the memory. One write strobe is equal to the eight bits of the Write data bus. **Exists:** (DWC_EQOS_CSR_PORT==3||DWC_EQOS_CSR_PORT==5) **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** aclk_i **Registered:** Yes
wlast_s_i	I	AXI Slave Write Last. This signal indicates the last transfer in a Write burst. This input is not used because the AXI master always provides the input about the required number of Write data transfers to the slave. **Exists:** DWC_EQOS_AXI_SLAVE&&!DWC_EQOS_AXI4_LITE_SLAVE **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** aclk_i **Registered:** N/A
wvalid_s_i	I	AXI Slave Write Valid. This signal indicates that valid Write data and strobes are available: ■ 1: Write data and strobes available ■ 0: Write data and strobes not available **Exists:** (DWC_EQOS_CSR_PORT==3||DWC_EQOS_CSR_PORT==5) **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** aclk_i **Registered:** No
wready_s_o	O	AXI Slave Write Ready. This signal indicates that the slave can accept the Write data: ■ 1: Slave ready ■ 0: Slave not ready **Exists:** (DWC_EQOS_CSR_PORT==3||DWC_EQOS_CSR_PORT==5) **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** aclk_i **Registered:** No

Port Name	I/O	Description
bid_s_o[(DWC_EQOS_GS_ID-1):0]	O	AXI Slave Response ID. The identification tag of the Write response. The bid value is equal to the awid value of the write transaction to which the slave is responding. **Exists:** (DWC_EQOS_CSR_PORT==3||DWC_EQOS_CSR_PORT==5) **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** aclk_i **Registered:** No
bresp_s_o[1:0]	O	AXI Slave Write Response. This signal indicates the status of the write transaction. The valid responses are OKAY(2'b00), EXOKAY(2'b01), SLVERR(2'b10), and DECERR(2'b11). The DWC_ether_qos core provides only OKAY and SLVERR responses on the bresp_s_o signal. It does not provide the EXOKAY and DECERR responses. **Exists:** (DWC_EQOS_CSR_PORT==3||DWC_EQOS_CSR_PORT==5) **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** aclk_i **Registered:** No
bvalid_s_o	O	AXI Slave Write Response valid. This signal indicates that a valid Write response is available: ■ 1: Write response available ■ 0: Write response not available **Exists:** (DWC_EQOS_CSR_PORT==3||DWC_EQOS_CSR_PORT==5) **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** aclk_i **Registered:** No
bready_s_i	I	AXI Slave Response Ready. This signal indicates that the master can accept the response information. ■ 1: Slave ready ■ 0: Slave not ready **Exists:** (DWC_EQOS_CSR_PORT==3||DWC_EQOS_CSR_PORT==5) **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** aclk_i **Registered:** No

Port Name	I/O	Description
arid_s_i[(DWC_EQOS_GS_ID-1):0]	I	AXI Slave Read Address ID. This signal is the identification tag for the Read address group of signals. **Exists:** (DWC_EQOS_CSR_PORT==3||DWC_EQOS_CSR_PORT==5) **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** aclk_i **Registered:** No
araddr_s_i[(DWC_EQOS_AXI_GM_ADDR_WIDTH-1):0]	I	AXI Slave Read Address. The Read address bus gives the initial address of a read burst transaction. It provides the start address of the burst and the control signals that are issued with the address. The control signals specify how the address is calculated for the remaining transfers in the burst. **Exists:** (DWC_EQOS_CSR_PORT==3||DWC_EQOS_CSR_PORT==5) **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** aclk_i **Registered:** No
arlen_s_i[(DWC_EQOS_BLW-1):0]	I	AXI Slave Read Burst Length. The burst length gives the exact number of transfers in a burst. This information determines the number of data transfers associated with the address. **Exists:** DWC_EQOS_AXI_SLAVE&&!DWC_EQOS_AXI4_LITE_SLAVE **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** aclk_i **Registered:** No
arsize_s_i[2:0]	I	AXI Slave Burst Size. This signal indicates the size of each transfer in the burst. **Exists:** DWC_EQOS_AXI_SLAVE&&!DWC_EQOS_AXI4_LITE_SLAVE **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** aclk_i **Registered:** No

Port Name	I/O	Description
arburst_s_i[1:0]	I	AXI Slave Burst Type. The burst type, along with the size information, specifies how the address is calculated for each transfer within the burst. **Exists:** DWC_EQOS_AXI_SLAVE&&!DWC_EQOS_AXI4_LITE_SLAVE **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** aclk_i **Registered:** No
arlock_s_i[1:0]	I	AXI Slave Read Locked Access. This signal indicates the type of locked access to the Read address channel of the AXI Slave. The DWC_ether_qos ignores the value of this signal. **Exists:** DWC_EQOS_AXI_SLAVE&&!DWC_EQOS_AXI4_LITE_SLAVE **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** aclk_i **Registered:** No
arcache_s_i[3:0]	I	AXI Slave Cache Type. This signal provides additional information about the cacheable characteristics of the transfer. The DWC_ether_qos core does not use this input because it supports only normal data transactions. **Exists:** DWC_EQOS_AXI_SLAVE&&!DWC_EQOS_AXI4_LITE_SLAVE **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** aclk_i **Registered:** N/A
arprot_s_i[2:0]	I	AXI Slave Protection Type. This signal provides protection unit information for the transaction. The DWC_ether_qos AXI slave does not support Exclusive access. The DWC_ether_qos core does not use this input because it supports only normal data transactions. **Exists:** (DWC_EQOS_CSR_PORT==3||DWC_EQOS_CSR_PORT==5) **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** aclk_i **Registered:** Yes

Port Name	I/O	Description
arvalid_s_i	I	AXI Slave Read Address Valid. When this signal is high, it indicates that the read address and control information is valid. This signal remains stable until the address acknowledge signal arready_s_o is high. ■ 1: Address and control information valid ■ 0: Address and control information not valid **Exists:** (DWC_EQOS_CSR_PORT==3||DWC_EQOS_CSR_PORT==5) **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** aclk_i **Registered:** No
arready_s_o	O	AXI Slave Read Address Ready. This signal indicates that the slave is ready to accept an address and associated control signals: ■ 1: Slave ready ■ 0: Slave not ready **Exists:** (DWC_EQOS_CSR_PORT==3||DWC_EQOS_CSR_PORT==5) **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** aclk_i **Registered:** No
rid_s_o[(DWC_EQOS_GS_ID-1):0]	O	AXI Slave Read ID Tag. This signal is the ID tag of the Read data group of signals. The rid value is generated by the slave. The rid value is equal to the arid value of the Read transaction to which it is responding. **Exists:** (DWC_EQOS_CSR_PORT==3||DWC_EQOS_CSR_PORT==5) **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** aclk_i **Registered:** No
rdata_s_o[(DWC_EQOS_CSR_DATAWIDTH-1):0]	O	AXI Slave Read Data. The Read data bus takes the value of the DWC_EQOS_DATAWIDTH parameter. **Exists:** (DWC_EQOS_CSR_PORT==3||DWC_EQOS_CSR_PORT==5) **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** aclk_i **Registered:** No

Port Name	I/O	Description
rdata_s_p_o[((DWC_EQOS_CSR_DATAWIDTH/8)-1):0]	O	AXI Slave Read Parity Data. The Read parity data bus takes the value of the DWC_EQOS_CSR_DATAWIDTH/8 parameter. **Exists:** DWC_EQOS_AXI_SLAVE&&DWC_EQOS_ASP_PPE **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** aclk_i **Registered:** No
rresp_s_o[1:0]	O	AXI Slave Read Response. This signal indicates the status of the read transfer. The valid responses are OKAY(2'b00), EXOKAY(2'b01), and SLVERR(2'b10). The DWC_ether_qos core provides only OKAY and SLVERR response on the rresp_s_o signal. It does not provide the EXOKAY and DECERR responses. **Exists:** (DWC_EQOS_CSR_PORT==3||DWC_EQOS_CSR_PORT==5) **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** aclk_i **Registered:** No
rlast_s_o	O	AXI Slave Read Last. This signal indicates the last transfer in a Read burst. **Exists:** DWC_EQOS_AXI_SLAVE&&!DWC_EQOS_AXI4_LITE_SLAVE **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** aclk_i **Registered:** No
rvalid_s_o	O	AXI Slave Read Valid. This signal indicates that the required Read data is available and the read transfer can be completed: ■ 1: Read data available ■ 0: Read data not available **Exists:** (DWC_EQOS_CSR_PORT==3||DWC_EQOS_CSR_PORT==5) **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** aclk_i **Registered:** No

Port Name	I/O	Description
rready_s_i	I	AXI Slave Read Ready. This signal indicates that the slave can accept the read data and response information: ■ 1: AXI master ready ■ 0: AXI master not ready **Exists:** (DWC_EQOS_CSR_PORT==3||DWC_EQOS_CSR_PORT==5) **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** aclk_i **Registered:** No

16.21 AXI Low Power Interface Signals

Table 16-21 AXI Low Power Interface Signals

Port Name	I/O	Description
csysreq_i	I	AXI Clock Controller System Low-Power Request. When this signal is low, it indicates that the system clock controller requested the AXI low power interface to enter the low-power state. When high, this signal indicates that the AXI low power interface should come out of the low-power state. **Exists:** DWC_EQOS_SYS==4||DWC_EQOS_SYS==5 **Power Domain:** MAC_OFF **Active State:** Low **Synchronous to:** aclk_i **Registered:** No
csysack_o	O	AXI Low-Power Request Acknowledgment. This signal is the acknowledgment to the low-power request from the system. **Exists:** DWC_EQOS_SYS==4||DWC_EQOS_SYS==5 **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** aclk_i **Registered:** Yes
cactive_o	O	AXI Clock Active. This signal indicates that the AXI low-power interface requires clock signal: ■ 1: Clock required ■ 0: Clock not required **Exists:** DWC_EQOS_SYS==4||DWC_EQOS_SYS==5 **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** clk_rx_i when self-initiating request to come out of low-power mode; aclk_i in all other cases **Registered:** No

16.22 AHB Master Interface Signals

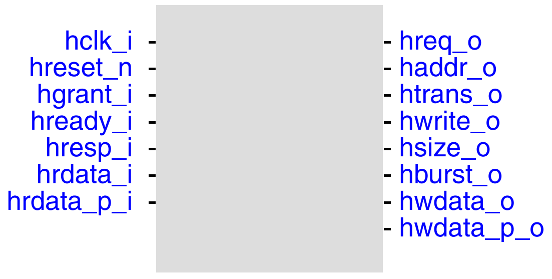


Table 16-22 AHB Master Interface Signals

Port Name	I/O	Description
hclk_i	I	AHB Master System Clock. This is the free-running AHB clock input provided by the Application. The DMA and the MTL modules also operate with this clock. The minimum valid frequency is 25 MHz (it is ~1.5 times MAC transmitter/receiver in SPRAM configurations) and the maximum valid frequency depends on frequency at which logic timing is met. **Exists:** DWC_EQOS_AHB_SUBSYS||DWC_EQOS_AHB_SLAVE **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** N/A **Registered:** N/A
hreset_n	I	AHB Master Power On System Reset. This signal is synchronous to hclk_i, the reset de-assertion must be synchronous to the hclk_i, the assertion can be synchronous or asynchronous based on the reset mode selected in configuration. **Exists:** DWC_EQOS_AHB_SUBSYS||DWC_EQOS_AHB_SLAVE **Power Domain:** MAC_OFF **Active State:** Low **Synchronous to:** hclk_i **Registered:** No
hgrant_i	I	AHB Grant to Master. Indicates that the EQOS-AHB subsystem is currently the highest priority master. Ownership of address and control signals changes at the end of a transfer, when the hready_i signal is high. Therefore, the subsystem gets access when both hgrant_i and hready_i signals are high. **Exists:** DWC_EQOS_SYS==3 **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** hclk_i **Registered:** No

Port Name	I/O	Description
hready_i	I	AHB Master Slave Ready. This signal indicates that a transfer has finished on the bus. The Slave being addressed may drive this signal low to extend a transfer. **Exists:** DWC_EQOS_SYS==3 **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** hclk_i **Registered:** No
hresp_i[1:0]	I	AHB Slave Response to Master. This signal provides information about the transfer status: ■ 00: OKAY (Transfer completed OK) ■ 01: ERROR (Error in current transfer) ■ 10: RETRY (The slave is busy and wants the master to retry the transfer) ■ 11: SPLIT (The slave accepted request, but the master should back off from bus until the slave is ready to serve) **Exists:** DWC_EQOS_SYS==3 **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** hclk_i **Registered:** No
hrdata_i[(DWC_EQOS_DATAWIDTH-1):0]	I	AHB Master Read Data. This signal transfers the data from the AHB slave to the EQOS-AHB during Read operations. **Exists:** DWC_EQOS_SYS==3 **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** hclk_i **Registered:** Yes
hrdata_p_i[(((DWC_EQOS_DATAWIDTH==32)?4:((DWC_EQOS_DATAWIDTH==64)?8:16))-1):0]	I	AHB Master Parity Read Data. This signal transfers the parity data from the AHB slave to the EQOS-AHB during Read operations. **Exists:** DWC_EQOS_AHB_SUBSYS&&DWC_EQOS_ASP_PPE **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** hclk_i **Registered:** Yes

Port Name	I/O	Description
hreq_o	O	AHB Master Request. The EQOS-AHB subsystem sets this signal to indicate that it requires the bus. This signal is processed by the arbiter to grant the bus. **Exists:** DWC_EQOS_SYS==3 **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** hclk_i **Registered:** Yes
haddr_o[31:0]	O	AHB Master Address. This is the 32-bit AHB address for current transaction. **Exists:** DWC_EQOS_SYS==3 **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** hclk_i **Registered:** Yes
htrans_o[1:0]	O	AHB Master Transfer Type. This signal indicates the AHB transfer type. The AHB master interface uses the following values: ■ 00: IDLE ■ 01: BUSY ■ 10: NONSEQUENTIAL ■ 11: SEQUENTIAL **Exists:** DWC_EQOS_SYS==3 **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** hclk_i **Registered:** Yes
hwrite_o	O	AHB Master Read or Write Signal. When this signal is high, it indicates a Write transfer. When this signal is low, it indicates a Read transfer. **Exists:** DWC_EQOS_SYS==3 **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** hclk_i **Registered:** Yes

Port Name	I/O	Description
hsize_o[2:0]	O	AHB Master Data Transfer Size. This signal indicates the size of the AHB data transfer: ■ 010: Word (32 bits) ■ 011: DWord (64 bits) ■ 100: LWord (128 bits) ■ All other encodings are not used. **Exists:** DWC_EQOS_SYS==3 **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** hclk_i **Registered:** Yes
hburst_o[2:0]	O	AHB Master Burst. This signal indicates the transfer type of an AHB Burst: ■ 000: SINGLE (Single Transfer) ■ 001: INCR (Incrementing burst of undefined length) ■ 011: INCR4 (4-beat incrementing burst) ■ 101: INCR8 (8-beat incrementing burst) ■ 111: INCR16 (16-beat incrementing burst) ■ All other encodings are not used. **Exists:** DWC_EQOS_SYS==3 **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** hclk_i **Registered:** Yes
hwdata_o[(DWC_EQOS_DATAWIDTH-1):0]	O	AHB Master Write Data Bus. This bus transfers the data from the EQOS-AHB subsystem to the AHB slaves. **Exists:** DWC_EQOS_SYS==3 **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** hclk_i **Registered:** Yes
hwdata_p_o[(((DWC_EQOS_DATAWIDTH==32)?4:((DWC_EQOS_DATAWIDTH==64)?8:16))-1):0]	O	AHB Master Parity for Write Data Bus. This bus transfers the parity data (generated om hwdata_o) from the EQOS-AHB subsystem to the AHB slaves. **Exists:** DWC_EQOS_AHB_SUBSYS&&DWC_EQOS_ASP_PPE **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** hclk_i **Registered:** Yes

16.23 AHB Slave Interface Signals

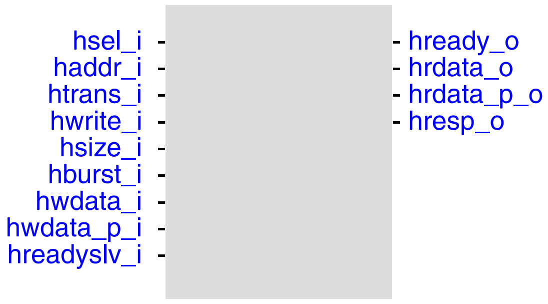


Table 16-23 AHB Slave Interface Signals

Port Name	I/O	Description
hsel_i	I	AHB Slave Select. This signal is a combinatorial decode of the address bus, indicating that the EQOS-AHB Subsystem is the target for the current transfer. All reads are completed without SPLIT or RETRY response for this address region. **Exists:** (DWC_EQOS_CSR_PORT==2) **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** CSR clock **Registered:** No
haddr_i[15:0]	I	AHB Address. This signal indicates the 16-bit AHB address for current transaction. **Note:** The CSR uses only 8K address space, that is, haddr_i[12:0]. You must connect Bits [15:13] of haddr_i to 0. If Bits[15:13] of haddr_i are not connected to 0, the address is considered as reserved address. **Exists:** (DWC_EQOS_CSR_PORT==2) **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** CSR clock **Registered:** No
htrans_i[1:0]	I	AHB Transfer. This signal indicates the type of current AHB transfer: ■ 00: IDLE ■ 01: BUSY ■ 10: NONSEQUENTIAL ■ 11: SEQUENTIAL **Exists:** (DWC_EQOS_CSR_PORT==2) **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** CSR clock **Registered:** No

Port Name	I/O	Description
hwrite_i	I	AHB Read or Write. When this signal is high, it indicates a Write transfer. When this signal is low, it indicates a Read transfer. **Exists:** (DWC_EQOS_CSR_PORT==2) **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** CSR clock **Registered:** No
hsize_i[2:0]	I	AHB Transfer Data Size. This signal indicates the size of the AHB transfer: ■ 000: Byte (8 bits) ■ 001: Halfword (16 bits) ■ 010: Word (32 bits) ■ All other encodings are considered as Word. **Exists:** (DWC_EQOS_CSR_PORT==2) **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** CSR clock **Registered:** No
hburst_i[2:0]	I	AHB Burst Length. This signal indicates the type of an AHB burst: ■ 000: SINGLE (Single Transfer) ■ 001: INCR (Incrementing burst of undefined length) ■ 010: WRAP4 (4-beat wrapping burst) ■ 011: INCR4 (4-beat incrementing burst) ■ 100: WRAP8 (8-beat wrapping burst) ■ 101: INCR8 (8-beat incrementing burst) ■ 110: WRAP16 (16-beat wrapping burst) ■ 111: INCR16 (16-beat incrementing burst) **Exists:** (DWC_EQOS_CSR_PORT==2) **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** CSR clock **Registered:** No
hwdata_i[(DWC_EQOS_CSR_DATAWIDTH-1):0]	I	AHB Write Data. This signal is the AHB Write data input to the slave port. **Exists:** (DWC_EQOS_CSR_PORT==2) **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** CSR clock **Registered:** Yes

Port Name	I/O	Description
hwdata_p_i[((DWC_EQOS_CSR_DATAWIDTH/8)-1):0]	I	AHB Write parity Data. This signal is the AHB parity data (generated on hwdata_i) input to the slave port. **Exists:** DWC_EQOS_AHB_SLAVE&&DWC_EQOS_ASP_PPE **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** CSR clock **Registered:** Yes
hreadyslv_i	I	AHB Master or Slave Ready. The hready of the AHB bus input to all master and slave ports. **Exists:** (DWC_EQOS_CSR_PORT==2) **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** CSR clock **Registered:** No
hready_o	O	AHB Slave Ready. The EQOS-AHB Subsystem slave indicates that the current transfer has finished. **Exists:** (DWC_EQOS_CSR_PORT==2) **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** CSR clock **Registered:** Yes
hrdata_o[(DWC_EQOS_CSR_DATAWIDTH-1):0]	O	AHB Slave Read Data. This is the AHB slave read data output from the slave port. **Exists:** (DWC_EQOS_CSR_PORT==2) **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** CSR clock **Registered:** Yes
hrdata_p_o[((DWC_EQOS_CSR_DATAWIDTH/8)-1):0]	O	AHB Slave Read Parity Data. This is the AHB slave read parity data (generated on hrdata_o) output from the slave port. **Exists:** DWC_EQOS_AHB_SLAVE&&DWC_EQOS_ASP_PPE **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** CSR clock **Registered:** Yes

Port Name	I/O	Description
hresp_o[1:0]	O	AHB Slave Response. The signal provides the information about transfer status. The following is the valid response: ■ 00: OKAY: Transfer completed OK ■ 01: ERROR: Transfer completed with ERROR ■ All other encodings are not used. **Exists:** (DWC_EQOS_CSR_PORT==2) **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** CSR clock **Registered:** Yes

16.24 MDC Interface Signals

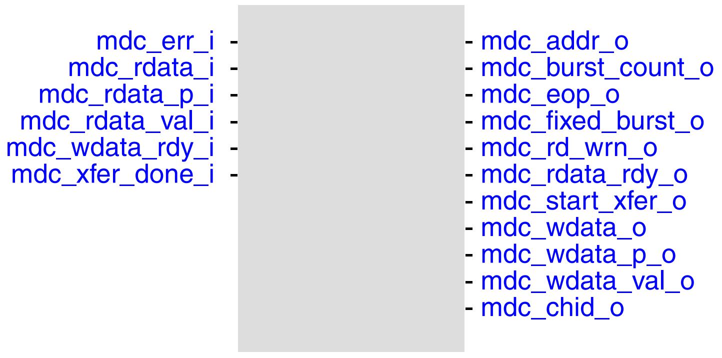


Table 16-24 MDC Interface Signals

Port Name	I/O	Description
mdc_err_i	I	Transfer Error. When this signal is high (for 1 clock), it indicates that an error occurred during application data transfer. The application must also terminate the transfer by driving the mdc_xfer_done_i signal high in next clock cycle. This signal must not be asserted when there is no application data transfer pending, that is when MDC interface is idle. **Exists:** DWC_EQOS_SYS == 2 **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** clk_app_i **Registered:** No
mdc_rdata_i[(DWC_EQOS_DATAWIDTH-1):0]	I	Read Data. This bus contains the data read from the application. The DMA samples the read data bus when the mdc_rdata_val_i and mdc_rdata_rdy_o signals are high. The DMA uses the mdc_rdata_rdy_o signal to indicate that it is ready to accept the data. When the mdc_rdata_rdy_o signal is high, the application can change the data on the mdc_rdata_i and assert the mdc_rdata_val_i signal to indicate availability of new data. **Exists:** DWC_EQOS_SYS == 2 **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** clk_app_i **Registered:** No

Port Name	I/O	Description
mdc_rdata_p_i[(((DWC_EQOS_DATAWIDTH==32)?4:((DWC_EQOS_DATAWIDTH==64)?8:16))-1):0]	I	Read Parity Data. This bus contains the parity data read from the application. The DMA samples the parity read data bus when the mdc_rdata_val_i and mdc_rdata_rdy_o signals are high. **Exists:** DWC_EQOS_DMA_SUBSYS&&DWC_EQOS_ASP_PPE **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** clk_app_i **Registered:** No
mdc_rdata_val_i	I	Read Data Valid. When the application drives this signal high, it indicates that the application has valid data on the mdc_rdata_i bus. The Read transfer is done when the mdc_rdata_val_i and mdc_rdata_rdy_o signals are asserted. **Exists:** DWC_EQOS_SYS == 2 **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** clk_app_i **Registered:** No
mdc_wdata_rdy_i	I	Write Data Ready. The application asserts this signal to indicate that it is ready to accept the data on the mdc_wdata_o bus. The Write transfer happens when the mdc_wdata_val_o and mdc_wdata_rdy_i signals are asserted. **Exists:** DWC_EQOS_SYS == 2 **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** clk_app_i **Registered:** No
mdc_xfer_done_i	I	Transfer Done. When this signal is high (for 1 clock), it indicates that the data transfers, initiated by asserting the mdc_start_xfer_o signal, is complete. For Read transfer, this signal should be asserted for one clock after the last Read data transfer is complete. For Write transfer, this signal should be asserted for one clock after the Write data transfer is complete. **Exists:** DWC_EQOS_SYS == 2 **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** clk_app_i **Registered:** No

Port Name	I/O	Description
mdc_addr_o[(DWC_EQOS_AXI_GM_ADDR_WIDTH-1):0]	O	Address. This bus give the address of the current transfer. The start address of a burst transfer is given along with the assertion of mdc_start_xfer_o signal. The address is incremented at the end of each Read or Write transfer, indicated by handshaking of the following signals: ■ mdc_wdata_rdy_i and mdc_wdata_val_o, or ■ mdc_rdata_val_i and mdc_rdata_rdy_o **Exists:** DWC_EQOS_DMA_SUBSYS||DWC_EQOS_MDC_INTF_SBD_IO_EN **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** clk_app_i **Registered:** Yes
mdc_burst_count_o[7:0]	O	Burst Count. These bits indicate the length of the burst transfer initiated by the DMA. ■ 0000_0000: 1 beat ■ 0000_0001: 2 beats ■ .. ■ 1111_1100: 255 beats ■ 1111_1111: 256 beats Each beat is 4, 8, or 16 bytes for 32-bit, 64-bit, or 128-bit wide bus, respectively. **Exists:** DWC_EQOS_DMA_SUBSYS||DWC_EQOS_MDC_INTF_SBD_IO_EN **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** clk_app_i **Registered:** Yes
mdc_eop_o	O	End of Packet. The signal indicates that the current data on the mdc_wdata_o bus corresponds to the last transfer of the packet. The DMA cannot supply more data as per requested burst length. The application must terminate the burst Transfer on receiving this signal by driving the mdc_xfer_done_i signal high. This signal is applicable only for the MDC write transactions. The application should sample this signal when the mdc_wdata_val_o signal is high. **Exists:** DWC_EQOS_SYS == 2 **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** clk_app_i **Registered:** No

Port Name	I/O	Description
mdc_fixed_burst_o	O	Fixed Burst Length. This signal reflects the value of the Fixed Burst (FB) bit of the DMA_Mode register. **Exists:** DWC_EQOS_SYS == 2 **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** clk_app_i **Registered:** Yes
mdc_rd_wrn_o	O	Read or Write Transfer. When this signal is high, it indicates a Read transfer. When this signal is low, it indicates a Write transfer. **Exists:** DWC_EQOS_DMA_SUBSYS||DWC_EQOS_MDC_INTF_SBD_IO_EN **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** clk_app_i **Registered:** Yes
mdc_rdata_rdy_o	O	Read Data Ready. The DMA asserts this signal when it is ready to accept the Read data on the mdc_rdata_i signal. The Read transfer is done when the mdc_rdata_val_i and mdc_rdata_rdy_o signals are asserted. **Exists:** DWC_EQOS_SYS == 2 **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** clk_app_i **Registered:** No
mdc_start_xfer_o	O	Start Transfer. When this signal is high (for 1 clock), it indicates the start of a transaction by the DMA. This signal also indicates the validity of the Values on mdc_addr_o, mdc_mixed_burst_o, mdc_rd_wrn_o, mdc_burst_count_o, and mdc_fixed_burst_o signals. **Exists:** DWC_EQOS_DMA_SUBSYS||DWC_EQOS_MDC_INTF_SBD_IO_EN **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** clk_app_i **Registered:** Yes

Port Name	I/O	Description
mdc_wdata_o[(DWC_EQOS_DATAWIDTH -1):0]	O	Write Data. This bus contains the data output for Write transfers. This data is valid when the DMA asserts the mdc_wdata_val_o signal. The Write transfer happens when the mdc_wdata_val_o and mdc_wdata_rdy_i signals are asserted. **Exists:** DWC_EQOS_SYS == 2 **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** clk_app_i **Registered:** No
mdc_wdata_p_o[(((DWC_EQOS_DATAWIDTH==32)?4:((DWC_EQOS_DATAWIDTH==64)?8:16))-1):0]	O	Write Parity Data. This bus contains the parity data output for Write transfers. This data is valid when the DMA asserts the mdc_wdata_val_o signal. The Write transfer happens when the mdc_wdata_val_o and mdc_wdata_rdy_i signals are asserted. **Exists:** DWC_EQOS_DMA_SUBSYS&&DWC_EQOS_ASP_PPE **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** clk_app_i **Registered:** No
mdc_wdata_val_o	O	Write Data Valid. When this signal is asserted, it indicates that the Write data is Valid on the mdc_wdata_o signal. The Write data transfer happens when the mdc_wdata_val_o and mdc_wdata_rdy_i signals are asserted. **Exists:** DWC_EQOS_SYS == 2 **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** clk_app_i **Registered:** No
mdc_chid_o[(ACTCHW-1):0]	O	Current Transfer DMA Channel ID. This bus contains the DMA channel ID of current transfer. This is valid with mdc_start_xfer_o pulse until cycle prior to next mdc_start_xfer_o pulse (for EQOS-AHB configurations this is changed when the AHB Bus is idle prior to the transfer). The LSb indicates Tx or Rx DMA Channel (1- Tx, 0- Rx). The MSb bits specifies the channel ID of Tx/Rx DMA indicated by LSb. **Exists:** DWC_EQOS_DMA_SUBSYS||DWC_EQOS_AHB_SUBSYS **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** clk_app_i **Registered:** Yes

16.25 TSO Memory Interface (TMI) Signals

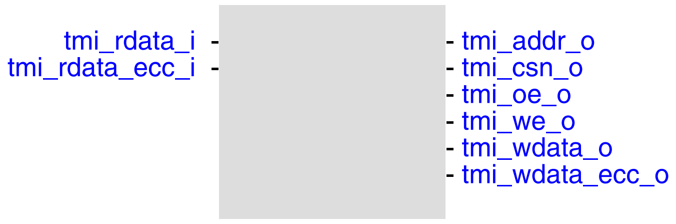


Table 16-25 TSO Memory Interface (TMI) Signals

Port Name	I/O	Description
tmi_addr_o[(DWC_EQOS_TSO_MEM_ADDR_WIDTH-1):0]	O	Address. This bus gives the address of the data transfer. The address width depends upon the size of the selected memory. **Exists:** DWC_EQOS_TSO_MEM_EN **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** Application clock **Registered:** Yes
tmi_csn_o	O	Chip Select. This signal is used as a chip select to select the TSO memory. **Exists:** DWC_EQOS_TSO_MEM_EN **Power Domain:** MAC_OFF **Active State:** Low **Synchronous to:** Application clock **Registered:** No
tmi_oe_o	O	Output Enable. The DMA drives this signal to enable the output data from the TSO memory. The TSO memory drives the valid Read data on the tmi_rdata_i signal only when this signal is high. The DMA samples the Read data, with this signal high, one clock after the address is given. **Exists:** DWC_EQOS_TSO_MEM_EN **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** Application clock **Registered:** No
tmi_we_o	O	Write Enable. When this signal is high, it indicates that a Write operation request is intended. **Exists:** DWC_EQOS_TSO_MEM_EN **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** Application clock **Registered:** No

Port Name	I/O	Description
tmi_wdata_o[(DWC_EQOS_DATAWIDTH-1):0]	O	Write Data. This bus carries the data to be written in the Tx FIFO. It is valid when tmi_we_o is high along with tmi_csn_o low. **Exists:** DWC_EQOS_TSO_MEM_EN **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** Application clock **Registered:** Yes
tmi_wdata_ecc_o[(DWC_EQOS_TSO_ECW-1):0]	O	TSO ECC Write Data. This bus carries the ECC data (computed on {tmi_addr_o,tmi_wdata_o}) to be written in the TSO Memory. It is valid when tmi_we_o is high along with tmi_csn_o low. **Exists:** DWC_EQOS_TSO_MEM_EN&&DWC_EQOS_ASP_ECC **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** Application clock **Registered:** Yes
tmi_rdata_i[(DWC_EQOS_DATAWIDTH-1):0]	I	Read Data. This bus carries the data read from the TSO Memory. It is valid when the tmi_oe_o signal is high. **Exists:** DWC_EQOS_TSO_MEM_EN **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** Application clock **Registered:** Yes
tmi_rdata_ecc_i[(DWC_EQOS_TSO_ECW-1):0]	I	TSO ECC Read Data. This bus carries the ECC data read from the TSO Memory. It is valid when the tmi_oe_o signal is high. **Exists:** DWC_EQOS_TSO_MEM_EN&&DWC_EQOS_ASP_ECC **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** Application clock **Registered:** Yes

16.26 Application Transmit Interface (ATI) Signals

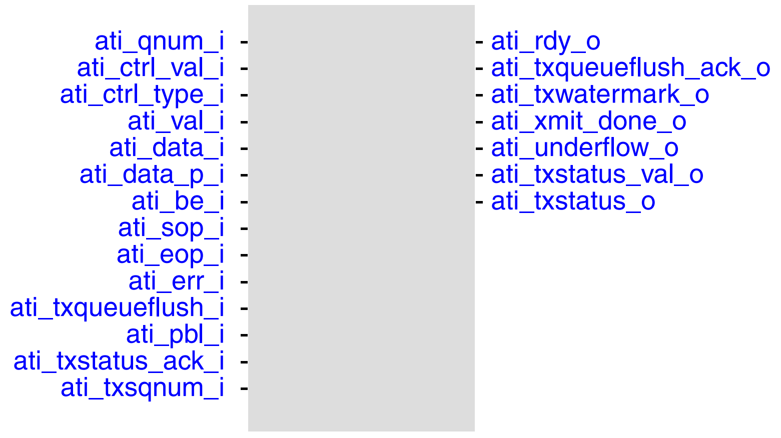


Table 16-26 Application Transmit Interface (ATI) Signals

Port Name	I/O	Description
ati_qnum_i[(DWC_EQOS_TXQW-1):0]	I	Transmit Queue Number. This signal indicates the Tx queue number with which the current ATI data or control transfer is associated. **Exists:** DWC_EQOS_MTL_SUBSYS&&DWC_EQOS_NUM_TXQ!=1 **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** clk_app_i **Registered:** Yes
ati_ctrl_val_i	I	Packet Control Data Valid. This signal should be asserted before the ati_sop_i and ati_val_i signals are asserted for packet transmission. When this signal is high, it qualifies the ati_data_i signal as Packet Control Word in association with the ati_ctrl_type_i signal. The type of the control word is indicated through the ati_ctrl_type_i signal. **Exists:** DWC_EQOS_SYS == 1 **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** clk_app_i **Registered:** No

Port Name	I/O	Description
ati_ctrl_type_i[(DWC_EQOS_ATI_CTL_WIDTH-1):0]	I	Packet Control Data Type. This signal determines the type of the control word Driven on the ati_data_i signal when the ati_ctrl_val_i signal is high. **Exists:** ((DWC_EQOS_DATAWIDTH==32&&(DWC_EQOS_SA_VLAN_INS_CTRL_EN||DWC_EQOS_OST_EN||DWC_EQOS_TBS))||(DWC_EQOS_DATAWIDTH==64&&(DWC_EQOS_OST_EN||DWC_EQOS_TBS)))&&DWC_EQOS_MTL_SUBSYS **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** clk_app_i **Registered:** No
ati_val_i	I	ATI Valid. This signal qualifies all packet transmission control signals and the data on the application transmit interface. **Exists:** DWC_EQOS_SYS == 1 **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** clk_app_i **Registered:** No
ati_data_i[(DWC_EQOS_DATAWIDTH-1):0]	I	Application Data. This is the Packet data from the application when the ati_val_i signal is high. In little-endian format, the LSB lane ati_data_i[7:0] is the first byte transmitted by the DWC_ether_qos core. In big-endian format, the MSB lane ati_data_i[DWC_EQOS_DATAWIDTH - 1: DWC_EQOS_DATAWIDTH - 8] is the first byte transmitted by the DWC_ether_qos core. This signal indicates the control information when the ati_ctrl_val_i signal is high along with the type of control indicated by the ati_ctrl_type_i signal. When the control word is being transmitted, there is no change in the format of the control word based on the endianness. The endianness is applicable only for the data. **Exists:** DWC_EQOS_SYS == 1 **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** clk_app_i **Registered:** Yes
ati_data_p_i[(((DWC_EQOS_DATAWIDTH==32)?4:((DWC_EQOS_DATAWIDTH==64)?8:16))-1):0]	I	Application Parity Data. This is the Parity data computed on ati_data_i bus. **Exists:** DWC_EQOS_MTL_SUBSYS&&DWC_EQOS_ASP_PPE **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** clk_app_i **Registered:** Yes

Port Name	I/O	Description
ati_be_i[(DWC_EQOS_BEW-1):0]	I	Number of Valid Byte Lanes. When the ari_eop_o signal is high, this signal indicates the number of valid bytes. The total number of valid bytes is equal to ati_be_o + 1 Bytes on the ati_data_i signal. This signal is valid only when the ati_val_i and ati_eop_i signals are high. When the ati_eop_i signal is low, the MTL considers all bytes of the ati_data_i signal as valid. When the ati_be_i signal width is 2, the following values indicate the lanes that have the data: ■ 11: All 4 byte lanes have data ■ 10: LS 3 byte lanes have data ■ 01: LS 2 byte lanes have data ■ 00: LS 1 byte lane has the data **Exists:** DWC_EQOS_SYS == 1 **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** clk_app_i **Registered:** Yes. If Transmit Checksum Engine is selected, then No.
ati_sop_i	I	Start of Packet. This signal indicates that the application is ready to Transfer on the first data of packet. This signal is valid only when the ati_val_i signal is high. **Exists:** DWC_EQOS_SYS == 1 **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** clk_app_i **Registered:** No
ati_eop_i	I	End of Packet. This signal indicates that the current data transferred to the MTL is the last data. This signal is valid only when the ati_val_i signal is high. **Exists:** DWC_EQOS_SYS == 1 **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** clk_app_i **Registered:** No
ati_err_i	I	Transmit Error. This signal indicates an error in the packet that is being transmitted. The MAC must corrupt the packet with CRC error. The application must drive this signal high along with the ati_val_i and ati_eop_i signals. **Exists:** DWC_EQOS_SYS == 1 **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** clk_app_i **Registered:** No

Port Name	I/O	Description
ati_rdy_o	O	ATI Ready. When this signal is high, it indicates that the MTL is ready to accept additional data or control from the application. This signal is low in the following conditions: ■ When Tx Queue is being flushed ■ When Tx Queue is full ■ When internally-generated checksum data is being written to Tx Queue **Note:** This signal is a combinational signal. It is generated based on the ati_qnum_i signal in configurations with multiple queues. To avoid combinational loop, the ati_qnum_i signal should be registered based on the ati_rdy_o signal. **Exists:** DWC_EQOS_SYS == 1 **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** clk_app_i **Registered:** No
ati_txqueueflush_i[(DWC_EQOS_NUM_TXQ-1):0]	I	TX Queue flush. This signal is available for each Tx queue. A Pulse on this signal flushes the contents of the queue. **Exists:** DWC_EQOS_SYS == 1 **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** clk_app_i **Registered:** No
ati_txqueueflush_ack_o[(DWC_EQOS_NUM_TXQ-1):0]	O	Acknowledgment for TX Queue flush. This signal is available for each Tx queue. A Pulse on this signal indicates that the transmit queue flush request through the ati_txqueueflush_i is completed. **Exists:** DWC_EQOS_SYS == 1 **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** clk_app_i **Registered:** No

Port Name	I/O	Description
ati_pbl_i[((DWC_EQOS_ATI_PBL_WIDTH *DWC_EQOS_NUM_TXQ)-1):0]	I	Number of Beats Requested By the Data Application to Transmit on Tx Queue. This signal indicates the space requested by the Application on the Tx Queue. This space is in terms of number of beats depending on the configurations: 4, 8, 16, or 32 bytes in 32-bit, 64-bit, 128-bit, or 256-bit data bus mode, respectively. The maximum limit of the number of beats is equal to half of the Tx Queue depth. For configurations with multiple Tx Queues, you should not drive values that are more than the half of the programmed values of Tx Queue depth. For example, if the Tx Queue size is programed to be 2K and Datawidth is 32, the Tx Queue depth is equal to 512. Therefore, you should not drive any value which requests more than 256 beats transfer, that is, you should not drive any value which is more than 255. The following is the relationship between ati_pbl_i and the number of beats: ■ 0: 1 number of beats ■ 1: 2 number of beats ■ and so ■ N: N+1 number of beats **Note:** The width of the ati_pbl_i port depends upon the number of Tx queues selected while configuring the core. It is given by (MTL Tx FIFO Address bus Width - 1) x Number of Tx Queues selected **Exists:** DWC_EQOS_SYS == 1 **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** clk_app_i **Registered:** No
ati_txwatermark_o[(DWC_EQOS_NUM_TXQ-1):0]	O	Indicates the availability of space in TX data FIFO as requested through ati_pbl_i. **Exists:** DWC_EQOS_SYS == 1 **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** clk_app_i **Registered:** No
ati_xmit_done_o[(DWC_EQOS_NUM_TXQ-1):0]	O	Transmit Done Status. This signal is available for each Tx queue. This signal indicates that the packet corresponding to a Tx queue is transmitted on the line. **Exists:** DWC_EQOS_SYS == 1 **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** clk_app_i **Registered:** Yes

Port Name	I/O	Description
ati_underflow_o[(DWC_EQOS_NUM_TXQ-1):0]	O	Tx Queue Underflow. This signal is available for each Tx queue. This signal indicates that the corresponding Tx queue did not have data to transmit which caused underflow. **Exists:** DWC_EQOS_SYS == 1 **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** clk_app_i **Registered:** Yes
ati_txstatus_val_o[(DWC_EQOS_NUM_TXQ-1):0]	O	Valid Status Available on the Transmit Queue. This signal is available for each Tx queue.This signal indicates that the packet status is available for corresponding Tx queue. When this signal is high, the Tx Status can be read by driving the ati_txstatus_ack_i and ati_txsqnum_i signals high. The ati_txsqnum_i signal indicates the Tx queue number for which the status is being read by the application. **Exists:** DWC_EQOS_MTL_SUBSYS&&(!DWC_EQOS_TX_STS_DROP) **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** clk_app_i **Registered:** No
ati_txstatus_o[17:0]	O	Transmit Status Word of the Selected Queue Indicated through ati_txsqnum_i. This signal indicates the packet status of the Tx queue. This is valid only when the ati_txstatus_ack_i signal is high for the Tx queue indicated through the ati_txsqnum_i signal. **Exists:** DWC_EQOS_MTL_SUBSYS&&(!DWC_EQOS_TX_STS_DROP) **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** clk_app_i **Registered:** No
ati_txstatus_ack_i	I	Acknowledgment for the Tx Status. The application drives this signal high to accept the valid (indicated by ati_txstatus_val_o) Tx status Available on ati_txstatus_o signal. When multiple Tx queues are present in your configuration, the ati_txsqnum_i signal indicates the acknowledged queue number. **Exists:** DWC_EQOS_MTL_SUBSYS&&(!DWC_EQOS_TX_STS_DROP) **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** clk_app_i **Registered:** No

Port Name	I/O	Description
ati_txsqnum_i[(DWC_EQOS_TXQW-1):0]	I	Queue Selection for Transmit Status Read. This signal is used to select the Tx queue number to read the status when the ati_txstatus_ack_i signal is high. **Exists:** DWC_EQOS_MTL_SUBSYS&&(DWC_EQOS_NUM_TXQ!=1)&&(!DWC_EQOS_TX_STS_DROP) **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** clk_app_i **Registered:** No

16.27 Application Receive Interface (ARI) Signals

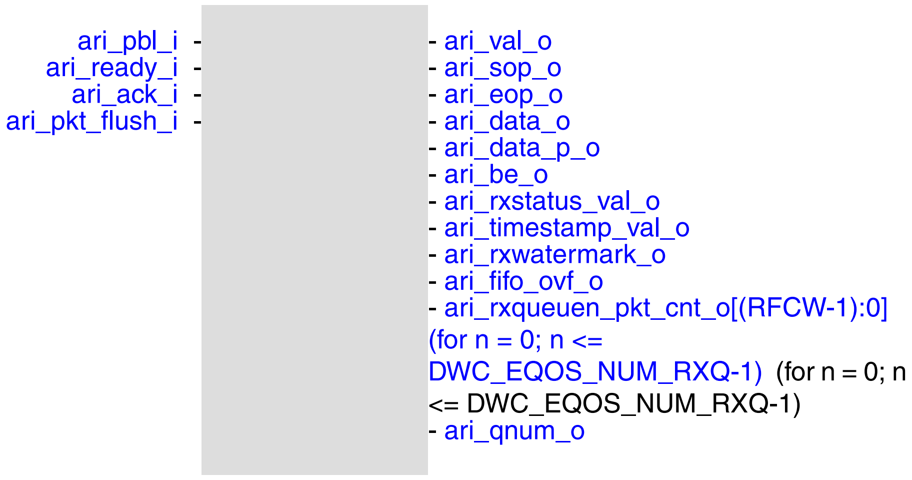


Table 16-27 Application Receive Interface (ARI) Signals

Port Name	I/O	Description
ari_pbl_i[((DWC_EQOS_NUM_RXQ*DWC_EQOS_ARI_PBL_WIDTH)-1):0]	I	Number of Beats Requested by the Application for Queue. This signal indicates the number of beats of the data requested by application from the MTL Rx queue. For configurations with multiple Rx Queues, you should not drive values that are more than the half of the programmed values of Rx queue depth. For example, if the Rx Queue size is programed to be 2K and Datawidth is 32, the Rx Queue depth is equal to 512. Therefore, you should not drive any value which requests more than 256 beats transfer, that is, you should not drive any value which is more than 255. The following is the relationship between ari_pbl_i and the number of beats: ■ 0: 1 number of beats ■ 1: 2 number of beats ■ and so ■ N: N+1 number of beats **Note:** The width of the ari_pbl_i port depends upon the number of Rx queues selected while configuring the core. It is given by (MTL Rx FIFO Address bus Width - 1) x Number of Rx Queues selected **Exists:** DWC_EQOS_SYS == 1 **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** clk_app_i **Registered:** No

Port Name	I/O	Description
ari_ready_i[(DWC_EQOS_NUM_RXQ-1):0]	I	Application Queue Ready Signal. This signal is available for each Rx queue. The application drives this signal high when it is ready to accept a single burst of data Transfer on the corresponding Rx queue. The burst length is as indicated by the application through ari_pbl_i signal. **Exists:** DWC_EQOS_MTL_SUBSYS&&DWC_EQOS_NUM_RXQ>1 **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** clk_app_i **Registered:** No
ari_val_o	O	ARI Valid. When this signal is high, it qualifies the data and other control Signals on the MTL receive application interface. In configurations with multiple queues, this signal corresponds to the currently-selected queue number indicated by the ari_qnum_o signal. **Exists:** DWC_EQOS_SYS == 1 **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** clk_app_i **Registered:** Yes
ari_sop_o	O	Start of Packet. This signal indicates that the current data transferred to the application corresponds to the first data of the packet. In configurations with multiple queues, this signal corresponds to the currently-selected queue number indicated by the ari_qnum_o signal. **Exists:** DWC_EQOS_SYS == 1 **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** clk_app_i **Registered:** Yes
ari_eop_o	O	End of Packet. This signal indicates that the current data transferred to the application corresponds to the last data of the packet. In configurations with multiple queues, this signal corresponds to the currently-selected queue number indicated by the ari_qnum_o signal. **Exists:** DWC_EQOS_SYS == 1 **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** clk_app_i **Registered:** Yes

Port Name	I/O	Description
ari_data_o[(DWC_EQOS_DATAWIDTH-1):0]	O	Packet Data. This is the packet data from the Rx Queue to the application. In little-endian format, the LSB lane ari_data_o[7:0] is the first byte received by the DWC_ether_qos core. In big-endian format, the MSB lane ari_data_o[DWC_EQOS_DATAWIDTH - 1:DWC_EQOS_DATAWIDTH - 8] is the first byte received by the DWC_ether_qos core. In configurations with multiple queues, this signal corresponds to the currently-selected queue number indicated by the ari_qnum_o signal. This signal gives the status when the ari_rxstatus_val_i signal is high. **Exists:** DWC_EQOS_SYS == 1 **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** clk_app_i **Registered:** Yes
ari_data_p_o[(((DWC_EQOS_DATAWIDTH==32)?4:((DWC_EQOS_DATAWIDTH==64)?8:16))-1):0]	O	Parity of Packet Data. This is the packet data parity from the Rx Queue to the application. In little-endian format, the LSB lane ari_data_p_o[0] is the first byte received by the DWC_ether_qos core. In big-endian format, the MSB lane ari_data_p_o[DWC_EQOS_DATAWIDTH/8 - 1 : DWC_EQOS_DATAWIDTH/8 - 8] is the first byte received by the DWC_ether_qos core. In configurations with multiple queues, this signal corresponds to the currently-selected queue number indicated by the ari_qnum_o signal. This signal gives the status when the ari_rxstatus_val_i signal is high. **Exists:** DWC_EQOS_MTL_SUBSYS&&DWC_EQOS_ASP_PPE **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** clk_app_i **Registered:** Yes

Port Name	I/O	Description
ari_be_o[(DWC_EQOS_BEW-1):0]	O	Number of Valid Byte Lanes. This signal indicates the number of valid byte lanes in the last data phase transfer (when the ari_eop_o signal is high) on the MTL receive application interface. In configurations with multiple queues, this signal corresponds to the currently-selected queue number indicated by the ari_qnum_o signal. This signal is valid only when the ari_val_o and ari_eop_o signals are high. When the ari_eop_o signal is low, the ari_be_o signal is always driven with all-ones by the MTL. When the width of this signal is 2, the following values indicate the lanes that have the data: ■ 11: All 4 byte lanes have data ■ 10: LS 3 byte lanes have data ■ 01: LS 2 byte lanes have data ■ 00: LS 1 byte lane has data **Exists:** DWC_EQOS_SYS == 1 **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** clk_app_i **Registered:** Yes
ari_ack_i	I	ARI Acknowledgment. This is the acknowledgment signal from the application. When the ari_val_o or ari_rxstatus_val_o signal is high, this signal indicates that the application has accepted the current data or Status on the ari_data_o signal. **Exists:** DWC_EQOS_SYS == 1 **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** clk_app_i **Registered:** No
ari_rxstatus_val_o	O	Valid Receive Status. This signal indicates that the ari_data_o bus contains the Receive status to the application. In configurations with multiple queues, this signal corresponds to the currently-selected queue number indicated by the ari_qnum_o signal. **Exists:** DWC_EQOS_SYS == 1 **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** clk_app_i **Registered:** Yes

Port Name	I/O	Description
ari_timestamp_val_o	O	Receive Timestamp Valid. This signal indicates that a valid timestamp value is available on the ARI data bus (ari_data_o). This signal is available only in the EQOS-MTL configuration. In 32-bit configuration, this signal is high for two clock cycles required to transfer two 32-bit words. The lower 32 bits of the timestamp (the Nanoseconds field) are given first the ari_data_o bus, followed by the upper word (the Seconds field). **Exists:** DWC_EQOS_MTL_SUBSYS&&DWC_EQOS_TIME_STAMPING **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** clk_app_i **Registered:** Yes
ari_rxwatermark_o[(DWC_EQOS_NUM_RXQ-1):0]	O	Indicates the availability of data in RX data FIFO as requested through ari_pbl_i. **Exists:** DWC_EQOS_SYS == 1 **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** clk_app_i **Registered:** Yes
ari_fifo_ovf_o[(DWC_EQOS_NUM_RXQ-1):0]	O	Rx Queue Overflow. This signal is available for all Rx queues. This indicates that the Rx packet is lost because of overflow in corresponding Rx queue. **Exists:** DWC_EQOS_SYS == 1 **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** clk_app_i **Registered:** Yes
ari_pkt_flush_i[(DWC_EQOS_NUM_DMA_RX_CH-1):0]	I	Packet Flush Request. This signal is available for all Rx queues. This signal indicates the packet flush request from the application to flush the current active packet in corresponding Rx queue. **Exists:** DWC_EQOS_SYS == 1 **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** clk_app_i **Registered:** No

Port Name	I/O	Description
ari_rxqueuen_pkt_cnt_o[(RFCW-1):0] (for n = 0; n <= DWC_EQOS_NUM_RXQ-1)	O	Rx Queue Packets Counter in Rx Queue. This status signal provides the number of packets present in the Rx queue. The width of the counter depends the maximum number of packets that can be stored in the Rx queue. For example, if the total Rx memory size is 2,048 bytes and the minimum packet size is 15 bytes, the maximum number of packets that can be stored in the Rx Queue is 128. Therefore, the width = 7. **Exists:** DWC_EQOS_MTL_SUBSYS && DWC_EQOS_NUM_RXQ>n **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** clk_app_i **Registered:** Yes
ari_qnum_o[(QNUMW-1):0]	O	Currently Selected Rx Queue Number. This signal indicates the queue that is selected by the Rx Arbiter in the MTL for current burst transfer. This signal qualifies the following signals: ■ ari_val_o ■ ari_sop_o ■ ari_eop_o ■ ari_data_o ■ ari_be_o ■ ari_rxstatus_val_o ■ ari_pkt_flush_i **Exists:** DWC_EQOS_MTL_SUBSYS&&DWC_EQOS_NUM_RXQ>1 **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** clk_app_i **Registered:** Yes

16.28 Rx Parser Interface Signals

rxp_rdata_i
rxp_rdata_ecc_i
rxp_addr_o
rxp_wdata_o
rxp_wdata_ecc_o
rxp_csn_o
rxp_we_o
rxp_oe_o

Table 16-28 Rx Parser Interface Signals

Port Name	I/O	Description
rxp_addr_o[(L2MEMSIZE-1):0]	O	Rx Parser Memory Address. This signal gives the address to the Rx Parser Memory Port. It is valid when the rxp_csn_o signal is low. The address width is displayed in coreConsultant after you configure the Rx Parser. The Rx Parser Address Bus Width field of the Flexible Rx Parser under Filtering dialog box displays the address width. **Exists:** DWC_EQOS_FRP_EN **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** Application clock **Registered:** Yes
rxp_wdata_o[95:0]	O	Rx Parser Memory Write Data. This bus carries the data to be written in the Rx Parser. It is valid when the rxp_csn_o signal is low and rxp_we_o signal is high. **Exists:** DWC_EQOS_FRP_EN **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** Application clock **Registered:** Yes
rxp_wdata_ecc_o[(DWC_EQOS_RXP_ECW-1):0]	O	Rx Parser Memory ECC Write Data. This bus carries the ECC data(computed on {rxp_addr_o,rxp_wdata_o}) to be written in the Rx Parser Memory. It is valid when the rxp_csn_o signal is low and rxp_we_o signal is high. The width of this data bus is 8-bit. **Exists:** DWC_EQOS_FRP_EN&&DWC_EQOS_ASP_ECC **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** Application clock **Registered:** Yes

Port Name	I/O	Description
rxp_csn_o	O	Rx Parser Memory Chip Select. This signal qualifies the rxp_addr_o, rxp_wdata_o, rxp_wdata_ecc_o and rxp_we_o signals. **Exists:** DWC_EQOS_FRP_EN **Power Domain:** MAC_OFF **Active State:** Low **Synchronous to:** Application clock **Registered:** No
rxp_we_o	O	Rx Parser Memory Write Enable. When this signal is high, it indicates that the rxp_wdata_o and rxp_wdata_ecc_o data should be written to the memory. **Exists:** DWC_EQOS_FRP_EN **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** Application clock **Registered:** Yes
rxp_rdata_i[95:0]	I	Rx Parser Memory Read data. This bus gives the address to the Rx Parser Memory read port. It is valid when the rxp_oe_o signal is high which is a clock cycle delayed version of rxp_csn_o being low. **Exists:** DWC_EQOS_FRP_EN **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** clk_tx_i **Registered:** Yes
rxp_rdata_ecc_i[(DWC_EQOS_RXP_ECW-1):0]	I	Rx Parser Memory ECC Read Data. This bus carries the ECC data read from the Rx Parser Memory. It is valid when the trc_rd_en_o signal is high. **Exists:** DWC_EQOS_FRP_EN&&DWC_EQOS_ASP_ECC **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** clk_tx_i **Registered:** No
rxp_oe_o	O	Rx Parser memory Read Enable. When this signal is high, it indicates that the rxp_rdata_i or rxp_rdata_ecc_i signal should contain valid data. **Exists:** DWC_EQOS_FRP_EN **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** clk_tx_i **Registered:** Yes

16.29 DPRAM Interface Signals

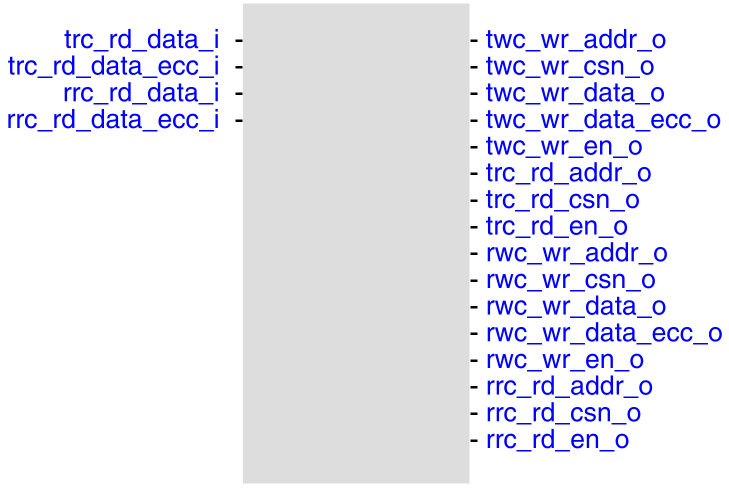


Table 16-29 DPRAM Interface Signals

Port Name	I/O	Description
twc_wr_addr_o[(DWC_EQOS_TX_FIFO_PTR_WIDTH-1):0]	O	Transmit FIFO Write Address. This signal gives the address to the Tx FIFO write port. It is valid when the twc_wr_csn_o signal is low. The address width is displayed in coreConsultant after you configure the Tx FIFO size. The MTL Tx FIFO Address Bus Width field of the Buffer Management dialog box displays the address width. **Exists:** DWC_EQOS_SYS!=0&&!DWC_EQOS_SPRAM **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** Application clock **Registered:** Yes
twc_wr_csn_o	O	Transmit FIFO Write Port Chip Select. This signal qualifies the twc_wr_addr_o, twc_wr_data_o, and twc_wr_en_o signals. **Exists:** DWC_EQOS_SYS!=0&&!DWC_EQOS_SPRAM **Power Domain:** MAC_OFF **Active State:** Low **Synchronous to:** Application clock **Registered:** No

Port Name	I/O	Description
twc_wr_data_o[(DWC_EQOS_MEMW-1):0]	O	Transmit FIFO Write Data. This bus carries the data to be written in the Tx FIFO. It is valid when the twc_wr_csn_o signal is low and twc_wr_en_o signal is high. The width of this data bus depends the data bus width selected during configuration. The additional bits (3, 4, or 5 in 32-bit, 64-bit, or 128-bit data) are used for storing the tag or byte enable bits for the valid data lanes. **Exists:** DWC_EQOS_SYS!=0&&!DWC_EQOS_SPRAM **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** Application clock **Registered:** Yes
twc_wr_data_ecc_o[(DWC_EQOS_TX_ECW-1):0]	O	Transmit FIFO ECC Write Data. This bus carries the ECC data(computed on {twc_wr_addr_o,twc_wr_data_o}) to be written in the Tx FIFO. It is valid when the twc_wr_csn_o signal is low and twc_wr_en_o signal is high. The width of this data bus depends the data bus width and the depth (address width) selected during configuration. **Exists:** DWC_EQOS_SYS!=0&&!DWC_EQOS_SPRAM&&DWC_EQOS_ASP_ECC **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** Application clock **Registered:** Yes
twc_wr_en_o	O	Transmit FIFO Write Enable. When this signal is high, it indicates that the twc_wr_data_o data should be written to the memory. **Exists:** DWC_EQOS_SYS!=0&&!DWC_EQOS_SPRAM **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** Application clock **Registered:** Yes
trc_rd_addr_o[(DWC_EQOS_TX_FIFO_PTR_WIDTH-1):0]	O	Transmit FIFO Read Address. This bus gives the address to the Tx FIFO read port. It is valid when the trc_rd_csn_o signal is low. The width of address bus depends the Tx FIFO depth and the data bus width. **Exists:** DWC_EQOS_SYS!=0&&!DWC_EQOS_SPRAM **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** clk_tx_i **Registered:** Yes

Port Name	I/O	Description
trc_rd_csn_o	O	Transmit FIFO Read Port Chip Select. This signal qualifies the trc_rd_addr_o signal. **Exists:** DWC_EQOS_SYS!=0&&!DWC_EQOS_SPRAM **Power Domain:** MAC_OFF **Active State:** Low **Synchronous to:** clk_tx_i **Registered:** No
trc_rd_data_i[(DWC_EQOS_MEMW-1):0]	I	Transmit FIFO Read Data. This bus carries the data read from the Tx FIFO. It is valid when the trc_rd_en_o signal is high. **Exists:** DWC_EQOS_SYS!=0&&!DWC_EQOS_SPRAM **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** clk_tx_i **Registered:** No
trc_rd_data_ecc_i[(DWC_EQOS_TX_ECW-1):0]	I	Transmit FIFO ECC Read Data. This bus carries the ECC data read from the Tx FIFO. It is valid when the trc_rd_en_o signal is high. **Exists:** DWC_EQOS_SYS!=0&&!DWC_EQOS_SPRAM&&DWC_EQOS_ASP_ECC **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** clk_tx_i **Registered:** No
trc_rd_en_o	O	Transmit FIFO Read Enable. When this signal is high, it indicates that the trc_rd_data_i signal should contain valid data. **Exists:** DWC_EQOS_SYS!=0&&!DWC_EQOS_SPRAM **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** clk_tx_i **Registered:** Yes

Port Name	I/O	Description
rwc_wr_addr_o[(DWC_EQOS_RX_FIFO_PTR_WIDTH-1):0]	O	Receive FIFO Write Address. This bus gives the address to the Rx FIFO write port. It is valid when the rwc_wr_csn_o signal is low. The address width is displayed in the coreConsultant after you configure the Rx FIFO size. The MTL Rx FIFO Address Bus Width field of the Buffer Management dialog box displays the address width. **Exists:** DWC_EQOS_SYS!=0&&!DWC_EQOS_SPRAM **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** clk_rx_i **Registered:** Yes
rwc_wr_csn_o	O	Receive FIFO Write Port Chip Select. This signal qualifies the rwc_wr_addr_o, rwc_wr_data_o, and rwc_wr_en_o signals. **Exists:** DWC_EQOS_SYS!=0&&!DWC_EQOS_SPRAM **Power Domain:** MAC_OFF **Active State:** Low **Synchronous to:** clk_rx_i **Registered:** No
rwc_wr_data_o[(DWC_EQOS_MEMW-1):0]	O	Receive FIFO Write Data. This bus carries the data to be written in the Rx FIFO. It is valid when the rwc_wr_csn_o is low and rwc_wr_en_o signals are high. The width of the FIFO write data bus depends the data bus width selected while configuring the core. The additional bits (3, 4, or 5 in 32-bit, 64-bit, or 128-bit data) are used for storing the tag or byte enable bits for the valid data lanes. **Exists:** DWC_EQOS_SYS!=0&&!DWC_EQOS_SPRAM **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** clk_rx_i **Registered:** Yes
rwc_wr_data_ecc_o[(DWC_EQOS_RX_ECW-1):0]	O	Receive FIFO ECC Write Data. This bus carries the ECC data(computed on {rwc_wr_addr_o,rwc_wr_data_o}) to be written in the Rx FIFO. It is valid when the rwc_wr_csn_o is low and rwc_wr_en_o signals are high. The width of the FIFO write data bus depends the data bus width and FIFO depth (address width) selected while configuring the core. **Exists:** DWC_EQOS_SYS!=0&&!DWC_EQOS_SPRAM&&DWC_EQOS_ASP_ECC **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** clk_rx_i **Registered:** Yes

Port Name	I/O	Description
rwc_wr_en_o	O	Receive FIFO Write Enable. When this signal is high, it indicates that the rwc_wr_data_o signal should be written into the memory. **Exists:** DWC_EQOS_SYS!=0&&!DWC_EQOS_SPRAM **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** clk_rx_i **Registered:** No
rrc_rd_addr_o[(DWC_EQOS_RX_FIFO_PTR_WIDTH-1):0]	O	Receive FIFO Read Address. This bus gives the address to the Rx FIFO read port. It is valid when the rrc_rd_csn_o signal is low. The width of address bus depends the Rx FIFO depth and the data bus width. **Exists:** DWC_EQOS_SYS!=0&&!DWC_EQOS_SPRAM **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** Application clock **Registered:** Yes
rrc_rd_data_i[(DWC_EQOS_MEMW-1):0]	I	Receive FIFO Read Data. This bus carries the data read from the Rx FIFO. It is valid when the rrc_rd_en_o signal is high. **Exists:** DWC_EQOS_SYS!=0&&!DWC_EQOS_SPRAM **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** Application clock **Registered:** No
rrc_rd_data_ecc_i[(DWC_EQOS_RX_ECW-1):0]	I	Receive FIFO ECC Read Data. This bus carries the ECC data read from the Rx FIFO. It is valid when the rrc_rd_en_o signal is high. **Exists:** DWC_EQOS_SYS!=0&&!DWC_EQOS_SPRAM&&DWC_EQOS_ASP_ECC **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** Application clock **Registered:** No
rrc_rd_csn_o	O	Receive Read Qualification. This signal qualifies the rrc_rd_addr_o signal. **Exists:** DWC_EQOS_SYS!=0&&!DWC_EQOS_SPRAM **Power Domain:** MAC_OFF **Active State:** Low **Synchronous to:** Application clock **Registered:** No

Port Name	I/O	Description
rrc_rd_en_o	O	Receive Read Enable. This signal indicates that rrc_rd_data_i signal should contain valid data. **Exists:** DWC_EQOS_SYS!=0&&!DWC_EQOS_SPRAM **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** Application clock **Registered:** Yes

16.30 SPRAM Interface Signals

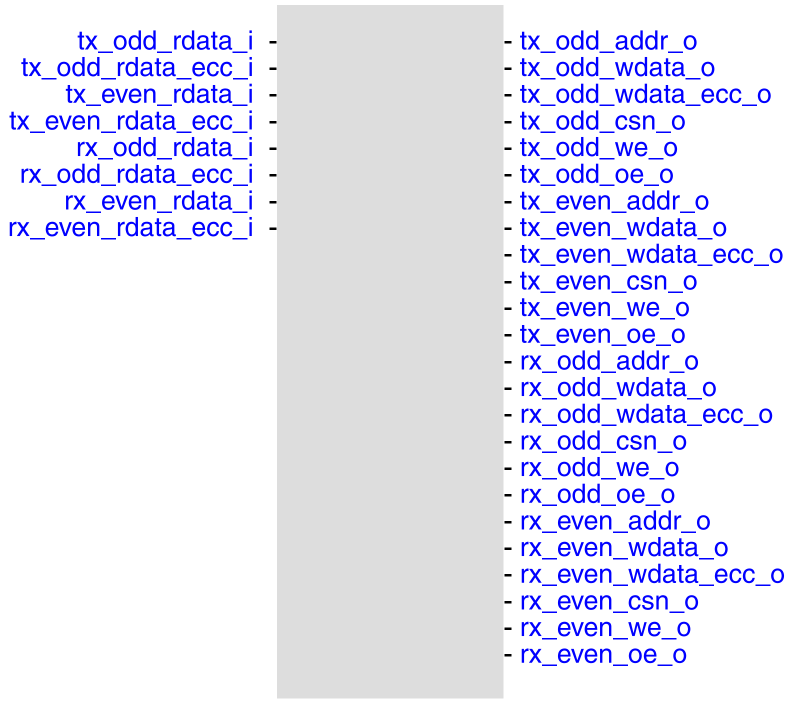


Table 16-30 SPRAM Interface Signals

Port Name	I/O	Description
tx_odd_addr_o[(DWC_EQOS_TX_FIFO_PTR_WIDTH-2):0]	O	Transmit FIFO Address, for odd memory. It is valid when tx_odd_csn_o is low. The address width is displayed in coreConsultant after you configure the Tx FIFO size. The MTL Tx FIFO Address Bus Width field of the Buffer Management dialog box displays the address width. **Exists:** DWC_EQOS_SYS!=0&&DWC_EQOS_SPRAM **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** Application clock **Registered:** No
tx_odd_wdata_o[(DWC_EQOS_MEMW-1):0]	O	Transmit FIFO write data, for odd memory. It is valid when both tx_odd_csn_o is low and tx_odd_we_o is high. The width of this signal depends the datawidth selected during configuration. The additional bits (3, 4, or 5 in 32-bit, 64-bit, or 128-bit data) are used for storing the tag or byte enable bits for the valid data lanes. **Exists:** DWC_EQOS_SYS!=0&&DWC_EQOS_SPRAM **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** Application clock **Registered:** No

Port Name	I/O	Description
tx_odd_wdata_ecc_o[(DWC_EQOS_TX_ECW-1):0]	O	Transmit FIFO ECC write data, for odd memory. It is valid when both tx_odd_csn_o is low and tx_odd_we_o is high. The width of this signal depends the datawidth and depth of FIFO (address width) selected during configuration. **Exists:** DWC_EQOS_SYS!=0&&DWC_EQOS_SPRAM&&DWC_EQOS_ASP_ECC **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** Application clock **Registered:** No
tx_odd_rdata_i[(DWC_EQOS_MEMW-1):0]	I	Transmit FIFO read data, for odd memory. It is valid when tx_odd_oe_o is asserted. The width of this signal depends the datawidth selected during configuration. The additional bits (3, 4, or 5 in 32-bit, 64-bit, or 128-bit data) are used for storing the tag or byte enable bits for the valid data lanes. **Exists:** DWC_EQOS_SYS!=0&&DWC_EQOS_SPRAM **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** Application clock **Registered:** No
tx_odd_rdata_ecc_i[(DWC_EQOS_TX_ECW-1):0]	I	Transmit FIFO ECC read data, for odd memory. It is valid when tx_odd_oe_o is asserted. The width of this signal depends the datawidth and depth of FIFO (address width) selected during configuration. **Exists:** DWC_EQOS_SYS!=0&&DWC_EQOS_SPRAM&&DWC_EQOS_ASP_ECC **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** Application clock **Registered:** No
tx_odd_csn_o	O	Transmit FIFO chip select, for odd memory. This signal qualifies tx_odd_addr_o and tx_odd_we_o **Exists:** DWC_EQOS_SYS!=0&&DWC_EQOS_SPRAM **Power Domain:** MAC_OFF **Active State:** Low **Synchronous to:** Application clock **Registered:** No

Port Name	I/O	Description
tx_odd_we_o	O	Transmit FIFO write enable, for odd memory. This signal controls the read/write operation. Indicates write operation when high and read operation when low. It is valid when tx_odd_csn_o is low. **Exists:** DWC_EQOS_SYS!=0&&DWC_EQOS_SPRAM **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** Application clock **Registered:** No
tx_odd_oe_o	O	Transmit FIFO output enable, for odd memory. This signal enables the output port of the memory **Exists:** DWC_EQOS_SYS!=0&&DWC_EQOS_SPRAM **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** Application clock **Registered:** No
tx_even_addr_o[(DWC_EQOS_TX_FIFO_PTR_WIDTH-2):0]	O	Transmit FIFO Address, for even memory. It is valid when tx_even_csn_o is low. The address width is displayed in coreConsultant after you configure the Tx FIFO size. The MTL Tx FIFO Address Bus Width field of the Buffer Management dialog box displays the address width. **Exists:** DWC_EQOS_SYS!=0&&DWC_EQOS_SPRAM **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** Application clock **Registered:** No
tx_even_wdata_o[(DWC_EQOS_MEMW-1):0]	O	Transmit FIFO write data, for even memory. It is valid when both tx_even_csn_o is low and tx_even_we_o is high. The width of this signal depends the datawidth selected during configuration. The additional bits (3, 4, or 5 in 32-bit, 64-bit, or 128-bit data) are used for storing the tag or byte enable bits for the valid data lanes. **Exists:** DWC_EQOS_SYS!=0&&DWC_EQOS_SPRAM **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** Application clock **Registered:** No

Port Name	I/O	Description
tx_even_wdata_ecc_o[(DWC_EQOS_TX_ECW-1):0]	O	Transmit FIFO ECC write data, for even memory. It is valid when both tx_even_csn_o is low and tx_even_we_o is high. The width of this signal depends the datawidth and depth of FIFO (address width) selected during configuration. **Exists:** DWC_EQOS_SYS!=0&&DWC_EQOS_SPRAM&&DWC_EQOS_ASP_ECC **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** Application clock **Registered:** No
tx_even_rdata_i[(DWC_EQOS_MEMW-1):0]	I	Transmit FIFO read data, for even memory. It is valid when tx_even_oe_o is asserted. The width of this signal depends the datawidth selected during configuration. The additional bits (3, 4, or 5 in 32-bit, 64-bit, or 128-bit data) are used for storing the tag or byte enable bits for the valid data lanes. **Exists:** DWC_EQOS_SYS!=0&&DWC_EQOS_SPRAM **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** Application clock **Registered:** No
tx_even_rdata_ecc_i[(DWC_EQOS_TX_ECW-1):0]	I	Transmit FIFO ECC read data, for even memory. It is valid when tx_even_oe_o is asserted. The width of this signal depends the datawidth and depth of FIFO (address width) selected during configuration. **Exists:** DWC_EQOS_SYS!=0&&DWC_EQOS_SPRAM&&DWC_EQOS_ASP_ECC **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** Application clock **Registered:** No
tx_even_csn_o	O	Transmit FIFO chip select, for even memory. This signal qualifies tx_even_addr_o and tx_even_we_o **Exists:** DWC_EQOS_SYS!=0&&DWC_EQOS_SPRAM **Power Domain:** MAC_OFF **Active State:** Low **Synchronous to:** Application clock **Registered:** No

Port Name	I/O	Description
tx_even_we_o	O	Transmit FIFO write enable, for even memory. This signal controls the read/write operation. Indicates write operation when high and read operation when low. It is valid when tx_even_csn_o is low. **Exists:** DWC_EQOS_SYS!=0&&DWC_EQOS_SPRAM **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** Application clock **Registered:** No
tx_even_oe_o	O	Transmit FIFO output enable, for even memory. This signal enables the output port of the memory **Exists:** DWC_EQOS_SYS!=0&&DWC_EQOS_SPRAM **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** Application clock **Registered:** No
rx_odd_addr_o[(DWC_EQOS_RX_FIFO_PTR_WIDTH-2):0]	O	Receive FIFO Address, for odd memory. It is valid when rx_odd_csn_o is low. The address width is displayed in coreConsultant after you configure the Rx FIFO size. The MTL Rx FIFO Address Bus Width field of the Buffer Management dialog box displays the address width. **Exists:** DWC_EQOS_SYS!=0&&DWC_EQOS_SPRAM **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** Application clock **Registered:** No
rx_odd_wdata_o[(DWC_EQOS_MEMW-1):0]	O	Receive FIFO write data, for odd memory. It is valid when both rx_odd_csn_o is low and rx_odd_we_o is high. The width of this signal depends the datawidth selected during configuration. The additional bits (3, 4, or 5 in 32-bit, 64-bit, or 128-bit data) are used for storing the tag or byte enable bits for the valid data lanes. **Exists:** DWC_EQOS_SYS!=0&&DWC_EQOS_SPRAM **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** Application clock **Registered:** No

Port Name	I/O	Description
rx_odd_wdata_ecc_o[(DWC_EQOS_RX_ECW-1):0]	O	Receive FIFO ECC write data, for odd memory. It is valid when both rx_odd_csn_o is low and rx_odd_we_o is high. The width of this signal depends the datawidth and depth of FIFO (address width) selected during configuration. **Exists:** DWC_EQOS_SYS!=0&&DWC_EQOS_SPRAM&&DWC_EQOS_ASP_ECC **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** Application clock **Registered:** No
rx_odd_rdata_i[(DWC_EQOS_MEMW-1):0]	I	Receive FIFO read data, for odd memory. It is valid when rx_odd_oe_o is asserted. The width of this signal depends the datawidth selected during configuration. The additional bits (3, 4, or 5 in 32-bit, 64-bit, or 128-bit data) are used for storing the tag or byte enable bits for the valid data lanes. **Exists:** DWC_EQOS_SYS!=0&&DWC_EQOS_SPRAM **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** Application clock **Registered:** No
rx_odd_rdata_ecc_i[(DWC_EQOS_RX_ECW-1):0]	I	Receive FIFO ECC read data, for odd memory. It is valid when rx_odd_oe_o is asserted. The width of this signal depends the datawidth and depth (address width) selected during configuration. **Exists:** DWC_EQOS_SYS!=0&&DWC_EQOS_SPRAM&&DWC_EQOS_ASP_ECC **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** Application clock **Registered:** No
rx_odd_csn_o	O	Receive FIFO chip select, for odd memory. This signal qualifies rx_odd_addr_o and rx_odd_we_o **Exists:** DWC_EQOS_SYS!=0&&DWC_EQOS_SPRAM **Power Domain:** MAC_OFF **Active State:** Low **Synchronous to:** Application clock **Registered:** No

Port Name	I/O	Description
rx_odd_we_o	O	Receive FIFO write enable, for odd memory. This signal controls the read/write operation. Indicates write operation when high and read operation when low. It is valid when rx_odd_csn_o is low. **Exists:** DWC_EQOS_SYS!=0&&DWC_EQOS_SPRAM **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** Application clock **Registered:** No
rx_odd_oe_o	O	Receive FIFO output enable, for odd memory. This signal enables the output port of the memory **Exists:** DWC_EQOS_SYS!=0&&DWC_EQOS_SPRAM **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** Application clock **Registered:** No
rx_even_addr_o[(DWC_EQOS_RX_FIFO_PTR_WIDTH-2):0]	O	Receive FIFO Address, for even memory. It is valid when rx_even_csn_o is low. The address width is displayed in coreConsultant after you configure the Rx FIFO size. The MTL Rx FIFO Address Bus Width field of the Buffer Management dialog box displays the address width. **Exists:** DWC_EQOS_SYS!=0&&DWC_EQOS_SPRAM **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** Application clock **Registered:** No
rx_even_wdata_o[(DWC_EQOS_MEMW-1):0]	O	Receive FIFO write data, for even memory. It is valid when both rx_even_csn_o is low and rx_even_we_o is high. The width of this signal depends the datawidth selected during configuration. The additional bits (3, 4, or 5 in 32-bit, 64-bit, or 128-bit data) are used for storing the tag or byte enable bits for the valid data lanes. **Exists:** DWC_EQOS_SYS!=0&&DWC_EQOS_SPRAM **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** Application clock **Registered:** No

Port Name	I/O	Description
rx_even_wdata_ecc_o[(DWC_EQOS_RX_ECW-1):0]	O	Receive FIFO ECC write data, for even memory. It is valid when both rx_even_csn_o is low and rx_even_we_o is high. The width of this signal depends the datawidth and depth of FIFO (address width) selected during configuration. **Exists:** DWC_EQOS_SYS!=0&&DWC_EQOS_SPRAM&&DWC_EQOS_ASP_ECC **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** Application clock **Registered:** No
rx_even_rdata_i[(DWC_EQOS_MEMW-1):0]	I	Receive FIFO read data, for even memory. It is valid when rx_even_oe_o is asserted. The width of this signal depends the datawidth selected during configuration. The additional bits (3, 4, or 5 in 32-bit, 64-bit, or 128-bit data) are used for storing the tag or byte enable bits for the valid data lanes. **Exists:** DWC_EQOS_SYS!=0&&DWC_EQOS_SPRAM **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** Application clock **Registered:** No
rx_even_rdata_ecc_i[(DWC_EQOS_RX_ECW-1):0]	I	Receive FIFO ECC read data, for even memory. It is valid when rx_even_oe_o is asserted. The width of this signal depends the datawidth and depth of FIFO (address width) selected during configuration. **Exists:** DWC_EQOS_SYS!=0&&DWC_EQOS_SPRAM&&DWC_EQOS_ASP_ECC **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** Application clock **Registered:** No
rx_even_csn_o	O	Receive FIFO chip select, for even memory. This signal qualifies rx_even_addr_o and rx_even_we_o **Exists:** DWC_EQOS_SYS!=0&&DWC_EQOS_SPRAM **Power Domain:** MAC_OFF **Active State:** Low **Synchronous to:** Application clock **Registered:** No

Port Name	I/O	Description
rx_even_we_o	O	Receive FIFO write enable, for even memory. This signal controls the read/write operation. Indicates write operation when high and read operation when low. It is valid when rx_even_csn_o is low. **Exists:** DWC_EQOS_SYS!=0&&DWC_EQOS_SPRAM **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** Application clock **Registered:** No
rx_even_oe_o	O	Receive FIFO output enable, for even memory. This signal enables the output port of the memory **Exists:** DWC_EQOS_SYS!=0&&DWC_EQOS_SPRAM **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** Application clock **Registered:** No

16.31 MAC Tx Interface Signals

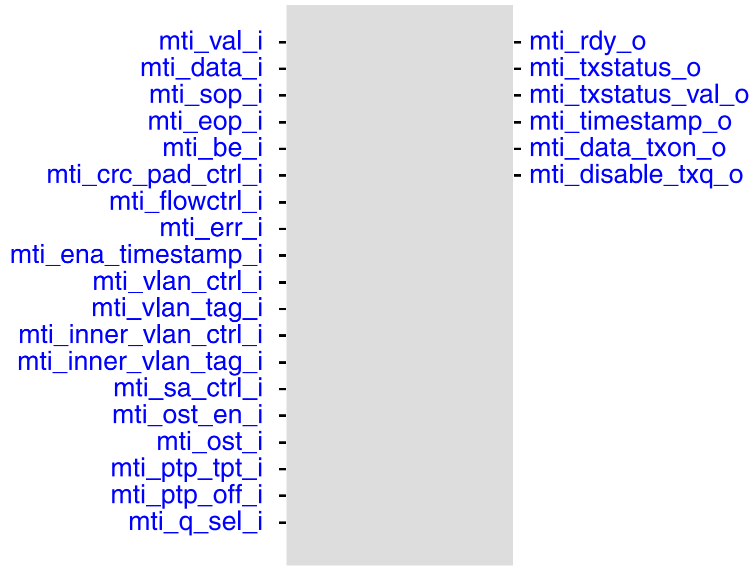


Table 16-31 MAC Tx Interface Signals

Port Name	I/O	Description
mti_val_i	I	Transmit Data Valid. The application drives this signal high to indicate to the MAC that data is available for transmission the mti_data_i bus. This signal also qualifies the validity of mti_be_i, mti_sop_i, and mti_eop_i signals. **Exists:** DWC_EQOS_SYS == 0 **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** clk_tx_i **Registered:** No
mti_data_i[(DWC_EQOS_DATAWIDTH-1):0]	I	Transmit Data Bus. The data for transmission is input to the MAC this bus. You can configure the MAC to process the data in the little-endian or bigendian format. In the little-endian format, the LSB lane mti_data_i[7:0] is the first byte transmitted by the MAC. In the big-endian format, the MSB lane mti_data_i[DWC_EQOS_DATAWIDTH-1: DWC_EQOS_DATAWIDTH-8] is the first byte transmitted by the MAC. **Exists:** DWC_EQOS_SYS == 0 **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** clk_tx_i **Registered:** Yes

Port Name	I/O	Description
mti_sop_i	I	Transmit Start of Packet. This signal indicates that the start of an Ethernet packet is being transferred to the MAC. **Exists:** DWC_EQOS_SYS == 0 **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** clk_tx_i **Registered:** No
mti_eop_i	I	Transmit End of Packet. This signal indicates that the end of an Ethernet packet is being transferred to the MAC. **Exists:** DWC_EQOS_SYS == 0 **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** clk_tx_i **Registered:** No
mti_be_i[((DWC_EQOS_DATAWIDTH/64)+1):0]	I	Transmit Byte Lanes This signal indicates the number of byte lanes valid on the mti_data_i bus when the mti_eop_i signal is high. Total number of valid bytes = mti_be_i bytes + 1 byte When the mti_eop_i signal is low, the MAC always considers all byte lanes to have valid data. **Exists:** DWC_EQOS_SYS == 0 **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** clk_tx_i **Registered:** Yes

Port Name	I/O	Description
mti_crc_pad_ctrl_i[1:0]	I	Transmit Cyclic Redundancy Check (CRC) and PAD Control. This signal controls the insertion of the FCS bytes and PAD bits in the packet being transmitted by the MAC. This signal is sampled only when the mti_sop_i signal is high. The MAC decodes the signal as follows: ■ 2'b00: CRC and Pad Insertion The MAC appends CRC at the end of transmitted packet having length greater than or equal to 60 bytes. The MAC automatically adds padding and CRC to a packet having length less than 60 bytes. ■ 2'b01: CRC Insertion (Disable Pad Insertion) The MAC appends CRC to the end of the transmitted packet, but it does not append padding. The application should ensure that the padding bytes are present in the packet being transferred. ■ 2'b10: Disable CRC Insertion The MAC does not append the CRC to the end of the transmitted packet. The application should ensure that the padding and CRC bytes are present in the packet being transferred. ■ 2'b11: CRC Replacement The MAC replaces the last four bytes of the transmitted packet with recalculated CRC bytes. The application should ensure that the padding and CRC bytes are present in the packet being transferred. **Exists:** DWC_EQOS_SYS == 0 **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** clk_tx_i **Registered:** No
mti_flowctrl_i[(DWC_EQOS_TXFCBW-1):0]	I	Transmit Flow Control. When this signal is high, it instructs the MAC to transmit the Pause packets in the full-duplex mode. In the half-duplex mode, the MAC enables the backpressure function until this signal is made low again. When Pause Flow Control (PFC) is enabled, the corresponding bit indicates the MAC to transmit PFC packet with pause quanta for programmed priorities for corresponding Rx queue. **Exists:** DWC_EQOS_SYS == 0 **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** clk_tx_i **Registered:** Yes
mti_rdy_o	O	Transmit Data Ready. When this signal is high, it indicates that the MAC is ready to accept the data input the mti_data_i bus. **Exists:** DWC_EQOS_SYS == 0 **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** clk_tx_i **Registered:** No

Port Name	I/O	Description
mti_err_i	I	Transmit Error Input. When this signal is high, it indicates to the MAC to corrupt the packet currently being transmitted by forcing the CRC error. This signal is sampled along with the mti_eop_i signal. **Exists:** DWC_EQOS_SYS == 0 **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** clk_tx_i **Registered:** No
mti_txstatus_o[31:0]	O	Transmit Status. This bus gives the transmission status of the last Ethernet frame sent to the MAC on the mti_data_i bus. **Exists:** DWC_EQOS_SYS == 0 **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** clk_tx_i **Registered:** Yes
mti_txstatus_val_o	O	Transmit Status Valid. This signal is set high for 1 clock cycle to indicate that mti_txstatus_o gives valid status for the last transmitted packet. **Exists:** DWC_EQOS_SYS == 0 **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** clk_tx_i **Registered:** No
mti_timestamp_o[63:0]	O	Transmit Packet Timestamping. This bus provides the timestamp of the packet transmitted by the MAC. The value this bus is valid only when the mti_txstatus_val_o signal is high. **Exists:** DWC_EQOS_TIME_STAMPING&&DWC_EQOS_CORE **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** clk_tx_i **Registered:** Yes

Port Name	I/O	Description
mti_ena_timestamp_i	I	Transmit Enable Timestamping. This signal directs the MAC to capture the timestamp for the Transmit packet. The application should drive this signal high only for those PTP packets that require timestamping. This signal is sampled only when mti_sop_i and mti_val_i signals are high. **Exists:** DWC_EQOS_TIME_STAMPING&&DWC_EQOS_CORE **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** clk_tx_i **Registered:** No
mti_vlan_ctrl_i[1:0]	I	VLAN Insertion Control. This encoded signal requests the MAC to perform VLAN tagging or untagging before transmitting the packets. If the packet is modified for VLAN tags, the MAC automatically recalculates and replaces the CRC bytes. This signal is sampled only when the mti_sop_i signal is high. The following list describes the values: ■ 2'b00: Do not add a VLAN tag. ■ 2'b01: Remove the VLAN tag from the packets before transmission. This option should be used only with the VLAN packets. ■ 2'b10: Insert a VLAN tag as indicated by mti_vlan_tag_i. This option should be used only with the VLAN packets. ■ 2'b11: Replace the VLAN tag in packets as indicated by mti_vlan_tag_i. This option should be used only with the VLAN packets. **Exists:** DWC_EQOS_SA_VLAN_INS_CTRL_EN&&DWC_EQOS_CORE **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** clk_tx_i **Registered:** No
mti_vlan_tag_i[15:0]	I	VLAN Tag for Insertion. This signal contains the VLAN Tag for Insertion or Replacement in the Transmit packet based on the mti_vlan_ctrl_i signal. **Exists:** DWC_EQOS_SA_VLAN_INS_CTRL_EN&&DWC_EQOS_CORE **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** clk_tx_i **Registered:** No

Port Name	I/O	Description
mti_inner_vlan_ctrl_i[1:0]	I	Inner VLAN Insertion Control. This encoded signal requests the MAC to perform inner VLAN tagging or untagging before transmitting the packets. The application needs to set appropriate mti_crc_pad_ctrl_i value when this control is enabled. This signal is sampled only when the mti_sop_i signal is asserted. The following list describes the values: ■ 2'b00: Do not add a inner VLAN tag. ■ 2'b01: Remove the inner VLAN tag from the packet before transmission. This option should be used only with the Double VLAN packets. ■ 2'b10: Insert a inner VLAN tag as indicated by mti_inner_vlan_tag_i signal. This option should be used only with the VLAN packets. ■ 2'b11: Replace the inner VLAN tag in packets as indicated by mti_inner_vlan_tag_i signal. This option should be used only with the Double VLAN packets. **Exists:** DWC_EQOS_SA_VLAN_INS_CTRL_EN&&DWC_EQOS_DOUBLE_VLAN_EN&&DWC_EQOS_CORE **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** clk_tx_i **Registered:** No
mti_inner_vlan_tag_i[15:0]	I	Inner VLAN Tag for Insertion. This signal contains the inner VLAN Tag for insertion or replacement in the Transmit packet based on the mti_inner_vlan_ctrl_i signal. **Exists:** DWC_EQOS_SA_VLAN_INS_CTRL_EN&&DWC_EQOS_DOUBLE_VLAN_EN&&DWC_EQOS_CORE **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** clk_tx_i **Registered:** No

Port Name	I/O	Description
mti_sa_ctrl_i[2:0]	I	Source Address Insertion Control. This encoded signal requests the MAC to add or replace the Source Address field in the Ethernet packet with the value given in the MAC Address 0 register. This signal is sampled only when the mti_sop_i signal is high. If the Source Address field is modified in a packet, the MAC automatically recalculates and replaces the CRC bytes. Bit 2 specifies the MAC Address Register (0 or 1) that is used for Source Address insertion or replacement. The following list describes the values of Bits [1:0]: ■ 2'b00: Do not include the source address. ■ 2'b01: Include or insert the source address. For reliable transmission, the application must provide frames without source addresses. ■ 2'b10: Replace the source address. For reliable transmission, the application must provide frames with source addresses. ■ 2'b11: Reserved **Exists:** DWC_EQOS_SA_VLAN_INS_CTRL_EN&&DWC_EQOS_CORE **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** clk_tx_i **Registered:** No
mti_data_txon_o	O	Transmit AV. This signal indicates that the MAC is transmitting a packet. **Exists:** DWC_EQOS_NUM_TXQ>1&&DWC_EQOS_CORE **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** clk_tx_i **Registered:** Yes
mti_ost_en_i	I	Transmit Enable One-Step Time Correction. When this signal is high, it directs the MAC to perform one-step time correction for the Transmit packet. It also validates the mti_ost_i signal. The application should drive this signal high only for those PTP packets that require one-step time correction. This signal is sampled only when the mti_sop_i and mti_val_i signals are high. **Exists:** DWC_EQOS_OST_EN&&DWC_EQOS_CORE **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** clk_tx_i **Registered:** No

Port Name	I/O	Description
mti_ost_i[63:0]	I	Transmit One-Step Time Correction Input. This bus contains the timestamp required for one-step time correction. This is validated by the mti_ost_en_i signal. **Exists:** DWC_EQOS_OST_EN&&DWC_EQOS_CORE **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** clk_tx_i **Registered:** No
mti_ptp_tpt_i[1:0]	I	Transmit PTP transport protocol type. This signal indicates the type of transport protocol over which PTP messages are sent. The following list describes the values of these bits:. ■ 2'b00: Reserved ■ 2'b01: PTP over Ethernet ■ 2'b10: PTP over UDP/IPv4 ■ 2'b11: PTP over UDP/IPv6. This signal is sampled only when the mti_sop_i and mti_val_i signals are high and is valid when mti_ost_en_i is high. **Exists:** DWC_EQOS_POU_OST_EN&&DWC_EQOS_CORE **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** clk_tx_i **Registered:** Yes
mti_ptp_off_i[13:0]	I	Transmit PTP offset. This signal indicates the starting byte location of the PTP message, for the packet sent on the MTI interface. This signal is sampled only when the signals mti_sop_i and mti_val_i are high and is valid when mti_ost_en_i is high and mti_ptp_tpt_i is 10 or 11. For example, for PTP message sent over UDP/IPv4 over untagged Ethernet packet, the offset is 42. **Exists:** DWC_EQOS_POU_OST_EN&&DWC_EQOS_CORE **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** clk_tx_i **Registered:** yes
mti_q_sel_i[(DWC_EQOS_TXQW-1):0]	I	Transmit Queue Select. This signal indicates the Tx queue from which the transmit packet arrives to MAC transmitter. This signal is sampled only when the mti_sop_i and mti_val_i signals are high and is valid when mti_ost_en_i is high. **Exists:** DWC_EQOS_CBTI_EN&&DWC_EQOS_CORE **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** clk_tx_i **Registered:** Yes

Port Name	I/O	Description
mti_disable_txq_o[(DWC_EQOS_NUM_TXQ-1):0]	O	MTI Disable Tx Queue Signal. This signal indicates that the corresponding Tx queue needs to be disabled for transmission. This happens when a PFC packet containing nonzero pause time for priorities corresponding to the Tx queue is received. **Exists:** DWC_EQOS_DCB_EN&&DWC_EQOS_CORE&&DWC_EQOS_NUM_TXQ>1 **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** clk_tx_i **Registered:** Yes

16.32 MAC Rx Interface Signals

- mri_val_o
- mri_data_o
- mri_be_o
- mri_sop_o
- mri_eop_o
- mri_rxstatus_o
- mri_timestamp_o
- mri_timestamp_val_o
- mri_q_sel_o

Table 16-32 MAC Rx Interface Signals

Port Name	I/O	Description
mri_val_o	O	Receive Data Valid. The MAC drives this signal high to indicate to the application (MTL) that the received packet data is available on the mri_data_o bus. This signal also qualifies the mri_data_o, mri_be_o, mri_sop_o, and mri_eop_o signals. **Exists:** DWC_EQOS_SYS == 0 **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** clk_rx_i **Registered:** Yes
mri_data_o[(DWC_EQOS_DATAWIDTH-1):0]	O	Receive Data Bus. This signal carries the received packet data from the MAC to the MTL. This signal is valid only when the mri_val_o signal is high. In big-endian format, the MSB lane mri_data_o[DWC_EQOS_DATAWIDTH-1:DWC_EQOS_DATAWIDTH-8] is the first byte received by the DWC_ether_qos core. In little-endian format, the LSB lane mri_data_o[7:0] is the first byte received by the DWC_ether_qos core. **Exists:** DWC_EQOS_SYS == 0 **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** clk_rx_i **Registered:** Yes

Port Name	I/O	Description
mri_be_o[((DWC_EQOS_DATAWIDTH/64)+1):0]	O	Receive Byte Lanes This signal indicates the number of byte lanes valid on the mri_data_o signal when the mri_eop_o signal is high. The total number of valid bytes is equal to mri_be_o + 1 bytes. The width of this bus depends on the width of the data bus. When the mri_eop_i signal is low, the MAC always drives all-ones on the mri_be_o signal. When the mri_be_o signal width is 2, the following values indicate the lanes that have the data: ■ 11: All 4 byte lanes have the data ■ 10: LS 3 byte lane has the data ■ 01: LS 2 byte lane has the data ■ 00: LS 1 byte lane has the data **Exists:** DWC_EQOS_SYS == 0 **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** clk_rx_i **Registered:** Yes
mri_sop_o	O	Receive Start of Packet. This signal indicates that the start of an Ethernet packet is being transferred from the MAC. **Exists:** DWC_EQOS_SYS == 0 **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** clk_rx_i **Registered:** Yes
mri_eop_o	O	Receive End of Packet. This signal indicates that the end of an Ethernet packet is being transferred from the MAC. **Exists:** DWC_EQOS_SYS == 0 **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** clk_rx_i **Registered:** Yes
mri_rxstatus_o[127:0]	O	Receive Status. This bus gives the status of the received Ethernet packet from the MAC the mri_data_i signal. This signal is valid only when the mri_eop_o signal is high. The width of the bus can be 112-bit or 128-bit. The status is 112-bit wide by default. The status width is 128-bit when you select the Enable Double VLAN Processing option. **Exists:** DWC_EQOS_SYS == 0 **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** clk_rx_i **Registered:** Yes

Port Name	I/O	Description
mri_timestamp_o[63:0]	O	Receive Packet Timestamping. This bus provides the timestamp of received packet. The value on this bus is valid only when the mri_eop_o signal is high. **Exists:** DWC_EQOS_TIME_STAMPING&&DWC_EQOS_CORE **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** clk_rx_i **Registered:** Yes
mri_timestamp_val_o	O	Receive Timestamp Valid. This signal validates the presence of timestamp for the Receive packet. **Exists:** DWC_EQOS_TIME_STAMPING&&DWC_EQOS_CORE **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** clk_rx_i **Registered:** Yes
mri_q_sel_o[(DWC_EQOS_RXQW-1):0]	O	Receive Queue Select. This signal indicates the Rx queue to which the received packet goes. **Exists:** DWC_EQOS_NUM_RXQ>1&&DWC_EQOS_CORE **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** clk_rx_i **Registered:** Yes

16.33 EST Memory Interface Signals

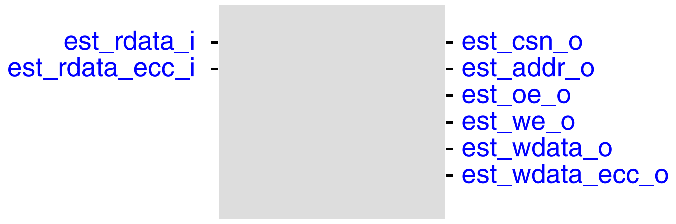


Table 16-33 EST Memory Interface Signals

Port Name	I/O	Description
est_csn_o	O	EST Memory Chip Select. This signal should be used as a chip select to EST Memory. **Exists:** DWC_EQOS_AV_EST **Power Domain:** MAC_OFF **Active State:** Low **Synchronous to:** clk_ptp_ref_i **Registered:** No
est_addr_o[DWC_EQOS_EMAW:0]	O	EST Memory Address Bus. EST Memory Address bus to be connected to External EST Memory. **Exists:** DWC_EQOS_AV_EST **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** clk_ptp_ref_i **Registered:** Yes
est_oe_o	O	EST Memory Output Enable. This signal should be used as a output enable to EST Memory. Memory block is expected to drive Read Data on est_rdata_i when this signal is High. **Exists:** DWC_EQOS_AV_EST **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** clk_ptp_ref_i **Registered:** No
est_we_o	O	EST Memory Write Enable. This signal should be used as a write enable to EST Memory. Memory block is expected to write data provided on est_wdata_o to the address provided on est_addr_o. **Exists:** DWC_EQOS_AV_EST **Power Domain:** MAC_OFF **Active State:** High **Synchronous to:** clk_ptp_ref_i **Registered:** No

Port Name	I/O	Description
est_wdata_o[(DWC_EQOS_EMW-1):0]	O	EST Memory Write Data. This is the write data bus to EST Memory. **Exists:** DWC_EQOS_AV_EST **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** clk_ptp_ref_i **Registered:** Yes
est_wdata_ecc_o[(DWC_EQOS_EST_ECW-1):0]	O	EST Memory ECC Write Data. This is the ECC write data bus to EST Memory. **Exists:** DWC_EQOS_AV_EST&&DWC_EQOS_ASP_ECC **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** clk_ptp_ref_i **Registered:** Yes
est_rdata_i[(DWC_EQOS_EMW-1):0]	I	EST Memory Read Data. This is the read data bus from EST Memory. **Exists:** DWC_EQOS_AV_EST **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** clk_ptp_ref_i **Registered:** Yes
est_rdata_ecc_i[(DWC_EQOS_EST_ECW-1):0]	I	EST Memory ECC Read Data. This is the ECC read data bus from EST Memory. **Exists:** DWC_EQOS_AV_EST&&DWC_EQOS_ASP_ECC **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** clk_ptp_ref_i **Registered:** Yes

16.34 Debug Bus Interface Signals

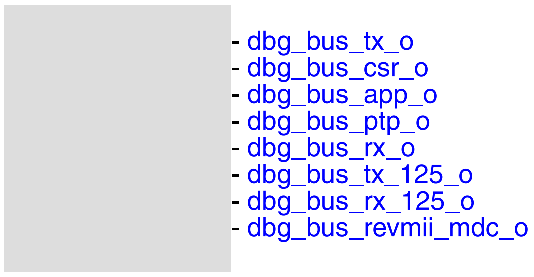


Table 16-34 Debug Bus Interface Signals

Port Name	I/O	Description
dbg_bus_tx_o[0:0]	O	Tx Clock Domain Debug Bus. This is the Debug Bus of signals from Tx Clock Domain. **Exists:** DWC_EQOS_DBG_BUS **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** clk_tx_i **Registered:** No
dbg_bus_csr_o[0:0]	O	CSR Clock Domain Debug Bus. This is the Debug Bus of signals from CSR Clock Domain. **Exists:** DWC_EQOS_DBG_BUS&&(DWC_EQOS_CSR_SLV_CLK||DWC_EQOS_CORE) **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** CSR Clock **Registered:** No
dbg_bus_app_o[0:0]	O	Application Clock Domain Debug Bus. This is the Debug Bus of signals from Application Clock Domain. **Exists:** DWC_EQOS_DBG_BUS&&DWC_EQOS_SYS>0 **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** Application Clock **Registered:** No

Port Name	I/O	Description
dbg_bus_ptp_o[0:0]	O	PTP Clock Domain Debug Bus. This is the Debug Bus of signals from PTP Clock Domain. **Exists:** DWC_EQOS_DBG_BUS&&(DWC_EQOS_AV_EST||DWC_EQOS_FLEXI_PPS_OUT_EN) **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** clk_ptp_ref_i **Registered:** No
dbg_bus_rx_o[0:0]	O	Rx Clock Domain Debug Bus. This is the Debug Bus of signals from Rx Clock Domain. **Exists:** DWC_EQOS_DBG_BUS **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** clk_rx_i **Registered:** No
dbg_bus_tx_125_o[0:0]	O	Tx 125MHz Clock Domain Debug Bus. This is the Debug Bus of signals from Tx 125MHz Clock Domain. **Exists:** DWC_EQOS_DBG_BUS&&(DWC_EQOS_PCS_EN||DWC_EQOS_SMII_EN) **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** clk_tx_125_i **Registered:** No
dbg_bus_rx_125_o[0:0]	O	Rx 125MHz Clock Domain Debug Bus. This is the Debug Bus of signals from Rx 125MHz Clock Domain. **Exists:** DWC_EQOS_DBG_BUS&&(DWC_EQOS_PCS_EN||DWC_EQOS_SMII_EN) **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** clk_rx_125_i **Registered:** No
dbg_bus_revmii_mdc_o[2:0]	O	RevMII MDC Clock Domain Debug Bus. This is the Debug Bus of signals from RevMII MDC Clock Domain. **Exists:** DWC_EQOS_DBG_BUS&&DWC_EQOS_REVMII_EN **Power Domain:** MAC_OFF **Active State:** N/A **Synchronous to:** revmii_mdc_i **Registered:** No

17 Register Descriptions

This chapter details all possible registers in the controller. They are arranged hierarchically into maps and blocks (banks). For configurable IP titles, your actual configuration might not contain all of these registers.

Attention: For configurable IP titles, do not use this document to determine the exact attributes of your register map. It is for reference purposes only.

When you configure the controller in coreConsultant, you must access the register attributes for your actual configuration at `workspace/report/ComponentRegisters.html` or `workspace/report/ComponentRegisters.xml` after you have completed the report creation activity. That report comes from the exact same source as this chapter but removes all the registers that are not in your actual configuration. This does not apply to non-configurable IP titles. In addition, all parameter expressions are evaluated to actual values. Therefore, the Offset and Memory Access values might change d pending on your actual configuration.

Some expressions might refer to TCL functions or procedures (sometimes identified as **<functionof>**) that coreConsultant uses to make calculations. The exact formula used by these TCL functions is not provided in this chapter. However, when you configure the controller in coreConsultant, all TCL functions and parameters are evaluated completely; and the resulting values are displayed where appropriate in the coreConsultant GUI reports.

Exists Expressions

The Exist expressions indicate the combination of configuration parameters required for a register, field, or block to exist in the memory map. The expression is only valid in the local context and does not indicate the conditions for existence of the parent. For example, the Exists expression for a bit field in a register assumes that the register exists and does not include the conditions for existence of the register.

Offset

The term *Offset* is synonymous with *Address.*

Memory Access Attributes

The Memory Access attribute is defined as `<ReadBehavior>/<WriteBehavior>` which are defined in the following table.

Table 17-1 Possible Read and Write Behaviors

Read (or Write) Behavior	Description
RC	A read clears this register field.
RS	A read sets this register field.
RM	A read modifies the contents of this register field.
Wo	You can only write to this register once field.
W1C	A write of 1 clears this register field.
W1S	A write of 1 sets this register field.
W1T	A write of 1 toggles this register field.
W0C	A write of 0 clears this register field.
W0S	A write of 0 sets this register field.
W0T	A write of 0 toggles this register field.
WC	Any write clears this register field.
WS	Any write sets this register field.
WM	Any write toggles this register field.
no Read Behavior attribute	You cannot read this register. It is Write-Only.
no Write Behavior attribute	You cannot write to this register. It is Read-Only.

Table 17-2 Memory Access Examples

Memory Access	Description
R	Read-only register field.
W	Write-only register field.
R/W	Read/write register field.
R/W1C	You can read this register field. Writing 1 clears it.
RC/W1C	Reading this register field clears it. Writing 1 clears it.
R/Wo	You can read this register field. You can only write to it once.

Special Optional Attributes

Some register fields might use the following optional attributes.

Table 17-3 Optional Attributes

Attribute	Description
Volatile	As defined by the IP-XACT specification. If true, indicates in the case of a write followed by read, or in the case of two consecutive reads, there is no guarantee as to what is returned by the read on the second transaction or that this return value is consistent with the write or read of the first transaction. The element implies there is some additional mechanism by which this field can acquire new values other than by reads/writes/resets and other access methods known to IP-XACT. For example, when the controller updates the register field contents.
Testable	As defined by the IP-XACT specification. Possible values are unconstrained, untestable, readOnly, writeAsRead, restore. Untestable means that this field is untestable by a simple automated register test. For example, the read-write access of the register is controlled by a pin or another register. readOnly means that you should not write to this register; only read from it. This might apply for a register that modifies the contents of another register.
Reset Mask	As defined by the IP-XACT specification. Indicates that this register field has an unknown reset value. For example, the reset value is set by another register or an input pin; or the register is implemented using RAM.
* Varies	Indicates that the memory access (or reset) attribute (read, write behavior) is not fixed. For example, the read-write access of the register is controlled by a pin or another register. Or when the access depends on some configuration parameter; in this case the post-configuration report in coreConsultant gives the actual access value.

Register definitions for each component memory map.

Table 17-4 Registers for the DWC_eqos_top_map Memory Map

Register	Offset	Description
The address block EQOS_MAC contains up to 190 Registers: MAC_Configuration to MAC_Log_Message_Interval EQOS_MAC		
"MAC_Configuration" on page 626	0x0	The MAC Configuration Register establishes the operating mode of the MAC.
"MAC_Ext_Configuration" on page 637	0x4	The MAC Extended Configuration Register establishes the operating mode of the MAC.
"MAC_Packet_Filter" on page 642	0x8	The MAC Packet Filter register contains the filter controls for receiving packets. Some of the controls...
"MAC_Watchdog_Timeout" on page 647	0xc	The Watchdog Timeout register controls the watchdog timeout for received packets.
"MAC_Hash_Table_Reg0" on page 649	0x10	The Hash Table Register 0 contains the first 32 bits of the hash table, when the width of the hash...

Register	Offset	Description
"MAC_Hash_Table_Reg1" on page 650	0x14	The Hash Table Register 1 contains the second 32 bits of the hash table. You can specify the width...
"MAC_Hash_Table_Reg2" on page 651	0x18	The Hash Table Register 2 contains the third 32 bits of the hash table. You can specify the width...
"MAC_Hash_Table_Reg3" on page 652	0x1c	The Hash Table Register 3 contains the fourth 32 bits of the hash table. You can specify the width...
"MAC_Hash_Table_Reg4" on page 653	0x20	The Hash Table Register 4 contains the fifth 32 bits of the hash table. You can specify the width...
"MAC_Hash_Table_Reg5" on page 654	0x24	The Hash Table Register 5 contains the sixth 32 bits of the hash table. You can specify the width...
"MAC_Hash_Table_Reg6" on page 655	0x28	The Hash Table Register 6 contains the seventh 32 bits of the hash table. You can specify the width...
"MAC_Hash_Table_Reg7" on page 656	0x2c	The Hash Table Register 7 contains the eighth 32 bits of the hash table. You can specify the width...
"MAC_VLAN_Tag" on page 657	0x50	The VLAN Tag register identifies the IEEE 802.1Q VLAN type packets.
"MAC_VLAN_Tag_Ctrl" on page 662	0x50	This register is the redefined format of the MAC VLAN Tag Register. It is used for indirect addressing....
"MAC_VLAN_Tag_Data" on page 667	0x54	This register holds the read/write data for Indirect Access of the Per VLAN Tag registers. During...
"MAC_VLAN_Tag_Filter(#i) (for i = 0; i <= DWC_EQOS_NRVF-1)" on page 670	0x54	This register contains VLAN Tag filter ${i} control information.
"MAC_VLAN_Hash_Table" on page 673	0x58	When VTHM bit of the MAC_VLAN_Tag register is set, the 16-bit VLAN Hash Table register is used for...
"MAC_VLAN_Incl(#i) (for i = 0; i <= DWC_EQOS_NUM_TXQ-1)" on page 675	0x60	The Tx Queue ${i} VLAN Tag Inclusion register contains the VLAN tag for insertion in the Transmit...
"MAC_VLAN_Incl" on page 677	0x60	The VLAN Tag Inclusion or Replacement register contains the VLAN tag for insertion or replacement...
"MAC_Inner_VLAN_Incl" on page 681	0x64	The Inner VLAN Tag Inclusion or Replacement register contains the inner VLAN tag to be inserted...
"MAC_Q0_Tx_Flow_Ctrl" on page 684	0x70	The Flow Control register controls the generation and reception of the Control (Pause Command) packets...
"MAC_Q(#i)_Tx_Flow_Ctrl (for i = 1; i <= DWC_EQOS_NUM_TXQ-1)" on page 688	(0x0004*i) +0x0070	This register controls the generation of PFC Control packets of priorities mapped as per the PSRQi...
"MAC_Rx_Flow_Ctrl" on page 691	0x90	The Receive Flow Control register controls the pausing of MAC Transmit based on the received Pause...

Register	Offset	Description
"MAC_RxQ_Ctrl4" on page 693	0x94	The Receive Queue Control 4 register controls the routing of unicast and multicast packets that...
"MAC_TxQ_Prty_Map0" on page 697	0x98	The Transmit Queue Priority Mapping 0 register contains the priority values assigned to Tx Queue...
"MAC_TxQ_Prty_Map1" on page 698	0x9c	The Transmit Queue Priority Mapping 1 register contains the priority values assigned to Tx Queue...
"MAC_RxQ_Ctrl0" on page 699	0xa0	The Receive Queue Control 0 register controls the queue management in the MAC Receiver. Note: In...
"MAC_RxQ_Ctrl1" on page 702	0xa4	The Receive Queue Control 1 register controls the routing of multicast, broadcast, AV, DCB, and...
"MAC_RxQ_Ctrl2" on page 707	0xa8	This register controls the routing of tagged packets based on the USP (user Priority) field of the...
"MAC_RxQ_Ctrl3" on page 709	0xac	This register controls the routing of tagged packets based on the USP (user Priority) field of the...
"MAC_Interrupt_Status" on page 711	0xb0	The Interrupt Status register contains the status of interrupts.
"MAC_Interrupt_Enable" on page 719	0xb4	The Interrupt Enable register contains the masks for generating the interrupts.
"MAC_Rx_Tx_Status" on page 723	0xb8	The Receive Transmit Status register contains the Receive and Transmit Error status.
"MAC_PMT_Control_Status" on page 727	0xc0	The PMT Control and Status Register.
"MAC_RWK_Packet_Filter" on page 731	0xc4	The Remote Wakeup Filter registers are implemented as 8, 16, or 32 indirect access registers (wkuppktfilter_reg#i)...
"RWK_Filter(#i)(#j)(#k)(#l)_Command (for i = 0; i <= (DWC_EQOS_NUM_PMT_RWK_FILT/4)-1)" on page 738	0xc4	RWK Filter Command This register controls the filter operation.
"RWK_Filter(#i)(#j)(#k)(#l)_Offset (for i = 0; i <= (DWC_EQOS_NUM_PMT_RWK_FILT/4)-1)" on page 736	0xc4	RWK Filter Offset This register defines the offset (within the packet) from which the filter examines...
"RWK_Filter(#i)(#j)_CRC (for i = 0; i <= (DWC_EQOS_NUM_PMT_RWK_FILT/2)-1)" on page 735	0xc4	RWK Filter CRC-16 This register contains the CRC-16 to be matched.
"RWK_Filter(#i)_Byte_Mask (for i = 0; i <= DWC_EQOS_NUM_PMT_RWK_FILT-1)" on page 734	0xc4	RWK Filter Byte Mask This register contains the Remote Wakeup Filter Byte Mask.
"MAC_LPI_Control_Status" on page 742	0xd0	The LPI Control and Status Register controls the LPI functions and provides the LPI interrupt status....

Register	Offset	Description
“MAC_LPI_Timers_Control” on page 747	0xd4	The LPI Timers Control register controls the timeout values in the LPI states. It specifies the...
“MAC_LPI_Entry_Timer” on page 748	0xd8	This register controls the Tx LPI entry timer. This counter is enabled only when bit[20](LPITE)...
“MAC_1US_Tic_Counter” on page 749	0xdc	This register controls the generation of the Reference time (1 microsecond tic) for all the LPI...
“MAC_AN_Control” on page 750	0xe0	The AN Control register enables and restarts auto-negotiation. It also enables PCS loopback. This...
“MAC_AN_Status” on page 753	0xe4	The AN Status register indicates the link and the auto-negotiation status.
“MAC_AN_Advertisement” on page 755	0xe8	The Auto-Negotiation Advertisement register indicates the link and the auto-negotiation...
“MAC_AN_Link_Partner_Ability” on page 757	0xec	The Auto-Negotiation Link Partner Ability register contains the advertised ability of the link...
“MAC_AN_Expansion” on page 760	0xf0	The Auto-Negotiation Expansion register indicates if the MAC received a new base page from the link...
“MAC_TBI_Extended_Status” on page 761	0xf4	The TBI Extended Status register indicates all modes of operation of the MAC.
“MAC_PHYIF_Control_Status” on page 763	0xf8	The PHY Interface Control and Status register indicates the status signals received by the SGMII,...
“MAC_Version” on page 767	0x110	The version register identifies the version of the DWC_ether_qos. This register contains two bytes:...
“MAC_Debug” on page 768	0x114	The Debug register provides the debug status of various MAC blocks.
“MAC_HW_Feature0” on page 770	0x11c	This register indicates the presence of first set of the optional features or functions of the DWC_ether_qos....
“MAC_HW_Feature1” on page 777	0x120	This register indicates the presence of second set of the optional features or functions of the...
“MAC_HW_Feature2” on page 783	0x124	This register indicates the presence of third set of the optional features or functions of the DWC_ether_qos....
“MAC_HW_Feature3” on page 787	0x128	This register indicates the presence of fourth set the optional features or functions of the DWC_ether_qos....
“MAC_DPP_FSM_Interrupt_Status” on page 792	0x140	This register contains the status of Automotive Safety related Data Path Parity Errors, Interface...
“MAC_AXI_SLV_DPE_Addr_Status” on page 800	0x144	This register indicates the CSR address corresponding to the CSR write data on which parity error...

Register	Offset	Description
"MAC_FSM_Control" on page 801	0x148	This register is used to control the FSM State parity and timeout error injection in Debug...
"MAC_FSM_ACT_Timer" on page 809	0x14c	This register is used to select the FSM and Interface Timeout values.
"SNPS_SCS_REG1" on page 811	0x150	Synopsys Reserved Register
"MAC_MDIO_Address" on page 812	0x200	The MDIO Address register controls the management cycles to external PHY through a management...
"MAC_MDIO_Data" on page 817	0x204	The MDIO Data register stores the Write data to be written to the PHY register located at the address...
"MAC_GPIO_Control" on page 818	0x208	The GPIO Control register controls the GPIO.
"MAC_GPIO_Status" on page 820	0x20c	The General Purpose IO register provides the control to drive the following: up to 16 bits of output...
"MAC_ARP_Address" on page 822	0x210	The ARP Address register contains the IPv4 Destination Address of the MAC.
"MAC_CSR_SW_Ctrl" on page 823	0x230	This register contains SW programmable controls for changing the CSR access response and status...
"MAC_FPE_CTRL_STS" on page 825	0x234	This register controls the operation of Frame Preemption.
"MAC_Ext_Cfg1" on page 828	0x238	This register contains Split mode control field and offset field for Split Header feature.
"MAC_Presn_Time_ns" on page 830	0x240	This register contains the 32-bit binary rollover equivalent time of the PTP System Time in ns ...
"MAC_Presn_Time_Updt" on page 831	0x244	This field holds the 32-bit value of MAC 1722 Presentation Time in ns, that should be added to the...
"MAC_Address0_High" on page 832	0x300	The MAC Address0 High register holds the upper 16 bits of the first 6-byte MAC address of the station....
"MAC_Address0_Low" on page 834	0x304	The MAC Address0 Low register holds the lower 32 bits of the 6-byte first MAC address of the...
"MAC_Address(#i)_High (for i = 1; i <= DWC_EQOS_ADD_MAC_ADDR_REG-1)" on page 835	(0x0008*i) +0x0300	The MAC Address1 High register holds the upper 16 bits of the second 6-byte MAC address of the station. If...
"MAC_Address(#i)_Low (for i = 1; i <= DWC_EQOS_ADD_MAC_ADDR_REG-1)" on page 838	(0x0008*i) +0x0304	The MAC Address1 Low register holds the lower 32 bits of the second 6-byte MAC address of the...
"MAC_Address(#i)_High (for i = 32; i <= 63)" on page 839	(0x0008*i) +0x0300	The MAC Address32 High register holds the upper 16 bits of the 33rd 6-byte MAC address of the station. If...
"MAC_Address(#i)_Low (for i = 32; i <= 63)" on page 841	(0x0008*i) +0x0304	The MAC Address32 Low register holds the lower 16 bits of the 33rd 6-byte MAC address of the...

Register	Offset	Description
"MAC_Address(#i)_High (for i = 64; i <= 127)" on page 842	(0x0008*i) +0x0300	The MAC Address32 High register holds the upper 16 bits of the 33rd 6-byte MAC address of the station. If...
"MAC_Address(#i)_Low (for i = 64; i <= 127)" on page 844	(0x0008*i) +0x0304	The MAC Address32 Low register holds the lower 16 bits of the 33rd 6-byte MAC address of the...
"MMC_Control" on page 845	0x700	This register establishes the operating mode of MMC.
"MMC_Rx_Interrupt" on page 848	0x704	This register maintains the interrupts generated from all Receive statistics counters. The MMC...
"MMC_Tx_Interrupt" on page 859	0x708	This register maintains the interrupts generated from all Transmit statistics counters. The MMC...
"MMC_Rx_Interrupt_Mask" on page 870	0x70c	This register maintains the masks for interrupts generated from all Receive statistics counters....
"MMC_Tx_Interrupt_Mask" on page 879	0x710	This register maintains the masks for interrupts generated from all Transmit statistics counters....
"Tx_Octet_Count_Good_Bad" on page 888	0x714	This register provides the number of bytes transmitted by the DWC_ether_qos, exclusive of preamble...
"Tx_Packet_Count_Good_Bad" on page 889	0x718	This register provides the number of good and bad packets transmitted by DWC_ether_qos, exclusive...
"Tx_Broadcast_Packets_Good" on page 890	0x71c	This register provides the number of good broadcast packets transmitted by DWC_ether_qos.
"Tx_Multicast_Packets_Good" on page 891	0x720	This register provides the number of good multicast packets transmitted by DWC_ether_qos.
"Tx_64Octets_Packets_Good_Bad" on page 892	0x724	This register provides the number of good and bad packets transmitted by DWC_ether_qos with length...
"Tx_65To127Octets_Packets_Good_Bad" on page 893	0x728	This register provides the number of good and bad packets transmitted by DWC_ether_qos with length...
"Tx_128To255Octets_Packets_Good_Bad" on page 894	0x72c	This register provides the number of good and bad packets transmitted by DWC_ether_qos with length...
"Tx_256To511Octets_Packets_Good_Bad" on page 895	0x730	This register provides the number of good and bad packets transmitted by DWC_ether_qos with length...
"Tx_512To1023Octets_Packets_Good_Bad" on page 896	0x734	This register provides the number of good and bad packets transmitted by DWC_ether_qos with length...
"Tx_1024ToMaxOctets_Packets_Good_Bad" on page 897	0x738	This register provides the number of good and bad packets transmitted by DWC_ether_qos with length...
"Tx_Unicast_Packets_Good_Bad" on page 898	0x73c	This register provides the number of good and bad unicast packets transmitted by...
"Tx_Multicast_Packets_Good_Bad" on page 899	0x740	This register provides the number of good and bad multicast packets transmitted by...

Register	Offset	Description
"Tx_Broadcast_Packets_Good_Bad" on page 900	0x744	This register provides the number of good and bad broadcast packets transmitted by...
"Tx_Underflow_Error_Packets" on page 901	0x748	This register provides the number of packets aborted by DWC_ether_qos because of packets underflow...
"Tx_Single_Collision_Good_Packets" on page 902	0x74c	This register provides the number of successfully transmitted packets by DWC_ether_qos after a single...
"Tx_Multiple_Collision_Good_Packets" on page 903	0x750	This register provides the number of successfully transmitted packets by DWC_ether_qos after multiple...
"Tx_Deferred_Packets" on page 904	0x754	This register provides the number of successfully transmitted by DWC_ether_qos after a deferral...
"Tx_Late_Collision_Packets" on page 905	0x758	This register provides the number of packets aborted by DWC_ether_qos because of late collision...
"Tx_Excessive_Collision_Packets" on page 906	0x75c	This register provides the number of packets aborted by DWC_ether_qos because of excessive (16)...
"Tx_Carrier_Error_Packets" on page 907	0x760	This register provides the number of packets aborted by DWC_ether_qos because of carrier sense error...
"Tx_Octet_Count_Good" on page 908	0x764	This register provides the number of bytes transmitted by DWC_ether_qos, exclusive of preamble,...
"Tx_Packet_Count_Good" on page 909	0x768	This register provides the number of good packets transmitted by DWC_ether_qos.
"Tx_Excessive_Deferral_Error" on page 910	0x76c	This register provides the number of packets aborted by DWC_ether_qos because of excessive deferral...
"Tx_Pause_Packets" on page 911	0x770	This register provides the number of good Pause packets transmitted by DWC_ether_qos.
"Tx_VLAN_Packets_Good" on page 912	0x774	This register provides the number of good VLAN packets transmitted by DWC_ether_qos.
"Tx_OSize_Packets_Good" on page 913	0x778	This register provides the number of packets transmitted by DWC_ether_qos without errors and with...
"Rx_Packets_Count_Good_Bad" on page 914	0x780	This register provides the number of good and bad packets received by DWC_ether_qos.
"Rx_Octet_Count_Good_Bad" on page 915	0x784	This register provides the number of bytes received by DWC_ther_qos, exclusive of preamble, in good...
"Rx_Octet_Count_Good" on page 916	0x788	This register provides the number of bytes received by DWC_ether_qos, exclusive of preamble, only...
"Rx_Broadcast_Packets_Good" on page 917	0x78c	This register provides the number of good broadcast packets received by DWC_ether_qos.

Register	Offset	Description
“Rx_Multicast_Packets_Good” on page 918	0x790	This register provides the number of good multicast packets received by DWC_ether_qos.
“Rx_CRC_Error_Packets” on page 919	0x794	This register provides the number of packets received by DWC_ether_qos with CRC error.
“Rx_Alignment_Error_Packets” on page 920	0x798	This register provides the number of packets received by DWC_ether_qos with alignment (dribble)...
“Rx_Runt_Error_Packets” on page 921	0x79c	This register provides the number of packets received by DWC_ether_qos with runt (length less than...
“Rx_Jabber_Error_Packets” on page 922	0x7a0	This register provides the number of giant packets received by DWC_ether_qos with length (including...
“Rx_Undersize_Packets_Good” on page 923	0x7a4	This register provides the number of packets received by DWC_ether_qos with length less than 64...
“Rx_Oversize_Packets_Good” on page 924	0x7a8	This register provides the number of packets received by DWC_ether_qos without errors, with length...
“Rx_64Octets_Packets_Good_Bad” on page 925	0x7ac	This register provides the number of good and bad packets received by DWC_ether_qos with length...
“Rx_65To127Octets_Packets_Good_Bad” on page 926	0x7b0	This register provides the number of good and bad packets received by DWC_ether_qos with length...
“Rx_128To255Octets_Packets_Good_Bad” on page 927	0x7b4	This register provides the number of good and bad packets received by DWC_ether_qos with length...
“Rx_256To511Octets_Packets_Good_Bad” on page 928	0x7b8	This register provides the number of good and bad packets received by DWC_ether_qos with length...
“Rx_512To1023Octets_Packets_Good_Bad” on page 929	0x7bc	This register provides the number of good and bad packets received by DWC_ether_qos with length...
“Rx_1024ToMaxOctets_Packets_Good_Bad” on page 930	0x7c0	This register provides the number of good and bad packets received by DWC_ether_qos with length...
“Rx_Unicast_Packets_Good” on page 931	0x7c4	This register provides the number of good unicast packets received by DWC_ether_qos.
“Rx_Length_Error_Packets” on page 932	0x7c8	This register provides the number of packets received by DWC_ether_qos with length error (Length...
“Rx_Out_Of_Range_Type_Packets” on page 933	0x7cc	This register provides the number of packets received by DWC_ether_qos with length field not equal...
“Rx_Pause_Packets” on page 934	0x7d0	This register provides the number of good and valid Pause packets received by DWC_ether_qos.
“Rx_FIFO_Overflow_Packets” on page 935	0x7d4	This register provides the number of missed received packets because of FIFO overflow in...

Register	Offset	Description
“Rx_VLAN_Packets_Good_Bad” on page 936	0x7d8	This register provides the number of good and bad VLAN packets received by DWC_ether_qos.
“Rx_Watchdog_Error_Packets” on page 937	0x7dc	This register provides the number of packets received by DWC_ether_qos with error because of watchdog...
“Rx_Receive_Error_Packets” on page 938	0x7e0	This register provides the number of packets received by DWC_ether_qos with Receive error or Packet...
“Rx_Control_Packets_Good” on page 939	0x7e4	This register provides the number of good control packets received by DWC_ether_qos.
“Tx_LPI_USEC_Cntr” on page 940	0x7ec	This register provides the number of microseconds Tx LPI is asserted by DWC_ether_qos.
“Tx_LPI_Tran_Cntr” on page 941	0x7f0	This register provides the number of times DWC_ether_qos has entered Tx LPI.
“Rx_LPI_USEC_Cntr” on page 942	0x7f4	This register provides the number of microseconds Rx LPI is sampled by DWC_ether_qos.
“Rx_LPI_Tran_Cntr” on page 943	0x7f8	This register provides the number of times DWC_ether_qos has entered Rx LPI.
“MMC_IPC_Rx_Interrupt_Mask” on page 944	0x800	This register maintains the mask for the interrupt generated from the receive IPC statistic counters....
“MMC_IPC_Rx_Interrupt” on page 954	0x808	This register maintains the interrupt that the receive IPC statistic counters generate. The MMC...
“RxIPv4_Good_Packets” on page 964	0x810	This register provides the number of good IPv4 datagrams received by DWC_ether_qos with the TCP,...
“RxIPv4_Header_Error_Packets” on page 965	0x814	RxIPv4 Header Error Packets This register provides the number of IPv4 datagrams received by DWC_ether_qos...
“RxIPv4_No_Payload_Packets” on page 966	0x818	This register provides the number of IPv4 datagram packets received by DWC_ether_qos that did not...
“RxIPv4_Fragmented_Packets” on page 967	0x81c	This register provides the number of good IPv4 datagrams received by DWC_ether_qos with...
“RxIPv4_UDP_Checksum_Disabled_Packets” on page 968	0x820	This register provides the number of good IPv4 datagrams received by DWC_ether_qos that had a UDP...
“RxIPv6_Good_Packets” on page 969	0x824	This register provides the number of good IPv6 datagrams received by DWC_ether_qos with the TCP,...
“RxIPv6_Header_Error_Packets” on page 970	0x828	This register provides the number of IPv6 datagrams received by DWC_ether_qos with header (length...
“RxIPv6_No_Payload_Packets” on page 971	0x82c	This register provides the number of IPv6 datagram packets received by DWC_ether_qos that did not...

Register	Offset	Description
“RxUDP_Good_Packets” on page 972	0x830	This register provides the number of good IP datagrams received by DWC_ether_qos with a good UDP...
“RxUDP_Error_Packets” on page 973	0x834	This register provides the number of good IP datagrams received by DWC_ether_qos whose UDP payload...
“RxTCP_Good_Packets” on page 974	0x838	This register provides the number of good IP datagrams received by DWC_ether_qos with a good TCP...
“RxTCP_Error_Packets” on page 975	0x83c	This register provides the number of good IP datagrams received by DWC_ether_qos whose TCP payload...
“RxICMP_Good_Packets” on page 976	0x840	This register provides the number of good IP datagrams received by DWC_ether_qos with a good ICMP...
“RxICMP_Error_Packets” on page 977	0x844	This register provides the number of good IP datagrams received by DWC_ether_qos whose ICMP payload...
“RxIPv4_Good_Octets” on page 978	0x850	This register provides the number of bytes received by DWC_ether_qos in good IPv4 datagrams encapsulating...
“RxIPv4_Header_Error_Octets” on page 979	0x854	This register provides the number of bytes received by DWC_ether_qos in IPv4 datagrams with header...
“RxIPv4_No_Payload_Octets” on page 980	0x858	This register provides the number of bytes received by DWC_ether_qos in IPv4 datagrams that did...
“RxIPv4_Fragmented_Octets” on page 981	0x85c	This register provides the number of bytes received by DWC_ether_qos in fragmented IPv4 datagrams....
“RxIPv4_UDP_Checksum_Disable_Octets” on page 982	0x860	This register provides the number of bytes received by DWC_ether_qos in a UDP segment that had the...
“RxIPv6_Good_Octets” on page 983	0x864	This register provides the number of bytes received by DWC_ether_qos in good IPv6 datagrams encapsulating...
“RxIPv6_Header_Error_Octets” on page 984	0x868	This register provides the number of bytes received by DWC_ether_qos in IPv6 datagrams with header...
“RxIPv6_No_Payload_Octets” on page 985	0x86c	This register provides the number of bytes received by DWC_ether_qos in IPv6 datagrams that did...
“RxUDP_Good_Octets” on page 986	0x870	This register provides the number of bytes received by DWC_ether_qos in a good UDP segment. This...
“RxUDP_Error_Octets” on page 987	0x874	This register provides the number of bytes received by DWC_ether_qos in a UDP segment that had checksum...
“RxTCP_Good_Octets” on page 988	0x878	This register provides the number of bytes received by DWC_ether_qos in a good TCP segment. This...
“RxTCP_Error_Octets” on page 989	0x87c	This register provides the number of bytes received by DWC_ether_qos in a TCP segment that had checksum...

Register	Offset	Description
"RxICMP_Good_Octets" on page 990	0x880	This register provides the number of bytes received by DWC_ether_qos in a good ICMP segment. This...
"RxICMP_Error_Octets" on page 991	0x884	This register provides the number of bytes received by DWC_ether_qos in a ICMP segment that had...
"MMC_FPE_Tx_Interrupt" on page 992	0x8a0	This register maintains the interrupts generated from all FPE related Transmit statistics counters....
"MMC_FPE_Tx_Interrupt_Mask" on page 994	0x8a4	This register maintains the masks for interrupts generated from all FPE related Transmit statistics...
"MMC_Tx_FPE_Fragment_Cntr" on page 996	0x8a8	This register provides the number of additional mPackets transmitted due to preemption.
"MMC_Tx_Hold_Req_Cntr" on page 997	0x8ac	This register provides the count of number of times a hold request is given to MAC
"MMC_FPE_Rx_Interrupt" on page 998	0x8c0	This register maintains the interrupts generated from all FPE related Receive statistics counters....
"MMC_FPE_Rx_Interrupt_Mask" on page 1000	0x8c4	This register maintains the masks for interrupts generated from all FPE related Receive statistics...
"MMC_Rx_Packet_Assembly_Err_Cntr" on page 1002	0x8c8	This register provides the number of MAC frames with reassembly errors on the Receiver, due to mismatch...
"MMC_Rx_Packet_SMD_Err_Cntr" on page 1003	0x8cc	This register provides the number of received MAC frames rejected due to unknown SMD value and MAC...
"MMC_Rx_Packet_Assembly_OK_Cntr" on page 1004	0x8d0	This register provides the number of MAC frames that were successfully reassembled and delivered...
"MMC_Rx_FPE_Fragment_Cntr" on page 1005	0x8d4	This register provides the number of additional mPackets received due to preemption.
"MAC_L3_L4_Control(#i) (for i = 0; i <= DWC_EQOS_L3_L4_FILTER_NUM-1)" on page 1006	(0x0030*i) +0x0900	The Layer 3 and Layer 4 Control register controls the operations of filter 0 of Layer 3 and Layer...
"MAC_Layer4_Address(#i) (for i = 0; i <= DWC_EQOS_L3_L4_FILTER_NUM-1)" on page 1012	(0x0030*i) +0x0904	The MAC_Layer4_Address(#i), MAC_L3_L4_Control(#i), MAC_Layer3_Addr0_Reg(#i), MAC_Layer3_Addr1_Reg(#i),...
"MAC_Layer3_Addr0_Reg(#i) (for i = 0; i <= DWC_EQOS_L3_L4_FILTER_NUM-1)" on page 1014	(0x0030*i) +0x0910	For IPv4 packets, the Layer 3 Address 0 Register 0 register contains the 32-bit IP Source Address...
"MAC_Layer3_Addr1_Reg(#i) (for i = 0; i <= DWC_EQOS_L3_L4_FILTER_NUM-1)" on page 1015	(0x0030*i) +0x0914	For IPv4 packets, the Layer 3 Address 1 Register 0 register contains the 32-bit IP Destination Address...

Register	Offset	Description
"MAC_Layer3_Addr2_Reg(#i) (for i = 0; i <= DWC_EQOS_L3_L4_FILTER_NUM-1)" on page 1016	(0x0030*i) +0x0918	The Layer 3 Address 2 Register 0 register is reserved for IPv4 packets. For IPv6 packets, it contains...
"MAC_Layer3_Addr3_Reg(#i) (for i = 0; i <= DWC_EQOS_L3_L4_FILTER_NUM-1)" on page 1017	(0x0030*i) +0x091C	The Layer 3 Address 3 Register 0 register is reserved for IPv4 packets. For IPv6 packets, it contains...
"MAC_Timestamp_Control" on page 1018	0xb00	This register controls the operation of the System Time generator and processing of PTP packets...
"MAC_Sub_Second_Increment" on page 1027	0xb04	This register specifies the value to be added to the internal system time register every cycle of...
"MAC_System_Time_Seconds" on page 1029	0xb08	The System Time Seconds register, along with System Time Nanoseconds register, indicates the current...
"MAC_System_Time_Nanoseconds" on page 1030	0xb0c	The System Time Nanoseconds register, along with System Time Seconds register, indicates the current...
"MAC_System_Time_Seconds_Update" on page 1031	0xb10	The System Time Seconds Update register, along with the System Time Nanoseconds Update register,...
"MAC_System_Time_Nanoseconds_Update" on page 1032	0xb14	MAC System Time Nanoseconds Update register.
"MAC_Timestamp_Addend" on page 1034	0xb18	Timestamp Addend register. This register value is used only when the system time is configured for...
"MAC_System_Time_Higher_Word_Seconds" on page 1035	0xb1c	System Time - Higher Word Seconds register.
"MAC_Timestamp_Status" on page 1036	0xb20	Timestamp Status register. All bits except Bits[27:25] gets cleared when the application reads this...
"MAC_Tx_Timestamp_Status_Nanoseconds" on page 1042	0xb30	This register contains the nanosecond part of timestamp captured for Transmit packets when Tx status...
"MAC_Tx_Timestamp_Status_Seconds" on page 1044	0xb34	The register contains the higher 32 bits of the timestamp (in seconds) captured when a PTP packet...
"MAC_Auxiliary_Control" on page 1045	0xb40	The Auxiliary Timestamp Control register controls the Auxiliary Timestamp snapshot.
"MAC_Auxiliary_Timestamp_Nanoseconds" on page 1047	0xb48	The Auxiliary Timestamp Nanoseconds register, along with MAC_Auxiliary_Timestamp_Seconds, gives...
"MAC_Auxiliary_Timestamp_Seconds" on page 1048	0xb4c	The Auxiliary Timestamp - Seconds register contains the lower 32 bits of the Seconds field of the...
"MAC_Timestamp_Ingress_Asym_Corr" on page 1049	0xb50	The MAC Timestamp Ingress Asymmetry Correction register contains the Ingress Asymmetry Correction...
"MAC_Timestamp_Egress_Asym_Corr" on page 1050	0xb54	The MAC Timestamp Egress Asymmetry Correction register contains the Egress Asymmetry Correction...

Register	Offset	Description
"MAC_Timestamp_Ingress_Corr_Nanosecond" on page 1051	0xb58	This register contains the correction value in nanoseconds to be used with the captured timestamp...
"MAC_Timestamp_Egress_Corr_Nanosecond" on page 1052	0xb5c	This register contains the correction value in nanoseconds to be used with the captured timestamp...
"MAC_Timestamp_Ingress_Corr_Subnanosec" on page 1053	0xb60	This register contains the sub-nanosecond part of the correction value to be used with the captured...
"MAC_Timestamp_Egress_Corr_Subnanosec" on page 1054	0xb64	This register contains the sub-nanosecond part of the correction value to be used with the captured...
"MAC_Timestamp_Ingress_Latency" on page 1055	0xb68	This register holds the Ingress MAC latency.
"MAC_Timestamp_Egress_Latency" on page 1056	0xb6c	This register holds the Egress MAC latency.
"MAC_PPS_Control" on page 1057	0xb70	PPS Control register. Bits[30:24] of this register are valid only when four Flexible PPS outputs...
"MAC_PPS(#i)_Target_Time_Seconds (for i = 0; i <= DWC_EQOS_PPS_OUT_NUM-1)" on page 1066	(0x0010*i) +0x0B80	The PPS Target Time Seconds register, along with PPS Target Time Nanoseconds register, is used to...
"MAC_PPS(#i)_Target_Time_Nanoseconds (for i = 0; i <= DWC_EQOS_PPS_OUT_NUM-1)" on page 1067	(0x0010*i) +0x0B84	PPS0 Target Time Nanoseconds register.
"MAC_PPS(#i)_Interval (for i = 0; i <= DWC_EQOS_PPS_OUT_NUM-1)" on page 1069	(0x0010*i) +0x0B88	The PPS0 Interval register contains the number of units of sub-second increment value between the...
"MAC_PPS(#i)_Width (for i = 0; i <= DWC_EQOS_PPS_OUT_NUM-1)" on page 1070	(0x0010*i) +0x0B8C	The PPS0 Width register contains the number of units of sub-second increment value between the rising...
"MAC_PTO_Control" on page 1071	0xbc0	This register controls the PTP Offload Engine operation. This register is available only when the...
"MAC_Source_Port_Identity0" on page 1075	0xbc4	This register contains Bits[31:0] of the 80-bit Source Port Identity of the PTP node. This register...
"MAC_Source_Port_Identity1" on page 1076	0xbc8	This register contains Bits[63:32] of the 80-bit Source Port Identity of the PTP node. This register...
"MAC_Source_Port_Identity2" on page 1077	0xbcc	This register contains Bits[79:64] of the 80-bit Source Port Identity of the PTP node. This register...
"MAC_Log_Message_Interval" on page 1078	0xbd0	This register contains the periodic intervals for automatic PTP packet generation. This register...

Register	Offset	Description
The address block EQOS_MTL contains up to 7 Registers: MTL_Operation_Mode to MTL_RxQ_DMA_Map1 EQOS_MTL		
"MTL_Operation_Mode" on page 1080	0xc00	The Operation Mode register establishes the Transmit and Receive operating modes and commands.
"MTL_DBG_CTL" on page 1083	0xc08	The FIFO Debug Access Control and Status register controls the operation mode of FIFO debug...
"MTL_DBG_STS" on page 1088	0xc0c	The FIFO Debug Status register contains the status of FIFO debug access.
"MTL_FIFO_Debug_Data" on page 1091	0xc10	The FIFO Debug Data register contains the data to be written to or read from the FIFOs.
"MTL_Interrupt_Status" on page 1092	0xc20	The software driver (application) reads this register during interrupt service routine or polling...
"MTL_RxQ_DMA_Map0" on page 1096	0xc30	The Receive Queue and DMA Channel Mapping 0 register is reserved in EQOS-CORE and EQOS-MTL...
"MTL_RxQ_DMA_Map1" on page 1101	0xc34	The Receive Queue and DMA Channel Mapping 1 register is reserved in EQOS-CORE and EQOS-MTL...
"MTL_TBS_CTRL" on page 1106	0xc40	This register controls the operation of Time Based Scheduling.
"MTL_EST_Control" on page 1108	0xc50	This register controls the operation of Enhancements to Scheduled Transmission (IEEE802.1Qbv).
"MTL_EST_Status" on page 1111	0xc58	This register provides Status related to Enhancements to Scheduled Transmission...
"MTL_EST_Sch_Error" on page 1115	0xc60	This register provides the One Hot encoded Queue Numbers that are having the Scheduling related...
"MTL_EST_Frm_Size_Error" on page 1116	0xc64	This register provides the One Hot encoded Queue Numbers that are having the Frame Size related...
"MTL_EST_Frm_Size_Capture" on page 1117	0xc68	This register captures the Frame Size and Queue Number of the first occurrence of the Frame Size...
"MTL_EST_Intr_Enable" on page 1119	0xc70	This register implements the Interrupt Enable bits for the various events that generate an interrupt....
"MTL_EST_GCL_Control" on page 1121	0xc80	This register provides the control information for reading/writing to the Gate Control lists.
"MTL_EST_GCL_Data" on page 1126	0xc84	This register holds the read data or write data in case of reads and writes respectively.
"MTL_FPE_CTRL_STS" on page 1127	0xc90	This register controls the operation of, and provides status for Frame Preemption...
"MTL_FPE_Advance" on page 1129	0xc94	This register holds the Hold and Release Advance time.

Register	Offset	Description
"MTL_RXP_Control_Status" on page 1130	0xca0	The MTL_RXP_Control_Status register establishes the operating mode of Rx Parser and provides some...
"MTL_RXP_Interrupt_Control_Status" on page 1132	0xca4	The MTL_RXP_Interrupt_Control_Status registers provides enable control for the interrupts and provides...
"MTL_RXP_Drop_Cnt" on page 1135	0xca8	The MTL_RXP_Drop_Cnt register provides the drop count of Rx Parser initiated drops.
"MTL_RXP_Error_Cnt" on page 1136	0xcac	The MTL_RXP_Error_Cnt register provides the Rx Parser related error occurrence count.
"MTL_RXP_Indirect_Acc_Control_Status" on page 1137	0xcb0	The MTL_RXP_Indirect_Acc_Control_Status register provides the Indirect Access control and status...
"MTL_RXP_Indirect_Acc_Data" on page 1140	0xcb4	The MTL_RXP_Indirect_Acc_Data registers holds the data associated to Indirect Access to Rx Parser...
"MTL_ECC_Control" on page 1141	0xcc0	The MTL_ECC_Control register establishes the operating mode of ECC related to MTL memories.
"MTL_Safety_Interrupt_Status" on page 1144	0xcc4	The MTL_Safety_Interrupt_Status registers provides Safety interrupt status.
"MTL_ECC_Interrupt_Enable" on page 1146	0xcc8	The MTL_ECC_Interrupt_Enable register provides enable bits for the ECC interrupts.
"MTL_ECC_Interrupt_Status" on page 1148	0xccc	The MTL_ECC_Interrupt_Status register provides MTL ECC Interrupt Status.
"MTL_ECC_Err_Sts_Rctl" on page 1153	0xcd0	The MTL_ECC_Err_Sts_Rctl register establishes the control for ECC Error status capture.
"MTL_ECC_Err_Addr_Status" on page 1156	0xcd4	The MTL_ECC_Err_Addr_Status register provides the memory addresses for the correctable and uncorrectable...
"MTL_ECC_Err_Cntr_Status" on page 1157	0xcd8	The MTL_ECC_Err_Cntr_Status register provides ECC Error count for Correctable and uncorrectable...
"MTL_DPP_Control" on page 1158	0xce0	The MTL_DPP_Control establishes the operating mode of Data Parity protection and error...
The address block EQOS_MTL_Q${i} contains up to 10 registers: MTL_TxQ0_Operation_Mode to MTL_RxQ0_Control EQOS_MTL_Q0		
"MTL_TxQ0_Operation_Mode" on page 1164	0xd00	The Queue 0 Transmit Operation Mode register establishes the Transmit queue operating modes and...
"MTL_TxQ0_Underflow" on page 1167	0xd04	The Queue 0 Underflow Counter register contains the counter for packets aborted because of Transmit...
"MTL_TxQ0_Debug" on page 1169	0xd08	The Queue 0 Transmit Debug register gives the debug status of various blocks related to the Transmit...

Register	Offset	Description
"MTL_TxQ0_ETS_Status" on page 1172	0xd14	The Queue 0 ETS Status register provides the average traffic transmitted in Queue 0.
"MTL_TxQ0_Quantum_Weight" on page 1173	0xd18	The Queue 0 Quantum or Weights register contains the quantum value for Deficit Weighted Round Robin...
"MTL_Q0_Interrupt_Control_Status" on page 1174	0xd2c	This register contains the interrupt enable and status bits for the queue 0 interrupts.
"MTL_RxQ0_Operation_Mode" on page 1177	0xd30	The Queue 0 Receive Operation Mode register establishes the Receive queue operating modes and command. The...
"MTL_RxQ0_Missed_Packet_Overflow_Cnt" on page 1182	0xd34	The Queue 0 Missed Packet and Overflow Counter register contains the counter for packets missed...
"MTL_RxQ0_Debug" on page 1184	0xd38	The Queue 0 Receive Debug register gives the debug status of various blocks related to the Receive...
"MTL_RxQ0_Control" on page 1186	0xd3c	The Queue Receive Control register controls the receive arbitration and passing of received packets...
The address block EQOS_MTL_Q${i} contains up to 14 registers: MTL_TxQ${i}_Operation_Mode to MTL_RxQ${i}_Control EQOS_MTL_Q1		
"MTL_TxQ(#i)_Operation_Mode (for i = 1; i <= DWC_EQOS_NUM_TXQ-1)" on page 1188	(0x0040*i) +0x0D00	The Queue 1 Transmit Operation Mode register establishes the Transmit queue operating modes and...
"MTL_TxQ(#i)_Underflow (for i = 1; i <= DWC_EQOS_NUM_TXQ-1)" on page 1191	(0x0040*i) +0x0D04	The Queue 1 Underflow Counter register contains the counter for packets aborted because of Transmit...
"MTL_TxQ(#i)_Debug (for i = 1; i <= DWC_EQOS_NUM_TXQ-1)" on page 1193	(0x0040*i) +0x0D08	The Queue 1 Transmit Debug register gives the debug status of various blocks related to the Transmit...
"MTL_TxQ(#i)_ETS_Control (for i = 1; i <= DWC_EQOS_NUM_TXQ-1)" on page 1196	(0x0040*i) +0x0D10	The Queue ETS Control register controls the enhanced transmission selection operation.
"MTL_TxQ(#i)_ETS_Status (for i = 1; i <= DWC_EQOS_NUM_TXQ-1)" on page 1198	(0x0040*i) +0x0D14	The Queue 1 ETS Status register provides the average traffic transmitted in Queue 1.
"MTL_TxQ(#i)_Quantum_Weight (for i = 1; i <= DWC_EQOS_NUM_TXQ-1)" on page 1199	(0x0040*i) +0x0D18	The Queue 1 idleSlopeCredit, Quantum or Weights register provides the average traffic transmitted...
"MTL_TxQ(#i)_SendSlopeCredit (for i = 1; i <= DWC_EQOS_NUM_TXQ-1)" on page 1201	(0x0040*i) +0x0D1C	The sendSlopeCredit register contains the sendSlope credit value required for the credit-based shaper...
"MTL_TxQ(#i)_HiCredit (for i = 1; i <= DWC_EQOS_NUM_TXQ-1)" on page 1202	(0x0040*i) +0x0D20	The hiCredit register contains the hiCredit value required for the credit-based shaper algorithm...
"MTL_TxQ(#i)_LoCredit (for i = 1; i <= DWC_EQOS_NUM_TXQ-1)" on page 1203	(0x0040*i) +0x0D24	The loCredit register contains the loCredit value required for the credit-based shaper algorithm...

Register	Offset	Description
"MTL_Q(#i)_Interrupt_Control_Status) (for i = 1; i <= max(DWC_EQOS_NUM_TXQ-1,DWC_EQOS_NUM_RXQ-1))" on page 1204	(0x0040*i) +0x0D2C	This register contains the interrupt enable and status bits for the queue 1 interrupts.
"MTL_RxQ(#i)_Operation_Mode (for i = 1; i <= DWC_EQOS_NUM_RXQ-1)" on page 1207	(0x0040*i) +0x0D30	The Queue 1 Receive Operation Mode register establishes the Receive queue operating modes and command. The...
"MTL_RxQ(#i)_Missed_Packet_Overflow_Cnt (for i = 1; i <= DWC_EQOS_NUM_RXQ-1)" on page 1212	(0x0040*i) +0x0D34	The Queue 1 Missed Packet and Overflow Counter register contains the counter for packets missed...
"MTL_RxQ(#i)_Debug (for i = 1; i <= DWC_EQOS_NUM_RXQ-1)" on page 1214	(0x0040*i) +0x0D38	The Queue 1 Receive Debug register gives the debug status of various blocks related to the Receive...
"MTL_RxQ(#i)_Control (for i = 1; i <= DWC_EQOS_NUM_RXQ-1)" on page 1216	(0x0040*i) +0x0D3C	The Queue Receive Control register controls the receive arbitration and passing of received packets...
The address block EQOS_DMA contains up to 9 Registers: DMA_Mode to AXI4_TxRx_AWAR_ACE_Control EQOS_DMA		
"DMA_Mode" on page 1218	0x1000	The Bus Mode register establishes the bus operating modes for the DMA.
"DMA_SysBus_Mode" on page 1223	0x1004	The System Bus mode register controls the behavior of the AHB or AXI master. It mainly controls...
"DMA_Interrupt_Status" on page 1229	0x1008	The application reads this Interrupt Status register during interrupt service routine or polling...
"DMA_Debug_Status0" on page 1233	0x100c	The Debug Status 0 register gives the Receive and Transmit process status for DMA Channel 0-Channel...
"DMA_Debug_Status1" on page 1238	0x1010	The Debug Status1 register gives the Receive and Transmit process status for DMA Channel 3-Channel...
"DMA_Debug_Status2" on page 1243	0x1014	The Debug Status Register 2 gives the Receive and Transmit process status for DMA Channel 7.
"AXI4_Tx_AR_ACE_Control" on page 1245	0x1020	This register is used to control the AXI4 Cache Coherency Signals for read transactions by all the...
"AXI4_Rx_AW_ACE_Control" on page 1248	0x1024	This register is used to control the AXI4 Cache Coherency Signals for write transactions by all...
"AXI4_TxRx_AWAR_ACE_Control" on page 1250	0x1028	This register is used to control the AXI4 Cache Coherency Signals for Descriptor write transactions...
"AXI_LPI_Entry_Interval" on page 1252	0x1040	This register is used to control the AXI LPI entry interval.
"DMA_TBS_CTRL" on page 1253	0x1050	This register is used to control the TBS attributes.
"DMA_Safety_Interrupt_Status" on page 1255	0x1080	This register indicates summary (whether error occured in DMA/MTL/MAC and correctable/uncorrectable)...

Register	Offset	Description
"DMA_ECC_Interrupt_enable" on page 1258	0x1084	This register is used to enable the Automotive Safety related TSO memory ECC error interrupt.
"DMA_ECC_Interrupt_Status" on page 1259	0x1088	This register indicates the Automotive Safety related TSO memory ECC error interrupt status.
The address block EQOS_DMA_CH${i} contains up to 22 registers: DMA_CH${i}_Control to DMA_CH${i}_Miss_Frame_Cnt EQOS_DMA_CH0		
"DMA_CH(#i)_Control (for i = 0; i <= DWC_EQOS_NUM_DMA_CSR_CH-1)" on page 1261	(0x0080*i) +0x1100	The DMA Channeli Control register specifies the MSS value for segmentation, length to skip between...
"DMA_CH(#i)_Tx_Control (for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1)" on page 1263	(0x0080*i) +0x1104	The DMA Channeli Transmit Control register controls the Tx features such as PBL, TCP segmentation,...
"DMA_CH(#i)_Rx_Control (for i = 0; i <= DWC_EQOS_NUM_DMA_RX_CH-1)" on page 1268	(0x0080*i) +0x1108	The DMA Channeli Receive Control register controls the Rx features such as PBL, buffer size, and...
"DMA_CH(#i)_TxDesc_List_HAddress (for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1)" on page 1272	(0x0080*i) +0x1110	The Channeli Tx Descriptor List HAddress register has the higher 8 or 16 bits of the start address...
"DMA_CH(#i)_TxDesc_List_Address (for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1)" on page 1273	(0x0080*i) +0x1114	The Channeli Tx Descriptor List Address register points the DMA to the start of Transmit descriptor...
"DMA_CH(#i)_RxDesc_List_HAddress (for i = 0; i <= DWC_EQOS_NUM_DMA_RX_CH-1)" on page 1274	(0x0080*i) +0x1118	The Channeli Rx Descriptor List HAddress register has the higher 8 or 16 bits of the start address...
"DMA_CH(#i)_RxDesc_List_Address (for i = 0; i <= DWC_EQOS_NUM_DMA_RX_CH-1)" on page 1275	(0x0080*i) +0x111C	The Channeli Rx Descriptor List Address register points the DMA to the start of Receive descriptor...
"DMA_CH(#i)_TxDesc_Tail_Pointer (for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1)" on page 1276	(0x0080*i) +0x1120	The Channeli Tx Descriptor Tail Pointer register points to an offset from the base and indicates...
"DMA_CH(#i)_RxDesc_Tail_Pointer (for i = 0; i <= DWC_EQOS_NUM_DMA_RX_CH-1)" on page 1277	(0x0080*i) +0x1128	The Channeli Rx Descriptor Tail Pointer Points to an offset from the base and indicates the location...
"DMA_CH(#i)_TxDesc_Ring_Length (for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1)" on page 1278	(0x0080*i) +0x112C	The Tx Descriptor Ring Length register contains the length of the Transmit descriptor ring.
"DMA_CH(#i)_RxDesc_Ring_Length (for i = 0; i <= DWC_EQOS_NUM_DMA_RX_CH-1)" on page 1279	(0x0080*i) +0x1130	The Channeli Rx Descriptor Ring Length register contains the length of the Receive descriptor circular...

Register	Offset	Description
"DMA_CH(#i)_Interrupt_Enable (for i = 0; i <= DWC_EQOS_NUM_DMA_CSR_CH-1)" on page 1280	(0x0080*i) +0x1134	The Channeli Interrupt Enable register enables the interrupts reported by the Status register.
"DMA_CH(#i)_Rx_Interrupt_Watchdog_Timer (for i = 0; i <= DWC_EQOS_NUM_DMA_RX_CH-1)" on page 1285	(0x0080*i) +0x1138	The Receive Interrupt Watchdog Timer register indicates the watchdog timeout for Receive Interrupt...
"DMA_CH(#i)_Slot_Function_Control_Status (for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1)" on page 1287	(0x0080*i) +0x113C	The Slot Function Control and Status register contains the control bits for slot function and the...
"DMA_CH(#i)_Current_App_TxDesc (for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1)" on page 1289	(0x0080*i) +0x1144	The Channeli Current Application Transmit Descriptor register points to the current Transmit descriptor...
"DMA_CH(#i)_Current_App_RxDesc (for i = 0; i <= DWC_EQOS_NUM_DMA_RX_CH-1)" on page 1290	(0x0080*i) +0x114C	The Channeli Current Application Receive Descriptor register points to the current Receive descriptor...
"DMA_CH(#i)_Current_App_TxBuffer_H (for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1)" on page 1291	(0x0080*i) +0x1150	The Channeli Current Application Transmit Buffer Address High register has the higher 8 or 16 bits...
"DMA_CH(#i)_Current_App_TxBuffer (for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1)" on page 1292	(0x0080*i) +0x1154	The Channeli Current Application Transmit Buffer Address register points to the current Tx buffer...
"DMA_CH(#i)_Current_App_RxBuffer_H (for i = 0; i <= DWC_EQOS_NUM_DMA_RX_CH-1)" on page 1293	(0x0080*i) +0x1158	The Channeli Current Application Receive Buffer Address High register has the higher 8 or 16 bits...
"DMA_CH(#i)_Current_App_RxBuffer (for i = 0; i <= DWC_EQOS_NUM_DMA_RX_CH-1)" on page 1294	(0x0080*i) +0x115C	The Channel 0 Current Application Receive Buffer Address register points to the current Rx buffer...
"DMA_CH(#i)_Status (for i = 0; i <= DWC_EQOS_NUM_DMA_CSR_CH-1)" on page 1295	(0x0080*i) +0x1160	The software driver (application) reads the Status register during interrupt service routine or...
"DMA_CH(#i)_Miss_Frame_Cnt (for i = 0; i <= DWC_EQOS_NUM_DMA_RX_CH-1)" on page 1303	(0x0080*i) +0x1164	This register has the number of packet counter that got dropped by the DMA either due to Bus Error...
"DMA_CH(#i)_RXP_Accept_Cnt (for i = 0; i <= DWC_EQOS_NUM_DMA_RX_CH-1)" on page 1305	(0x0080*i) +0x1168	The DMA_CH(#i)_RXP_Accept_Cnt registers provides the count of the number of frames accepted by Rx...
"DMA_CH(#i)_RX_ERI_Cnt (for i = 0; i <= DWC_EQOS_NUM_DMA_RX_CH-1)" on page 1306	(0x0080*i) +0x1168	The DMA_CH(#i)_RX_ERI_Cnt registers provides the count of the number of times ERI was...

17.1 EQOS_MAC Registers

17.1.1 MAC_Configuration

- **Description:** The MAC Configuration Register establishes the operating mode of the MAC.
- **Size:** 32 bits
- **Offset:** 0x0
- **Exists:** Always

31	30:28	27	26:24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6:5	4	3:2	1	0
ARPEN	SARC	IPC	IPG	GPSLCE	S2KP	CST	ACS	WD	BE	JD	JE	PS	FES	DM	LM	ECRSFD	DO	DCRS	DR	Reserved_7	BL	DC	PRELEN	TE	RE

Table 17-5 Fields for Register: MAC_Configuration

Bits	Name	Memory Access	Description
31	ARPEN	R/W	ARP Offload Enable When this bit is set, the MAC can recognize an incoming ARP request packet and schedules the ARP packet for transmission. It forwards the ARP packet to the application and also indicate the events in the RxStatus. When this bit is reset, the MAC receiver does not recognize any ARP packet and indicates them as Type frame in the RxStatus. This bit is available only when the Enable IPv4 ARP Offload is selected. **Values:** ■ 0x0 (DISABLE): ARP Offload is disabled ■ 0x1 (ENABLE): ARP Offload is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_ARP_EN

Bits	Name	Memory Access	Description
30:28	SARC	R/W	Source Address Insertion or Replacement Control This field controls the source address insertion or replacement for all transmitted packets. Bit 30 specifies which MAC Address register (0 or 1) is used for source address insertion or replacement based on the values of Bits[29:28]: **2'b0x**: ■ The mti_sa_ctrl_i and ati_sa_ctrl_i input signals control the SA field generation. **2'b10:** ■ If Bit 30 is set to 0, the MAC inserts the content of the MAC Address 0 registers in the SA field of all transmitted packets. ■ If Bit 30 is set to 1 and the Enable MAC Address Register 1 option is selected while configuring the core, the MAC inserts the content of the MAC Address 1 registers in the SA field of all transmitted packets. **2'b11**: ■ If Bit 30 is set to 0, the MAC replaces the content of the MAC Address 0 registers in the SA field of all transmitted packets. ■ If Bit 30 is set to 1 and the MAC Address Register 1 is enabled, the MAC replaces the content of the MAC Address 1 registers in the SA field of all transmitted packets. **Note**: ■ Changes to this field take effect only on the start of a packet. If you write to this register field when a packet is being transmitted, only the subsequent packet can use the updated value, that is, the current packet does not use the updated value. **Values:** ■ 0x0 (SA_CTRL_IN): mti_sa_ctrl_i and ati_sa_ctrl_i input signals control the SA field generation ■ 0x2 (MAC0_INS_SA): Contents of MAC Addr-0 inserted in SA field ■ 0x3 (MAC0_REP_SA): Contents of MAC Addr-0 replaces SA field ■ 0x6 (MAC1_INS_SA): Contents of MAC Addr-1 inserted in SA field ■ 0x7 (MAC1_REP_SA): Contents of MAC Addr-1 replaces SA field **Value After Reset:** 0x0 **Exists:** DWC_EQOS_SA_VLAN_INS_CTRL_EN

Bits	Name	Memory Access	Description
27	IPC	R/W	Checksum Offload When set, this bit enables the IPv4 header checksum checking and IPv4 or IPv6 TCP, UDP, or ICMP payload checksum checking. When this bit is reset, the COE function in the receiver is disabled. The Layer 3 and Layer 4 Packet Filter and Enable Split Header features automatically selects the IPC Full Checksum Offload Engine on the Receive side. When any of these features are enabled, you must set the IPC bit. **Values:** ■ 0x0 (DISABLE): IP header/payload checksum checking is disabled ■ 0x1 (ENABLE): IP header/payload checksum checking is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_RX_COE
26:24	IPG	R/W	Inter-Packet Gap These bits control the minimum IPG between packets during transmission. This range of minimum IPG is valid in full-duplex mode. In the half-duplex mode, the minimum IPG can be configured only for 64-bit times (IPG = 100). Lower values are not considered. When a JAM pattern is being transmitted because of backpressure activation, the MAC does not consider the minimum IPG. The above function (IPG less than 96 bit times) is valid only when EIPGEN bit in MAC_Ext_Configuration register is reset. When EIPGEN is set, then the minimum IPG (greater than 96 bit times) is controlled as per the description given in EIPG field in MAC_Ext_Configuration register. **Values:** ■ 0x0 (IPG96): 96 bit times IPG ■ 0x1 (IPG88): 88 bit times IPG ■ 0x2 (IPG80): 80 bit times IPG ■ 0x3 (IPG72): 72 bit times IPG ■ 0x4 (IPG64): 64 bit times IPG ■ 0x5 (IPG56): 56 bit times IPG ■ 0x6 (IPG48): 48 bit times IPG ■ 0x7 (IPG40): 40 bit times IPG **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
23	GPSLCE	R/W	Giant Packet Size Limit Control Enable When this bit is set, the MAC considers the value in GPSL field in MAC_Ext_Configuration register to declare a received packet as Giant packet. This field must be programmed to more than 1,518 bytes. Otherwise, the MAC considers 1,518 bytes as giant packet limit. When this bit is reset, the MAC considers a received packet as Giant packet when its size is greater than 1,518 bytes (1522 bytes for tagged packet). The watchdog timeout limit, Jumbo Packet Enable and 2K Packet Enable have higher precedence over this bit, that is the MAC considers a received packet as Giant packet when its size is greater than 9,018 bytes (9,022 bytes for tagged packet) with Jumbo Packet Enabled and greater than 2,000 bytes with 2K Packet Enabled. The watchdog timeout, if enabled, terminates the received packet when watchdog limit is reached. Therefore, the programmed giant packet limit should be less than the watchdog limit to get the giant packet status. **Values:** ■ 0x0 (DISABLE): Giant Packet Size Limit Control is disabled ■ 0x1 (ENABLE): Giant Packet Size Limit Control is enabled **Value After Reset:** 0x0 **Exists:** Always
22	S2KP	R/W	IEEE 802.3as Support for 2K Packets When this bit is set, the MAC considers all packets with up to 2,000 bytes length as normal packets. When the JE bit is not set, the MAC considers all received packets of size more than 2K bytes as Giant packets. When this bit is reset and the JE bit is not set, the MAC considers all received packets of size more than 1,518 bytes (1,522 bytes for tagged) as giant packets. For more information about how the setting of this bit and the JE bit impact the Giant packet status, see the Table, Gaint Packet Status based on S2KP and JE Bits. **Note**: When the JE bit is set, setting this bit has no effect on the giant packet status. **Values:** ■ 0x0 (DISABLE): Support upto 2K packet is disabled ■ 0x1 (ENABLE): Support upto 2K packet is Enabled **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
21	CST	R/W	CRC stripping for Type packets When this bit is set, the last four bytes (FCS) of all packets of Ether type (type field greater than 1,536) are stripped and dropped before forwarding the packet to the application. **Note**: For information about how the settings of the ACS bit and this bit impact the packet length, see the Table, Packet Length based on the CST and ACS Bits. **Values:** ■ 0x0 (DISABLE): CRC stripping for Type packets is disabled ■ 0x1 (ENABLE): CRC stripping for Type packets is enabled **Value After Reset:** 0x0 **Exists:** Always
20	ACS	R/W	Automatic Pad or CRC Stripping When this bit is set, the MAC strips the Pad or FCS field on the incoming packets only if the value of the length field is less than 1,536 bytes. All received packets with length field greater than or equal to 1,536 bytes are passed to the application without stripping the Pad or FCS field. When this bit is reset, the MAC passes all incoming packets to the application, without any modification. **Note**: For information about how the settings of CST bit and this bit impact the packet length, see the Table, Packet Length based on the CST and ACS Bit . **Values:** ■ 0x0 (DISABLE): Automatic Pad or CRC Stripping is disabled ■ 0x1 (ENABLE): Automatic Pad or CRC Stripping is enabled **Value After Reset:** 0x0 **Exists:** Always
19	WD	R/W	Watchdog Disable When this bit is set, the MAC disables the watchdog timer on the receiver. The MAC can receive packets of up to 16,383 bytes. When this bit is reset, the MAC does not allow more than 2,048 bytes (10,240 if JE is set high) of the packet being received. The MAC cuts off any bytes received after 2,048 bytes. **Values:** ■ 0x0 (ENABLE): Watchdog is enabled ■ 0x1 (DISABLE): Watchdog is disabled **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
18	BE	R/W	Packet Burst Enable When this bit is set, the MAC allows packet bursting during transmission in the GMII half-duplex mode. **Values:** ■ 0x0 (DISABLE): Packet Burst is disabled ■ 0x1 (ENABLE): Packet Burst is enabled **Value After Reset:** 0x0 **Exists:** !(DWC_EQOS_M110_ONLY || DWC_EQOS_FDUPLX_ONLY)
17	JD	R/W	Jabber Disable When this bit is set, the MAC disables the jabber timer on the transmitter. The MAC can transfer packets of up to 16,383 bytes. When this bit is reset, if the application sends more than 2,048 bytes of data (10,240 if JE is set high) during transmission, the MAC does not send rest of the bytes in that packet. **Values:** ■ 0x0 (ENABLE): Jabber is enabled ■ 0x1 (DISABLE): Jabber is disabled **Value After Reset:** 0x0 **Exists:** Always
16	JE	R/W	Jumbo Packet Enable When this bit is set, the MAC allows jumbo packets of 9,018 bytes (9,022 bytes for VLAN tagged packets) without reporting a giant packet error in the Rx packet status. **Values:** ■ 0x0 (DISABLE): Jumbo packet is disabled ■ 0x1 (ENABLE): Jumbo packet is enabled **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
15	PS	* Varies	Port Select This bit selects the Ethernet line speed. This bit, along with Bit 14, selects the exact line speed. In the 10/100 Mbps-only (always 1) or 1000 Mbps-only (always 0) configurations, this bit is read-only (RO) with appropriate value. In default 10/100/1000 Mbps configurations, this bit is read-write (R/W). The mac_speed_o[1] signal reflects the value of this bit. **Values:** ■ 0x0 (1000_2500M): For 1000 or 2500 Mbps operations ■ 0x1 (10_100M): For 10 or 100 Mbps operations **Value After Reset:** (DWC_EQOS_M110_ONLY?\"0x1\":\"0x0\") **Exists:** Always **Memory Access:** ((!DWC_EQOS_M110_ONLY&&!DWC_EQOS_M1000_ONLY)?\"read-write\":\"read-only\")
14	FES	R/W	Speed This bit selects the speed mode. The mac_speed_o[0] signal reflects the value of this bit. **Values:** ■ 0x0 (10_1000M): 10 Mbps when PS bit is 1 and 1 Gbps when PS bit is 0 ■ 0x1 (100_2500M): 100 Mbps when PS bit is 1 and 2.5 Gbps when PS bit is 0 **Value After Reset:** 0x0 **Exists:** Always
13	DM	* Varies	Duplex Mode When this bit is set, the MAC operates in the full-duplex mode in which it can transmit and receive simultaneously. This bit is RO with default value of 1'b1 in the full-duplex-only configurations. **Values:** ■ 0x0 (HDUPLX): Half-duplex mode ■ 0x1 (FDUPLX): Full-duplex mode **Value After Reset:** ((DWC_EQOS_FDUPLX_ONLY)?\"1\":\"0\") **Exists:** Always **Memory Access:** ((DWC_EQOS_FDUPLX_ONLY)?\"read-only\":\"read-write\")

Bits	Name	Memory Access	Description
12	LM	R/W	Loopback Mode When this bit is set, the MAC operates in the loopback mode at GMII or MII. The (G)MII Rx clock input (clk_rx_i) is required for the loopback to work properly. This is because the Tx clock is not internally looped back. **Values:** ■ 0x0 (DISABLE): Loopback is disabled ■ 0x1 (ENABLE): Loopback is enabled **Value After Reset:** 0x0 **Exists:** Always
11	ECRSFD	R/W	Enable Carrier Sense Before Transmission in Full-Duplex Mode When this bit is set, the MAC transmitter checks the CRS signal before packet transmission in the full-duplex mode. The MAC starts the transmission only when the CRS signal is low. When this bit is reset, the MAC transmitter ignores the status of the CRS signal. **Values:** ■ 0x0 (DISABLE): ECRSFD is disabled ■ 0x1 (ENABLE): ECRSFD is enabled **Value After Reset:** 0x0 **Exists:** Always
10	DO	R/W	Disable Receive Own When this bit is set, the MAC disables the reception of packets when the gmii_txen_o is asserted in the half-duplex mode. When this bit is reset, the MAC receives all packets given by the PHY. This bit is not applicable in the full-duplex mode. **Values:** ■ 0x0 (ENABLE): Enable Receive Own ■ 0x1 (DISABLE): Disable Receive Own **Value After Reset:** 0x0 **Exists:** !(DWC_EQOS_FDUPLX_ONLY)

Bits	Name	Memory Access	Description
9	DCRS	R/W	Disable Carrier Sense During Transmission When this bit is set, the MAC transmitter ignores the (G)MII CRS signal during packet transmission in the half-duplex mode. As a result, no errors are generated because of Loss of Carrier or No Carrier during transmission. When this bit is reset, the MAC transmitter generates errors because of Carrier Sense. The MAC can even abort the transmission. **Values:** ■ 0x0 (ENABLE): Enable Carrier Sense During Transmission ■ 0x1 (DISABLE): Disable Carrier Sense During Transmission **Value After Reset:** 0x0 **Exists:** !(DWC_EQOS_FDUPLX_ONLY)
8	DR	R/W	Disable Retry When this bit is set, the MAC attempts only one transmission. When a collision occurs on the GMII or MII interface, the MAC ignores the current packet transmission and reports a Packet Abort with excessive collision error in the Tx packet status. When this bit is reset, the MAC retries based on the settings of the BL field. This bit is applicable only in the half-duplex mode. **Values:** ■ 0x0 (ENABLE): Enable Retry ■ 0x1 (DISABLE): Disable Retry **Value After Reset:** 0x0 **Exists:** !(DWC_EQOS_FDUPLX_ONLY)
7	Reserved_7	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
6:5	BL	R/W	Back-Off Limit The back-off limit determines the random integer number (r) of slot time delays (4,096 bit times for 1000/2500 Mbps; 512 bit times for 10/100 Mbps) for which the MAC waits before rescheduling a transmission attempt during retries after a collision. n = retransmission attempt. The random integer r takes the value in the range 0 <= r < 2^k This bit is applicable only in the half-duplex mode. **Values:** ■ 0x0 (MIN_N_10): k = min(n,10) ■ 0x1 (MIN_N_8): k = min(n,8) ■ 0x2 (MIN_N_4): k = min(n,4) ■ 0x3 (MIN_N_1): k = min(n,1) **Value After Reset:** 0x0 **Exists:** !(DWC_EQOS_FDUPLX_ONLY)
4	DC	R/W	Deferral Check When this bit is set, the deferral check function is enabled in the MAC. The MAC issues a Packet Abort status, along with the excessive deferral error bit set in the Tx packet status, when the Tx state machine is deferred for more than 24,288 bit times in 10 or 100 Mbps mode. If the MAC is configured for 1000/2500 Mbps operation, the threshold for deferral is 155,680 bits times. Deferral begins when the transmitter is ready to transmit, but it is prevented because of an active carrier sense signal (CRS) on GMII or MII. The defer time is not cumulative. For example, if the transmitter defers for 10,000 bit times because the CRS signal is active and the CRS signal becomes inactive, the transmitter transmits and collision happens. Because of collision, the transmitter needs to back off and then defer again after back off completion. In such a scenario, the deferral timer is reset to 0, and it is restarted. When this bit is reset, the deferral check function is disabled and the MAC defers until the CRS signal goes inactive. This bit is applicable only in the half-duplex mode. **Values:** ■ 0x0 (DISABLE): Deferral check function is disabled ■ 0x1 (ENABLE): Deferral check function is enabled **Value After Reset:** 0x0 **Exists:** !(DWC_EQOS_FDUPLX_ONLY)

Bits	Name	Memory Access	Description
3:2	PRELEN	R/W	Preamble Length for Transmit packets These bits control the number of preamble bytes that are added to the beginning of every Tx packet. The preamble reduction occurs only when the MAC is operating in the full-duplex mode. **Values:** ■ 0x0 (7BYTES): 7 bytes of preamble ■ 0x1 (5BYTES): 5 bytes of preamble ■ 0x2 (3BYTES): 3 bytes of preamble ■ 0x3 (RESERVED): Reserved **Value After Reset:** 0x0 **Exists:** Always
1	TE	R/W	Transmitter Enable When this bit is set, the Tx state machine of the MAC is enabled for transmission on the GMII or MII interface. When this bit is reset, the MAC Tx state machine is disabled after it completes the transmission of the current packet. The Tx state machine does not transmit any more packets. **Values:** ■ 0x0 (DISABLE): Transmitter is disabled ■ 0x1 (ENABLE): Transmitter is enabled **Value After Reset:** 0x0 **Exists:** Always
0	RE	R/W	Receiver Enable When this bit is set, the Rx state machine of the MAC is enabled for receiving packets from the GMII or MII interface. When this bit is reset, the MAC Rx state machine is disabled after it completes the reception of the current packet. The Rx state machine does not receive any more packets from the GMII or MII interface. **Values:** ■ 0x0 (DISABLE): Receiver is disabled ■ 0x1 (ENABLE): Receiver is enabled **Value After Reset:** 0x0 **Exists:** Always

17.1.2 MAC_Ext_Configuration

- **Description:** The MAC Extended Configuration Register establishes the operating mode of the MAC.
- **Size:** 32 bits
- **Offset:** 0x4
- **Exists:** Always

31	30	29:25	24	23	22:20	19	18	17	16	15:14	13:0
FHE	Reserved_30	EIPG	EIPGEN	Reserved_23	HDSMS	PDC	USP	SPEN	DCRCC	Reserved_15_14	GPSL

Table 17-6 Fields for Register: MAC_Ext_Configuration

Bits	Name	Memory Access	Description
31	FHE	R/W	Flexible Header Enable When this is set then it is expected that all the incoming packets from the Line to have 4B Flexible Header in the Rx path (except for the pause and PFC packets). Similarly, in the Tx path all the packets from application are expected to have 4B Flexible header. The position of the Flexible Header is always fixed at 12Bytes from the beginning of the packet (after MAC DA/SA). **Values:** ■ 0x0 (DISABLE): Flexible Header is disabled ■ 0x1 (ENABLE): Flexible Header is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_FHE
30	Reserved_30	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
29:25	EIPG	R/W	Extended Inter-Packet Gap The value in this field is applicable when the EIPGEN bit is set. This field (as Most Significant bits), along with IPG field in MAC_Configuration register, gives the minimum IPG greater than 96 bit times in steps of 8 bit times: {EIPG, IPG} 8'h00 - 104 bit times 8'h01 - 112 bit times 8'h02 - 120 bit times ----------------------- 8'hFF - 2144 bit times **Value After Reset:** 0x0 **Exists:** Always
24	EIPGEN	R/W	Extended Inter-Packet Gap Enable When this bit is set, the MAC interprets EIPG field and IPG field in MAC_Configuration register together as minimum IPG greater than 96 bit times in steps of 8 bit times. When this bit is reset, the MAC ignores EIPG field and interprets IPG field in MAC_Configuration register as minimum IPG less than or equal to 96 bit times in steps of 8 bit times. Note: The extended Inter-Packet Gap feature must be enabled when operating in Full-Duplex mode only. There may be undesirable effects on back-pressure function and frame transmission if it is enabled in Half-Duplex mode. **Values:** ■ 0x0 (DISABLE): Extended Inter-Packet Gap is disabled ■ 0x1 (ENABLE): Extended Inter-Packet Gap is enabled **Value After Reset:** 0x0 **Exists:** Always
23	Reserved_23	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
22:20	HDSMS	R/W	Maximum Size for Splitting the Header Data These bits indicate the maximum header size allowed for splitting the header data in the received packet. **Values:** ■ 0x0 (64BYTES): Maximum Size for Splitting the Header Data is 64 bytes ■ 0x1 (128BYTES): Maximum Size for Splitting the Header Data is 128 bytes ■ 0x2 (256BYTES): Maximum Size for Splitting the Header Data is 256 bytes ■ 0x3 (512BYTES): Maximum Size for Splitting the Header Data is 512 bytes ■ 0x4 (1024BYTES): Maximum Size for Splitting the Header Data is 1024 bytes ■ 0x5 (RSVD): Reserved **Value After Reset:** 0x0 **Exists:** DWC_EQOS_SPLIT_HDR_EN
19	PDC	R/W	Packet Duplication Control When this bit is set, the received packet with Multicast/Broadcast Destination address is routed to multiple Receive DMA Channels. The Receive DMA Channels is identified by the DCS field of MAC_Address(#i)_High register corresponding to the MAC Address register that matches the Multicast/Broadcast Destination address in the received packet. The DCS field is interpreted to be a one-hot value, each bit corresponding to the Receive DMA Channel. When this bit is reset, the received packet is routed to single Receive DMA Channel. The Receive DMA Channel is identified by the DCS field of MAC_Address(#i)_High register corresponding to the MAC Address register that matches the Destination address in the received packet. The DCS field is interpreted as a binary value. **Values:** ■ 0x0 (DISABLE): Packet Duplication Control is disabled ■ 0x1 (ENABLE): Packet Duplication Control is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_PDUP

Bits	Name	Memory Access	Description
18	USP	R/W	Unicast Slow Protocol Packet Detect When this bit is set, the MAC detects the Slow Protocol packets with unicast address of the station specified in the MAC_Address0_High and MAC_Address0_Low registers. The MAC also detects the Slow Protocol packets with the Slow Protocols multicast address (01-80-C2-00-00-02). When this bit is reset, the MAC detects only Slow Protocol packets with the Slow Protocol multicast address specified in the IEEE 802.3-2015, Section 5. **Values:** ■ 0x0 (DISABLE): Unicast Slow Protocol Packet Detection is disabled ■ 0x1 (ENABLE): Unicast Slow Protocol Packet Detection is enabled **Value After Reset:** 0x0 **Exists:** Always
17	SPEN	R/W	Slow Protocol Detection Enable When this bit is set, MAC processes the Slow Protocol packets (Ether Type 0x8809) and provides the Rx status. The MAC discards the Slow Protocol packets with invalid sub-types. When this bit is reset, the MAC forwards all error-free Slow Protocol packets to the application. The MAC considers such packets as normal Type packets. **Values:** ■ 0x0 (DISABLE): Slow Protocol Detection is disabled ■ 0x1 (ENABLE): Slow Protocol Detection is enabled **Value After Reset:** 0x0 **Exists:** Always
16	DCRCC	R/W	Disable CRC Checking for Received Packets When this bit is set, the MAC receiver does not check the CRC field in the received packets. When this bit is reset, the MAC receiver always checks the CRC field in the received packets. **Values:** ■ 0x0 (ENABLE): CRC Checking is enabled ■ 0x1 (DISABLE): CRC Checking is disabled **Value After Reset:** 0x0 **Exists:** Always
15:14	Reserved_15_14	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
13:0	GPSL	R/W	Giant Packet Size Limit If the received packet size is greater than the value programmed in this field in units of bytes, the MAC declares the received packet as Giant packet. The value programmed in this field must be greater than or equal to 1,518 bytes. Any other programmed value is considered as 1,518 bytes. For VLAN tagged packets, the MAC adds 4 bytes to the programmed value. When the Enable Double VLAN Processing option is selected, the MAC adds 8 bytes to the programmed value for double VLAN tagged packets. The value in this field is applicable when the GPSLCE bit is set in MAC_Configuration register. **Value After Reset:** 0x0 **Exists:** Always

17.1.3 MAC_Packet_Filter

- **Description:** The MAC Packet Filter register contains the filter controls for receiving packets. Some of the controls from this register go to the address check block of the MAC which performs the first level of address filtering. The second level of filtering is performed on the incoming packet based on other controls such as Pass Bad Packets and Pass Control Packets.
- **Size:** 32 bits
- **Offset:** 0x8
- **Exists:** Always

31	30:22	21	20	19:17	16	15:11	10	9	8	7:6	5	4	3	2	1	0
RA	Reserved_30_22	DNTU	IPFE	Reserved_19_17	VTFE	Reserved_15_11	HPF	SAF	SAIF	PCF	DBF	PM	DAIF	HMC	HUC	PR

Table 17-7 Fields for Register: MAC_Packet_Filter

Bits	Name	Memory Access	Description
31	RA	R/W	Receive All When this bit is set, the MAC Receiver module passes all received packets to the application, irrespective of whether they pass the address filter or not. The result of the SA or DA filtering is updated (pass or fail) in the corresponding bit in the Rx Status Word. When this bit is reset, the Receiver module passes only those packets to the application that pass the SA or DA address filter. **Values:** ■ 0x0 (DISABLE): Receive All is disabled ■ 0x1 (ENABLE): Receive All is enabled **Value After Reset:** 0x0 **Exists:** Always
30:22	Reserved_30_22	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
21	DNTU	R/W	Drop Non-TCP/UDP over IP Packets When this bit is set, the MAC drops the non-TCP or UDP over IP packets. The MAC forward only those packets that are processed by the Layer 4 filter. When this bit is reset, the MAC forwards all non-TCP or UDP over IP packets. **Values:** ■ 0x0 (FWD): Forward Non-TCP/UDP over IP Packets ■ 0x1 (DROP): Drop Non-TCP/UDP over IP Packets **Value After Reset:** 0x0 **Exists:** DWC_EQOS_L3_L4_FILTER_EN
20	IPFE	R/W	Layer 3 and Layer 4 Filter Enable When this bit is set, the MAC drops packets that do not match the enabled Layer 3 and Layer 4 filters. If Layer 3 or Layer 4 filters are not enabled for matching, this bit does not have any effect. When this bit is reset, the MAC forwards all packets irrespective of the match status of the Layer 3 and Layer 4 fields. **Values:** ■ 0x0 (DISABLE): Layer 3 and Layer 4 Filters are disabled ■ 0x1 (ENABLE): Layer 3 and Layer 4 Filters are enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_L3_L4_FILTER_EN
19:17	Reserved_19_17	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
16	VTFE	R/W	VLAN Tag Filter Enable When this bit is set, the MAC drops the VLAN tagged packets that do not match the VLAN Tag. When this bit is reset, the MAC forwards all packets irrespective of the match status of the VLAN Tag. **Values:** ■ 0x0 (DISABLE): VLAN Tag Filter is disabled ■ 0x1 (ENABLE): VLAN Tag Filter is enabled **Value After Reset:** 0x0 **Exists:** Always
15:11	Reserved_15_11	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
10	HPF	R/W	Hash or Perfect Filter When this bit is set, the address filter passes a packet if it matches either the perfect filtering or hash filtering as set by the HMC or HUC bit. When this bit is reset and the HUC or HMC bit is set, the packet is passed only if it matches the Hash filter. **Values:** ■ 0x0 (DISABLE): Hash or Perfect Filter is disabled ■ 0x1 (ENABLE): Hash or Perfect Filter is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_HASH_EN
9	SAF	R/W	Source Address Filter Enable When this bit is set, the MAC compares the SA field of the received packets with the values programmed in the enabled SA registers. If the comparison fails, the MAC drops the packet. When this bit is reset, the MAC forwards the received packet to the application with updated SAF bit of the Rx Status depending on the SA address comparison. **Note**: According to the IEEE specification, Bit 47 of the SA is reserved. However, in DWC_ether_qos, the MAC compares all 48 bits. The software driver should take this into consideration while programming the MAC address registers for SA. **Values:** ■ 0x0 (DISABLE): SA Filtering is disabled ■ 0x1 (ENABLE): SA Filtering is enabled **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_ADD_MAC_ADDR_REG>0)
8	SAIF	R/W	SA Inverse Filtering When this bit is set, the Address Check block operates in the inverse filtering mode for SA address comparison. If the SA of a packet matches the values programmed in the SA registers, it is marked as failing the SA Address filter. When this bit is reset, if the SA of a packet does not match the values programmed in the SA registers, it is marked as failing the SA Address filter. **Values:** ■ 0x0 (DISABLE): SA Inverse Filtering is disabled ■ 0x1 (ENABLE): SA Inverse Filtering is enabled **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_ADD_MAC_ADDR_REG>0)

Bits	Name	Memory Access	Description
7:6	PCF	R/W	Pass Control Packets These bits control the forwarding of all control packets (including unicast and multicast Pause packets). **Values:** ■ 0x0 (FLTR_ALL): MAC filters all control packets from reaching the application ■ 0x1 (FW_XCPT_PAU): MAC forwards all control packets except Pause packets to the application even if they fail the Address filter ■ 0x2 (FW_ALL): MAC forwards all control packets to the application even if they fail the Address filter ■ 0x3 (FW_PASS): MAC forwards the control packets that pass the Address filter **Value After Reset:** 0x0 **Exists:** Always
5	DBF	R/W	Disable Broadcast Packets When this bit is set, the AFM module blocks all incoming broadcast packets. In addition, it overrides all other filter settings. When this bit is reset, the AFM module passes all received broadcast packets. **Values:** ■ 0x0 (ENABLE): Enable Broadcast Packets ■ 0x1 (DISABLE): Disable Broadcast Packets **Value After Reset:** 0x0 **Exists:** Always
4	PM	R/W	Pass All Multicast When this bit is set, it indicates that all received packets with a multicast destination address (first bit in the destination address field is '1') are passed. When this bit is reset, filtering of multicast packet depends on HMC bit. **Values:** ■ 0x0 (DISABLE): Pass All Multicast is disabled ■ 0x1 (ENABLE): Pass All Multicast is enabled **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
3	DAIF	R/W	DA Inverse Filtering When this bit is set, the Address Check block operates in inverse filtering mode for the DA address comparison for both unicast and multicast packets. When this bit is reset, normal filtering of packets is performed. **Values:** ■ 0x0 (DISABLE): DA Inverse Filtering is disabled ■ 0x1 (ENABLE): DA Inverse Filtering is enabled **Value After Reset:** 0x0 **Exists:** Always
2	HMC	R/W	Hash Multicast When this bit is set, the MAC performs the destination address filtering of received multicast packets according to the hash table. When this bit is reset, the MAC performs the perfect destination address filtering for multicast packets, that is, it compares the DA field with the values programmed in DA registers. **Values:** ■ 0x0 (DISABLE): Hash Multicast is disabled ■ 0x1 (ENABLE): Hash Multicast is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_HASH_EN
1	HUC	R/W	Hash Unicast When this bit is set, the MAC performs the destination address filtering of unicast packets according to the hash table. When this bit is reset, the MAC performs a perfect destination address filtering for unicast packets, that is, it compares the DA field with the values programmed in DA registers. **Values:** ■ 0x0 (DISABLE): Hash Unicast is disabled ■ 0x1 (ENABLE): Hash Unicast is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_HASH_EN
0	PR	R/W	Promiscuous Mode When this bit is set, the Address Filtering module passes all incoming packets irrespective of the destination or source address. The SA or DA Filter Fails status bits of the Rx Status Word are always cleared when PR is set. **Values:** ■ 0x0 (DISABLE): Promiscuous Mode is disabled ■ 0x1 (ENABLE): Promiscuous Mode is enabled **Value After Reset:** 0x0 **Exists:** Always

17.1.4 MAC_Watchdog_Timeout

- **Description:** The Watchdog Timeout register controls the watchdog timeout for received packets.
- **Size:** 32 bits
- **Offset:** 0xc
- **Exists:** Always

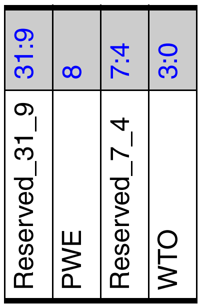


Table 17-8 Fields for Register: MAC_Watchdog_Timeout

Bits	Name	Memory Access	Description
31:9	Reserved_31_9	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
8	PWE	R/W	Programmable Watchdog Enable When this bit is set and the WD bit of the MAC_Configuration register is reset, the WTO field is used as watchdog timeout for a received packet. When this bit is cleared, the watchdog timeout for a received packet is controlled by setting of WD and JE bits in MAC_Configuration register. **Values:** ■ 0x0 (DISABLE): Programmable Watchdog is disabled ■ 0x1 (ENABLE): Programmable Watchdog is enabled **Value After Reset:** 0x0 **Exists:** Always
7:4	Reserved_7_4	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
3:0	WTO	R/W	Watchdog Timeout When the PWE bit is set and the WD bit of the MAC_Configuration register is reset, this field is used as watchdog timeout for a received packet. If the length of a received packet exceeds the value of this field, such packet is terminated and declared as an error packet. **Note**: When the PWE bit is set, the value in this field should be more than 1,522 (0x05F2). Otherwise, the IEEE 802.3-specified valid tagged packets are declared as error packets and then dropped. **Values:** ■ 0x0 (2KBYTES): 2 KB ■ 0x1 (3KBYTES): 3 KB ■ 0x2 (4KBYTES): 4 KB ■ 0x3 (5KBYTES): 5 KB ■ 0x4 (6KBYTES): 6 KB ■ 0x5 (7KBYTES): 7 KB ■ 0x6 (8KBYTES): 8 KB ■ 0x7 (9KBYTES): 9 KB ■ 0x8 (10KBYTES): 10 KB ■ 0x9 (11KBYTES): 11 KB ■ 0xa (12KBYTES): 12 KB ■ 0xb (13KBYTES): 13 KB ■ 0xc (14KBYTES): 14 KB ■ 0xd (15KBYTES): 15 KB ■ 0xe (16383BYTES): 16383 Bytes ■ 0xf (RESERVED): Reserved **Value After Reset:** 0x0 **Exists:** Always

17.1.5 MAC_Hash_Table_Reg0

- **Description:** The Hash Table Register 0 contains the first 32 bits of the hash table, when the width of the hash table is 128 or 256 bits. You can specify the width of the hash table by using the Hash Table Size option in coreConsultant.
  The Hash table is used for group address filtering. For hash filtering, the content of the destination address in the incoming packet is passed through the CRC logic and the upper six (seven in 128-bit Hash or eight in 256-bit Hash) bits of the CRC register are used to index the content of the Hash table. The most significant bits determines the register to be used (Hash Table Register X), and the least significant five bits determine the bit within the register. For example, a hash value of 6'b100000 (in 64-bit Hash) selects Bit 0 of the Hash Table Register 1, a value of 7b'1110000 (in 128-bit Hash) selects Bit 16 of the Hash Table Register 3 and a value of 8b'10111111 (in 256-bit Hash) selects Bit 31 of the Hash Table Register 5.
  The hash value of the destination address is calculated in the following way:
  - Calculate the 32-bit CRC for the DA (See IEEE 802.3, Section 3.2.8 for the steps to calculate CRC32).
  - Perform bitwise reversal for the value obtained in Step 1.
  - Take the upper 6 (or 7 or 8) bits from the value obtained in Step 2.

  If the corresponding bit value of the register is 1'b1, the packet is accepted. Otherwise, it is rejected. If the PM bit is set in MAC_Packet_Filter, all multicast packets are accepted regardless of the multicast hash values.
  If the Hash Table register is configured to be double-synchronized to the (G)MII clock domain, the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the Hash Table Register X registers are written.
  If double-synchronization is enabled, consecutive writes to this register should be performed after at least four clock cycles in the destination clock domain.
- **Size:** 32 bits
- **Offset:** 0x10
- **Exists:** (DWC_EQOS_HASH_EN)

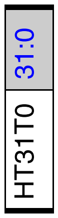


Table 17-9 Fields for Register: MAC_Hash_Table_Reg0

Bits	Name	Memory Access	Description
31:0	HT31T0	R/W	MAC Hash Table First 32 Bits This field contains the first 32 Bits [31:0] of the Hash table. **Value After Reset:** 0x0 **Exists:** Always

17.1.6 MAC_Hash_Table_Reg1

- **Description:** The Hash Table Register 1 contains the second 32 bits of the hash table. You can specify the width of the hash table by using the Hash Table Size option in coreConsultant.
  The Hash table is used for group address filtering. For hash filtering, the content of the destination address in the incoming packet is passed through the CRC logic and the upper six (seven in 128-bit Hash or eight in 256-bit Hash) bits of the CRC register are used to index the content of the Hash table. The most significant bits determines the register to be used (Hash Table Register X), and the least significant five bits determine the bit within the register. For example, a hash value of 6'b100000 (in 64-bit Hash) selects Bit 0 of the Hash Table Register 1, a value of 7b'1110000 (in 128-bit Hash) selects Bit 16 of the Hash Table Register 3 and a value of 8b'10111111 (in 256-bit Hash) selects Bit 31 of the Hash Table Register 5.
  The hash value of the destination address is calculated in the following way:
  - Calculate the 32-bit CRC for the DA (See IEEE 802.3, Section 3.2.8 for the steps to calculate CRC32).
  - Perform bitwise reversal for the value obtained in Step 1.
  - Take the upper 6 (or 7 or 8) bits from the value obtained in Step 2.

  If the corresponding bit value of the register is 1'b1, the packet is accepted. Otherwise, it is rejected. If the PM bit is set in MAC_Packet_Filter, all multicast packets are accepted regardless of the multicast hash values.
  If the Hash Table register is configured to be double-synchronized to the (G)MII clock domain, the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the Hash Table Register X registers are written.
  If double-synchronization is enabled, consecutive writes to this register should be performed after at least four clock cycles in the destination clock domain.
- **Size:** 32 bits
- **Offset:** 0x14
- **Exists:** (DWC_EQOS_HASH_TABLE > 32)

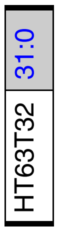


Table 17-10 Fields for Register: MAC_Hash_Table_Reg1

Bits	Name	Memory Access	Description
31:0	HT63T32	R/W	MAC Hash Table Second 32 Bits This field contains the second 32 Bits [63:32] of the Hash table. **Value After Reset:** 0x0 **Exists:** Always

17.1.7 MAC_Hash_Table_Reg2

- **Description:** The Hash Table Register 2 contains the third 32 bits of the hash table. You can specify the width of the hash table by using the Hash Table Size option in coreConsultant.
  The Hash table is used for group address filtering. For hash filtering, the content of the destination address in the incoming packet is passed through the CRC logic and the upper six (seven in 128-bit Hash or eight in 256-bit Hash) bits of the CRC register are used to index the content of the Hash table. The most significant bits determines the register to be used (Hash Table Register X), and the least significant five bits determine the bit within the register. For example, a hash value of 6'b100000 (in 64-bit Hash) selects Bit 0 of the Hash Table Register 1, a value of 7b'1110000 (in 128-bit Hash) selects Bit 16 of the Hash Table Register 3 and a value of 8b'10111111 (in 256-bit Hash) selects Bit 31 of the Hash Table Register 5.
  The hash value of the destination address is calculated in the following way:
  - Calculate the 32-bit CRC for the DA (See IEEE 802.3, Section 3.2.8 for the steps to calculate CRC32).
  - Perform bitwise reversal for the value obtained in Step 1.
  - Take the upper 6 (or 7 or 8) bits from the value obtained in Step 2.

  If the corresponding bit value of the register is 1'b1, the packet is accepted. Otherwise, it is rejected. If the PM bit is set in MAC_Packet_Filter, all multicast packets are accepted regardless of the multicast hash values.
  If the Hash Table register is configured to be double-synchronized to the (G)MII clock domain, the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the Hash Table Register X registers are written.
  If double-synchronization is enabled, consecutive writes to this register should be performed after at least four clock cycles in the destination clock domain.
- **Size:** 32 bits
- **Offset:** 0x18
- **Exists:** (DWC_EQOS_HASH_TABLE > 64)

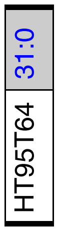


Table 17-11 Fields for Register: MAC_Hash_Table_Reg2

Bits	Name	Memory Access	Description
31:0	HT95T64	R/W	MAC Hash Table Third 32 Bits This field contains the third 32 Bits [95:64] of the Hash table. **Value After Reset:** 0x0 **Exists:** Always

17.1.8 MAC_Hash_Table_Reg3

- **Description:** The Hash Table Register 3 contains the fourth 32 bits of the hash table. You can specify the width of the hash table by using the Hash Table Size option in coreConsultant.
The Hash table is used for group address filtering. For hash filtering, the content of the destination address in the incoming packet is passed through the CRC logic and the upper six (seven in 128-bit Hash or eight in 256-bit Hash) bits of the CRC register are used to index the content of the Hash table. The most significant bits determines the register to be used (Hash Table Register X), and the least significant five bits determine the bit within the register. For example, a hash value of 6'b100000 (in 64-bit Hash) selects Bit 0 of the Hash Table Register 1, a value of 7b'1110000 (in 128-bit Hash) selects Bit 16 of the Hash Table Register 3 and a value of 8b'10111111 (in 256-bit Hash) selects Bit 31 of the Hash Table Register 5.
The hash value of the destination address is calculated in the following way:
    - Calculate the 32-bit CRC for the DA (See IEEE 802.3, Section 3.2.8 for the steps to calculate CRC32).
    - Perform bitwise reversal for the value obtained in Step 1.
    - Take the upper 6 (or 7 or 8) bits from the value obtained in Step 2.

  If the corresponding bit value of the register is 1'b1, the packet is accepted. Otherwise, it is rejected. If the PM bit is set in MAC_Packet_Filter, all multicast packets are accepted regardless of the multicast hash values.
If the Hash Table register is configured to be double-synchronized to the (G)MII clock domain, the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the Hash Table Register X registers are written.
If double-synchronization is enabled, consecutive writes to this register should be performed after at least four clock cycles in the destination clock domain.
- **Size:** 32 bits
- **Offset:** 0x1c
- **Exists:** (DWC_EQOS_HASH_TABLE > 64)

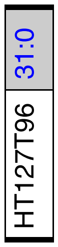


Table 17-12 Fields for Register: MAC_Hash_Table_Reg3

Bits	Name	Memory Access	Description
31:0	HT127T96	R/W	MAC Hash Table Fourth 32 Bits This field contains the fourth 32 Bits [127:96] of the Hash table. **Value After Reset:** 0x0 **Exists:** Always

17.1.9 MAC_Hash_Table_Reg4

- **Description:** The Hash Table Register 4 contains the fifth 32 bits of the hash table. You can specify the width of the hash table by using the Hash Table Size option in coreConsultant.
  The Hash table is used for group address filtering. For hash filtering, the content of the destination address in the incoming packet is passed through the CRC logic and the upper six (seven in 128-bit Hash or eight in 256-bit Hash) bits of the CRC register are used to index the content of the Hash table. The most significant bits determines the register to be used (Hash Table Register X), and the least significant five bits determine the bit within the register. For example, a hash value of 6'b100000 (in 64-bit Hash) selects Bit 0 of the Hash Table Register 1, a value of 7b'1110000 (in 128-bit Hash) selects Bit 16 of the Hash Table Register 3 and a value of 8b'10111111 (in 256-bit Hash) selects Bit 31 of the Hash Table Register 5.
  The hash value of the destination address is calculated in the following way:
  - Calculate the 32-bit CRC for the DA (See IEEE 802.3, Section 3.2.8 for the steps to calculate CRC32).
  - Perform bitwise reversal for the value obtained in Step 1.
  - Take the upper 6 (or 7 or 8) bits from the value obtained in Step 2.

  If the corresponding bit value of the register is 1'b1, the packet is accepted. Otherwise, it is rejected. If the PM bit is set in MAC_Packet_Filter, all multicast packets are accepted regardless of the multicast hash values.
  If the Hash Table register is configured to be double-synchronized to the (G)MII clock domain, the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the Hash Table Register X registers are written.
  If double-synchronization is enabled, consecutive writes to this register should be performed after at least four clock cycles in the destination clock domain.
- **Size:** 32 bits
- **Offset:** 0x20
- **Exists:** (DWC_EQOS_HASH_TABLE > 128)

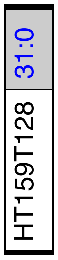


Table 17-13 Fields for Register: MAC_Hash_Table_Reg4

Bits	Name	Memory Access	Description
31:0	HT159T128	R/W	MAC Hash Table Fifth 32 Bits This field contains the fifth 32 Bits [159:128] of the Hash table. **Value After Reset:** 0x0 **Exists:** Always

17.1.10 MAC_Hash_Table_Reg5

- **Description:** The Hash Table Register 5 contains the sixth 32 bits of the hash table. You can specify the width of the hash table by using the Hash Table Size option in coreConsultant.
  The Hash table is used for group address filtering. For hash filtering, the content of the destination address in the incoming packet is passed through the CRC logic and the upper six (seven in 128-bit Hash or eight in 256-bit Hash) bits of the CRC register are used to index the content of the Hash table. The most significant bits determines the register to be used (Hash Table Register X), and the least significant five bits determine the bit within the register. For example, a hash value of 6'b100000 (in 64-bit Hash) selects Bit 0 of the Hash Table Register 1, a value of 7b'1110000 (in 128-bit Hash) selects Bit 16 of the Hash Table Register 3 and a value of 8b'10111111 (in 256-bit Hash) selects Bit 31 of the Hash Table Register 5.
  The hash value of the destination address is calculated in the following way:
  - Calculate the 32-bit CRC for the DA (See IEEE 802.3, Section 3.2.8 for the steps to calculate CRC32).
  - Perform bitwise reversal for the value obtained in Step 1.
  - Take the upper 6 (or 7 or 8) bits from the value obtained in Step 2.

  If the corresponding bit value of the register is 1'b1, the packet is accepted. Otherwise, it is rejected. If the PM bit is set in MAC_Packet_Filter, all multicast packets are accepted regardless of the multicast hash values.
  If the Hash Table register is configured to be double-synchronized to the (G)MII clock domain, the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the Hash Table Register X registers are written.
  If double-synchronization is enabled, consecutive writes to this register should be performed after at least four clock cycles in the destination clock domain.
- **Size:** 32 bits
- **Offset:** 0x24
- **Exists:** (DWC_EQOS_HASH_TABLE > 128)

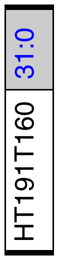


Table 17-14 Fields for Register: MAC_Hash_Table_Reg5

Bits	Name	Memory Access	Description
31:0	HT191T160	R/W	MAC Hash Table Sixth 32 Bits This field contains the sixth 32 Bits [191:160] of the Hash table. **Value After Reset:** 0x0 **Exists:** Always

17.1.11 MAC_Hash_Table_Reg6

- **Description:** The Hash Table Register 6 contains the seventh 32 bits of the hash table. You can specify the width of the hash table by using the Hash Table Size option in coreConsultant.
  The Hash table is used for group address filtering. For hash filtering, the content of the destination address in the incoming packet is passed through the CRC logic and the upper six (seven in 128-bit Hash or eight in 256-bit Hash) bits of the CRC register are used to index the content of the Hash table. The most significant bits determines the register to be used (Hash Table Register X), and the least significant five bits determine the bit within the register. For example, a hash value of 6'b100000 (in 64-bit Hash) selects Bit 0 of the Hash Table Register 1, a value of 7b'1110000 (in 128-bit Hash) selects Bit 16 of the Hash Table Register 3 and a value of 8b'10111111 (in 256-bit Hash) selects Bit 31 of the Hash Table Register 5.
  The hash value of the destination address is calculated in the following way:
  - Calculate the 32-bit CRC for the DA (See IEEE 802.3, Section 3.2.8 for the steps to calculate CRC32).
  - Perform bitwise reversal for the value obtained in Step 1.
  - Take the upper 6 (or 7 or 8) bits from the value obtained in Step 2.

  If the corresponding bit value of the register is 1'b1, the packet is accepted. Otherwise, it is rejected. If the PM bit is set in MAC_Packet_Filter, all multicast packets are accepted regardless of the multicast hash values.
  If the Hash Table register is configured to be double-synchronized to the (G)MII clock domain, the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the Hash Table Register X registers are written.
  If double-synchronization is enabled, consecutive writes to this register should be performed after at least four clock cycles in the destination clock domain.
- **Size:** 32 bits
- **Offset:** 0x28
- **Exists:** (DWC_EQOS_HASH_TABLE > 128)

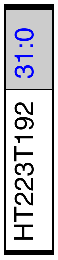


Table 17-15 Fields for Register: MAC_Hash_Table_Reg6

Bits	Name	Memory Access	Description
31:0	HT223T192	R/W	MAC Hash Table Seventh 32 Bits This field contains the seventh 32 Bits [223:192] of the Hash table. **Value After Reset:** 0x0 **Exists:** Always

17.1.12 MAC_Hash_Table_Reg7

- **Description:** The Hash Table Register 7 contains the eighth 32 bits of the hash table. You can specify the width of the hash table by using the Hash Table Size option in coreConsultant.
  The Hash table is used for group address filtering. For hash filtering, the content of the destination address in the incoming packet is passed through the CRC logic and the upper six (seven in 128-bit Hash or eight in 256-bit Hash) bits of the CRC register are used to index the content of the Hash table. The most significant bits determines the register to be used (Hash Table Register X), and the least significant five bits determine the bit within the register. For example, a hash value of 6'b100000 (in 64-bit Hash) selects Bit 0 of the Hash Table Register 1, a value of 7b'1110000 (in 128-bit Hash) selects Bit 16 of the Hash Table Register 3 and a value of 8b'10111111 (in 256-bit Hash) selects Bit 31 of the Hash Table Register 5.
  The hash value of the destination address is calculated in the following way:
  - Calculate the 32-bit CRC for the DA (See IEEE 802.3, Section 3.2.8 for the steps to calculate CRC32).
  - Perform bitwise reversal for the value obtained in Step 1.
  - Take the upper 6 (or 7 or 8) bits from the value obtained in Step 2.

  If the corresponding bit value of the register is 1'b1, the packet is accepted. Otherwise, it is rejected. If the PM bit is set in MAC_Packet_Filter, all multicast packets are accepted regardless of the multicast hash values.
  If the Hash Table register is configured to be double-synchronized to the (G)MII clock domain, the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the Hash Table Register X registers are written.
  If double-synchronization is enabled, consecutive writes to this register should be performed after at least four clock cycles in the destination clock domain.
- **Size:** 32 bits
- **Offset:** 0x2c
- **Exists:** (DWC_EQOS_HASH_TABLE > 128)

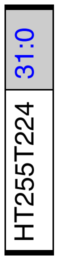


Table 17-16 Fields for Register: MAC_Hash_Table_Reg7

Bits	Name	Memory Access	Description
31:0	HT255T224	R/W	MAC Hash Table Eighth 32 Bits This field contains the eighth 32 Bits [255:224] of the Hash table. **Value After Reset:** 0x0 **Exists:** Always

17.1.13 MAC_VLAN_Tag

- **Description:** The VLAN Tag register identifies the IEEE 802.1Q VLAN type packets.
- **Size:** 32 bits
- **Offset:** 0x50
- **Exists:** !DWC_EQOS_ERVFE

31	30	29:28	27	26	25	24	23	22:21	20	19	18	17	16	15:0
EIVLRXS	Reserved_30	EIVLS	ERIVLT	EDVLP	VTHM	EVLRXS	Reserved_23	EVLS	DOVLTC	ERSVLM	ESVL	VTIM	ETV	VL

Table 17-17 Fields for Register: MAC_VLAN_Tag

Bits	Name	Memory Access	Description
31	EIVLRXS	R/W	Enable Inner VLAN Tag in Rx Status When this bit is set, the MAC provides the inner VLAN Tag in the Rx status. When this bit is reset, the MAC does not provide the inner VLAN Tag in Rx status. **Values:** ■ 0x0 (DISABLE): Inner VLAN Tag in Rx status is disabled ■ 0x1 (ENABLE): Inner VLAN Tag in Rx status is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_DOUBLE_VLAN_EN
30	Reserved_30	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
29:28	EIVLS	R/W	Enable Inner VLAN Tag Stripping on Receive This field indicates the stripping operation on inner VLAN Tag in received packet. **Values:** ■ 0x0 (DONOT): Do not strip ■ 0x1 (IFPASS): Strip if VLAN filter passes ■ 0x2 (IFFAIL): Strip if VLAN filter fails ■ 0x3 (ALWAYS): Always strip **Value After Reset:** 0x0 **Exists:** DWC_EQOS_DOUBLE_VLAN_EN

Bits	Name	Memory Access	Description
27	ERIVLT	R/W	Enable Inner VLAN Tag When this bit and the EDVLP field are set, the MAC receiver enables operation on the inner VLAN Tag (if present). When this bit is reset, the MAC receiver enables operation on the outer VLAN Tag (if present). The ERSVLM bit determines which VLAN type is enabled for filtering or matching. **Values:** ■ 0x0 (DISABLE): Inner VLAN tag is disabled ■ 0x1 (ENABLE): Inner VLAN tag is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_DOUBLE_VLAN_EN
26	EDVLP	R/W	Enable Double VLAN Processing When this bit is set, the MAC enables processing of up to two VLAN Tags on Tx and Rx (if present). When this bit is reset, the MAC enables processing of up to one VLAN Tag on Tx and Rx (if present). **Values:** ■ 0x0 (DISABLE): Double VLAN Processing is disabled ■ 0x1 (ENABLE): Double VLAN Processing is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_DOUBLE_VLAN_EN
25	VTHM	R/W	VLAN Tag Hash Table Match Enable When this bit is set, the most significant four bits of CRC of VLAN Tag (ones-complement of most significant four bits of CRC of VLAN Tag when ETV bit is reset) are used to index the content of the MAC_VLAN_Hash_Table register. A value of 1 in the VLAN Hash Table register, corresponding to the index, indicates that the packet matched the VLAN hash table. When the ETV bit is set, the CRC of the 12-bit VLAN Identifier (VID) is used for comparison. When the ETV bit is reset, the ones-complement of the CRC of the 16-bit VLAN tag is used for comparison. When this bit is reset, the VLAN Hash Match operation is not performed. **Values:** ■ 0x0 (DISABLE): VLAN Tag Hash Table Match is disabled ■ 0x1 (ENABLE): VLAN Tag Hash Table Match is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_VLAN_HASH_EN

Bits	Name	Memory Access	Description
24	EVLRXS	R/W	Enable VLAN Tag in Rx status When this bit is set, MAC provides the outer VLAN Tag in the Rx status. When this bit is reset, the MAC does not provide the outer VLAN Tag in Rx status. **Values:** ■ 0x0 (DISABLE): VLAN Tag in Rx status is disabled ■ 0x1 (ENABLE): VLAN Tag in Rx status is enabled **Value After Reset:** 0x0 **Exists:** Always
23	Reserved_23	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
22:21	EVLS	R/W	Enable VLAN Tag Stripping on Receive This field indicates the stripping operation on the outer VLAN Tag in received packet. **Values:** ■ 0x0 (DONOT): Do not strip ■ 0x1 (IFPASS): Strip if VLAN filter passes ■ 0x2 (IFFAIL): Strip if VLAN filter fails ■ 0x3 (ALWAYS): Always strip **Value After Reset:** 0x0 **Exists:** Always
20	DOVLTC	R/W	Disable VLAN Type Check When this bit is set, the MAC does not check whether the VLAN Tag specified by the ERIVLT bit is of type S-VLAN or C-VLAN. When this bit is reset, the MAC filters or matches the VLAN Tag specified by the ERIVLT bit only when VLAN Tag type is similar to the one specified by the ERSVLM bit. **Values:** ■ 0x0 (ENABLE): VLAN Type Check is enabled ■ 0x1 (DISABLE): VLAN Type Check is disabled **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
19	ERSVLM	R/W	Enable Receive S-VLAN Match When this bit is set, the MAC receiver enables filtering or matching for S-VLAN (Type = 0x88A8) packets. When this bit is reset, the MAC receiver enables filtering or matching for C-VLAN (Type = 0x8100) packets. The ERIVLT bit determines the VLAN tag position considered for filtering or matching. **Values:** ■ 0x0 (DISABLE): Receive S-VLAN Match is disabled ■ 0x1 (ENABLE): Receive S-VLAN Match is enabled **Value After Reset:** 0x0 **Exists:** Always
18	ESVL	R/W	Enable S-VLAN When this bit is set, the MAC transmitter and receiver consider the S-VLAN packets (Type = 0x88A8) as valid VLAN tagged packets. **Values:** ■ 0x0 (DISABLE): S-VLAN is disabled ■ 0x1 (ENABLE): S-VLAN is enabled **Value After Reset:** 0x0 **Exists:** Always
17	VTIM	R/W	VLAN Tag Inverse Match Enable When this bit is set, this bit enables the VLAN Tag inverse matching. The packets without matching VLAN Tag are marked as matched. When reset, this bit enables the VLAN Tag perfect matching. The packets with matched VLAN Tag are marked as matched. **Values:** ■ 0x0 (DISABLE): VLAN Tag Inverse Match is disabled ■ 0x1 (ENABLE): VLAN Tag Inverse Match is enabled **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
16	ETV	R/W	Enable 12-Bit VLAN Tag Comparison When this bit is set, a 12-bit VLAN identifier is used for comparing and filtering instead of the complete 16-bit VLAN tag. Bits[11:0] of VLAN tag are compared with the corresponding field in the received VLAN-tagged packet. Similarly, when enabled, only 12 bits of the VLAN tag in the received packet are used for hash-based VLAN filtering. When this bit is reset, all 16 bits of the 15th and 16th bytes of the received VLAN packet are used for comparison and VLAN hash filtering. **Values:** ■ 0x0 (DISABLE): 12-Bit VLAN Tag Comparison is disabled ■ 0x1 (ENABLE): 12-Bit VLAN Tag Comparison is enabled **Value After Reset:** 0x0 **Exists:** Always
15:0	VL	R/W	VLAN Tag Identifier for Receive Packets This field contains the 802.1Q VLAN tag to identify the VLAN packets. This VLAN tag identifier is compared to the 15th and 16th bytes of the packets being received for VLAN packets. The following list describes the bits of this field: ■ Bits[15:13]: User Priority ■ Bit 12: Canonical Format Indicator (CFI) or Drop Eligible Indicator (DEI) ■ Bits[11:0]: VLAN Identifier (VID) field of VLAN tag When the ETV bit is set, only the VID is used for comparison. If this field ([11:0] if ETV is set) is all zeros, the MAC does not check the 15th and 16th bytes for VLAN tag comparison and declares all packets with Type field value of 0x8100 or 0x88a8 as VLAN packets. **Value After Reset:** 0x0 **Exists:** Always

17.1.14 MAC_VLAN_Tag_Ctrl

- **Description:** This register is the redefined format of the MAC VLAN Tag Register. It is used for indirect addressing. It contains the address offset, command type and Busy Bit for CSR access of the Per VLAN Tag registers.
- **Size:** 32 bits
- **Offset:** 0x50
- **Exists:** DWC_EQOS_ERVFE

31	30	29:28	27	26	25	24	23	22:21	20:19	18	17	16:7	6:2	1	0
EIVLRXS	Reserved_30	EIVLS	ERIVLT	EDVLP	VTHM	EVLRXS	Reserved_23	EVLS	Reserved_20_19	ESVL	VTIM	Reserved_16_7	OFS	CT	OB

Table 17-18 Fields for Register: MAC_VLAN_Tag_Ctrl

Bits	Name	Memory Access	Description
31	EIVLRXS	R/W	Enable Inner VLAN Tag in Rx Status When this bit is set, the MAC provides the inner VLAN Tag in the Rx status. When this bit is reset, the MAC does not provide the inner VLAN Tag in Rx status. **Values:** ■ 0x0 (DISABLE): Inner VLAN Tag in Rx status is disabled ■ 0x1 (ENABLE): Inner VLAN Tag in Rx status is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_DOUBLE_VLAN_EN
30	Reserved_30	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
29:28	EIVLS	R/W	Enable Inner VLAN Tag Stripping on Receive This field indicates the stripping operation on inner VLAN Tag in received packet. **Values:** ■ 0x0 (DONOT): Do not strip ■ 0x1 (IFPASS): Strip if VLAN filter passes ■ 0x2 (IFFAIL): Strip if VLAN filter fails ■ 0x3 (ALWAYS): Always strip **Value After Reset:** 0x0 **Exists:** DWC_EQOS_DOUBLE_VLAN_EN

Bits	Name	Memory Access	Description
27	ERIVLT	R/W	**Values:** ■ 0x0 (DISABLE): Inner VLAN tag is disabled ■ 0x1 (ENABLE): Inner VLAN tag is enabled **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_DOUBLE_VLAN_EN&&DWC_EQOS_VLAN_HASH_EN)
26	EDVLP	R/W	Enable Double VLAN Processing When this bit is set, the MAC enables processing of up to two VLAN Tags on Tx and Rx (if present). When this bit is reset, the MAC enables processing of up to one VLAN Tag on Tx and Rx (if present). **Values:** ■ 0x0 (DISABLE): Double VLAN Processing is disabled ■ 0x1 (ENABLE): Double VLAN Processing is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_DOUBLE_VLAN_EN
25	VTHM	R/W	VLAN Tag Hash Table Match Enable When this bit is set, the most significant four bits of CRC of VLAN Tag (ones-complement of most significant four bits of CRC of VLAN Tag when ETV bit is reset) are used to index the content of the MAC_VLAN_Hash_Table register. A value of 1 in the VLAN Hash Table register, corresponding to the index, indicates that the packet matched the VLAN hash table. When the ETV bit is set, the CRC of the 12-bit VLAN Identifier (VID) is used for comparison. When the ETV bit is reset, the ones-complement of the CRC of the 16-bit VLAN tag is used for comparison. When this bit is reset, the VLAN Hash Match operation is not performed. **Values:** ■ 0x0 (DISABLE): VLAN Tag Hash Table Match is disabled ■ 0x1 (ENABLE): VLAN Tag Hash Table Match is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_VLAN_HASH_EN

Bits	Name	Memory Access	Description
24	EVLRXS	R/W	Enable VLAN Tag in Rx status When this bit is set, MAC provides the outer VLAN Tag in the Rx status. When this bit is reset, the MAC does not provide the outer VLAN Tag in Rx status. **Values:** ■ 0x0 (DISABLE): VLAN Tag in Rx status is disabled ■ 0x1 (ENABLE): VLAN Tag in Rx status is enabled **Value After Reset:** 0x0 **Exists:** Always
23	Reserved_23	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
22:21	EVLS	R/W	Enable VLAN Tag Stripping on Receive This field indicates the stripping operation on the outer VLAN Tag in received packet. **Values:** ■ 0x0 (DONOT): Do not strip ■ 0x1 (IFPASS): Strip if VLAN filter passes ■ 0x2 (IFFAIL): Strip if VLAN filter fails ■ 0x3 (ALWAYS): Always strip **Value After Reset:** 0x0 **Exists:** Always
20:19	Reserved_20_19	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
18	ESVL	R/W	Enable S-VLAN When this bit is set, the MAC transmitter and receiver consider the S-VLAN packets (Type = 0x88A8) as valid VLAN tagged packets. **Values:** ■ 0x0 (DISABLE): S-VLAN is disabled ■ 0x1 (ENABLE): S-VLAN is enabled **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
17	VTIM	R/W	VLAN Tag Inverse Match Enable When this bit is set, this bit enables the VLAN Tag inverse matching. The packets without matching VLAN Tag are marked as matched. When reset, this bit enables the VLAN Tag perfect matching. The packets with matched VLAN Tag are marked as matched. **Values:** ■ 0x0 (DISABLE): VLAN Tag Inverse Match is disabled ■ 0x1 (ENABLE): VLAN Tag Inverse Match is enabled **Value After Reset:** 0x0 **Exists:** Always
16:7	Reserved_16_7	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
6:2	OFS	R/W	Offset This field holds the address offset of the MAC VLAN Tag Filter Register which the application is trying to access. The width of the field depends on the number of MAC VLAN Tag Registers enabled. **Value After Reset:** 0x0 **Exists:** Always
1	CT	R/W	Command Type This bit indicates if the current register access is a read or a write. When set, it indicate a read operation. When reset, it indicates a write operation. **Values:** ■ 0x0 (WRITE): Write operation ■ 0x1 (READ): Read operation **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
0	OB	R/W	Operation Busy This bit is set along with a read or write command for initiating the indirect access to per VLAN Tag Filter register. This bit is reset when the read or write command to per VLAN Tag Filter indirect access register is complete. The next indirect register access can be initiated only after this bit is reset. During a write operation, the bit is reset only after the data has been written into the Per VLAN Tag register. During a read operation, the data should be read from the MAC_VLAN_Tag_Data register only after this bit is reset. **Values:** ■ 0x0 (DISABLE): Operation Busy is disabled ■ 0x1 (ENABLE): Operation Busy is enabled **Value After Reset:** 0x0 **Exists:** Always **Testable:** untestable

17.1.15 MAC_VLAN_Tag_Data

- **Description:** This register holds the read/write data for Indirect Access of the Per VLAN Tag registers. During the read access, this field contains valid read data only after the OB bit is reset. During the write access, this field should be valid prior to setting the OB bit in the MAC_VLAN_Tag_Ctrl Register.
- **Size:** 32 bits
- **Offset:** 0x54
- **Exists:** DWC_EQOS_ERVFE

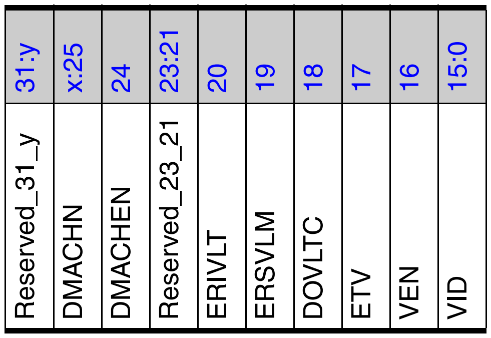


Table 17-19 Fields for Register: MAC_VLAN_Tag_Data

Bits	Name	Memory Access	Description
31:y	Reserved_31_y	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
x:25	DMACHN	R/W	DMA Channel Number The DMA Channel number to which the VLAN Tagged Frame is to be routed if it passes this VLAN Tag Filter is programmed in this field. If the Routing based on VLAN Tag Filter is not necessary, this field need not be programmed. **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_SYS>1) **Range Variable[x]:** DWC_EQOS_NUM_DMA_RX_CH>4?3:DWC_EQOS_NUM_DMA_RX_CH>2?2:1 + 24

Bits	Name	Memory Access	Description
24	DMACHEN	R/W	DMA Channel Number Enable This bit is the Enable for the DMA Channel Number value programmed in the field DMACH. When this bit is reset, the Routing does not occur based on VLAN Filter result. The frame is routed based on DA Based DMA Channel Routing. **Values:** ■ 0x0 (DISABLE): DMA Channel Number is disabled ■ 0x1 (ENABLE): DMA Channel Number is enabled **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_SYS>1)
23:21	Reserved_23_21	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
20	ERIVLT	R/W	Enable Inner VLAN Tag Comparison This bit is valid only when VLAN Tag Enable of the Filter is set. When this bit and the EDVLP field are set, the MAC receiver enables operation on the inner VLAN Tag (if present). When this bit is reset, the MAC receiver enables operation on the outer VLAN Tag (if present). **Values:** ■ 0x0 (DISABLE): Inner VLAN tag comparison is disabled ■ 0x1 (ENABLE): Inner VLAN tag comparison is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_DOUBLE_VLAN_EN
19	ERSVLM	R/W	Enable S-VLAN Match for received Frames This bit is valid only when VLAN Tag Enable of the Filter is set. When this bit is set, the MAC receiver enables filtering or matching for S-VLAN (Type = 0x88A8) packets. When this bit is reset, the MAC receiver enables filtering or matching for C-VLAN (Type = 0x8100) packets. **Values:** ■ 0x0 (DISABLE): Receive S-VLAN Match is disabled ■ 0x1 (ENABLE): Receive S-VLAN Match is enabled **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
18	DOVLTC	R/W	Disable VLAN Type Comparison This bit is valid only when VLAN Tag Enable of the Filter is set. When this bit is set, the MAC does not check whether the VLAN Tag specified by the Enable Inner VLAN Tag Comparison bit is of type S-VLAN or C-VLAN. When this bit is reset, the MAC filters or matches the VLAN Tag specified by the Enable Inner VLAN Tag Comparison bit only when VLAN Tag type is similar to the one specified by the Enable S-VLAN Match for received Frames bit. **Values:** ■ 0x0 (ENABLE): VLAN type comparison is enabled ■ 0x1 (DISABLE): VLAN type comparison is disabled **Value After Reset:** 0x0 **Exists:** Always
17	ETV	R/W	12bits or 16bits VLAN comparison This bit is valid only when VEN of the Filter is set. When this bit is set, a 12-bit VLAN identifier is used for comparing and filtering instead of the complete 16-bit VLAN tag. Bits [11:0] of VLAN tag are compared with the corresponding field in the received VLAN-tagged packet. **Values:** ■ 0x0 (16BIT): 16 bit VLAN comparison ■ 0x1 (12BIT): 12 bit VLAN comparison **Value After Reset:** 0x0 **Exists:** Always
16	VEN	R/W	VLAN Tag Enable This bit is used to enable or disable the VLAN Tag. When this bit is set, the MAC compares the VLAN Tag of received packet with the VLAN Tag ID. When this bit is reset, no comparison is performed irrespective of the programming of the other fields. **Values:** ■ 0x0 (DISABLE): VLAN Tag is disabled ■ 0x1 (ENABLE): VLAN Tag is enabled **Value After Reset:** 0x0 **Exists:** Always
15:0	VID	R/W	VLAN Tag ID This field holds the VLAN Tag value which is used by the MAC for perfect comparison. It is valid when VLAN Tag Enable is set. **Value After Reset:** 0x0 **Exists:** Always

17.1.16 MAC_VLAN_Tag_Filter(#i) (for i = 0; i <= DWC_EQOS_NRVF-1)

- **Description:** This register contains VLAN Tag filter ${i} control information.
- **Size:** 32 bits
- **Offset:** 0x54
- **Exists:** DWC_EQOS_ERVFE&&(DWC_EQOS_NRVF> 0)

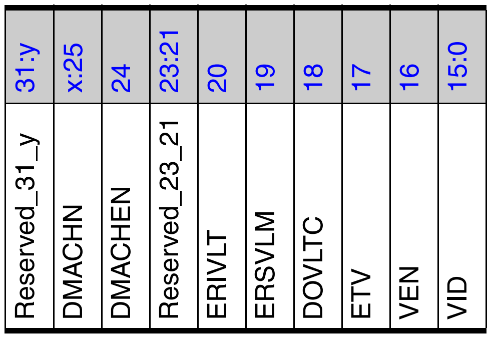


Table 17-20 Fields for Register: MAC_VLAN_Tag_Filter(#i) (for i = 0; i <= DWC_EQOS_NRVF-1)

Bits	Name	Memory Access	Description
31:y	Reserved_31_y	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
x:25	DMACHN	R/W	DMA Channel Number The DMA Channel number to which the VLAN Tagged Frame is to be routed if it passes this VLAN Tag Filter is programmed in this field. If the Routing based on VLAN Tag Filter is not necessary, this field need not be programmed. **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_SYS>1) **Range Variable[x]:** DWC_EQOS_NUM_DMA_RX_CH>4?3:DWC_EQOS_NUM_DMA_RX_CH>2?2:1 + 24
24	DMACHEN	R/W	DMA Channel Number Enable This bit is the Enable for the DMA Channel Number value programmed in the field DMACH. When this bit is reset, the Routing does not occur based on VLAN Filter result. The frame is routed based on DA Based DMA Channel Routing. **Values:** ■ 0x0 (DISABLE): DMA Channel Number is disabled ■ 0x1 (ENABLE): DMA Channel Number is enabled **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_SYS>1)

Bits	Name	Memory Access	Description
23:21	Reserved_23_21	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
20	ERIVLT	R/W	Enable Inner VLAN Tag Comparison This bit is valid only when VLAN Tag Enable of the Filter is set. When this bit and the EDVLP field are set, the MAC receiver enables operation on the inner VLAN Tag (if present). When this bit is reset, the MAC receiver enables operation on the outer VLAN Tag (if present). **Values:** ■ 0x0 (DISABLE): Inner VLAN tag comparison is disabled ■ 0x1 (ENABLE): Inner VLAN tag comparison is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_DOUBLE_VLAN_EN
19	ERSVLM	R/W	Enable S-VLAN Match for received Frames This bit is valid only when VLAN Tag Enable of the Filter is set. When this bit is set, the MAC receiver enables filtering or matching for S-VLAN (Type = 0x88A8) packets. When this bit is reset, the MAC receiver enables filtering or matching for C-VLAN (Type = 0x8100) packets. **Values:** ■ 0x0 (DISABLE): Receive S-VLAN Match is disabled ■ 0x1 (ENABLE): Receive S-VLAN Match is enabled **Value After Reset:** 0x0 **Exists:** Always
18	DOVLTC	R/W	Disable VLAN Type Comparison This bit is valid only when VLAN Tag Enable of the Filter is set. When this bit is set, the MAC does not check whether the VLAN Tag specified by the Enable Inner VLAN Tag Comparison bit is of type S-VLAN or C-VLAN. When this bit is reset, the MAC filters or matches the VLAN Tag specified by the Enable Inner VLAN Tag Comparison bit only when VLAN Tag type is similar to the one specified by the Enable S-VLAN Match for received Frames bit. **Values:** ■ 0x0 (ENABLE): VLAN type comparison is enabled ■ 0x1 (DISABLE): VLAN type comparison is disabled **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
17	ETV	R/W	12bits or 16bits VLAN comparison This bit is valid only when VEN of the Filter is set. When this bit is set, a 12-bit VLAN identifier is used for comparing and filtering instead of the complete 16-bit VLAN tag. Bits [11:0] of VLAN tag are compared with the corresponding field in the received VLAN-tagged packet. **Values:** ■ 0x0 (16BIT): 16 bit VLAN comparison ■ 0x1 (12BIT): 12 bit VLAN comparison **Value After Reset:** 0x0 **Exists:** Always
16	VEN	R/W	VLAN Tag Enable This bit is used to enable or disable the VLAN Tag. When this bit is set, the MAC compares the VLAN Tag of received packet with the VLAN Tag ID. When this bit is reset, no comparison is performed irrespective of the programming of the other fields. **Values:** ■ 0x0 (DISABLE): VLAN Tag is disabled ■ 0x1 (ENABLE): VLAN Tag is enabled **Value After Reset:** 0x0 **Exists:** Always
15:0	VID	R/W	VLAN Tag ID This field holds the VLAN Tag value which is used by the MAC for perfect comparison. It is valid when VLAN Tag Enable is set. **Value After Reset:** 0x0 **Exists:** Always

17.1.17 MAC_VLAN_Hash_Table

- **Description:** When VTHM bit of the MAC_VLAN_Tag register is set, the 16-bit VLAN Hash Table register is used for group address filtering based on the VLAN tag. For hash filtering, the content of the 16-bit VLAN tag or 12-bit VLAN ID (based on the ETV bit of MAC_VLAN_Tag Register) in the incoming packet is passed through the CRC logic. When ETV bit of MAC_VLAN_Tag register is set, the upper four bits of the calculated CRC are used to index the contents of the VLAN Hash table. When ETV bit of MAC_VLAN_Tag register is reset, the ones-complement of upper four bits of the calculated CRC are used to index the contents of the VLAN Hash table. For example, when ETV bit is set a hash value of 4b'1000 selects Bit 8 of the VLAN Hash table, whereas when ETV bit is reset a hash value of 4b'1000 selects Bit 7 of the VLAN Hash table.
  The hash value of the destination address is calculated in the following way:
  - Calculate the 32-bit CRC for the VLAN tag or ID (For steps to calculate CRC32, see Section 3.2.8 of IEEE 802.3).
  - Perform bitwise reversal for the value obtained in step 1.
  - Take the upper four bits from the value obtained in step 2.

  If the VLAN hash Table register is configured to be double-synchronized to the (G)MII clock domain, the synchronization is triggered only when Bits[15:8] (in little-endian mode) or Bits[7:0] (in big-endian mode) of this register are written.
  - If double-synchronization is enabled, consecutive writes to this register should be performed after at least four clock cycles in the destination clock domain.
- **Size:** 32 bits
- **Offset:** 0x58
- **Exists:** (DWC_EQOS_VLAN_HASH_EN)

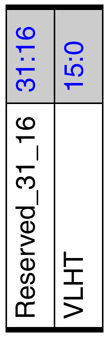


Table 17-21 Fields for Register: MAC_VLAN_Hash_Table

Bits	Name	Memory Access	Description
31:16	Reserved_31_16	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
15:0	VLHT	R/W	VLAN Hash Table This field contains the 16-bit VLAN Hash Table. **Value After Reset:** 0x0 **Exists:** Always

17.1.18 MAC_VLAN_Incl(#i) (for i = 0; i <= DWC_EQOS_NUM_TXQ-1)

- **Description:** The Tx Queue ${i} VLAN Tag Inclusion register contains the VLAN tag for insertion in the Transmit packets from Tx Queue ${i}. It also contains the VLAN tag insertion controls.
- **Size:** 32 bits
- **Offset:** 0x60
- **Exists:** DWC_EQOS_CBTI_EN

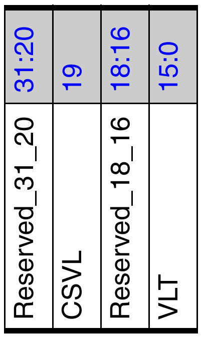


Table 17-22 Fields for Register: MAC_VLAN_Incl(#i) (for i = 0; i <= DWC_EQOS_NUM_TXQ-1)

Bits	Name	Memory Access	Description
31:20	Reserved_31_20	R/W	**Exists:** Always
19	CSVL	R	C-VLAN or S-VLAN When this bit is set, S-VLAN type (0x88A8) is inserted in the 13th and 14th bytes of transmitted packets. When this bit is reset, C-VLAN type (0x8100) is inserted in the 13th and 14th bytes of transmitted packets. **Values:** ■ 0x0 (C-VLAN): C-VLAN type (0x8100) is inserted ■ 0x1 (S-VLAN): S-VLAN type (0x88A8) is inserted **Value After Reset:** 0x0 **Exists:** Always
18:16	Reserved_18_16	R/W	**Exists:** Always

Bits	Name	Memory Access	Description
15:0	VLT	R/W	VLAN Tag for Transmit Packets This field contains the value of the VLAN tag to be inserted. The value must only be changed when the transmit lines are inactive or during the initialization phase. Bits[15:13] are the User Priority field, Bit 12 is the CFI/DEI field, and Bits[11:0] are the VID field in the VLAN tag. The following list describes the bits of this field: ■ Bits[15:13]: User Priority ■ Bit 12: Canonical Format Indicator (CFI) or Drop Eligible Indicator (DEI) ■ Bits[11:0]: VLAN Identifier (VID) field of VLAN tag **Exists:** Always

17.1.19 MAC_VLAN_Incl

- **Description:** The VLAN Tag Inclusion or Replacement register contains the VLAN tag for insertion or replacement in the Transmit packets. It also contains the VLAN tag insertion controls.
- **Size:** 32 bits
- **Offset:** 0x60
- **Exists:** (DWC_EQOS_SA_VLAN_INS_CTRL_EN)

31	30	29:y	x:24	23:22	21	20	19	18	17:16	15:0
BUSY	RDWR	Reserved_29_y	ADDR	Reserved_23_22	CBTI	VLTI	CSVL	VLP	VLC	VLT

Table 17-23 Fields for Register: MAC_VLAN_Incl

Bits	Name	Memory Access	Description
31	BUSY	R	Busy This bit indicates the status of the read/write operation of indirect access to the queue/channel specific VLAN inclusion register. For write operation write to a register is complete when this bit is reset. For read operation the read data is valid when the bit is reset. The application must make sure that this bit is reset before attempting subsequent access to this register. **Values:** ■ 0x0 (INACTIVE): Busy status not detected ■ 0x1 (ACTIVE): Busy status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_CBTI_EN **Testable:** untestable

Bits	Name	Memory Access	Description
30	RDWR	R/W	Read write control This bit controls the read or write operation for indirectly accessing the queue/channel specific VLAN Inclusion register. When set indicates write operation and when reset indicates read operation. This does not have any effect when CBTI is reset. **Values:** ■ 0x0 (READ): Read operation of indirect access ■ 0x1 (WRITE): Write operation of indirect access **Value After Reset:** 0x0 **Exists:** DWC_EQOS_CBTI_EN
29:y	Reserved_29_y	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[y]:** DWC_EQOS_TXQW + 24
x:24	ADDR	R/W	Address This field selects one of the queue/channel specific VLAN Inclusion register for read/write access. This does not have any effect when CBTI is reset. **Value After Reset:** 0x0 **Exists:** DWC_EQOS_CBTI_EN **Range Variable[x]:** DWC_EQOS_TXQW + 23
23:22	Reserved_23_22	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
21	CBTI	R/W	Channel based tag insertion When this bit is set, outer VLAN tag is inserted for every packets transmitted by the MAC. The tag value is taken from the queue/channel specific VLAN tag register. The VLTI, VLP, VLC, and VLT fields of this register are ignored when this bit is set. When this bit is set, a write operation to byte 3 of this register initiates the read/write access to the indirect register. When reset, outer VLAN operation is based on the setting of VLTI, VLP, VLC and VLT fields of this register. **Values:** ■ 0x0 (DISABLE): Channel based tag insertion is disabled ■ 0x1 (ENABLE): Channel based tag insertion is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_CBTI_EN

Bits	Name	Memory Access	Description
20	VLTI	R/W	VLAN Tag Input When this bit is set, it indicates that the VLAN tag to be inserted or replaced in Tx packet should be taken from the following: ■ The mti_vlan_tag_i signal in EQOS-CORE configurations ■ The control word in EQOS-MTL configurations ■ The Tx descriptor in EQOS-AHB, EQOS-AXI, or EQOS-DMA configurations **Values:** ■ 0x0 (DISABLE): VLAN Tag Input is disabled ■ 0x1 (ENABLE): VLAN Tag Input is enabled **Value After Reset:** 0x0 **Exists:** Always
19	CSVL	R/W	C-VLAN or S-VLAN When this bit is set, S-VLAN type (0x88A8) is inserted or replaced in the 13th and 14th bytes of transmitted packets. When this bit is reset, C-VLAN type (0x8100) is inserted or replaced in the 13th and 14th bytes of transmitted packets. **Values:** ■ 0x0 (C-VLAN): C-VLAN type (0x8100) is inserted or replaced ■ 0x1 (S-VLAN): S-VLAN type (0x88A8) is inserted or replaced **Value After Reset:** 0x0 **Exists:** Always
18	VLP	R/W	VLAN Priority Control When this bit is set, the control bits[17:16] are used for VLAN deletion, insertion, or replacement. When this bit is reset, the mti_vlan_ctrl_i control input is used and bits[17:16] are ignored. **Values:** ■ 0x0 (DISABLE): VLAN Priority Control is disabled ■ 0x1 (ENABLE): VLAN Priority Control is enabled **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
17:16	VLC	R/W	VLAN Tag Control in Transmit Packets ■ 2'b00: No VLAN tag deletion, insertion, or replacement ■ 2'b01: VLAN tag deletion The MAC removes the VLAN type (bytes 13 and 14) and VLAN tag (bytes 15 and 16) of all transmitted packets with VLAN tags. ■ 2'b10: VLAN tag insertion The MAC inserts VLT in bytes 15 and 16 of the packet after inserting the Type value (0x8100 or 0x88a8) in bytes 13 and 14. This operation is performed on all transmitted packets, irrespective of whether they already have a VLAN tag. ■ 2'b11: VLAN tag replacement The MAC replaces VLT in bytes 15 and 16 of all VLAN-type transmitted packets (Bytes 13 and 14 are 0x8100 or 0x88a8). **Note**: Changes to this field take effect only on the start of a packet. If you write this register field when a packet is being transmitted, only the subsequent packet can use the updated value, that is, the current packet does not use the updated value. **Values:** ■ 0x0 (NONE): No VLAN tag deletion, insertion, or replacement ■ 0x1 (DELETE): VLAN tag deletion ■ 0x2 (INSERT): VLAN tag insertion ■ 0x3 (REPLACE): VLAN tag replacement **Value After Reset:** 0x0 **Exists:** Always
15:0	VLT	R/W	VLAN Tag for Transmit Packets This field contains the value of the VLAN tag to be inserted or replaced. The value must only be changed when the transmit lines are inactive or during the initialization phase. Bits[15:13] are the User Priority field, Bit 12 is the CFI/DEI field, and Bits[11:0] are the VID field in the VLAN tag. The following list describes the bits of this field: ■ Bits[15:13]: User Priority ■ Bit 12: Canonical Format Indicator (CFI) or Drop Eligible Indicator (DEI) ■ Bits[11:0]: VLAN Identifier (VID) field of VLAN tag **Value After Reset:** 0x0 **Exists:** Always

17.1.20 MAC_Inner_VLAN_Incl

- **Description:** The Inner VLAN Tag Inclusion or Replacement register contains the inner VLAN tag to be inserted or replaced in the Transmit packet. It also contains the inner VLAN tag insertion controls.
- **Size:** 32 bits
- **Offset:** 0x64
- **Exists:** (DWC_EQOS_SA_VLAN_INS_CTRL_EN&&DWC_EQOS_DOUBLE_VLAN_EN)

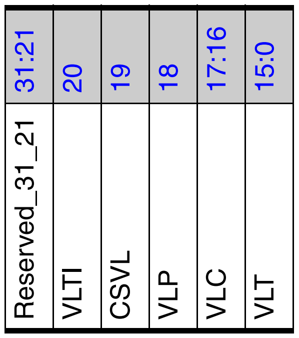


Table 17-24 Fields for Register: MAC_Inner_VLAN_Incl

Bits	Name	Memory Access	Description
31:21	Reserved_31_21	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
20	VLTI	R/W	VLAN Tag Input When this bit is set, it indicates that the VLAN tag to be inserted or replaced in Tx packet should be taken from the following: ■ The mti_vlan_tag_i signal in EQOS-CORE configurations ■ The control word in EQOS-MTL configurations ■ The Tx descriptor in EQOS-AHB, EQOS-AXI, or EQOS-DMA configurations **Values:** ■ 0x0 (DISABLE): VLAN Tag Input is disabled ■ 0x1 (ENABLE): VLAN Tag Input is enabled **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
19	CSVL	R/W	C-VLAN or S-VLAN When this bit is set, S-VLAN type (0x88A8) is inserted or replaced in the 13th and 14th bytes of transmitted packets. When this bit is reset, C-VLAN type (0x8100) is inserted or replaced in the 13th and 14th bytes of transmitted packets. **Values:** ■ 0x0 (C-VLAN): C-VLAN type (0x8100) is inserted ■ 0x1 (S-VLAN): S-VLAN type (0x88A8) is inserted **Value After Reset:** 0x0 **Exists:** Always
18	VLP	R/W	VLAN Priority Control When this bit is set, the VLC field is used for VLAN deletion, insertion, or replacement. When this bit is reset, the mti_vlan_ctrl_i control input is used and the VLC field is ignored. **Values:** ■ 0x0 (DISABLE): VLAN Priority Control is disabled ■ 0x1 (ENABLE): VLAN Priority Control is enabled **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
17:16	VLC	R/W	VLAN Tag Control in Transmit Packets ■ 2'b00: No VLAN tag deletion, insertion, or replacement ■ 2'b01: VLAN tag deletion The MAC removes the VLAN type (bytes 17 and 18) and VLAN tag (bytes 19 and 20) of all transmitted packets with VLAN tags. ■ 2'b10: VLAN tag insertion The MAC inserts VLT in bytes 19 and 20 of the packet after inserting the Type value (0x8100 or 0x88a8) in bytes 17 and 18. This operation is performed on all transmitted packets, irrespective of whether they already have a VLAN tag. ■ 2'b11: VLAN tag replacement The MAC replaces VLT in bytes 19 and 20 of all VLAN-type transmitted packets (Bytes 17 and 18 are 0x8100 or 0x88a8). **Note**: Changes to this field take effect only on the start of a packet. If you write this register field when a packet is being transmitted, only the subsequent packet can use the updated value, that is, the current packet does not use the updated value. **Values:** ■ 0x0 (NONE): No VLAN tag deletion, insertion, or replacement ■ 0x1 (DELETE): VLAN tag deletion ■ 0x2 (INSERT): VLAN tag insertion ■ 0x3 (REPLACE): VLAN tag replacement **Value After Reset:** 0x0 **Exists:** Always
15:0	VLT	R/W	VLAN Tag for Transmit Packets This field contains the value of the VLAN tag to be inserted or replaced. The value must only be changed when the transmit lines are inactive or during the initialization phase. Bits[15:13] are the User Priority field, Bit 12 is the CFI/DEI field, and Bits[11:0] are the VID field in the VLAN tag. The following list describes the bits of this field: ■ Bits[15:13]: User Priority ■ Bit 12: Canonical Format Indicator (CFI) or Drop Eligible Indicator (DEI) ■ Bits[11:0]: VLAN Identifier (VID) field of VLAN tag **Value After Reset:** 0x0 **Exists:** Always

17.1.21 MAC_Q0_Tx_Flow_Ctrl

- **Description:** The Flow Control register controls the generation and reception of the Control (Pause Command) packets by the Flow control module of the MAC. A Write to a register with the Busy bit set to 1 triggers the Flow Control block to generate a Pause packet. The fields of the control packet are selected as specified in the 802.3x specification, and the Pause Time value from this register is used in the Pause Time field of the control packet. The Busy bit remains set until the control packet is transferred onto the cable. The application must make sure that the Busy bit is cleared before writing to the register.
  When the PFCE bit in the MAC_Rx_Flow_Ctrl register is enabled, this register controls the generation of Priority Flow Control (PFC) frames with priorities mapped according to PSRQ0 in the MAC_RxQ_Ctrl2 register.
- **Size:** 32 bits
- **Offset:** 0x70
- **Exists:** Always

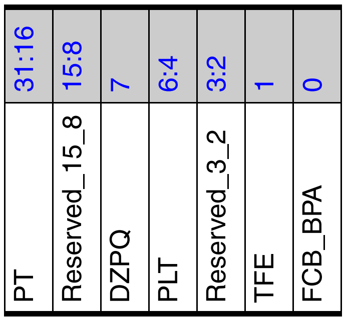


Table 17-25 Fields for Register: MAC_Q0_Tx_Flow_Ctrl

Bits	Name	Memory Access	Description
31:16	PT	R/W	Pause Time This field holds the value to be used in the Pause Time field in the Tx control packet. If the Pause Time bits are configured to be double-synchronized to the (G)MII clock domain, consecutive writes to this register should be performed only after at least four clock cycles in the destination clock domain. **Value After Reset:** 0x0 **Exists:** Always
15:8	Reserved_15_8	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
7	DZPQ	R/W	Disable Zero-Quanta Pause When this bit is set, it disables the automatic generation of the zero-quanta Pause packets on de-assertion of the flow-control signal from the FIFO layer (MTL or external sideband flow control signal sbd_flowctrl_i or mti_flowctrl_i). When this bit is reset, normal operation with automatic zero-quanta Pause packet generation is enabled. **Values:** ■ 0x0 (ENABLE): Zero-Quanta Pause packet generation is enabled ■ 0x1 (DISABLE): Zero-Quanta Pause packet generation is disabled **Value After Reset:** 0x0 **Exists:** Always
6:4	PLT	R/W	Pause Low Threshold This field configures the threshold of the Pause timer at which the input flow control signal mti_flowctrl_i (or sbd_flowctrl_i) is checked for automatic retransmission of the Pause packet. The threshold values should be always less than the Pause Time configured in Bits[31:16]. For example, if PT = 100H (256 slot times), and PLT = 001, a second Pause packet is automatically transmitted if the mti_flowctrl_i signal is asserted at 228 (256-28) slot times after the first Pause packet is transmitted. The following list provides the threshold values for different values. The slot time is defined as the time taken to transmit 512 bits (64 bytes) on the GMII or MII interface. This (approximate) computation is based on the packet size (64, 1518, 2000, 9018, 16384, or 32768) + 2 Pause Packet Size + IPG in Slot Times. **Values:** ■ 0x0 (PT4): Pause Time minus 4 Slot Times (PT -4 slot times) ■ 0x1 (PT28): Pause Time minus 28 Slot Times (PT -28 slot times) ■ 0x2 (PT36): Pause Time minus 36 Slot Times (PT -36 slot times) ■ 0x3 (PT144): Pause Time minus 144 Slot Times (PT -144 slot times) ■ 0x4 (PT256): Pause Time minus 256 Slot Times (PT -256 slot times) ■ 0x5 (PT512): Pause Time minus 512 Slot Times (PT -512 slot times) ■ 0x6 (RSVD): Reserved **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
3:2	Reserved_3_2	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
1	TFE	R/W	Transmit Flow Control Enable **Full-Duplex Mode:** In the full-duplex mode, when this bit is set, the MAC enables the flow control operation to Tx Pause packets. When this bit is reset, the flow control operation in the MAC is disabled, and the MAC does not transmit any Pause packets. **Half-Duplex Mode:** In the half-duplex mode, when this bit is set, the MAC enables the backpressure operation. When this bit is reset, the backpressure feature is disabled. **Values:** ■ 0x0 (DISABLE): Transmit Flow Control is disabled ■ 0x1 (ENABLE): Transmit Flow Control is enabled **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
0	FCB_BPA	R/W	Flow Control Busy or Backpressure Activate This bit initiates a Pause packet in the full-duplex mode and activates the backpressure function in the half-duplex mode if the TFE bit is set. **Full-Duplex Mode:** In the full-duplex mode, this bit should be read as 1'b0 before writing to this register. To initiate a Pause packet, the application must set this bit to 1'b1. During Control packet transfer, this bit continues to be set to indicate that a packet transmission is in progress. When Pause packet transmission is complete, the MAC resets this bit to 1'b0. You should not write to this register until this bit is cleared. **Half-Duplex Mode:** When this bit is set (and TFE bit is set) in the half-duplex mode, the MAC asserts the backpressure. During backpressure, when the MAC receives a new packet, the transmitter starts sending a JAM pattern resulting in a collision. This control register bit is logically ORed with the mti_flowctrl_i input signal for the backpressure function. When the MAC is configured for the full-duplex mode, the BPA is automatically disabled. Access restriction applies. Setting 1 sets. Self-cleared. Setting 0 has no effect. **Values:** ■ 0x0 (DISABLE): Flow Control Busy or Backpressure Activate is disabled ■ 0x1 (ENABLE): Flow Control Busy or Backpressure Activate is enabled **Value After Reset:** 0x0 **Exists:** Always **Testable:** untestable

17.1.22 MAC_Q(#i)_Tx_Flow_Ctrl (for i = 1; i <= DWC_EQOS_NUM_TXQ-1)

- **Description:** This register controls the generation of PFC Control packets of priorities mapped as per the PSRQ*i* field in the MAC_RxQ_Ctrl2/MAC_RxQ_Ctrl3 registers.
- **Size:** 32 bits
- **Offset:** (0x0004*i)+0x0070
- **Exists:** (DWC_EQOS_NUM_RXQ>1 && DWC_EQOS_DCB_EN)

31:16	15:8	7	6:4	3:2	1	0
PT	Reserved_15_8	DZPQ	PLT	Reserved_3_2	TFE	FCB_BPA

Table 17-26 Fields for Register: MAC_Q(#i)_Tx_Flow_Ctrl (for i = 1; i <= DWC_EQOS_NUM_TXQ-1)

Bits	Name	Memory Access	Description
31:16	PT	R/W	Pause Time This field holds the value to be used in the Pause Time field in the Tx control packet. If the Pause Time bits are configured to be double-synchronized to the (G)MII clock domain, consecutive writes to this register should be performed only after at least four clock cycles in the destination clock domain. **Value After Reset:** 0x0 **Exists:** Always
15:8	Reserved_15_8	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
7	DZPQ	R/W	Disable Zero-Quanta Pause When this bit is set, it disables the automatic generation of the zero-quanta Pause packets on de-assertion of the flow-control signal from the FIFO layer (MTL or external sideband flow control signal sbd_flowctrl_i or mti_flowctrl_i). When this bit is reset, normal operation with automatic zero-quanta Pause packet generation is enabled. **Values:** ■ 0x0 (ENABLE): Zero-Quanta Pause packet generation is enabled ■ 0x1 (DISABLE): Zero-Quanta Pause packet generation is disabled **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
6:4	PLT	R/W	Pause Low Threshold This field configures the threshold of the Pause timer at which the input flow control signal mti_flowctrl_i (or sbd_flowctrl_i) is checked for automatic retransmission of the Pause packet. The threshold values should be always less than the Pause Time configured in Bits[31:16]. For example, if PT = 100H (256 slot times), and PLT = 001, a second Pause packet is automatically transmitted if the mti_flowctrl_i signal is asserted at 228 (256-28) slot times after the first Pause packet is transmitted. The following list provides the threshold values for different values. The slot time is defined as the time taken to transmit 512 bits (64 bytes) on the GMII or MII interface. This (approximate) computation is based on the packet size (64, 1518, 2000, 9018, 16384, or 32768) + 2 Pause Packet Size + IPG in Slot Times. **Values:** ■ 0x0 (PT4): Pause Time minus 4 Slot Times (PT -4 slot times) ■ 0x1 (PT28): Pause Time minus 28 Slot Times (PT -28 slot times) ■ 0x2 (PT36): Pause Time minus 36 Slot Times (PT -36 slot times) ■ 0x3 (PT144): Pause Time minus 144 Slot Times (PT -144 slot times) ■ 0x4 (PT256): Pause Time minus 256 Slot Times (PT -256 slot times) ■ 0x5 (PT512): Pause Time minus 512 Slot Times (PT -512 slot times) ■ 0x6 (RSVD): Reserved **Value After Reset:** 0x0 **Exists:** Always
3:2	Reserved_3_2	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
1	TFE	R/W	Transmit Flow Control Enable When this bit is set in full-duplex mode, the MAC enables the flow control operation to Tx Pause packets. When this bit is reset, the flow control operation in the MAC is disabled, and the MAC does not transmit any Pause packets. **Values:** ■ 0x0 (DISABLE): Transmit Flow Control is disabled ■ 0x1 (ENABLE): Transmit Flow Control is enabled **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
0	FCB_BPA	R/W	Flow Control Busy This bit initiates a PFC packet if the TFE bit is set. To initiate a PFC packet, the application must set this bit to 1'b1. During Control packet transfer, this bit continues to be set to indicate that a packet transmission is in progress. When PFC packet transmission is complete, the MAC resets this bit to 1'b0. You should not write to this register until this bit is cleared. Access restriction applies. Setting 1 sets. Self-cleared. Setting 0 has no effect. **Values:** ■ 0x0 (DISABLE): Flow Control Busy or Backpressure Activate is disabled ■ 0x1 (ENABLE): Flow Control Busy or Backpressure Activate is enabled **Value After Reset:** 0x0 **Exists:** Always **Testable:** untestable

17.1.23 MAC_Rx_Flow_Ctrl

- **Description:** The Receive Flow Control register controls the pausing of MAC Transmit based on the received Pause packet.
- **Size:** 32 bits
- **Offset:** 0x90
- **Exists:** Always

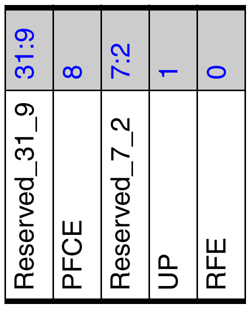


Table 17-27 Fields for Register: MAC_Rx_Flow_Ctrl

Bits	Name	Memory Access	Description
31:9	Reserved_31_9	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
8	PFCE	R/W	Priority Based Flow Control Enable When this bit is set, it enables generation and reception of priority-based flow control (PFC) packets. When this bit is reset, it enables generation and reception of 802.3x Pause control packets. **Values:** ■ 0x0 (DISABLE): Priority Based Flow Control is disabled ■ 0x1 (ENABLE): Priority Based Flow Control is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_DCB_EN
7:2	Reserved_7_2	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
1	UP	R/W	Unicast Pause Packet Detect A pause packet is processed when it has the unique multicast address specified in the IEEE 802.3. When this bit is set, the MAC can also detect Pause packets with unicast address of the station. This unicast address should be as specified in MAC_Address0_High and MAC_Address0_Low. When this bit is reset, the MAC only detects Pause packets with unique multicast address. **Note**: The MAC does not process a Pause packet if the multicast address is different from the unique multicast address. This is also applicable to the received PFC packet when the Priority Flow Control (PFC) is enabled. The unique multicast address (0x01_80_C2_00_00_01) is as specified in IEEE 802.1 Qbb-2011. **Values:** ■ 0x0 (DISABLE): Unicast Pause Packet Detect disabled ■ 0x1 (ENABLE): Unicast Pause Packet Detect enabled **Value After Reset:** 0x0 **Exists:** Always
0	RFE	R/W	Receive Flow Control Enable When this bit is set and the MAC is operating in full-duplex mode, the MAC decodes the received Pause packet and disables its transmitter for a specified (Pause) time. When this bit is reset or the MAC is operating in half-duplex mode, the decode function of the Pause packet is disabled. When PFC is enabled, flow control is enabled for PFC packets. The MAC decodes the received PFC packet and disables the Transmit queue, with matching priorities, for a duration of received Pause time. **Values:** ■ 0x0 (DISABLE): Receive Flow Control is disabled ■ 0x1 (ENABLE): Receive Flow Control is enabled **Value After Reset:** 0x0 **Exists:** Always

17.1.24 MAC_RxQ_Ctrl4

- **Description:** The Receive Queue Control 4 register controls the routing of unicast and multicast packets that fail the Destination or Source address filter to the Rx queues.
- **Size:** 32 bits
- **Offset:** 0x94
- **Exists:** (DWC_EQOS_NUM_RXQ>1)

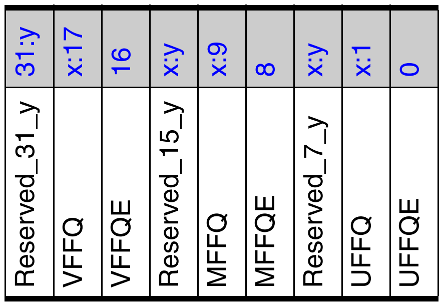


Table 17-28 Fields for Register: MAC_RxQ_Ctrl4

Bits	Name	Memory Access	Description
31:y	Reserved_31_y	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
x:17	VFFQ	R/W	VLAN Tag Filter Fail Packets Queue This field holds the Rx queue number to which the tagged packets failing the Destination or Source Address filter (and UFFQE/MFFQE not enabled) or failing the VLAN tag filter must be routed to. This field is valid only when the VFFQE bit is set. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[x]:** DWC_EQOS_NUM_RXQ>4?3:DWC_EQOS_NUM_RXQ>2?2 :1 + 16

Bits	Name	Memory Access	Description
16	VFFQE	R/W	VLAN Tag Filter Fail Packets Queuing Enable When this bit is set, the tagged packets which fail the Destination or Source address filter or fail the VLAN tag filter, are routed to the Rx Queue Number programmed in the VFFQ. When this bit is reset, the tagged packets which fail the Destination or Source address filter or fail the VLAN tag filter are routed based on other routing options. This bit is valid only when the RA bit of the MAC_Packet_Filter register is set. **Values:** ■ 0x0 (DISABLE): VLAN tag Filter Fail Packets Queuing is disabled ■ 0x1 (ENABLE): VLAN tag Filter Fail Packets Queuing is enabled **Value After Reset:** 0x0 **Exists:** Always
x:y	Reserved_15_y	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[x]:** DWC_EQOS_NUM_RXQ>4?4:DWC_EQOS_NUM_RXQ>2?5:6 + DWC_EQOS_NUM_RXQ>4?12:DWC_EQOS_NUM_RXQ>2?11:10 - 1 **Range Variable[y]:** DWC_EQOS_NUM_RXQ>4?12:DWC_EQOS_NUM_RXQ>2?11:10
x:9	MFFQ	R/W	Multicast Address Filter Fail Packets Queue. This field holds the Rx queue number to which the Multicast packets failing the Destination or Source Address filter are routed to. This field is valid only when the MFFQE bit is set. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[x]:** DWC_EQOS_NUM_RXQ>4?3:DWC_EQOS_NUM_RXQ>2?2:1 + 8

Bits	Name	Memory Access	Description
8	MFFQE	R/W	Multicast Address Filter Fail Packets Queuing Enable. When this bit is set, the Multicast packets which fail the Destination or Source address filter is routed to the Rx Queue Number programmed in the MFFQ. When this bit is reset, the Multicast packets which fail the Destination or Source address filter is routed based on other routing options. This bit is valid only when the RA bit of the MAC_Packet_Filter register is set. **Values:** ■ 0x0 (DISABLE): Multicast Address Filter Fail Packets Queuing is disabled ■ 0x1 (ENABLE): Multicast Address Filter Fail Packets Queuing is enabled **Value After Reset:** 0x0 **Exists:** Always
x:y	Reserved_7_y	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[x]:** DWC_EQOS_NUM_RXQ>4?4:DWC_EQOS_NUM_RXQ>2?5:6 + DWC_EQOS_NUM_RXQ>4?4:DWC_EQOS_NUM_RXQ>2?3:2 - 1 **Range Variable[y]:** DWC_EQOS_NUM_RXQ>4?4:DWC_EQOS_NUM_RXQ>2?3:2
x:1	UFFQ	R/W	Unicast Address Filter Fail Packets Queue. This field holds the Rx queue number to which the Unicast packets failing the Destination or Source Address filter are routed to. This field is valid only when the UFFQE bit is set. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[x]:** DWC_EQOS_NUM_RXQ>4?3:DWC_EQOS_NUM_RXQ>2?2:1

Bits	Name	Memory Access	Description
0	UFFQE	R/W	Unicast Address Filter Fail Packets Queuing Enable. When this bit is set, the Unicast packets which fail the Destination or Source address filter is routed to the Rx Queue Number programmed in the UFFQ. When this bit is reset, the Unicast packets which fail the Destination or Source address filter is routed based on other routing options. This bit is valid only when the RA bit of the MAC_Packet_Filter register is set. **Values:** ■ 0x0 (DISABLE): Unicast Address Filter Fail Packets Queuing is disabled ■ 0x1 (ENABLE): Unicast Address Filter Fail Packets Queuing is enabled **Value After Reset:** 0x0 **Exists:** Always

17.1.25 MAC_TxQ_Prty_Map0

- **Description:** The Transmit Queue Priority Mapping 0 register contains the priority values assigned to Tx Queue 0 through Tx Queue 3.
- **Size:** 32 bits
- **Offset:** 0x98
- **Exists:** (DWC_EQOS_DCB_EN&&(DWC_EQOS_NUM_TXQ>1))

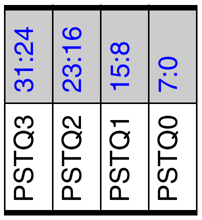


Table 17-29 Fields for Register: MAC_TxQ_Prty_Map0

Bits	Name	Memory Access	Description
31:24	PSTQ3	R/W	Priorities Selected in Transmit Queue 3 This bit is similar to the PSTQ0 bit. **Value After Reset:** 0x0 **Exists:** DWC_EQOS_NUM_TXQ>3
23:16	PSTQ2	R/W	Priorities Selected in Transmit Queue 2 This bit is similar to the PSTQ0 bit. **Value After Reset:** 0x0 **Exists:** DWC_EQOS_NUM_TXQ>2
15:8	PSTQ1	R/W	Priorities Selected in Transmit Queue 1 This bit is similar to the PSTQ0 bit. **Value After Reset:** 0x0 **Exists:** DWC_EQOS_NUM_TXQ!=1
7:0	PSTQ0	R/W	Priorities Selected in Transmit Queue 0 This field holds the priorities assigned to Tx Queue 0 by the software. This field determines if Tx Queue 0 should be blocked from transmitting specified pause time when a PFC packet is received with priorities matching the priorities programmed in this field. If the content of this field is not mutually exclusive to corresponding fields of other Transmit queues, that is, same priority is mapped to multiple Tx queues, the MAC blocks all queues with matching priority for specified time. **Value After Reset:** 0x0 **Exists:** Always

17.1.26 MAC_TxQ_Prty_Map1

- **Description:** The Transmit Queue Priority Mapping 1 register contains the priority values assigned to Tx Queue 4 through Tx Queue 7.
- **Size:** 32 bits
- **Offset:** 0x9c
- **Exists:** (DWC_EQOS_DCB_EN&&(DWC_EQOS_NUM_TXQ>4))

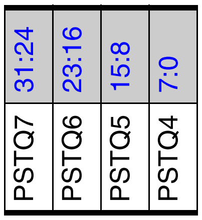


Table 17-30 **Fields for Register: MAC_TxQ_Prty_Map1**

Bits	Name	Memory Access	Description
31:24	PSTQ7	R/W	Priorities Selected in Transmit Queue 7 This bit is similar to the PSTQ4 bit. **Value After Reset:** 0x0 **Exists:** DWC_EQOS_NUM_TXQ>7
23:16	PSTQ6	R/W	Priorities Selected in Transmit Queue 6 This bit is similar to the PSTQ4 bit. **Value After Reset:** 0x0 **Exists:** DWC_EQOS_NUM_TXQ>6
15:8	PSTQ5	R/W	Priorities Selected in Transmit Queue 5 This bit is similar to the PSTQ4 bit. **Value After Reset:** 0x0 **Exists:** DWC_EQOS_NUM_TXQ>5
7:0	PSTQ4	R/W	Priorities Selected in Transmit Queue 4 This field holds the priorities assigned to Tx Queue 4 by the software. This field determines if Tx Queue 4 should be blocked from transmitting specified pause time when a PFC packet is received with priorities matching the priorities programmed in this field. If the content of this field is not mutually exclusive to corresponding fields of other Transmit queues, that is, same priority is mapped to multiple Tx queues, the MAC blocks all queues with matching priority for specified time. **Value After Reset:** 0x0 **Exists:** Always

17.1.27 MAC_RxQ_Ctrl0

- **Description:** The Receive Queue Control 0 register controls the queue management in the MAC Receiver.**Note:** In multiple Rx queues configuration, all the queues are disabled by default. Enable the Rx queue by programming the corresponding field in this register.
- **Size:** 32 bits
- **Offset:** 0xa0
- **Exists:** (DWC_EQOS_NUM_RXQ>1)

31:16	15:14	13:12	11:10	9:8	7:6	5:4	3:2	1:0
Reserved_31_16	RXQ7EN	RXQ6EN	RXQ5EN	RXQ4EN	RXQ3EN	RXQ2EN	RXQ1EN	RXQ0EN

Table 17-31 Fields for Register: MAC_RxQ_Ctrl0

Bits	Name	Memory Access	Description
31:16	Reserved_31_16	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
15:14	RXQ7EN	R/W	Receive Queue 7 Enable This field is similar to the RXQ0EN field. **Values:** ■ 0x0 (DISABLE): Queue not enabled ■ 0x1 (EN_AV): Queue enabled for AV ■ 0x2 (EN_DCB_GEN): Queue enabled for DCB/Generic ■ 0x3 (RSVD): Reserved **Value After Reset:** 0x0 **Exists:** DWC_EQOS_NUM_RXQ==8
13:12	RXQ6EN	R/W	Receive Queue 6 Enable This field is similar to the RXQ0EN field. **Values:** ■ 0x0 (DISABLE): Queue not enabled ■ 0x1 (EN_AV): Queue enabled for AV ■ 0x2 (EN_DCB_GEN): Queue enabled for DCB/Generic ■ 0x3 (RSVD): Reserved **Value After Reset:** 0x0 **Exists:** DWC_EQOS_NUM_RXQ>=7

Bits	Name	Memory Access	Description
11:10	RXQ5EN	R/W	Receive Queue 5 Enable This field is similar to the RXQ0EN field. **Values:** ■ 0x0 (DISABLE): Queue not enabled ■ 0x1 (EN_AV): Queue enabled for AV ■ 0x2 (EN_DCB_GEN): Queue enabled for DCB/Generic ■ 0x3 (RSVD): Reserved **Value After Reset:** 0x0 **Exists:** DWC_EQOS_NUM_RXQ>=6
9:8	RXQ4EN	R/W	Receive Queue 4 Enable This field is similar to the RXQ0EN field. **Values:** ■ 0x0 (DISABLE): Queue not enabled ■ 0x1 (EN_AV): Queue enabled for AV ■ 0x2 (EN_DCB_GEN): Queue enabled for DCB/Generic ■ 0x3 (RSVD): Reserved **Value After Reset:** 0x0 **Exists:** DWC_EQOS_NUM_RXQ>=5
7:6	RXQ3EN	R/W	Receive Queue 3 Enable This field is similar to the RXQ0EN field. **Values:** ■ 0x0 (DISABLE): Queue not enabled ■ 0x1 (EN_AV): Queue enabled for AV ■ 0x2 (EN_DCB_GEN): Queue enabled for DCB/Generic ■ 0x3 (RSVD): Reserved **Value After Reset:** 0x0 **Exists:** DWC_EQOS_NUM_RXQ>=4
5:4	RXQ2EN	R/W	Receive Queue 2 Enable This field is similar to the RXQ0EN field. **Values:** ■ 0x0 (DISABLE): Queue not enabled ■ 0x1 (EN_AV): Queue enabled for AV ■ 0x2 (EN_DCB_GEN): Queue enabled for DCB/Generic ■ 0x3 (RSVD): Reserved **Value After Reset:** 0x0 **Exists:** DWC_EQOS_NUM_RXQ>=3

Bits	Name	Memory Access	Description
3:2	RXQ1EN	R/W	Receive Queue 1 Enable This field is similar to the RXQ0EN field. **Values:** ■ 0x0 (DISABLE): Queue not enabled ■ 0x1 (EN_AV): Queue enabled for AV ■ 0x2 (EN_DCB_GEN): Queue enabled for DCB/Generic ■ 0x3 (RSVD): Reserved **Value After Reset:** 0x0 **Exists:** DWC_EQOS_NUM_RXQ!=1
1:0	RXQ0EN	R/W	Receive Queue 0 Enable This field indicates whether Rx Queue 0 is enabled for AV or DCB. **Values:** ■ 0x0 (DISABLE): Queue not enabled ■ 0x1 (EN_AV): Queue enabled for AV ■ 0x2 (EN_DCB_GEN): Queue enabled for DCB/Generic ■ 0x3 (RSVD): Reserved **Value After Reset:** 0x0 **Exists:** Always

17.1.28 MAC_RxQ_Ctrl1

- **Description:** The Receive Queue Control 1 register controls the routing of multicast, broadcast, AV, DCB, and untagged packets to the Rx queues.
- **Size:** 32 bits
- **Offset:** 0xa4
- **Exists:** (DWC_EQOS_NUM_RXQ>1)

31:y	23	26:24	x:22	21	20	19	18:16	15	14:12	11	10:8	7	6:4	3	2:0
Reserved_31_y	Rsvd	FPRQ	TPQC	TACPQE	MCBCQEN	Reserved_19	MCBCQ	Reserved_15	UPQ	Reserved_11	DCBCPQ	Reserved_7	PTPQ	Reserved_3	AVCPQ

Table 17-32 Fields for Register: MAC_RxQ_Ctrl1

Bits	Name	Memory Access	Description
31:y	Reserved_31_y	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
23			**Reserved Field:** Yes
26:24	FPRQ	R/W	Frame Preemption Residue Queue This field holds the Rx queue number to which the residual preemption frames must be forwarded. Preemption frames that are tagged and pass the SA/DA/VLAN filtering are routed based on PSRQ and all other frames are treated as residual frames and is routed to the queue number mentioned in this field. The Queue-0 is used as a default queue for express frames, so this field cannot be programmed to a value 0. **Value After Reset:** 0x0 **Exists:** DWC_EQOS_FPE

Bits	Name	Memory Access	Description
x:22	TPQC	R/W	Tagged PTP over Ethernet Packets Queuing Control. This field controls the routing of the VLAN Tagged PTPoE packets. If DWC_EQOS_AV_ENABLE is selected in the configuration, the following programmable options are allowed. ■ 2'b00: VLAN Tagged PTPoE packets are routed as generic VLAN Tagged packet (based on PSRQ for only non-AV enabled Rx Queues). ■ 2'b01: VLAN Tagged PTPoE packets are routed to Rx Queue specified by PTPQ field (That Rx Queue can be enabled for AV or non-AV traffic). ■ 2'b10: VLAN Tagged PTPoE packets are routed to only AV enabled Rx Queues based on PSRQ. ■ 2'b11: Reserved If DWC_EQOS_AV_ENABLE is not selected in the configuration, the following programmable options are allowed. ■ 1'b0: VLAN Tagged PTPoE packets are routed as generic VLAN Tagged packet (based on PSRQ for DCB/Generic enabled Rx Queues). ■ 1'b1: VLAN Tagged PTPoE packets are routed to Rx Queues specified by PTPQ field. **Value After Reset:** 0x0 **Exists:** DWC_EQOS_TIME_STAMPING **Range Variable[x]:** DWC_EQOS_AV_ENABLE?2:1 + 21
21	TACPQE	R/W	Tagged AV Control Packets Queuing Enable. When set, the MAC routes the received Tagged AV Control packets to the Rx queue specified by AVCPQ field. When reset, the MAC routes the received Tagged AV Control packets based on the tag priority matching the PSRQ fields in MAC_RxQ_Ctrl2 and MAC_RxQ_Ctrl3 registers. **Values:** ■ 0x0 (DISABLE): Tagged AV Control Packets Queuing is disabled ■ 0x1 (ENABLE): Tagged AV Control Packets Queuing is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_AV_ENABLE

Bits	Name	Memory Access	Description
20	MCBCQEN	R/W	Multicast and Broadcast Queue Enable This bit specifies that Multicast or Broadcast packets routing to the Rx Queue is enabled and the Multicast or Broadcast packets must be routed to Rx Queue specified in MCBCQ field. **Values:** ■ 0x0 (DISABLE): Multicast and Broadcast Queue is disabled ■ 0x1 (ENABLE): Multicast and Broadcast Queue is enabled **Value After Reset:** 0x0 **Exists:** Always
19	Reserved_19	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
18:16	MCBCQ	R/W	Multicast and Broadcast Queue This field specifies the Rx Queue onto which Multicast or Broadcast Packets are routed. Any Rx Queue enabled for Generic/DCB/AV traffic can be used to route the Multicast or Broadcast Packets. **Values:** ■ 0x0 (QUEUE0): Receive Queue 0 ■ 0x1 (QUEUE1): Receive Queue 1 ■ 0x2 (QUEUE2): Receive Queue 2 ■ 0x3 (QUEUE3): Receive Queue 3 ■ 0x4 (QUEUE4): Receive Queue 4 ■ 0x5 (QUEUE5): Receive Queue 5 ■ 0x6 (QUEUE6): Receive Queue 6 ■ 0x7 (QUEUE7): Receive Queue 7 **Value After Reset:** 0x0 **Exists:** Always
15	Reserved_15	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
14:12	UPQ	R/W	Untagged Packet Queue This field indicates the Rx Queue to which Untagged Packets are to be routed. Any Rx Queue enabled for Generic/DCB/AV traffic can be used to route the Untagged Packets. **Values:** ■ 0x0 (QUEUE0): Receive Queue 0 ■ 0x1 (QUEUE1): Receive Queue 1 ■ 0x2 (QUEUE2): Receive Queue 2 ■ 0x3 (QUEUE3): Receive Queue 3 ■ 0x4 (QUEUE4): Receive Queue 4 ■ 0x5 (QUEUE5): Receive Queue 5 ■ 0x6 (QUEUE6): Receive Queue 6 ■ 0x7 (QUEUE7): Receive Queue 7 **Value After Reset:** 0x0 **Exists:** Always
11	Reserved_11	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
10:8	DCBCPQ	R/W	DCB Control Packets Queue This field specifies the Rx queue on which the received DCB control packets are routed. The DCB data packets are routed based on the PSRQ field of the Transmit Flow Control Register of corresponding queue. **Values:** ■ 0x0 (QUEUE0): Receive Queue 0 ■ 0x1 (QUEUE1): Receive Queue 1 ■ 0x2 (QUEUE2): Receive Queue 2 ■ 0x3 (QUEUE3): Receive Queue 3 ■ 0x4 (QUEUE4): Receive Queue 4 ■ 0x5 (QUEUE5): Receive Queue 5 ■ 0x6 (QUEUE6): Receive Queue 6 ■ 0x7 (QUEUE7): Receive Queue 7 **Value After Reset:** 0x0 **Exists:** DWC_EQOS_DCB_EN
7	Reserved_7	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
6:4	PTPQ	R/W	PTP Packets Queue This field specifies the Rx queue on which the PTP packets sent over the Ethernet payload (not over IPv4 or IPv6) are routed. When the AV8021ASMEN bit of MAC_Timestamp_Control register is set, only untagged PTP over Ethernet packets are routed on an Rx Queue. If the bit is not set, then based on programming of TPQC field, both tagged and untagged PTPoE packets can be routed to this Rx Queue. **Values:** ■ 0x0 (QUEUE0): Receive Queue 0 ■ 0x1 (QUEUE1): Receive Queue 1 ■ 0x2 (QUEUE2): Receive Queue 2 ■ 0x3 (QUEUE3): Receive Queue 3 ■ 0x4 (QUEUE4): Receive Queue 4 ■ 0x5 (QUEUE5): Receive Queue 5 ■ 0x6 (QUEUE6): Receive Queue 6 ■ 0x7 (QUEUE7): Receive Queue 7 **Value After Reset:** 0x0 **Exists:** DWC_EQOS_TIME_STAMPING
3	Reserved_3	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
2:0	AVCPQ	R/W	AV Untagged Control Packets Queue This field specifies the Receive queue on which the received AV tagged and untagged control packets are routed. The AV tagged (when TACPQE bit is set) and untagged control packets are routed to Receive queue specified by this field. **Values:** ■ 0x0 (QUEUE0): Receive Queue 0 ■ 0x1 (QUEUE1): Receive Queue 1 ■ 0x2 (QUEUE2): Receive Queue 2 ■ 0x3 (QUEUE3): Receive Queue 3 ■ 0x4 (QUEUE4): Receive Queue 4 ■ 0x5 (QUEUE5): Receive Queue 5 ■ 0x6 (QUEUE6): Receive Queue 6 ■ 0x7 (QUEUE7): Receive Queue 7 **Value After Reset:** 0x0 **Exists:** DWC_EQOS_AV_ENABLE

17.1.29 MAC_RxQ_Ctrl2

- **Description:** This register controls the routing of tagged packets based on the USP (user Priority) field of the received packets to the RxQueues 0 to 3.
- **Size:** 32 bits
- **Offset:** 0xa8
- **Exists:** (DWC_EQOS_NUM_RXQ>1)

31:24	23:16	15:8	7:0
PSRQ3	PSRQ2	PSRQ1	PSRQ0

Table 17-33 Fields for Register: MAC_RxQ_Ctrl2

Bits	Name	Memory Access	Description
31:24	PSRQ3	R/W	Priorities Selected in the Receive Queue 3 This field decides the priorities assigned to Rx Queue 3. All packets with priorities that match the values set in this field are routed to Rx Queue 3. For example, if PSRQ3[6, 3] are set, packets with USP field equal to 3 or 6 are routed to Rx Queue 3. The software must ensure that the content of this field is mutually exclusive to the PSRQ fields for other queues, that is, the same priority is not mapped to multiple Rx queues. When the DCB feature is selected, this field also determines the priorities to be included in the PFC packet sent to remote station when Rx Queue 3 crosses the flow control threshold settings. **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_NUM_RXQ>3)

Bits	Name	Memory Access	Description
23:16	PSRQ2	R/W	Priorities Selected in the Receive Queue 2 This field decides the priorities assigned to Rx Queue 2. All packets with priorities that match the values set in this field are routed to Rx Queue 2. For example, if PSRQ2[1, 0] are set, packets with USP field equal to 1 or 0 are routed to Rx Queue 2. The software must ensure that the content of this field is mutually exclusive to the PSRQ fields for other queues, that is, the same priority is not mapped to multiple Rx queues. When the DCB feature is selected, this field also determines the priorities to be included in the PFC packet sent to remote station when Rx Queue 2 crosses the flow control threshold settings. **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_NUM_RXQ>2)
15:8	PSRQ1	R/W	Priorities Selected in the Receive Queue 1 This field decides the priorities assigned to Rx Queue 1. All packets with priorities that match the values set in this field are routed to Rx Queue 1. For example, if PSRQ1[4] is set, packets with USP field equal to 4 are routed to Rx Queue 1. The software must ensure that the content of this field is mutually exclusive to the PSRQ fields for other queues, that is, the same priority is not mapped to multiple Rx queues. When the DCB feature is selected, this field also determines the priorities to be included in the PFC packet sent to remote station when Rx Queue 1 crosses the flow control threshold settings. **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_NUM_RXQ>1)
7:0	PSRQ0	R/W	Priorities Selected in the Receive Queue 0 This field decides the priorities assigned to Rx Queue 0. All packets with priorities that match the values set in this field are routed to Rx Queue 0. For example, if PSRQ0[5] is set, packets with USP field equal to 5 are routed to Rx Queue 0. The software must ensure that the content of this field is mutually exclusive to the PSRQ fields for other queues, that is, the same priority is not mapped to multiple Rx queues. When the DCB feature is selected, this field also determines the priorities to be included in the PFC packet sent to remote station when Rx Queue 0 crosses the flow control threshold settings. **Value After Reset:** 0x0 **Exists:** Always

17.1.30 MAC_RxQ_Ctrl3

- **Description:** This register controls the routing of tagged packets based on the USP (user Priority) field of the received packets to the RxQueues 4 to 7.
- **Size:** 32 bits
- **Offset:** 0xac
- **Exists:** (DWC_EQOS_NUM_RXQ>4)

31:24	23:16	15:8	7:0
PSRQ7	PSRQ6	PSRQ5	PSRQ4

Table 17-34 Fields for Register: MAC_RxQ_Ctrl3

Bits	Name	Memory Access	Description
31:24	PSRQ7	R/W	Priorities Selected in the Receive Queue 7 This field decides the priorities assigned to Rx Queue 7. All packets with priorities that match the values set in this field are routed to Rx Queue 7. For example, if PSRQ7[7, 4] are set, packets with USP field equal to 7 or 4 are routed to Rx Queue 7. The software must ensure that the content of this field is mutually exclusive to the PSRQ fields for other queues, that is, the same priority is not mapped to multiple Rx queues. When the DCB feature is selected, this field also determines the priorities to be included in the PFC packet sent to remote station when Rx Queue 7 crosses the flow control threshold settings. **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_NUM_RXQ>7)

Bits	Name	Memory Access	Description
23:16	PSRQ6	R/W	Priorities Selected in the Receive Queue 6 This field decides the priorities assigned to Rx Queue 6. All packets with priorities that match the values set in this field are routed to Rx Queue 6. For example, if PSRQ6[5] are set, packets with USP field equal to 5 are routed to Rx Queue 6. The software must ensure that the content of this field is mutually exclusive to the PSRQ fields for other queues, that is, the same priority is not mapped to multiple Rx queues. When the DCB feature is selected, this field also determines the priorities to be included in the PFC packet sent to remote station when Rx Queue 6 crosses the flow control threshold settings. **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_NUM_RXQ>6)
15:8	PSRQ5	R/W	Priorities Selected in the Receive Queue 5 This field decides the priorities assigned to Rx Queue 5. All packets with priorities that match the values set in this field are routed to Rx Queue 5. For example, if PSRQ5[6] is set, packets with USP field equal to 6 are routed to Rx Queue 5. The software must ensure that the content of this field is mutually exclusive to the PSRQ fields for other queues, that is, the same priority is not mapped to multiple Rx queues. When the DCB feature is selected, this field also determines the priorities to be included in the PFC packet sent to remote station when Rx Queue 5 crosses the flow control threshold settings. **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_NUM_RXQ>5)
7:0	PSRQ4	R/W	Priorities Selected in the Receive Queue 4 This field decides the priorities assigned to Rx Queue 4. All packets with priorities that match the values set in this field are routed to Rx Queue 4. For example, if PSRQ4[7:4] is set, packets with USP field equal to 7, 6, 5, or 4 are routed to Rx Queue 4. The software must ensure that the content of this field is mutually exclusive to the PSRQ fields for other queues, that is, the same priority is not mapped to multiple Rx queues. When the DCB feature is selected, this field also determines the priorities to be included in the PFC packet sent to remote station when Rx Queue 4 crosses the flow control threshold settings. **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_NUM_RXQ>4)

17.1.31 MAC_Interrupt_Status

- **Description:** The Interrupt Status register contains the status of interrupts.
- **Size:** 32 bits
- **Offset:** 0xb0
- **Exists:** Always

31:21	20	19	18	17	16	15	14	13	12	11	10	9	8	7:6	5	4	3	2	1	0
Reserved_31_21	MFRIS	MFTIS	MDIOIS	FPEIS	Reserved_16	GPIIS	RXSTSIS	TXSTSIS	TSIS	MMCRXIPIS	MMCTXIS	MMCRXIS	MMCIS	Reserved_7_6	LPIIS	PMTIS	PHYIS	PCSANCIS	PCSLCHGIS	RGSMIIIS

Table 17-35 Fields for Register: MAC_Interrupt_Status

Bits	Name	Memory Access	Description
31:21	Reserved_31_21	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
20	MFRIS	R	MMC FPE Receive Interrupt Status This bit is set high when an interrupt is generated in the MMC FPE Receive Interrupt Register. This bit is cleared when all bits in this interrupt register are cleared. This bit is valid only when you select the Enable MAC Management Counters (MMC) option along with FPE support. **Values:** ■ 0x0 (INACTIVE): MMC FPE Receive Interrupt status not active ■ 0x1 (ACTIVE): MMC FPE Receive Interrupt status active **Value After Reset:** 0x0 **Exists:** DWC_EQOS_FPE&&DWC_EQOS_MMC_FPE_EN

Bits	Name	Memory Access	Description
19	MFTIS	R	MMC FPE Transmit Interrupt Status This bit is set high when an interrupt is generated in the MMC FPE Transmit Interrupt Register. This bit is cleared when all bits in this interrupt register are cleared. This bit is valid only when you select the Enable MAC Management Counters (MMC) option along with FPE support. **Values:** ■ 0x0 (INACTIVE): MMC FPE Transmit Interrupt status not active ■ 0x1 (ACTIVE): MMC FPE Transmit Interrupt status active **Value After Reset:** 0x0 **Exists:** DWC_EQOS_FPE&&DWC_EQOS_MMC_FPE_EN
18	MDIOIS	R	MDIO Interrupt Status This bit indicates an interrupt event after the completion of MDIO operation. To reset this bit, the application has to read this bit/Write 1 to this bit when RCWE bit of MAC_CSR_SW_Ctrl register is set. Access restriction applies. Clears on read (or write of 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): MDIO Interrupt status not active ■ 0x1 (ACTIVE): MDIO Interrupt status active **Value After Reset:** 0x0 **Exists:** DWC_EQOS_SMA_EN||DWC_EQOS_REVMII_EN
17	FPEIS	R	Frame Preemption Interrupt Status This bit indicates an interrupt event during the operation of Frame Preemption (Bits[19:16] of MAC_FPE_CTRL_STS register is set). To reset this bit, the application must clear the event in MAC_FPE_CTRL_STS that has caused the Interrupt. **Values:** ■ 0x0 (INACTIVE): Frame Preemption Interrupt status not active ■ 0x1 (ACTIVE): Frame Preemption Interrupt status active **Value After Reset:** 0x0 **Exists:** DWC_EQOS_FPE
16	Reserved_16	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
15	GPIIS	R	GPI Interrupt Status When the GPIO feature is enabled, this bit is set when any active event (LL or LH) occurs on the GPIS field of the MAC_GPIO_Status register and the corresponding GPIE bit is enabled in the MAC_GPIO_Control register. This bit is cleared on reading lane 0 (GPIS) of the MAC_GPIO_Status register. **Values:** ■ 0x0 (INACTIVE): GPI Interrupt status not active ■ 0x1 (ACTIVE): GPI Interrupt status active **Value After Reset:** 0x0 **Exists:** DWC_EQOS_GPIO_EN
14	RXSTSIS	R	Receive Status Interrupt This bit indicates the status of received packets. This bit is set when the RWT bit is set in the MAC_Rx_Tx_Status register. This bit is cleared when the corresponding interrupt source bit is read (or corresponding interrupt source bit is written to 1 when RCWE bit of MAC_CSR_SW_Ctrl register is set) in the MAC_Rx_Tx_Status register. **Values:** ■ 0x0 (INACTIVE): Receive Interrupt status not active ■ 0x1 (ACTIVE): Receive Interrupt status active **Value After Reset:** 0x0 **Exists:** Always
13	TXSTSIS	R	Transmit Status Interrupt This bit indicates the status of transmitted packets. This bit is set when any of the following bits is set in the MAC_Rx_Tx_Status register: ■ Excessive Collision (EXCOL) ■ Late Collision (LCOL) ■ Excessive Deferral (EXDEF) ■ Loss of Carrier (LCARR) ■ No Carrier (NCARR) ■ Jabber Timeout (TJT) This bit is cleared when the corresponding interrupt source bit is read (or corresponding interrupt source bit is written to 1 when RCWE bit of MAC_CSR_SW_Ctrl register is set) in the MAC_Rx_Tx_Status register. **Values:** ■ 0x0 (INACTIVE): Transmit Interrupt status not active ■ 0x1 (ACTIVE): Transmit Interrupt status active **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
12	TSIS	R	Timestamp Interrupt Status If the Timestamp feature is enabled, this bit is set when any of the following conditions is true: ■ The system time value is equal to or exceeds the value specified in the Target Time High and Low registers. ■ There is an overflow in the Seconds register. ■ The Target Time Error occurred, that is, programmed target time already elapsed. If the Auxiliary Snapshot feature is enabled, this bit is set when the auxiliary snapshot trigger is asserted. In configurations other than EQOS_CORE, when drop transmit status is enabled in MTL, this bit is set when the captured transmit timestamp is updated in the MAC_Tx_Timestamp_Status_Nanoseconds and Mac_TxTimestamp_Status_Seconds registers. When PTP offload feature is enabled, this bit is set when the captured transmit timestamp is updated in the MAC_Tx_Timestamp_Status_Nanoseconds and MAC_Tx_Timestamp_Status_Seconds registers, for PTO generated Delay Request and Pdelay request packets. This bit is cleared when the corresponding interrupt source bit is read (or corresponding interrupt source bit is written to 1 when RCWE bit of MAC_CSR_SW_Ctrl register is set) in the MAC_Timestamp_Status register. **Values:** ■ 0x0 (INACTIVE): Timestamp Interrupt status not active ■ 0x1 (ACTIVE): Timestamp Interrupt status active **Value After Reset:** 0x0 **Exists:** ((DWC_EQOS_TIME_STAMPING&&(DWC_EQOS_SYSTIME_SOURCE!=1||!DWC_EQOS_CORE))||DWC_EQOS_ADV_TIME_AUX_SNAP||DWC_EQOS_PTO_EN)

Bits	Name	Memory Access	Description
11	MMCRXIPIS	R	MMC Receive Checksum Offload Interrupt Status This bit is set high when an interrupt is generated in the MMC Receive Checksum Offload Interrupt Register. This bit is cleared when all bits in this interrupt register are cleared. This bit is valid only when you select the Enable MAC Management Counters (MMC) and Enable Receive TCP/IP Checksum Check options. **Values:** ■ 0x0 (INACTIVE): MMC Receive Checksum Offload Interrupt status not active ■ 0x1 (ACTIVE): MMC Receive Checksum Offload Interrupt status active **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_IPC_RX_PKTS_ATLEAST_ONE||DWC_EQOS_MMC_IPC_RX_DGRAM_BCNT_ATLEAST_ONE||DWC_EQOS_MMC_IPC_RX_PAYLD_BCNT_ATLEAST_ONE
10	MMCTXIS	R	MMC Transmit Interrupt Status This bit is set high when an interrupt is generated in the MMC Transmit Interrupt Register. This bit is cleared when all bits in this interrupt register are cleared. This bit is valid only when you select the Enable MAC Management Counters (MMC) option. **Values:** ■ 0x0 (INACTIVE): MMC Transmit Interrupt status not active ■ 0x1 (ACTIVE): MMC Transmit Interrupt status active **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_MMC_TX_EN_ATLEAST_ONE||DWC_EQOS_MMC_TX_LPI_ATLEAST_ONE)
9	MMCRXIS	R	MMC Receive Interrupt Status This bit is set high when an interrupt is generated in the MMC Receive Interrupt Register. This bit is cleared when all bits in this interrupt register are cleared. This bit is valid only when you select the Enable MAC Management Counters (MMC) option. **Values:** ■ 0x0 (INACTIVE): MMC Receive Interrupt status not active ■ 0x1 (ACTIVE): MMC Receive Interrupt status active **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_MMC_RX_EN_ATLEAST_ONE||DWC_EQOS_MMC_RX_LPI_ATLEAST_ONE)

Bits	Name	Memory Access	Description
8	MMCIS	R	MMC Interrupt Status This bit is set high when Bit 11, Bit 10, or Bit 9 is set high. This bit is cleared only when all these bits are low. This bit is valid only when you select the Enable MAC Management Counters (MMC) option. **Values:** ■ 0x0 (INACTIVE): MMC Interrupt status not active ■ 0x1 (ACTIVE): MMC Interrupt status active **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_EN
7:6	Reserved_7_6	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
5	LPIIS	R	LPI Interrupt Status When the Energy Efficient Ethernet feature is enabled, this bit is set for any LPI state entry or exit in the MAC Transmitter or Receiver. This bit is cleared when the corresponding interrupt source bit of MAC_LPI_Control_Status register is read (or corresponding interrupt source bit of MAC_LPI_Control_Status register is written to 1 when RCWE bit of MAC_CSR_SW_Ctrl register is set). **Values:** ■ 0x0 (INACTIVE): LPI Interrupt status not active ■ 0x1 (ACTIVE): LPI Interrupt status active **Value After Reset:** 0x0 **Exists:** DWC_EQOS_EEE_EN
4	PMTIS	R	PMT Interrupt Status This bit is set when a Magic packet or Wake-on-LAN packet is received in the power-down mode (RWKPRCVD and MGKPRCVD bits in MAC_PMT_Control_Status register). This bit is cleared when corresponding interrupt source bit are cleared because of a Read operation to the MAC_PMT_Control_Status register (or corresponding interrupt source bit of MAC_PMT_Control_Status register is written to 1 when RCWE bit of MAC_CSR_SW_Ctrl register is set). This bit is valid only when you select the Enable Power Management option. **Values:** ■ 0x0 (INACTIVE): PMT Interrupt status not active ■ 0x1 (ACTIVE): PMT Interrupt status active **Value After Reset:** 0x0 **Exists:** DWC_EQOS_PMT_EN

Bits	Name	Memory Access	Description
3	PHYIS	R	PHY Interrupt This bit is set when rising edge is detected on the phy_intr_i input. This bit is cleared when this register is read (or this bit is written to 1 when RCWE bit of MAC_CSR_SW_Ctrl register is set). **Values:** ■ 0x0 (INACTIVE): PHY Interrupt not detected ■ 0x1 (ACTIVE): PHY Interrupt detected **Value After Reset:** 0x0 **Exists:** Always
2	PCSANCIS	R	PCS Auto-Negotiation Complete This bit is set when auto-negotiation is completed in the TBI, RTBI, or SGMII PHY interface (see ANC bit in the MAC_AN_Status register). This bit is cleared when the MAC_AN_Status register is read (or ANC bit of MAC_AN_Status register is written to 1 when RCWE bit of MAC_CSR_SW_Ctrl register is set). This bit is valid only when you select the optional TBI, RTBI, or SGMII PHY interface. **Values:** ■ 0x0 (INACTIVE): PCS Auto-Negotiation has not Completed ■ 0x1 (ACTIVE): PCS Auto-Negotiation Completed **Value After Reset:** 0x0 **Exists:** DWC_EQOS_TBI_EN||DWC_EQOS_SGMII_EN||DWC_EQOS_RTBI_EN
1	PCSLCHGIS	R	PCS Link Status Changed This bit is set because of any change in Link Status in the TBI, RTBI, or SGMII PHY interface (See LS bit in MAC_AN_Status register). This bit is cleared when the MAC_AN_Status register is read (or LS bit of MAC_AN_Status register is written to 1 when RCWE bit of MAC_CSR_SW_Ctrl register is set). This bit is valid only when you select the optional TBI, RTBI, or SGMII PHY interface. **Values:** ■ 0x0 (INACTIVE): PCS Link Status has not Changed ■ 0x1 (ACTIVE): PCS Link Status Changed **Value After Reset:** 0x0 **Exists:** DWC_EQOS_TBI_EN||DWC_EQOS_SGMII_EN||DWC_EQOS_RTBI_EN **Testable:** untestable

Bits	Name	Memory Access	Description
0	RGSMIIIS	R	RGMII or SMII Interrupt Status This bit is set because of any change in value of the Link Status of RGMII or SMII interface (LNKSTS bit in MAC_PHYIF_Control_Status register). This bit is cleared when the MAC_PHYIF_Control_Status register is read (or LNKSTS bit of MAC_PHYIF_Control_Status register is written to 1 when RCWE bit of MAC_CSR_SW_Ctrl register is set). This bit is valid only when you select the optional RGMII or SMII PHY interface. **Values:** ■ 0x0 (INACTIVE): RGMII or SMII Interrupt Status is not active ■ 0x1 (ACTIVE): RGMII or SMII Interrupt Status is active **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_RGMII_EN||DWC_EQOS_SMII_EN) **Testable:** untestable

17.1.32 MAC_Interrupt_Enable

- **Description:** The Interrupt Enable register contains the masks for generating the interrupts.
- **Size:** 32 bits
- **Offset:** 0xb4
- **Exists:** Always

31:19	18	17	16	15	14	13	12	11:6	5	4	3	2	1	0
Reserved_31_19	MDIOIE	FPEIE	Reserved_16	Reserved_15	RXSTSIE	TXSTSIE	TSIE	Reserved_11_6	LPIIE	PMTIE	PHYIE	PCSANCIE	PCSLCHGIE	RGSMIIIE

Table 17-36 Fields for Register: MAC_Interrupt_Enable

Bits	Name	Memory Access	Description
31:19	Reserved_31_19	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
18	MDIOIE	R/W	MDIO Interrupt Enable When this bit is set, it enables the assertion of the interrupt when MDIOIS field is set in the MAC_Interrupt_Status register. **Values:** ■ 0x0 (DISABLE): MDIO Interrupt is disabled ■ 0x1 (ENABLE): MDIO Interrupt is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_SMA_EN||DWC_EQOS_REVMII_EN
17	FPEIE	R/W	Frame Preemption Interrupt Enable When this bit is set, it enables the assertion of the interrupt when FPEIS field is set in the MAC_Interrupt_Status register. **Values:** ■ 0x0 (DISABLE): Frame Preemption Interrupt is disabled ■ 0x1 (ENABLE): Frame Preemption Interrupt is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_FPE
16	Reserved_16	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
15	Reserved_15	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
14	RXSTSIE	R/W	Receive Status Interrupt Enable When this bit is set, it enables the assertion of the interrupt signal because of the setting of RXSTSIS bit in the MAC_Interrupt_Status register. **Values:** ■ 0x0 (DISABLE): Receive Status Interrupt is disabled ■ 0x1 (ENABLE): Receive Status Interrupt is enabled **Value After Reset:** 0x0 **Exists:** Always
13	TXSTSIE	R/W	Transmit Status Interrupt Enable When this bit is set, it enables the assertion of the interrupt signal because of the setting of TXSTSIS bit in the MAC_Interrupt_Status register. **Values:** ■ 0x0 (DISABLE): Timestamp Status Interrupt is disabled ■ 0x1 (ENABLE): Timestamp Status Interrupt is enabled **Value After Reset:** 0x0 **Exists:** Always
12	TSIE	R/W	Timestamp Interrupt Enable When this bit is set, it enables the assertion of the interrupt signal because of the setting of TSIS bit in MAC_Interrupt_Status register. **Values:** ■ 0x0 (DISABLE): Timestamp Interrupt is disabled ■ 0x1 (ENABLE): Timestamp Interrupt is enabled **Value After Reset:** 0x0 **Exists:** ((DWC_EQOS_TIME_STAMPING&&(DWC_EQOS_SYSTIME_SOURCE!=1||!DWC_EQOS_CORE))||DWC_EQOS_ADV_TIME_AUX_SNAP||DWC_EQOS_PTO_EN)
11:6	Reserved_11_6	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
5	LPIIE	R/W	LPI Interrupt Enable When this bit is set, it enables the assertion of the interrupt signal because of the setting of LPIIS bit in MAC_Interrupt_Status register. **Values:** ■ 0x0 (DISABLE): LPI Interrupt is disabled ■ 0x1 (ENABLE): LPI Interrupt is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_EEE_EN
4	PMTIE	R/W	PMT Interrupt Enable When this bit is set, it enables the assertion of the interrupt signal because of the setting of PMTIS bit in MAC_Interrupt_Status register. **Values:** ■ 0x0 (DISABLE): PMT Interrupt is disabled ■ 0x1 (ENABLE): PMT Interrupt is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_PMT_EN
3	PHYIE	R/W	PHY Interrupt Enable When this bit is set, it enables the assertion of the interrupt signal because of the setting of PHYIS bit in MAC_Interrupt_Status register. **Values:** ■ 0x0 (DISABLE): PHY Interrupt is disabled ■ 0x1 (ENABLE): PHY Interrupt is enabled **Value After Reset:** 0x0 **Exists:** Always
2	PCSANCIE	R/W	PCS AN Completion Interrupt Enable When this bit is set, it enables the assertion of the interrupt signal because of the setting of the PCSANCIS bit in MAC_Interrupt_Status register. **Values:** ■ 0x0 (DISABLE): PCS AN Completion Interrupt is disabled ■ 0x1 (ENABLE): PCS AN Completion Interrupt is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_TBI_EN||DWC_EQOS_SGMII_EN||DWC_EQOS_RTBI_EN

Bits	Name	Memory Access	Description
1	PCSLCHGIE	R/W	PCS Link Status Interrupt Enable When this bit is set, it enables the assertion of the interrupt signal because of the setting of the PCSLCHGIS bit in MAC_Interrupt_Status register. **Values:** ■ 0x0 (DISABLE): PCS Link Status Interrupt is disabled ■ 0x1 (ENABLE): PCS Link Status Interrupt is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_TBI_EN||DWC_EQOS_SGMII_EN||DWC_EQOS_RTBI_EN
0	RGSMIIIE	R/W	RGMII or SMII Interrupt Enable When this bit is set, it enables the assertion of the interrupt signal because of the setting of RGSMIIIS bit in MAC_Interrupt_Status register. **Values:** ■ 0x0 (DISABLE): RGMII or SMII Interrupt is disabled ■ 0x1 (ENABLE): RGMII or SMII Interrupt is enabled **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_RGMII_EN||DWC_EQOS_SMII_EN)

17.1.33 MAC_Rx_Tx_Status

- **Description:** The Receive Transmit Status register contains the Receive and Transmit Error status.
- **Size:** 32 bits
- **Offset:** 0xb8
- **Exists:** Always

31:9	8	7:6	5	4	3	2	1	0
Reserved_31_9	RWT	Reserved_7_6	EXCOL	LCOL	EXDEF	LCARR	NCARR	TJT

Table 17-37 Fields for Register: MAC_Rx_Tx_Status

Bits	Name	Memory Access	Description
31:9	Reserved_31_9	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
8	RWT	R	Receive Watchdog Timeout This bit is set when a packet with length greater than 2,048 bytes is received (10, 240 bytes when Jumbo Packet mode is enabled) and the WD bit is reset in the MAC_Configuration register. This bit is set when a packet with length greater than 16,383 bytes is received and the WD bit is set in the MAC_Configuration register. Access restriction applies. Clears on read (or write of 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): No receive watchdog timeout ■ 0x1 (ACTIVE): Receive watchdog timed out **Value After Reset:** 0x0 **Exists:** Always
7:6	Reserved_7_6	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
5	EXCOL	R	Excessive Collisions When the DTXSTS bit is set in the MTL_Operation_Mode register, this bit indicates that the transmission aborted after 16 successive collisions while attempting to transmit the current packet. If the DR bit is set in the MAC_Configuration register, this bit is set after the first collision and the packet transmission is aborted. Access restriction applies. Clears on read (or write of 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): No collision ■ 0x1 (ACTIVE): Excessive collision is sensed **Value After Reset:** 0x0 **Exists:** !DWC_EQOS_FDUPLX_ONLY&&!DWC_EQOS_CORE
4	LCOL	R	Late Collision When the DTXSTS bit is set in the MTL_Operation_Mode register, this bit indicates that the packet transmission aborted because a collision occurred after the collision window (64 bytes including Preamble in MII mode; 512 bytes including Preamble and Carrier Extension in GMII mode). This bit is not valid if the Underflow error occurs. Access restriction applies. Clears on read (or write of 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): No collision ■ 0x1 (ACTIVE): Late collision is sensed **Value After Reset:** 0x0 **Exists:** !DWC_EQOS_FDUPLX_ONLY&&!DWC_EQOS_CORE

Bits	Name	Memory Access	Description
3	EXDEF	R	Excessive Deferral When the DTXSTS bit is set in the MTL_Operation_Mode register and the DC bit is set in the MAC_Configuration register, this bit indicates that the transmission ended because of excessive deferral of over 24,288 bit times (155,680 in 1000/2500 Mbps mode or when Jumbo packet is enabled). Access restriction applies. Clears on read (or write of 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): No Excessive deferral ■ 0x1 (ACTIVE): Excessive deferral **Value After Reset:** 0x0 **Exists:** !DWC_EQOS_FDUPLX_ONLY&&!DWC_EQOS_CORE
2	LCARR	R	Loss of Carrier When the DTXSTS bit is set in the MTL_Operation_Mode register, this bit indicates that the loss of carrier occurred during packet transmission, that is, the phy_crs_i signal was inactive for one or more transmission clock periods during packet transmission. This bit is valid only for packets transmitted without collision. Access restriction applies. Clears on read (or write of 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): Carrier is present ■ 0x1 (ACTIVE): Loss of carrier **Value After Reset:** 0x0 **Exists:** !DWC_EQOS_FDUPLX_ONLY&&!DWC_EQOS_CORE
1	NCARR	R	No Carrier When the DTXSTS bit is set in the MTL_Operation_Mode register, this bit indicates that the carrier signal from the PHY is not present at the end of preamble transmission. Access restriction applies. Clears on read (or write of 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): Carrier is present ■ 0x1 (ACTIVE): No carrier **Value After Reset:** 0x0 **Exists:** !DWC_EQOS_FDUPLX_ONLY&&!DWC_EQOS_CORE

Bits	Name	Memory Access	Description
0	TJT	R	Transmit Jabber Timeout This bit indicates that the Transmit Jabber Timer expired which happens when the packet size exceeds 2,048 bytes (10,240 bytes when the Jumbo packet is enabled) and JD bit is reset in the MAC_Configuration register. This bit is set when the packet size exceeds 16,383 bytes and the JD bit is set in the MAC_Configuration register. Access restriction applies. Clears on read (or write of 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): No Transmit Jabber Timeout ■ 0x1 (ACTIVE): Transmit Jabber Timeout occurred **Value After Reset:** 0x0 **Exists:** Always

17.1.34 MAC_PMT_Control_Status

- **Description:** The PMT Control and Status Register.
- **Size:** 32 bits
- **Offset:** 0xc0
- **Exists:** (DWC_EQOS_PMT_EN)

31	30:29	28:24	23:11	10	9	8:7	6	5	4:3	2	1	0
RWKFILTRST	Reserved_30_29	RWKPTR	Reserved_23_11	RWKPFE	GLBLUCAST	Reserved_8_7	RWKPRCVD	MGKPRCVD	Reserved_4_3	RWKPKTEN	MGKPKTEN	PWRDWN

Table 17-38 Fields for Register: MAC_PMT_Control_Status

Bits	Name	Memory Access	Description
31	RWKFILTRST	R/W	Remote Wake-Up Packet Filter Register Pointer Reset When this bit is set, the remote wake-up packet filter register pointer is reset to 3'b000. It is automatically cleared after 1 clock cycle. Access restriction applies. Setting 1 sets. Self-cleared. Setting 0 has no effect. **Values:** ■ 0x0 (DISABLE): Remote Wake-Up Packet Filter Register Pointer is not Reset ■ 0x1 (ENABLE): Remote Wake-Up Packet Filter Register Pointer is Reset **Value After Reset:** 0x0 **Exists:** DWC_EQOS_PMT_RWK_EN **Testable:** untestable
30:29	Reserved_30_29	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
28:24	RWKPTR	R	Remote Wake-up FIFO Pointer This field gives the current value (0 to 7, 15, or 31 when 4, 8, or 16 Remote Wake-up Packet Filters are selected) of the Remote Wake-up Packet Filter register pointer. When the value of this pointer is equal to maximum for the selected number of Remote Wake-up Packet Filters, the contents of the Remote Wake-up Packet Filter Register are transferred to the clk_rx_i domain when a Write occurs to that register. **Value After Reset:** 0x0 **Exists:** DWC_EQOS_PMT_RWK_EN **Testable:** untestable
23:11	Reserved_23_11	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
10	RWKPFE	R/W	Remote Wake-up Packet Forwarding Enable When this bit is set along with RWKPKTEN, the MAC receiver drops all received frames until it receives the expected Wake-up frame. All frames after that event including the received wake-up frame are forwarded to application. This bit is then self-cleared on receiving the wake-up packet. The application can also clear this bit before the expected wake-up frame is received. In such cases, the MAC reverts to the default behavior where packets received are forwarded to the application. This bit must only be set when RWKPKTEN is set high and PWRDWN is set low. The setting of this bit has no effect when PWRDWN is set high. **Note:** If Magic Packet Enable and Wake-Up Frame Enable are both set along with setting of this bit and Magic Packet is received prior to wake-up frame, this bit is self-cleared on receiving Magic Packet, the received Magic packet is dropped, and all frames after received Magic Packet are forwarded to application. Access restriction applies. Setting 1 sets. Self-cleared. Setting 0 has no effect. **Values:** ■ 0x0 (DISABLE): Remote Wake-up Packet Forwarding is disabled ■ 0x1 (ENABLE): Remote Wake-up Packet Forwarding is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_PMT_RWK_EN

Bits	Name	Memory Access	Description
9	GLBLUCAST	R/W	Global Unicast When this bit set, any unicast packet filtered by the MAC (DAF) address recognition is detected as a remote wake-up packet. **Values:** ■ 0x0 (DISABLE): Global unicast is disabled ■ 0x1 (ENABLE): Global unicast is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_PMT_RWK_EN
8:7	Reserved_8_7	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
6	RWKPRCVD	R	Remote Wake-Up Packet Received When this bit is set, it indicates that the power management event is generated because of the reception of a remote wake-up packet. This bit is cleared when this register is read. Access restriction applies. Clears on read (or write of 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): Remote wake-up packet is received ■ 0x1 (ACTIVE): Remote wake-up packet is received **Value After Reset:** 0x0 **Exists:** DWC_EQOS_PMT_RWK_EN
5	MGKPRCVD	R	Magic Packet Received When this bit is set, it indicates that the power management event is generated because of the reception of a magic packet. This bit is cleared when this register is read. Access restriction applies. Clears on read (or write of 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): No Magic packet is received ■ 0x1 (ACTIVE): Magic packet is received **Value After Reset:** 0x0 **Exists:** DWC_EQOS_PMT_MGK_EN
4:3	Reserved_4_3	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
2	RWKPKTEN	R/W	Remote Wake-Up Packet Enable When this bit is set, a power management event is generated when the MAC receives a remote wake-up packet. **Values:** ■ 0x0 (DISABLE): Remote wake-up packet is disabled ■ 0x1 (ENABLE): Remote wake-up packet is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_PMT_RWK_EN
1	MGKPKTEN	R/W	Magic Packet Enable When this bit is set, a power management event is generated when the MAC receives a magic packet. **Values:** ■ 0x0 (DISABLE): Magic Packet is disabled ■ 0x1 (ENABLE): Magic Packet is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_PMT_MGK_EN
0	PWRDWN	R/W	Power Down When this bit is set, the MAC receiver drops all received packets until it receives the expected magic packet or remote wake-up packet. This bit is then self-cleared and the power-down mode is disabled. The software can clear this bit before the expected magic packet or remote wake-up packet is received. The packets received by the MAC after this bit is cleared are forwarded to the application. This bit must only be set when the Magic Packet Enable, Global Unicast, or Remote Wake-Up Packet Enable bit is set high. **Note**: You can gate-off the CSR clock during the power-down mode. However, when the CSR clock is gated-off, you cannot perform any read or write operations on this register. Therefore, the Software cannot clear this bit. Access restriction applies. Setting 1 sets. Self-cleared. Setting 0 has no effect. **Values:** ■ 0x0 (DISABLE): Power down is disabled ■ 0x1 (ENABLE): Power down is enabled **Value After Reset:** 0x0 **Exists:** Always **Testable:** untestable

17.1.35 MAC_RWK_Packet_Filter

- **Description:** The Remote Wakeup Filter registers are implemented as 8, 16, or 32 indirect access registers (wkuppktfilter_reg#i) based on whether 4, 8, or 16 Remote Wakeup Filters are selected in the configuration and accessed by application through MAC_RWK_Packet_Filter register. When the Remote Wakeup Filters are to be programmed, the entire set of wkuppktfilter_reg registers must be written. The wkuppktfilter_reg register is programmed by sequentially writing the eight, sixteen or thirty-two register values in MAC_RWK_Packet_Filter register for wkuppktfilter_reg0, wkuppktfilter_reg1, ..., wkuppktfilter_reg31 respectively. The wkuppktfilter_reg register is read in a similar way. The MAC updates the wkuppktfilter_reg register current pointer value in RWKPTR field of MAC_PMT_Control_Status register.
  The Remote Wakeup Filters are arranged in blocks of 4 filters each and each such block have eight 32-bit wide registers, viz. wkuppktfilter_reg0-7, wkuppktfilter_reg8-15, wkuppktfilter_reg16-23 and wkuppktfilter_reg24-31. The fields of Remote Wakeup Filter are described as follows:
  Filter i Byte Mask: The filter i byte mask register defines the bytes of the packet that are examined by filter i (0, 1, 2, 3, .., 15) to determine whether or not a packet is a wake-up packet.

  - The MSB (31st bit) must be zero.
  - Bit j[30:0] is the byte mask.
  - If Bit j (byte number) of the byte mask is set, the CRC block processes the Filter i Offset + j of the incoming packet; otherwise Filter i Offset + j is ignored.

  Filter i Command: The 4-bit filter i command controls the filter i operation.

  - Bit 3 specifies the address type, defining the destination address type of the pattern. When the bit is set, the pattern applies to only multicast packets; when the bit is reset, the pattern applies only to unicast packet.
  - Bit 2 (Inverse Mode), when set, reverses the logic of the CRC16 hash function signal, to reject a packet with matching CRC-16 value. Bit 2, along with Bit 1, allows a MAC to reject a subset of remote wake-up packets by creating filter logic such as "Pattern 1 AND NOT Pattern 2".
  - Bit 1 (And_Previous) implements the Boolean logic. When set, the result of the current entry is logically ANDed with the result of the previous filter. This AND logic allows a filter pattern longer than 32 bytes by splitting the mask among two, three, or four filters. This depends on the number of filters that have the And_Previous bit set. The details are provided below:
    - The And_Previous bit setting is applicable within a set of 4 filters.
    - Setting of And_Previous bit of filter that is not enabled has no effect, that is setting And_Previous bit of lowest number filter in the set of 4 filters has no effect. For example, setting of And_Previous bit of Filter 0 has no effect.
    - If And_Previous bit is set for filter to form AND chained filter, the AND chain breaks at the point any filter is not enabled. For example: If Filter 2 And_Previous bit is set (bit 1 in Filter 2 command is set) but Filter 1 is not enabled (bit 0 in Filter 1 command is reset), then only Filter 2 result is considered. If Filter 2 And_Previous bit is set (bit 1 in Filter 2 command is set), Filter 3 And_Previous bit is set (bit 1 in Filter 3 command is set), but Filter 1 is not enabled (bit 0 in Filter 1 command is reset), then only Filter 2 result ANDed with Filter 3 result is considered. If Filter 2 And_Previous bit is set (bit 1 in Filter 2 command is set), Filter 3 And_Previous bit is set (bit 1 in Filter 3 command is set), but Filter 2 is not enabled (bit 0 in Filter 2 command is reset), then since setting of Filter 2 And_Previous bit has no effect only Filter 1 result ORed with Filter 3 result is considered.
    - If filters chained by And_Previous bit setting have complementary programming, then a frame may never pass the AND chained filter. For example, if Filter 2 And_Previous bit is set (bit 1 in

Filter 2 command is set), Filter 1 Address_Type bit is set (bit 3 in Filter 1 command is set) indicating multicast detection and Filter 2 Address_Type bit is reset (bit 3 in Filter 2 command is reset) indicating unicast detection or vice versa, a remote wakeup frame does not pass the AND chained filter as a remote wakeup frame cannot be of both unicast and multicast address type.

- Bit 0 is the enable for filter i. If Bit 0 is not set, filter i is disabled.

Filter i Offset: The filter i offset register defines the offset (within the packet) from which the filter i examines the packets.

- This 8-bit pattern-offset is the offset for the filter i first byte to be examined.
- The minimum allowed offset is 12, which refers to the 13th byte of the packet.
- The offset value 0 refers to the first byte of the packet.

Filter i CRC-16: The filter i CRC-16 register contains the CRC-16 value calculated from the pattern and the byte mask programmed in the Remote Wakeup filter register.

- The 16-bit CRC calculation uses the following polynomial: G(x) = x^16 + x^15 + x^2 + 1
- Each mask, used in the hash function calculation, is compared with a 16-bit value associated with that mask. Each filter has the following:
  - 32-bit Mask: Each bit in this mask corresponds to one byte in the detected packet. If the bit is 1, the corresponding byte is taken into the CRC-16 calculation.
  - 8-bit Offset Pointer: Specifies the byte to start the CRC-16 computation. The pointer and the mask are used together to locate the bytes to be used in the CRC-16 calculations.

The structure of the RWK Filter registers is shown in "Figure 10-11 Remote Wake-Up Packet Filter Register".

- **Size:** 32 bits
- **Offset:** 0xc4
- **Exists:** (DWC_EQOS_PMT_RWK_EN)

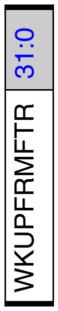


Table 17-39 Fields for Register: MAC_RWK_Packet_Filter

Bits	Name	Memory Access	Description
31:0	WKUPFRMFTR	R/W	RWK Packet Filter This field contains the various controls of RWK Packet filter. **Value After Reset:** 0x0 **Exists:** Always **Testable:** untestable

17.1.36 RWK_Filter(#i)_Byte_Mask (for i = 0; i <= DWC_EQOS_NUM_PMT_RWK_FILT-1)

- **Description:** RWK Filter Byte Mask This register contains the Remote Wakeup Filter Byte Mask.
- **Size:** 32 bits
- **Offset:** 0xc4
- **Exists:** DWC_EQOS_PMT_RWK_EN&&(DWC_EQOS_NUM_PMT_RWK_FILT > 0)

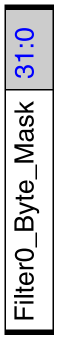


Table 17-40 Fields for Register: RWK_Filter(#i)_Byte_Mask (for i = 0; i <= DWC_EQOS_NUM_PMT_RWK_FILT-1)

Bits	Name	Memory Access	Description
31:0	Filter0_Byte_Mask	R/W	Filter${i} 32-bit Mask Each bit in this mask corresponds to one byte in the detected packet. If the bit is 1', the corresponding byte is taken into the CRC16 calculation. **Value After Reset:** 0x0 **Exists:** Always

17.1.37 RWK_Filter(#i)(#j)_CRC (for i = 0; i <= (DWC_EQOS_NUM_PMT_RWK_FILT/2)-1)

- **Description:** RWK Filter CRC-16
  This register contains the CRC-16 to be matched.
- **Size:** 32 bits
- **Offset:** 0xc4
- **Exists:** DWC_EQOS_PMT_RWK_EN&&(DWC_EQOS_NUM_PMT_RWK_FILT > 0)

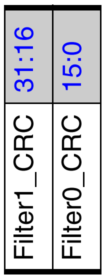


Table 17-41 Fields for Register: RWK_Filter(#i)(#j)_CRC (for i = 0; i <= (DWC_EQOS_NUM_PMT_RWK_FILT/2)-1)

Bits	Name	Memory Access	Description
31:16	Filter1_CRC	R/W	Filter${j} CRC-16 This filter CRC-16 contains the CRC_16 value of the pattern. ■ The 16-bit CRC calculation uses the following polynomial: G(x) = x^16 + x^15 + x^2 + 1 **Value After Reset:** 0x0 **Exists:** Always
15:0	Filter0_CRC	R/W	Filter${i} CRC-16 This filter CRC-16 contains the CRC_16 value of the pattern. ■ The 16-bit CRC calculation uses the following polynomial: G(x) = x^16 + x^15 + x^2 + 1 **Value After Reset:** 0x0 **Exists:** Always

17.1.38 RWK_Filter(#i)(#j)(#k)(#l)_Offset (for i = 0; i <= (DWC_EQOS_NUM_PMT_RWK_-FILT/4)-1)

- **Description:** RWK Filter Offset
  This register defines the offset (within the packet) from which the filter examines the packets.
- **Size:** 32 bits
- **Offset:** 0xc4
- **Exists:** DWC_EQOS_PMT_RWK_EN&&(DWC_EQOS_NUM_PMT_RWK_FILT > 0)

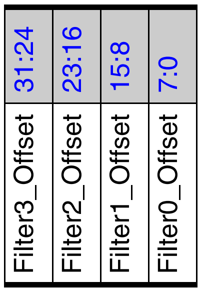


Table 17-42 Fields for Register: RWK_Filter(#i)(#j)(#k)(#l)_Offset (for i = 0; i <= (DWC_EQOS_NUM_PMT_RWK_FILT/4)-1)

Bits	Name	Memory Access	Description
31:24	Filter3_Offset	R/W	Filter${l} Offset This filter offset defines the offset (within the packet) from which the filter examines the packets. ■ This 8-bit pattern-offset is the offset for the filter first byte to be examined. ■ The minimum allowed offset is 12, which refers to the 13th byte of the packet. ■ The offset value 0 refers to the first byte of the packet. **Value After Reset:** 0x0 **Exists:** Always
23:16	Filter2_Offset	R/W	Filter${k} Offset This filter offset defines the offset (within the packet) from which the filter examines the packets. ■ This 8-bit pattern-offset is the offset for the filter first byte to be examined. ■ The minimum allowed offset is 12, which refers to the 13th byte of the packet. ■ The offset value 0 refers to the first byte of the packet. **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
15:8	Filter1_Offset	R/W	Filter${j} Offset This filter offset defines the offset (within the packet) from which the filter examines the packets. ■ This 8-bit pattern-offset is the offset for the filter first byte to be examined. ■ The minimum allowed offset is 12, which refers to the 13th byte of the packet. ■ The offset value 0 refers to the first byte of the packet. **Value After Reset:** 0x0 **Exists:** Always
7:0	Filter0_Offset	R/W	Filter${i} Offset This filter offset defines the offset (within the packet) from which the filter examines the packets. ■ This 8-bit pattern-offset is the offset for the filter first byte to be examined. ■ The minimum allowed offset is 12, which refers to the 13th byte of the packet. ■ The offset value 0 refers to the first byte of the packet. **Value After Reset:** 0x0 **Exists:** Always

17.1.39 RWK_Filter(#i)(#j)(#k)(#l)_Command (for i = 0; i <= (DWC_EQOS_NUM_PMT_RWK_-FILT/4)-1)

- **Description:** RWK Filter Command
  This register controls the filter operation.
- **Size:** 32 bits
- **Offset:** 0xc4
- **Exists:** DWC_EQOS_PMT_RWK_EN&&(DWC_EQOS_NUM_PMT_RWK_FILT > 0)

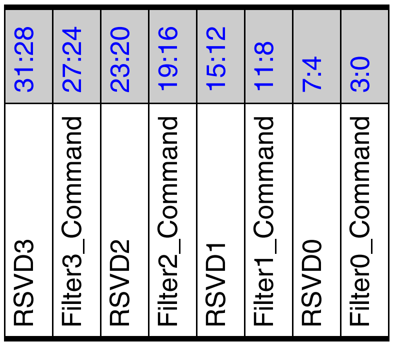


Table 17-43 Fields for Register: RWK_Filter(#i)(#j)(#k)(#l)_Command (for i = 0; i <= (DWC_EQOS_NUM_PMT_RWK_FILT/4)-1)

Bits	Name	Memory Access	Description
31:28	RSVD3	R	Reserved **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
27:24	Filter3_Command	R/W	Filter${l} Command The 4-bit filter command controls the filter operation. ■ Bit 3 specifies the address type, defining the destination address type of the pattern. When the bit is set, the pattern applies to only multicast packets; when the bit is reset, the pattern applies only to unicast packet. ■ Bit 2 (Inverse Mode), when set, reverses the logic of the CRC16 hash function signal, to reject a packet with matching CRC_16 value. ■ Bit 2, along with Bit 1, allows a MAC to reject a subset of remote wake-up packets by creating filter logic such as "Pattern 1 AND NOT Pattern 2". ■ Bit 1 (And_Previous) implements the Boolean logic. When set, the result of the current entry is logically ANDed with the result of the previous filter. This AND logic allows a filter pattern longer than 32 bytes by splitting the mask among two, three, or four filters. This depends on the number of filters that have the And_Previous bit set. ■ Bit 0 is the enable for filter. If Bit 0 is not set, filter is disabled. **Value After Reset:** 0x0 **Exists:** Always
23:20	RSVD2	R	Reserved **Value After Reset:** 0x0 **Exists:** Always
19:16	Filter2_Command	R/W	Filter${k} Command The 4-bit filter command controls the filter operation. ■ Bit 3 specifies the address type, defining the destination address type of the pattern. When the bit is set, the pattern applies to only multicast packets; when the bit is reset, the pattern applies only to unicast packet. ■ Bit 2 (Inverse Mode), when set, reverses the logic of the CRC16 hash function signal, to reject a packet with matching CRC_16 value. ■ Bit 2, along with Bit 1, allows a MAC to reject a subset of remote wake-up packets by creating filter logic such as "Pattern 1 AND NOT Pattern 2". ■ Bit 1 (And_Previous) implements the Boolean logic. When set, the result of the current entry is logically ANDed with the result of the previous filter. This AND logic allows a filter pattern longer than 32 bytes by splitting the mask among two, three, or four filters. This depends on the number of filters that have the And_Previous bit set. ■ Bit 0 is the enable for filter. If Bit 0 is not set, filter is disabled. **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
15:12	RSVD1	R	Reserved **Value After Reset:** 0x0 **Exists:** Always
11:8	Filter1_Command	R/W	Filter${j} Command The 4-bit filter command controls the filter operation. ■ Bit 3 specifies the address type, defining the destination address type of the pattern. When the bit is set, the pattern applies to only multicast packets; when the bit is reset, the pattern applies only to unicast packet. ■ Bit 2 (Inverse Mode), when set, reverses the logic of the CRC16 hash function signal, to reject a packet with matching CRC_16 value. ■ Bit 2, along with Bit 1, allows a MAC to reject a subset of remote wake-up packets by creating filter logic such as "Pattern 1 AND NOT Pattern 2". ■ Bit 1 (And_Previous) implements the Boolean logic. When set, the result of the current entry is logically ANDed with the result of the previous filter. This AND logic allows a filter pattern longer than 32 bytes by splitting the mask among two, three, or four filters. This depends on the number of filters that have the And_Previous bit set. ■ Bit 0 is the enable for filter. If Bit 0 is not set, filter is disabled. **Value After Reset:** 0x0 **Exists:** Always
7:4	RSVD0	R	Reserved **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
3:0	Filter0_Command	R/W	Filter${i} Command The 4-bit filter command controls the filter operation. ■ Bit 3 specifies the address type, defining the destination address type of the pattern. When the bit is set, the pattern applies to only multicast packets; when the bit is reset, the pattern applies only to unicast packet. ■ Bit 2 (Inverse Mode), when set, reverses the logic of the CRC16 hash function signal, to reject a packet with matching CRC_16 value. ■ Bit 2, along with Bit 1, allows a MAC to reject a subset of remote wake-up packets by creating filter logic such as "Pattern 1 AND NOT Pattern 2". ■ Bit 1 (And_Previous) implements the Boolean logic. When set, the result of the current entry is logically ANDed with the result of the previous filter. This AND logic allows a filter pattern longer than 32 bytes by splitting the mask among two, three, or four filters. This depends on the number of filters that have the And_Previous bit set. ■ Bit 0 is the enable for filter. If Bit 0 is not set, filter is disabled. **Value After Reset:** 0x0 **Exists:** Always

17.1.40 MAC_LPI_Control_Status

- **Description:** The LPI Control and Status Register controls the LPI functions and provides the LPI interrupt status. The status bits are cleared when this register is read.
- **Size:** 32 bits
- **Offset:** 0xd0
- **Exists:** (DWC_EQOS_EEE_EN)

31:22	21	20	19	18	17	16	15:10	9	8	7:4	3	2	1	0
Reserved_31_22	LPITCSE	LPIATE	LPITXA	PLSEN	PLS	LPIEN	Reserved_15_10	RLPIST	TLPIST	Reserved_7_4	RLPIEX	RLPIEN	TLPIEX	TLPIEN

Table 17-44 Fields for Register: MAC_LPI_Control_Status

Bits	Name	Memory Access	Description
31:22	Reserved_31_22	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
21	LPITCSE	R/W	LPI Tx Clock Stop Enable When this bit is set, the MAC asserts sbd_tx_clk_gating_ctrl_o signal high after it enters Tx LPI mode to indicate that the Tx clock to MAC can be stopped. When this bit is reset, the MAC does not assert sbd_tx_clk_gating_ctrl_o signal high after it enters Tx LPI mode. If RGMII Interface is selected, the Tx clock is required for transmitting the LPI pattern. The Tx Clock cannot be gated and so the LPITCSE bit cannot be programmed. **Values:** ■ 0x0 (DISABLE): LPI Tx Clock Stop is disabled ■ 0x1 (ENABLE): LPI Tx Clock Stop is enabled **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
20	LPIATE	R/W	LPI Timer Enable This bit controls the automatic entry of the MAC Transmitter into and exit out of the LPI state. When LPIATE, LPITXA and LPIEN bits are set, the MAC Transmitter enters LPI state only when the complete MAC TX data path is IDLE for a period indicated by the MAC_LPI_Entry_Timer register. After entering LPI state, if the data path becomes non-IDLE (due to a new packet being accepted for transmission), the Transmitter exits LPI state but does not clear LPIEN bit. This enables the re-entry into LPI state when it is IDLE again. When LPIATE is 0, the LPI Auto timer is disabled and MAC Transmitter enters LPI state based on the settings of LPITXA and LPIEN bit descriptions. **Values:** ■ 0x0 (DISABLE): LPI Timer is disabled ■ 0x1 (ENABLE): LPI Timer is enabled **Value After Reset:** 0x0 **Exists:** Always
19	LPITXA	R/W	LPI Tx Automate This bit controls the behavior of the MAC when it is entering or coming out of the LPI mode on the Transmit side. This bit is not functional in the EQOS-CORE configurations in which the Tx clock gating is done during the LPI mode. If the LPITXA and LPIEN bits are set to 1, the MAC enters the LPI mode only after all outstanding packets (in the core) and pending packets (in the application interface) have been transmitted. The MAC comes out of the LPI mode when the application sends any packet for transmission or the application issues a Tx FIFO Flush command. In addition, the MAC automatically clears the LPIEN bit when it exits the LPI state. If Tx FIFO Flush is set in the FTQ bit of MTL_TxQ0_Operation_Mode register, when the MAC is in the LPI mode, it exits the LPI mode. When this bit is 0, the LPIEN bit directly controls behavior of the MAC when it is entering or coming out of the LPI mode. **Values:** ■ 0x0 (DISABLE): LPI Tx Automate is disabled ■ 0x1 (ENABLE): LPI Tx Automate is enabled **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
18	PLSEN	R/W	PHY Link Status Enable This bit enables the link status received on the RGMII, SGMII, or SMII Receive paths to be used for activating the LPI LS TIMER. When this bit is set, the MAC uses the link-status bits of the MAC_PHYIF_Control_Status register and the PLS bit for the LPI LS Timer trigger. When this bit is reset, the MAC ignores the link-status bits of the MAC_PHYIF_Control_Status register and takes only the PLS bit. **Values:** ■ 0x0 (DISABLE): PHY Link Status is disabled ■ 0x1 (ENABLE): PHY Link Status is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_RGMII_EN||DWC_EQOS_SMII_EN||DWC_EQOS_SGMII_EN
17	PLS	R/W	PHY Link Status This bit indicates the link status of the PHY. The MAC Transmitter asserts the LPI pattern only when the link status is up (OKAY) at least for the time indicated by the LPI LS TIMER. When this bit is set, the link is considered to be okay (UP) and when this bit is reset, the link is considered to be down. **Values:** ■ 0x0 (DISABLE): link is down ■ 0x1 (ENABLE): link is okay (UP) **Value After Reset:** 0x0 **Exists:** Always
16	LPIEN	R/W	LPI Enable When this bit is set, it instructs the MAC Transmitter to enter the LPI state. When this bit is reset, it instructs the MAC to exit the LPI state and resume normal transmission. This bit is cleared when the LPITXA bit is set and the MAC exits the LPI state because of the arrival of a new packet for transmission. **Values:** ■ 0x0 (DISABLE): LPI state is disabled ■ 0x1 (ENABLE): LPI state is enabled **Value After Reset:** 0x0 **Exists:** Always
15:10	Reserved_15_10	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
9	RLPIST	R	Receive LPI State When this bit is set, it indicates that the MAC is receiving the LPI pattern on the GMII or MII interface. **Values:** ■ 0x0 (INACTIVE): Receive LPI state not detected ■ 0x1 (ACTIVE): Receive LPI state detected **Value After Reset:** 0x0 **Exists:** Always
8	TLPIST	R	Transmit LPI State When this bit is set, it indicates that the MAC is transmitting the LPI pattern on the GMII or MII interface. **Values:** ■ 0x0 (INACTIVE): Transmit LPI state not detected ■ 0x1 (ACTIVE): Transmit LPI state detected **Value After Reset:** 0x0 **Exists:** Always
7:4	Reserved_7_4	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
3	RLPIEX	R	Receive LPI Exit When this bit is set, it indicates that the MAC Receiver has stopped receiving the LPI pattern on the GMII or MII interface, exited the LPI state, and resumed the normal reception. This bit is cleared by a read into this register (or this bit is written to 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). **Note**: This bit may not be set if the MAC stops receiving the LPI pattern for a very short duration, such as, less than three clock cycles of CSR clock. **Values:** ■ 0x0 (INACTIVE): Receive LPI exit not detected ■ 0x1 (ACTIVE): Receive LPI exit detected **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
2	RLPIEN	R	Receive LPI Entry When this bit is set, it indicates that the MAC Receiver has received an LPI pattern and entered the LPI state. This bit is cleared by a read into this register (or this bit is written to 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). **Note**: This bit may not be set if the MAC stops receiving the LPI pattern for a very short duration, such as, less than three clock cycles of CSR clock. **Values:** ■ 0x0 (INACTIVE): Receive LPI entry not detected ■ 0x1 (ACTIVE): Receive LPI entry detected **Value After Reset:** 0x0 **Exists:** Always
1	TLPIEX	R	Transmit LPI Exit When this bit is set, it indicates that the MAC transmitter exited the LPI state after the application cleared the LPIEN bit and the LPI TW Timer has expired. This bit is cleared by a read into this register (or this bit is written to 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). **Values:** ■ 0x0 (INACTIVE): Transmit LPI exit not detected ■ 0x1 (ACTIVE): Transmit LPI exit detected **Value After Reset:** 0x0 **Exists:** Always
0	TLPIEN	R	Transmit LPI Entry When this bit is set, it indicates that the MAC Transmitter has entered the LPI state because of the setting of the LPIEN bit. This bit is cleared by a read into this register (or this bit is written to 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). **Values:** ■ 0x0 (INACTIVE): Transmit LPI entry not detected ■ 0x1 (ACTIVE): Transmit LPI entry detected **Value After Reset:** 0x0 **Exists:** Always

17.1.41 MAC_LPI_Timers_Control

- **Description:** The LPI Timers Control register controls the timeout values in the LPI states. It specifies the time for which the MAC transmits the LPI pattern and also the time for which the MAC waits before resuming the normal transmission.
- **Size:** 32 bits
- **Offset:** 0xd4
- **Exists:** (DWC_EQOS_EEE_EN)

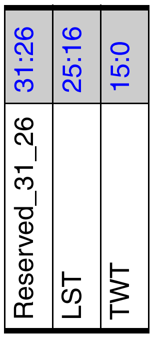


Table 17-45 Fields for Register: MAC_LPI_Timers_Control

Bits	Name	Memory Access	Description
31:26	Reserved_31_26	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
25:16	LST	R/W	LPI LS Timer This field specifies the minimum time (in milliseconds) for which the link status from the PHY should be up (OKAY) before the LPI pattern can be transmitted to the PHY. The MAC does not transmit the LPI pattern even when the LPIEN bit is set unless the LPI LS Timer reaches the programmed terminal count. The default value of the LPI LS Timer is 1000 (1 sec) as defined in the IEEE standard. **Value After Reset:** 0x3e8 **Exists:** Always
15:0	TWT	R/W	LPI TW Timer This field specifies the minimum time (in microseconds) for which the MAC waits after it stops transmitting the LPI pattern to the PHY and before it resumes the normal transmission. The TLPIEX status bit is set after the expiry of this timer. **Value After Reset:** 0x0 **Exists:** Always

17.1.42 MAC_LPI_Entry_Timer

- **Description:** This register controls the Tx LPI entry timer. This counter is enabled only when bit[20](LPITE) bit of MAC_LPI_Control_Status is set to 1.
- **Size:** 32 bits
- **Offset:** 0xd8
- **Exists:** (DWC_EQOS_EEE_EN)

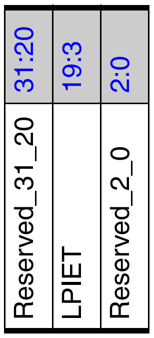


Table 17-46 Fields for Register: MAC_LPI_Entry_Timer

Bits	Name	Memory Access	Description
31:20	Reserved_31_20	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
19:3	LPIET	R/W	LPI Entry Timer This field specifies the time in microseconds the MAC waits to enter LPI mode, after it has transmitted all the frames. This field is valid and used only when LPITE and LPITXA are set to 1. Bits [2:0] are read-only so that the granularity of this timer is in steps of 8 micro-seconds. **Value After Reset:** 0x0 **Exists:** Always
2:0	Reserved_2_0	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always

17.1.43 MAC_1US_Tic_Counter

- **Description:** This register controls the generation of the Reference time (1 microsecond tic) for all the LPI timers. This timer has to be programmed by the software initially.
- **Size:** 32 bits
- **Offset:** 0xdc
- **Exists:** (DWC_EQOS_EEE_EN)

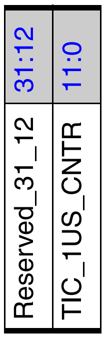


Table 17-47 Fields for Register: MAC_1US_Tic_Counter

Bits	Name	Memory Access	Description
31:12	Reserved_31_12	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
11:0	TIC_1US_CNTR	R/W	1US TIC Counter The application must program this counter so that the number of clock cycles of CSR clock is 1us. (Subtract 1 from the value before programming). For example if the CSR clock is 100MHz then this field needs to be programmed to value 100 - 1 = 99 (which is 0x63). This is required to generate the 1US events that are used to update some of the EEE related counters. **Value After Reset:** 0x63 **Exists:** Always

17.1.44 MAC_AN_Control

- **Description:** The AN Control register enables and restarts auto-negotiation. It also enables PCS loop-back. This register is optional.
- **Size:** 32 bits
- **Offset:** 0xe0
- **Exists:** ((DWC_EQOS_TBI_EN||DWC_EQOS_SGMII_EN||DWC_EQOS_RTBI_EN))

31:19	18	17	16	15	14	13	12	11:10	9	8:0
Reserved_31_19	SGMRAL	LR	ECD	Reserved_15	ELE	Reserved_13	ANE	Reserved_11_10	RAN	Reserved_8_0

Table 17-48 Fields for Register: MAC_AN_Control

Bits	Name	Memory Access	Description
31:19	Reserved_31_19	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
18	SGMRAL	R/W	SGMII RAL Control When this bit is set, the SGMII RAL block operates in the speed configured in the Speed and Port Select bits of the MAC_Configuration register. This is useful when the SGMII interface is used in a direct MAC to MAC connection (without a PHY) and any MAC must reconfigure the speed. When this bit is reset, the SGMII RAL block operates according to the link speed status received on SGMII (from the PHY). **Values:** ■ 0x0 (DISABLE): SGMII RAL Control is disabled ■ 0x1 (ENABLE): SGMII RAL Control is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_SGMII_EN

Bits	Name	Memory Access	Description
17	LR	R/W	Lock to Reference When this bit is set, the PHY locks its PLL to the 125 MHz reference clock. This bit controls the pcs_lck_ref_o signal on the TBI, RTBI, or SGMII interface. **Values:** ■ 0x0 (DISABLE): Lock to Reference is disabled ■ 0x1 (ENABLE): Lock to Reference is enabled **Value After Reset:** 0x0 **Exists:** Always
16	ECD	R/W	Enable Comma Detect When this bit is set, the PHY is enabled for comma detection and word resynchronization. When this bit is reset, the comma detection and word resynchronization will be done internally by DWC_ether_qos PCS block. This bit controls the pcs_en_cdet_o signal on the TBI, RTBI, or SGMII interface. **Values:** ■ 0x0 (DISABLE): Comma detect is disabled ■ 0x1 (ENABLE): Comma detect is enabled **Value After Reset:** 0x0 **Exists:** Always
15	Reserved_15	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
14	ELE	R/W	External Loopback Enable When this bit is set, it enables the PHY to loopback the Transmit data into the Receive path. The pcs_ewrap_o signal is asserted high when this bit is set. **Values:** ■ 0x0 (DISABLE): External Loopback is disabled ■ 0x1 (ENABLE): External Loopback is enabled **Value After Reset:** 0x0 **Exists:** Always
13	Reserved_13	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
12	ANE	R/W	Auto-Negotiation Enable When this bit is set, it enables the MAC to perform auto-negotiation with the link partner. When this is reset, the auto-negotiation is disabled. **Values:** ■ 0x0 (DISABLE): Auto-Negotiation is disabled ■ 0x1 (ENABLE): Auto-Negotiation is enabled **Value After Reset:** 0x0 **Exists:** Always
11:10	Reserved_11_10	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
9	RAN	R/W	Restart Auto-Negotiation When this bit is set, auto-negotiation is restarted if Bit 12 (ANE) is set. This bit is self-clearing after auto-negotiation starts. This bit should be cleared for normal operation. Access restriction applies. Setting 1 sets. Self-cleared. Setting 0 has no effect. **Values:** ■ 0x0 (DISABLE): Do not Restart Auto-Negotiation ■ 0x1 (ENABLE): Restart Auto-Negotiation **Value After Reset:** 0x0 **Exists:** Always **Testable:** untestable
8:0	Reserved_8_0	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always

17.1.45 MAC_AN_Status

- **Description:** The AN Status register indicates the link and the auto-negotiation status.
- **Size:** 32 bits
- **Offset:** 0xe4
- **Exists:** ((DWC_EQOS_TBI_EN||DWC_EQOS_SGMII_EN||DWC_EQOS_RTBI_EN))

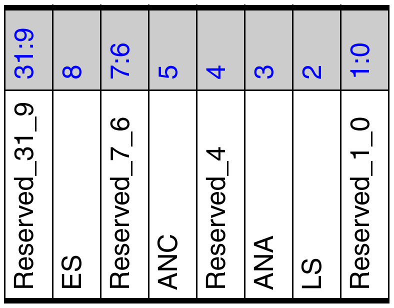


Table 17-49 Fields for Register: MAC_AN_Status

Bits	Name	Memory Access	Description
31:9	Reserved_31_9	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
8	ES	R	Extended Status This bit is tied to high if the TBI or RTBI interface is selected while configuring the core indicating that the MAC supports extended status information in the MAC_TBI_Extended_Status register. This bit is tied to low if the SGMII interface is selected and the TBI or RTBI interface is not selected while configuring the core indicating that MAC_TBI_Extended_Status register is not present. ■ 1: TBI or RTBI ■ 0: SGMII without TBI or **Values:** ■ 0x0 (INACTIVE): No Extended status ■ 0x1 (ACTIVE): Extended status **Value After Reset:** ((DWC_EQOS_TBI_EN||DWC_EQOS_RTBI_EN)?\"0x1\":\"0x0\") **Exists:** Always
7:6	Reserved_7_6	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
5	ANC	R	Auto-Negotiation Complete When this bit is set, it indicates that the auto-negotiation process is complete. This bit is cleared when auto-negotiation is again initiated. **Values:** ■ 0x0 (INACTIVE): Auto-Negotiation is not complete ■ 0x1 (ACTIVE): Auto-Negotiation is complete **Value After Reset:** 0x0 **Exists:** Always
4	Reserved_4	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
3	ANA	R	Auto-Negotiation Ability This bit is always high because the MAC supports auto-negotiation. **Values:** ■ 0x0 (INACTIVE): MAC does not posses Auto-Negotiation Ability ■ 0x1 (ACTIVE): MAC posses Auto-Negotiation Ability **Value After Reset:** 0x1 **Exists:** Always
2	LS	R	Link Status When this bit is set, it indicates that the link is up between the MAC and the TBI, RTBI, or SGMII interface. When this bit is reset, it indicates that the link is down between the MAC and the TBI, RTBI, or SGMII interface. **Values:** ■ 0x0 (INACTIVE): Link is down ■ 0x1 (ACTIVE): Link is up **Value After Reset:** 0x0 **Exists:** Always **Testable:** untestable
1:0	Reserved_1_0	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always

17.1.46 MAC_AN_Advertisement

- **Description:** The Auto-Negotiation Advertisement register indicates the link and the auto-negotiation status.
- **Size:** 32 bits
- **Offset:** 0xe8
- **Exists:** (DWC_EQOS_TBI_EN||DWC_EQOS_RTBI_EN)

31:16	15	14	13:12	11:9	8:7	6	5	4:0
Reserved_31_16	NP	Reserved_14	RFE	Reserved_11_9	PSE	HD	FD	Reserved_4_0

Table 17-50 Fields for Register: MAC_AN_Advertisement

Bits	Name	Memory Access	Description
31:16	Reserved_31_16	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
15	NP	R	Next Page Support This bit is always low because the MAC does not support the next page. **Values:** ■ 0x0 (INACTIVE): MAC does not support the next page **Value After Reset:** 0x0 **Exists:** Always
14	Reserved_14	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
13:12	RFE	R/W	Remote Fault Encoding These bits provide a remote fault encoding, indicating to a link partner that a fault or error condition has occurred. The encoding of these bits is defined in IEEE 802.3z, Section 37.2.1.5. **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
11:9	Reserved_11_9	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
8:7	PSE	R/W	Pause Encoding These bits provide an encoding for the Pause bits, indicating that the MAC is capable of configuring the Pause function as defined in IEEE 802.3x. The encoding of these bits is defined in IEEE 802.3z, Section 37.2.1.4. **Value After Reset:** 0x3 **Exists:** Always
6	HD	R/W	Half-Duplex When set high, this bit indicates that the MAC supports the half-duplex mode. This bit is always low (and RO) when the MAC is configured for the full-duplex-only mode. **Values:** ■ 0x0 (DISABLE): Full-duplex only mode ■ 0x1 (ENABLE): Supports Half-duplex mode **Value After Reset:** ((DWC_EQOS_TBI_EN||DWC_EQOS_RTBI_EN)?\"0x1\":\"0x0\") **Exists:** !DWC_EQOS_FDUPLX_ONLY
5	FD	R/W	Full-Duplex When set high, this bit indicates that the MAC supports the full-duplex mode. **Values:** ■ 0x0 (DISABLE): Doesnot support Full-duplex mode ■ 0x1 (ENABLE): Supports full-duplex mode **Value After Reset:** 0x1 **Exists:** Always
4:0	Reserved_4_0	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always

17.1.47 MAC_AN_Link_Partner_Ability

- **Description:** The Auto-Negotiation Link Partner Ability register contains the advertised ability of the link partner.
- **Size:** 32 bits
- **Offset:** 0xec
- **Exists:** (DWC_EQOS_TBI_EN||DWC_EQOS_RTBI_EN)

31:16	15	14	13:12	11:9	8:7	6	5	4:0
Reserved_31_16	NP	ACK	RFE	Reserved_11_9	PSE	HD	FD	Reserved_4_0

Table 17-51 Fields for Register: MAC_AN_Link_Partner_Ability

Bits	Name	Memory Access	Description
31:16	Reserved_31_16	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
15	NP	R	Next Page Support When this bit is set, it indicates that more next page information is available. When this bit is reset, it indicates that the next page exchange is not desired. **Values:** ■ 0x0 (INACTIVE): Next page support not available ■ 0x1 (ACTIVE): Next page support available **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
14	ACK	R	Acknowledge When this bit is set, the auto-negotiation function uses this bit to indicate that the link partner has successfully received the base page of the MAC. When this bit is reset, it indicates that the link partner did not successfully receive the base page of the MAC. **Values:** ■ 0x0 (INACTIVE): link partner did not successfully receive the base page ■ 0x1 (ACTIVE): Acknowledge - link partner did not successfully receive the base page **Value After Reset:** 0x0 **Exists:** Always
13:12	RFE	R	Remote Fault Encoding These bits provide a remote fault encoding, indicating a fault or error condition of the link partner. The encoding of these bits is defined in IEEE 802.3z, Section 37.2.1.5. **Value After Reset:** 0x0 **Exists:** Always
11:9	Reserved_11_9	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
8:7	PSE	R	Pause Encoding These bits provide an encoding for the Pause bits. These bits indicate that the link partner is capable of configuring the Pause function as defined in the IEEE 802.3x. The encoding of these bits is defined in IEEE 802.3z, Section 37.2.1.4. **Value After Reset:** 0x0 **Exists:** Always
6	HD	R	Half-Duplex When this bit is set, it indicates that the link partner has the ability to operate in the half-duplex mode. When this bit is reset, it indicates that the link partner does not have the ability to operate in the half-duplex mode. **Values:** ■ 0x0 (INACTIVE): Full-duplex only mode ■ 0x1 (ACTIVE): Supports Half-duplex mode **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
5	FD	R	Full-Duplex When this bit is set, it indicates that the link partner has the ability to operate in the full-duplex mode. When this bit is reset, it indicates that the link partner does not have the ability to operate in the full-duplex mode. **Values:** ■ 0x0 (INACTIVE): Doesnot support Full-duplex mode ■ 0x1 (ACTIVE): Supports full-duplex mode **Value After Reset:** 0x0 **Exists:** Always
4:0	Reserved_4_0	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always

17.1.48 MAC_AN_Expansion

- **Description:** The Auto-Negotiation Expansion register indicates if the MAC received a new base page from the link partner. This register is optional.
- **Size:** 32 bits
- **Offset:** 0xf0
- **Exists:** (DWC_EQOS_TBI_EN||DWC_EQOS_RTBI_EN)

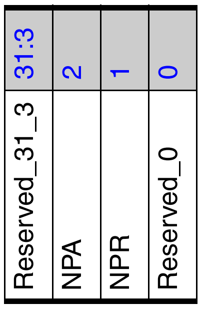


Table 17-52 Fields for Register: MAC_AN_Expansion

Bits	Name	Memory Access	Description
31:3	Reserved_31_3	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
2	NPA	R	Next Page Ability This bit is always low because the MAC does not support the next page function. **Values:** ■ 0x0 (INACTIVE): Next page support not available **Value After Reset:** 0x0 **Exists:** Always
1	NPR	R	New Page Received When this bit is set, it indicates that the MAC has received a new page. This bit is cleared when read. **Values:** ■ 0x0 (INACTIVE): Next page not received ■ 0x1 (ACTIVE): Next page received **Value After Reset:** 0x0 **Exists:** Always
0	Reserved_0	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always

17.1.49 MAC_TBI_Extended_Status

- **Description:** The TBI Extended Status register indicates all modes of operation of the MAC.
- **Size:** 32 bits
- **Offset:** 0xf4
- **Exists:** (DWC_EQOS_TBI_EN||DWC_EQOS_RTBI_EN)

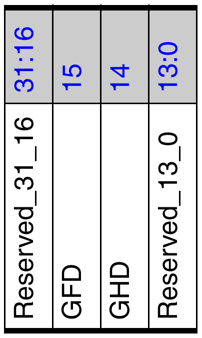


Table 17-53 Fields for Register: MAC_TBI_Extended_Status

Bits	Name	Memory Access	Description
31:16	Reserved_31_16	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
15	GFD	R	1000BASE-X Full-Duplex Capable When this bit is set, the MAC is able to perform the full-duplex and 1000BASE-X operations. **Values:** ■ 0x0 (INACTIVE): 1000BASE-X Full-Duplex Incapable ■ 0x1 (ACTIVE): 1000BASE-X Full-Duplex Capable **Value After Reset:** 0x1 **Exists:** Always
14	GHD	R	1000BASE-X Half-Duplex Capable When this bit is set, the MAC is able to perform the half-duplex and 1000BASE-X operations. This bit is always low when the MAC is configured for the full-duplex-only operation while configuring the core. **Values:** ■ 0x0 (INACTIVE): 1000BASE-X Half-Duplex Incapable ■ 0x1 (ACTIVE): 1000BASE-X Half-Duplex Capable **Value After Reset:** (((DWC_EQOS_TBI_EN||DWC_EQOS_RTBI_EN)&&!DWC_EQOS_FDUPLX_ONLY)?\"0x1\":\"0x0\") **Exists:** Always

Bits	Name	Memory Access	Description
13:0	Reserved_13_0	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always

17.1.50 MAC_PHYIF_Control_Status

- **Description:** The PHY Interface Control and Status register indicates the status signals received by the SGMII, RGMII, or SMII interface (selected at reset) from the PHY. This register is optional.
- **Size:** 32 bits
- **Offset:** 0xf8
- **Exists:** (DWC_EQOS_SGMII_EN||DWC_EQOS_RGMII_EN||DWC_EQOS_SMII_EN)

31:22	21	20	19	18:17	16	15:5	4	3	2	1	0
Reserved_31_22	FALSCARDET	JABTO	LNKSTS	LNKSPEED	LNKMOD	Reserved_15_5	SMIDRXS	Reserved_3	SFTERR	LUD	TC

Table 17-54 Fields for Register: MAC_PHYIF_Control_Status

Bits	Name	Memory Access	Description
31:22	Reserved_31_22	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
21	FALSCARDET	R	False Carrier Detected This bit indicates whether the SMII PHY detected false carrier (1'b1). **Values:** ■ 0x0 (INACTIVE): No False carrier ■ 0x1 (ACTIVE): False carrier detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_SMII_EN **Testable:** untestable
20	JABTO	R	Jabber Timeout This bit indicates the jabber timeout error (1'b1) in the received packet. **Values:** ■ 0x0 (INACTIVE): No jabber timeout error ■ 0x1 (ACTIVE): Jabber timeout error **Value After Reset:** 0x0 **Exists:** DWC_EQOS_SMII_EN **Testable:** untestable

Bits	Name	Memory Access	Description
19	LNKSTS	R	Link Status This bit indicates whether the link is up (1'b1) or down (1'b0). **Values:** ■ 0x0 (INACTIVE): Link down ■ 0x1 (ACTIVE): Link up **Value After Reset:** 0x0 **Exists:** Always **Testable:** untestable
18:17	LNKSPEED	R	Link Speed This bit indicates the current speed of the link. Bit 2 is reserved when the MAC is configured for the SMII PHY interface. **Values:** ■ 0x0 (2500K): 2.5 MHz ■ 0x1 (25M): 25 MHz ■ 0x2 (125M): 125 MHz ■ 0x3 (RSVD): Reserved **Value After Reset:** ((DWC_EQOS_SGMII_EN&&!(DWC_EQOS_RGMII_EN||DWC_EQOS_SMII_EN))?\"0x2\":\"0x0\") **Exists:** Always **Testable:** untestable
16	LNKMOD	R	Link Mode This bit indicates the current mode of operation of the link. **Values:** ■ 0x0 (HDUPLX): Half-duplex mode ■ 0x1 (FDUPLX): Full-duplex mode **Value After Reset:** 0x0 **Exists:** Always **Testable:** untestable
15:5	Reserved_15_5	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
4	SMIDRXS	R/W	Delay SMII Rx Data Sampling with respect to the SMII SYNC Signal When this bit is set, the first bit of the SMII Rx data is sampled one cycle after the SMII SYNC signal. When reset, the first bit of the SMII Rx data is sampled along with the SMII SYNC signal. **Values:** ■ 0x0 (DISABLE): Disable Delay SMII Rx Data Sampling with respect to the SMII SYNC Signal ■ 0x1 (ENABLE): Enable Delay SMII Rx Data Sampling with respect to the SMII SYNC Signal **Value After Reset:** 0x0 **Exists:** DWC_EQOS_SMII_EN&&!DWC_EQOS_SSSMII_EN
3	Reserved_3	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
2	SFTERR	R/W	SMII Force Transmit Error When set, this bit indicates to the PHY to force a transmit error in the SMII packet being transmitted. **Values:** ■ 0x0 (DISABLE): SMII force transmit error is disabled ■ 0x1 (ENABLE): SMII force transmit error is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_SMII_EN
1	LUD	R/W	Link Up or Down This bit indicates whether the link is up or down during transmission of configuration in the RGMII, SGMII, or SMII interface. **Values:** ■ 0x0 (LINKDOWN): Link down ■ 0x1 (LINKUP): Link up **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
0	TC	R/W	Transmit Configuration in RGMII, SGMII, or SMII When set, this bit enables the transmission of duplex mode, link speed, and link up or down information to the PHY in the RGMII, SMII, or SGMII port. When this bit is reset, no such information is driven to the PHY. The details of this feature are provided in the following sections: ■ "Reduced Gigabit Media Independent Interface" ■ "Serial Media Independent Interface" ■ "Serial Gigabit Media Independent Interface" **Values:** ■ 0x0 (DISABLE): Disable Transmit Configuration in RGMII, SGMII, or SMII ■ 0x1 (ENABLE): Enable Transmit Configuration in RGMII, SGMII, or SMII **Value After Reset:** 0x0 **Exists:** Always

17.1.51 MAC_Version

- **Description:** The version register identifies the version of the DWC_ether_qos. This register contains two bytes: one that Synopsys uses to identify the core release number, and the other that you set while configuring the core.
- **Size:** 32 bits
- **Offset:** 0x110
- **Exists:** Always

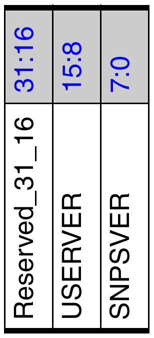


Table 17-55 Fields for Register: MAC_Version

Bits	Name	Memory Access	Description
31:16	Reserved_31_16	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
15:8	USERVER	R	User-defined Version (configured with coreConsultant) **Value After Reset:** (DWC_EQOS_USER_VER) **Exists:** Always
7:0	SNPSVER	R	Synopsys-defined Version **Value After Reset:** 0x51 **Exists:** Always

17.1.52 MAC_Debug

- **Description:** The Debug register provides the debug status of various MAC blocks.
- **Size:** 32 bits
- **Offset:** 0x114
- **Exists:** Always

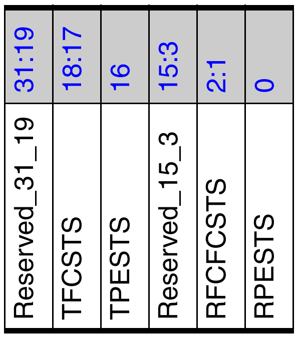


Table 17-56 Fields for Register: MAC_Debug

Bits	Name	Memory Access	Description
31:19	Reserved_31_19	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
18:17	TFCSTS	R	MAC Transmit Packet Controller Status This field indicates the state of the MAC Transmit Packet Controller module. **Values:** ■ 0x0 (IDLE): Idle state ■ 0x1 (WAITING): Waiting for one of the following: Status of the previous packet OR IPG or back off period to be over ■ 0x2 (GEN_TX_PAU): Generating and transmitting a Pause control packet (in full-duplex mode) ■ 0x3 (TRNSFR): Transferring input packet for transmission **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
16	TPESTS	R	MAC GMII or MII Transmit Protocol Engine Status When this bit is set, it indicates that the MAC GMII or MII transmit protocol engine is actively transmitting data, and it is not in the Idle state. **Values:** ■ 0x0 (INACTIVE): MAC GMII or MII Transmit Protocol Engine Status not detected ■ 0x1 (ACTIVE): MAC GMII or MII Transmit Protocol Engine Status detected **Value After Reset:** 0x0 **Exists:** Always
15:3	Reserved_15_3	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
2:1	RFCFCSTS	R	MAC Receive Packet Controller FIFO Status When this bit is set, this field indicates the active state of the small FIFO Read and Write controllers of the MAC Receive Packet Controller module. **Value After Reset:** 0x0 **Exists:** Always
0	RPESTS	R	MAC GMII or MII Receive Protocol Engine Status When this bit is set, it indicates that the MAC GMII or MII receive protocol engine is actively receiving data, and it is not in the Idle state. **Values:** ■ 0x0 (INACTIVE): MAC GMII or MII Receive Protocol Engine Status not detected ■ 0x1 (ACTIVE): MAC GMII or MII Receive Protocol Engine Status detected **Value After Reset:** 0x0 **Exists:** Always

17.1.53 MAC_HW_Feature0

- **Description:** This register indicates the presence of first set of the optional features or functions of the DWC_ether_qos. The software driver can use this register to dynamically enable or disable the programs related to the optional blocks.
  Note: *All bits are set or reset according to the features selected while configuring the core in coreConsultant.*
- **Size:** 32 bits
- **Offset:** 0x11c
- **Exists:** Always

31	30:28	27	26:25	24	23	22:18	17	16	15	14	13	12	11:10	9	8	7	6	5	4	3	2	1	0
Reserved_31	ACTPHYSEL	SAVLANINS	TSSTSSEL	MACADR64SEL	MACADR32SEL	ADDMACADRSEL	Reserved_17	RXCOESEL	Reserved_15	TXCOESEL	EEESEL	TSSEL	Reserved_11_10	ARPOFFSEL	MMCSEL	MGKSEL	RWKSEL	SMASEL	VLHASH	PCSSEL	HDSEL	GMIISEL	MIISEL

Table 17-57 Fields for Register: MAC_HW_Feature0

Bits	Name	Memory Access	Description
31	Reserved_31	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
30:28	ACTPHYSEL	R	Active PHY Selected When you have multiple PHY interfaces in your configuration, this field indicates the sampled value of phy_intf_sel_i during reset de-assertion. **Values:** ■ 0x0 (GMII_MII): GMII or MII ■ 0x1 (RGMII): RGMII ■ 0x2 (SGMII): SGMII ■ 0x3 (TBI): TBI ■ 0x4 (RMII): RMII ■ 0x5 (RTBI): RTBI ■ 0x6 (SMII): SMII ■ 0x7 (REVMIII): RevMII **Value After Reset:** ((DWC_EQOS_SINGLE_PHY_INTF)?((DWC_EQOS_RGMII_INTF_ONLY)?\"0x1\":(DWC_EQOS_SGMII_INTF_ONLY)?\"0x2\":(DWC_EQOS_RMII_INTF_ONLY)?\"0x4\":(DWC_EQOS_TBI_INTF_ONLY)?\"0x3\":(DWC_EQOS_RTBI_INTF_ONLY)?\"0x5\":(DWC_EQOS_SMII_INTF_ONLY)?\"0x6\":(DWC_EQOS_REVMII_INTF_ONLY)?\"0x7\":\"0x0\"):\"0x0\") **Exists:** Always
27	SAVLANINS	R	Source Address or VLAN Insertion Enable This bit is set to 1 when the Enable SA and VLAN Insertion on Tx option is selected **Values:** ■ 0x0 (INACTIVE): Source Address or VLAN Insertion Enable option is not selected ■ 0x1 (ACTIVE): Source Address or VLAN Insertion Enable option is selected **Value After Reset:** ((DWC_EQOS_SA_VLAN_INS_CTRL_EN)?\"1\":\"0\") **Exists:** Always

Bits	Name	Memory Access	Description
26:25	TSSTSSEL	R	Timestamp System Time Source This bit indicates the source of the Timestamp system time: This bit is set to 1 when the Enable IEEE 1588 Timestamp Support option is selected **Values:** ■ 0x0 (INTRNL): Internal ■ 0x1 (EXTRNL): External ■ 0x2 (BOTH): Both ■ 0x3 (RSVD): Reserved **Value After Reset:** ((DWC_EQOS_SYSTIME_SOURCE==2)?\"3\":(DWC_EQOS_SYSTIME_SOURCE==1)?\"2\":(DWC_EQOS_TIME_STAMPING)?\"1\":\"0\") **Exists:** Always
24	MACADR64SEL	R	MAC Addresses 64-127 Selected This bit is set to 1 when the Enable Additional 64 MAC Address Registers (64-127) option is selected **Values:** ■ 0x0 (INACTIVE): MAC Addresses 64-127 Select option is not selected ■ 0x1 (ACTIVE): MAC Addresses 64-127 Select option is selected **Value After Reset:** ((DWC_EQOS_ADD64_MAC_ADDR_REG)?\"1\":\"0\") **Exists:** Always
23	MACADR32SEL	R	MAC Addresses 32-63 Selected This bit is set to 1 when the Enable Additional 32 MAC Address Registers (32-63) option is selected **Values:** ■ 0x0 (INACTIVE): MAC Addresses 32-63 Select option is not selected ■ 0x1 (ACTIVE): MAC Addresses 32-63 Select option is selected **Value After Reset:** ((DWC_EQOS_ADD32_MAC_ADDR_REG)?\"1\":\"0\") **Exists:** Always
22:18	ADDMACADRSEL	R	MAC Addresses 1-31 Selected This bit is set to 1 when the non-zero value is selected for Enable Additional 1-31 MAC Address Registers option **Value After Reset:** (DWC_EQOS_ADD_MAC_ADDR_REG) **Exists:** Always
17	Reserved_17	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
16	RXCOESEL	R	Receive Checksum Offload Enabled This bit is set to 1 when the Enable Receive TCP/IP Checksum Check option is selected **Values:** ■ 0x0 (INACTIVE): Receive Checksum Offload Enable option is not selected ■ 0x1 (ACTIVE): Receive Checksum Offload Enable option is selected **Value After Reset:** ((DWC_EQOS_RX_COE)?\"1\":\"0\") **Exists:** Always
15	Reserved_15	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
14	TXCOESEL	R	Transmit Checksum Offload Enabled This bit is set to 1 when the Enable Transmit TCP/IP Checksum Insertion option is selected **Values:** ■ 0x0 (INACTIVE): Transmit Checksum Offload Enable option is not selected ■ 0x1 (ACTIVE): Transmit Checksum Offload Enable option is selected **Value After Reset:** ((DWC_EQOS_TX_COE)?\"1\":\"0\") **Exists:** Always
13	EEESEL	R	Energy Efficient Ethernet Enabled This bit is set to 1 when the Enable Energy Efficient Ethernet (EEE) option is selected **Values:** ■ 0x0 (INACTIVE): Energy Efficient Ethernet Enable option is not selected ■ 0x1 (ACTIVE): Energy Efficient Ethernet Enable option is selected **Value After Reset:** ((DWC_EQOS_EEE_EN)?\"1\":\"0\") **Exists:** Always
12	TSSEL	R	IEEE 1588-2008 Timestamp Enabled This bit is set to 1 when the Enable IEEE 1588 Timestamp Support option is selected **Values:** ■ 0x0 (INACTIVE): IEEE 1588-2008 Timestamp Enable option is not selected ■ 0x1 (ACTIVE): IEEE 1588-2008 Timestamp Enable option is selected **Value After Reset:** ((DWC_EQOS_TIME_STAMPING)?\"1\":\"0\") **Exists:** Always

Bits	Name	Memory Access	Description
11:10	Reserved_11_10	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
9	ARPOFFSEL	R	ARP Offload Enabled This bit is set to 1 when the Enable IPv4 ARP Offload option is selected **Values:** ■ 0x0 (INACTIVE): ARP Offload Enable option is not selected ■ 0x1 (ACTIVE): ARP Offload Enable option is selected **Value After Reset:** ((DWC_EQOS_ARP_EN)?\"1\":\"0\") **Exists:** Always
8	MMCSEL	R	RMON Module Enable This bit is set to 1 when the Enable MAC Management Counters (MMC) option is selected **Values:** ■ 0x0 (INACTIVE): RMON Module Enable option is not selected ■ 0x1 (ACTIVE): RMON Module Enable option is selected **Value After Reset:** ((DWC_EQOS_MMC_EN)?\"1\":\"0\") **Exists:** Always
7	MGKSEL	R	PMT Magic Packet Enable This bit is set to 1 when the Enable Magic Packet Detection option is selected **Values:** ■ 0x0 (INACTIVE): PMT Magic Packet Enable option is not selected ■ 0x1 (ACTIVE): PMT Magic Packet Enable option is selected **Value After Reset:** ((DWC_EQOS_PMT_MGK_EN)?\"1\":\"0\") **Exists:** Always
6	RWKSEL	R	PMT Remote Wake-up Packet Enable This bit is set to 1 when the Enable Remote Wake-Up Packet Detection option is selected **Values:** ■ 0x0 (INACTIVE): PMT Remote Wake-up Packet Enable option is not selected ■ 0x1 (ACTIVE): PMT Remote Wake-up Packet Enable option is selected **Value After Reset:** ((DWC_EQOS_PMT_RWK_EN)?\"1\":\"0\") **Exists:** Always

Bits	Name	Memory Access	Description
5	SMASEL	R	SMA (MDIO) Interface This bit is set to 1 when the Enable Station Management (MDIO Interface) option is selected **Values:** ■ 0x0 (INACTIVE): SMA (MDIO) Interface not selected ■ 0x1 (ACTIVE): SMA (MDIO) Interface selected **Value After Reset:** ((DWC_EQOS_SMA_EN)?\"1\":\"0\") **Exists:** Always
4	VLHASH	R	VLAN Hash Filter Selected This bit is set to 1 when the Enable VLAN Hash Table Based Filtering option is selected **Values:** ■ 0x0 (INACTIVE): VLAN Hash Filter not selected ■ 0x1 (ACTIVE): VLAN Hash Filter selected **Value After Reset:** ((DWC_EQOS_VLAN_HASH_EN)?\"1\":\"0\") **Exists:** Always
3	PCSSEL	R	PCS Registers (TBI, SGMII, or RTBI PHY interface) This bit is set to 1 when the TBI, SGMII, or RTBI PHY interface option is selected **Values:** ■ 0x0 (INACTIVE): No PCS Registers (TBI, SGMII, or RTBI PHY interface) ■ 0x1 (ACTIVE): PCS Registers (TBI, SGMII, or RTBI PHY interface) **Value After Reset:** ((DWC_EQOS_PCS_EN)?\"1\":\"0\") **Exists:** Always
2	HDSEL	R	Half-duplex Support This bit is set to 1 when the half-duplex mode is selected **Values:** ■ 0x0 (INACTIVE): No Half-duplex support ■ 0x1 (ACTIVE): Half-duplex support **Value After Reset:** ((!DWC_EQOS_FDUPLX_ONLY)?\"1\":\"0\") **Exists:** Always
1	GMIISEL	R	1000 Mbps Support This bit is set to 1 when 1000 Mbps is selected as the Mode of Operation **Values:** ■ 0x0 (INACTIVE): No 1000 Mbps support ■ 0x1 (ACTIVE): 1000 Mbps support **Value After Reset:** ((DWC_EQOS_OP_MODE!=1)?\"1\":\"0\") **Exists:** Always

Bits	Name	Memory Access	Description
0	MIISEL	R	10 or 100 Mbps Support This bit is set to 1 when 10/100 Mbps is selected as the Mode of Operation **Values:** ■ 0x0 (INACTIVE): No 10 or 100 Mbps support ■ 0x1 (ACTIVE): 10 or 100 Mbps support **Value After Reset:** ((DWC_EQOS_OP_MODE!=2)?\"1\":\"0\") **Exists:** Always

17.1.54 MAC_HW_Feature1

- **Description:** This register indicates the presence of second set of the optional features or functions of the DWC_ether_qos. The software driver can use this register to dynamically enable or disable the programs related to the optional blocks.
  Note: *All bits are set or reset according to the features selected while configuring the core in coreConsultant.*
- **Size:** 32 bits
- **Offset:** 0x120
- **Exists:** Always

31	30:27	26	25:24	23	22	21	20	19	18	17	16	15:14	13	12	11	10:6	5	4:0
Reserved_31	L3L4FNUM	Reserved_26	HASHTBLSZ	POUOST	Reserved_22	RAVSEL	AVSEL	DBGMEMA	TSOEN	SPHEN	DCBEN	ADDR64	ADVTHWORD	PTOEN	OSTEN	TXFIFOSIZE	SPRAM	RXFIFOSIZE

Table 17-58 Fields for Register: MAC_HW_Feature1

Bits	Name	Memory Access	Description
31	Reserved_31	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
30:27	L3L4FNUM	R	Total number of L3 or L4 Filters This field indicates the total number of L3 or L4 filters: **Values:** ■ 0x0 (NOFILT): No L3 or L4 Filter ■ 0x1 (1FILT): 1 L3 or L4 Filter ■ 0x2 (2FILT): 2 L3 or L4 Filters ■ 0x3 (3FILT): 3 L3 or L4 Filters ■ 0x4 (4FILT): 4 L3 or L4 Filters ■ 0x5 (5FILT): 5 L3 or L4 Filters ■ 0x6 (6FILT): 6 L3 or L4 Filters ■ 0x7 (7FILT): 7 L3 or L4 Filters ■ 0x8 (8FILT): 8 L3 or L4 Filters **Value After Reset:** (DWC_EQOS_L3_L4_FILTER_NUM) **Exists:** Always
26	Reserved_26	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
25:24	HASHTBLSZ	R	Hash Table Size This field indicates the size of the hash table: **Values:** ■ 0x0 (NO_HT): No hash table ■ 0x1 (64): 64 ■ 0x2 (128): 128 ■ 0x3 (256): 256 **Value After Reset:** ((DWC_EQOS_HASH_TABLE>128)?\"3\":(DWC_EQOS_HASH_TABLE>64)?\"2\":(DWC_EQOS_HASH_EN)?\"1\":\"0\") **Exists:** Always
23	POUOST	R	One Step for PTP over UDP/IP Feature Enable This bit is set to 1 when the Enable One step timestamp for PTP over UDP/IP feature is selected. **Values:** ■ 0x0 (INACTIVE): One Step for PTP over UDP/IP Feature is not selected ■ 0x1 (ACTIVE): One Step for PTP over UDP/IP Feature is selected **Value After Reset:** ((DWC_EQOS_POU_OST_EN)?\"1\":\"0\") **Exists:** Always
22	Reserved_22	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
21	RAVSEL	R	Rx Side Only AV Feature Enable This bit is set to 1 when the Enable Audio Video Bridging option on Rx Side Only is selected. **Values:** ■ 0x0 (INACTIVE): Rx Side Only AV Feature is not selected ■ 0x1 (ACTIVE): Rx Side Only AV Feature is selected **Value After Reset:** ((DWC_EQOS_AV_ENABLE&&!(DWC_EQOS_NUM_TXQ>1&&(DWC_EQOS_TX_Q1_AV_EN||DWC_EQOS_TX_Q2_AV_EN||DWC_EQOS_TX_Q3_AV_EN||DWC_EQOS_TX_Q4_AV_EN||DWC_EQOS_TX_Q5_AV_EN||DWC_EQOS_TX_Q6_AV_EN||DWC_EQOS_TX_Q7_AV_EN)))?\"1\":\"0\") **Exists:** Always

Bits	Name	Memory Access	Description
20	AVSEL	R	AV Feature Enable This bit is set to 1 when the Enable Audio Video Bridging option is selected. **Values:** ■ 0x0 (INACTIVE): AV Feature is not selected ■ 0x1 (ACTIVE): AV Feature is selected **Value After Reset:** ((DWC_EQOS_AV_ENABLE)?\"1\":\"0\") **Exists:** Always
19	DBGMEMA	R	DMA Debug Registers Enable This bit is set to 1 when the Debug Mode Enable option is selected **Values:** ■ 0x0 (INACTIVE): DMA Debug Registers option is not selected ■ 0x1 (ACTIVE): DMA Debug Registers option is selected **Value After Reset:** ((DWC_EQOS_DBG_FIFO_ACCESS)?\"1\":\"0\") **Exists:** Always
18	TSOEN	R	TCP Segmentation Offload Enable This bit is set to 1 when the Enable TCP Segmentation Offloading for TCP/IP Packets option is selected **Values:** ■ 0x0 (INACTIVE): TCP Segmentation Offload Feature is not selected ■ 0x1 (ACTIVE): TCP Segmentation Offload Feature is selected **Value After Reset:** ((DWC_EQOS_TSO_EN)?\"1\":\"0\") **Exists:** Always
17	SPHEN	R	Split Header Feature Enable This bit is set to 1 when the Enable Split Header Structure option is selected **Values:** ■ 0x0 (INACTIVE): Split Header Feature is not selected ■ 0x1 (ACTIVE): Split Header Feature is selected **Value After Reset:** ((DWC_EQOS_SPLIT_HDR_EN)?\"1\":\"0\") **Exists:** Always

Bits	Name	Memory Access	Description
16	DCBEN	R	DCB Feature Enable This bit is set to 1 when the Enable Data Center Bridging option is selected **Values:** ■ 0x0 (INACTIVE): DCB Feature is not selected ■ 0x1 (ACTIVE): DCB Feature is selected **Value After Reset:** ((DWC_EQOS_DCB_EN)?\"1\":\"0\") **Exists:** Always
15:14	ADDR64	R	Address Width. This field indicates the configured address width: **Values:** ■ 0x0 (32): 32 ■ 0x1 (40): 40 ■ 0x2 (48): 48 ■ 0x3 (RSVD): Reserved **Value After Reset:** ((DWC_EQOS_ADDRWIDTH==48)?2:((DWC_EQOS_ADDRWIDTH==40)?1:0)) **Exists:** Always
13	ADVTHWORD	R	IEEE 1588 High Word Register Enable This bit is set to 1 when the Add IEEE 1588 Higher Word Register option is selected **Values:** ■ 0x0 (INACTIVE): IEEE 1588 High Word Register option is not selected ■ 0x1 (ACTIVE): IEEE 1588 High Word Register option is selected **Value After Reset:** ((DWC_EQOS_ADV_TIME_HIGH_WORD)?\"1\":\"0\") **Exists:** Always
12	PTOEN	R	PTP Offload Enable This bit is set to 1 when the Enable PTP Timestamp Offload Feature is selected. **Values:** ■ 0x0 (INACTIVE): PTP Offload feature is not selected ■ 0x1 (ACTIVE): PTP Offload feature is selected **Value After Reset:** ((DWC_EQOS_PTO_EN)?\"1\":\"0\") **Exists:** Always

Bits	Name	Memory Access	Description
11	OSTEN	R	One-Step Timestamping Enable This bit is set to 1 when the Enable One-Step Timestamp Feature is selected. **Values:** ■ 0x0 (INACTIVE): One-Step Timestamping feature is not selected ■ 0x1 (ACTIVE): One-Step Timestamping feature is selected **Value After Reset:** ((DWC_EQOS_OST_EN)?\"1\":\"0\") **Exists:** Always
10:6	TXFIFOSIZE	R	MTL Transmit FIFO Size This field contains the configured value of MTL Tx FIFO in bytes expressed as Log to base 2 minus 7, that is, Log2(TXFIFO_SIZE) -7: **Values:** ■ 0x0 (128B): 128 bytes ■ 0x1 (256B): 256 bytes ■ 0x2 (512B): 512 bytes ■ 0x3 (1024B): 1024 bytes ■ 0x4 (2048B): 2048 bytes ■ 0x5 (4096B): 4096 bytes ■ 0x6 (8192B): 8192 bytes ■ 0x7 (16384B): 16384 bytes ■ 0x8 (32KB): 32 KB ■ 0x9 (64KB): 64 KB ■ 0xa (128KB): 128 KB ■ 0xb (RSVD): Reserved **Value After Reset:** (DWC_EQOS_TXFIFOSZ) **Exists:** Always
5	SPRAM	R	Single Port RAM Enable This bit is set to 1 when the Use single port RAM Feature is selected. **Values:** ■ 0x0 (INACTIVE): Single Port RAM feature is not selected ■ 0x1 (ACTIVE): Single Port RAM feature is selected **Value After Reset:** ((DWC_EQOS_SPRAM)?\"1\":\"0\") **Exists:** Always

Bits	Name	Memory Access	Description
4:0	RXFIFOSIZE	R	MTL Receive FIFO Size This field contains the configured value of MTL Rx FIFO in bytes expressed as Log to base 2 minus 7, that is, Log2(RXFIFO_SIZE) -7: **Values:** ■ 0x0 (128B): 128 bytes ■ 0x1 (256B): 256 bytes ■ 0x2 (512B): 512 bytes ■ 0x3 (1024B): 1024 bytes ■ 0x4 (2048B): 2048 bytes ■ 0x5 (4096B): 4096 bytes ■ 0x6 (8192B): 8192 bytes ■ 0x7 (16384B): 16384 bytes ■ 0x8 (32KB): 32 KB ■ 0x9 (64KB): 64 KB ■ 0xa (128KB): 128 KB ■ 0xb (256KB): 256 KB ■ 0xc (RSVD): Reserved **Value After Reset:** (DWC_EQOS_RXFIFOSZ) **Exists:** Always

17.1.55 MAC_HW_Feature2

- **Description:** This register indicates the presence of third set of the optional features or functions of the DWC_ether_qos. The software driver can use this register to dynamically enable or disable the programs related to the optional blocks.
- **Size:** 32 bits
- **Offset:** 0x124
- **Exists:** Always

31	30:28	27	26:24	23:22	21:18	17:16	15:12	11:10	9:6	5:4	3:0
Reserved_31	AUXSNAPNUM	Reserved_27	PPSOUTNUM	Reserved_23_22	TXCHCNT	Reserved_17_16	RXCHCNT	Reserved_11_10	TXQCNT	Reserved_5_4	RXQCNT

Table 17-59 Fields for Register: MAC_HW_Feature2

Bits	Name	Memory Access	Description
31	Reserved_31	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
30:28	AUXSNAPNUM	R	Number of Auxiliary Snapshot Inputs This field indicates the number of auxiliary snapshot inputs: **Values:** ■ 0x0 (NO_AUXI): No auxiliary input ■ 0x1 (1_AUXI): 1 auxiliary input ■ 0x2 (2_AUXI): 2 auxiliary input ■ 0x3 (3_AUXI): 3 auxiliary input ■ 0x4 (4_AUXI): 4 auxiliary input ■ 0x5 (RSVD): Reserved **Value After Reset:** (DWC_EQOS_AUX_SNAP_IN_NUM) **Exists:** Always
27	Reserved_27	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
26:24	PPSOUTNUM	R	Number of PPS Outputs This field indicates the number of PPS outputs: **Values:** ■ 0x0 (NO_PPSO): No PPS output ■ 0x1 (1_PPSO): 1 PPS output ■ 0x2 (2_PPSO): 2 PPS output ■ 0x3 (3_PPSO): 3 PPS output ■ 0x4 (4_PPSO): 4 PPS output ■ 0x5 (RSVD): Reserved **Value After Reset:** (DWC_EQOS_PPS_OUT_NUM) **Exists:** Always
23:22	Reserved_23_22	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
21:18	TXCHCNT	R	Number of DMA Transmit Channels This field indicates the number of DMA Transmit channels: **Values:** ■ 0x0 (1TXCH): 1 MTL Tx Channel ■ 0x1 (2TXCH): 2 MTL Tx Channels ■ 0x2 (3TXCH): 3 MTL Tx Channels ■ 0x3 (4TXCH): 4 MTL Tx Channels ■ 0x4 (5TXCH): 5 MTL Tx Channels ■ 0x5 (6TXCH): 6 MTL Tx Channels ■ 0x6 (7TXCH): 7 MTL Tx Channels ■ 0x7 (8TXCH): 8 MTL Tx Channels **Value After Reset:** ((DWC_EQOS_NUM_DMA_TX_CH>7)?\"7\":(DWC_EQOS_NUM_DMA_TX_CH>6)?\"6\":(DWC_EQOS_NUM_DMA_TX_CH>5)?\"5\":(DWC_EQOS_NUM_DMA_TX_CH>4)?\"4\":(DWC_EQOS_NUM_DMA_TX_CH>3)?\"3\":(DWC_EQOS_NUM_DMA_TX_CH>2)?\"2\":(DWC_EQOS_NUM_DMA_TX_CH>1)?\"1\":\"0\") **Exists:** Always
17:16	Reserved_17_16	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
15:12	RXCHCNT	R	Number of DMA Receive Channels This field indicates the number of DMA Receive channels: **Values:** ■ 0x0 (1RXCH): 1 MTL Rx Channel ■ 0x1 (2RXCH): 2 MTL Rx Channels ■ 0x2 (3RXCH): 3 MTL Rx Channels ■ 0x3 (4RXCH): 4 MTL Rx Channels ■ 0x4 (5RXCH): 5 MTL Rx Channels ■ 0x5 (6RXCH): 6 MTL Rx Channels ■ 0x6 (7RXCH): 7 MTL Rx Channels ■ 0x7 (8RXCH): 8 MTL Rx Channels **Value After Reset:** ((DWC_EQOS_NUM_DMA_RX_CH>7)?\"7\":(DWC_EQOS_NUM_DMA_RX_CH>6)?\"6\":(DWC_EQOS_NUM_DMA_RX_CH>5)?\"5\":(DWC_EQOS_NUM_DMA_RX_CH>4)?\"4\":(DWC_EQOS_NUM_DMA_RX_CH>3)?\"3\":(DWC_EQOS_NUM_DMA_RX_CH>2)?\"2\":(DWC_EQOS_NUM_DMA_RX_CH>1)?\"1\":\"0\") **Exists:** Always
11:10	Reserved_11_10	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
9:6	TXQCNT	R	Number of MTL Transmit Queues This field indicates the number of MTL Transmit queues: **Values:** ■ 0x0 (1TXQ): 1 MTL Tx Queue ■ 0x1 (2TXQ): 2 MTL Tx Queues ■ 0x2 (3TXQ): 3 MTL Tx Queues ■ 0x3 (4TXQ): 4 MTL Tx Queues ■ 0x4 (5TXQ): 5 MTL Tx Queues ■ 0x5 (6TXQ): 6 MTL Tx Queues ■ 0x6 (7TXQ): 7 MTL Tx Queues ■ 0x7 (8TXQ): 8 MTL Tx Queues **Value After Reset:** ((DWC_EQOS_NUM_TXQ>7)?\"7\":(DWC_EQOS_NUM_TXQ>6)?\"6\":(DWC_EQOS_NUM_TXQ>5)?\"5\":(DWC_EQOS_NUM_TXQ>4)?\"4\":(DWC_EQOS_NUM_TXQ>3)?\"3\":(DWC_EQOS_NUM_TXQ>2)?\"2\":(DWC_EQOS_NUM_TXQ>1)?\"1\":\"0\") **Exists:** Always
5:4	Reserved_5_4	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
3:0	RXQCNT	R	Number of MTL Receive Queues This field indicates the number of MTL Receive queues: **Values:** ■ 0x0 (1RXQ): 1 MTL Rx Queue ■ 0x1 (2RXQ): 2 MTL Rx Queues ■ 0x2 (3RXQ): 3 MTL Rx Queues ■ 0x3 (4RXQ): 4 MTL Rx Queues ■ 0x4 (5RXQ): 5 MTL Rx Queues ■ 0x5 (6RXQ): 6 MTL Rx Queues ■ 0x6 (7RXQ): 7 MTL Rx Queues ■ 0x7 (8RXQ): 8 MTL Rx Queues **Value After Reset:** ((DWC_EQOS_NUM_RXQ>7)?\"7\":(DWC_EQOS_NUM_RXQ>6)?\"6\":(DWC_EQOS_NUM_RXQ>5)?\"5\":(DWC_EQOS_NUM_RXQ>4)?\"4\":(DWC_EQOS_NUM_RXQ>3)?\"3\":(DWC_EQOS_NUM_RXQ>2)?\"2\":(DWC_EQOS_NUM_RXQ>1)?\"1\":\"0\") **Exists:** Always

17.1.56 MAC_HW_Feature3

- **Description:** This register indicates the presence of fourth set the optional features or functions of the DWC_ether_qos. The software driver can use this register to dynamically enable or disable the programs related to the optional blocks.
- **Size:** 32 bits
- **Offset:** 0x128
- **Exists:** Always

31:30	29:28	27	26	25:22	21:20	19:17	16	15	14:13	12:11	10	9	8:6	5	4	3	2:0
Reserved_31_30	ASP	TBSSEL	FPESEL	Reserved_25_22	ESTWID	ESTDEP	ESTSEL	Reserved_15	FRPES	FRPBS	FRPSEL	PDUPSEL	Reserved_7_6	DVLAN	CBTISEL	Reserved_3	NRVF

Table 17-60 Fields for Register: MAC_HW_Feature3

Bits	Name	Memory Access	Description
31:30	Reserved_31_30	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
29:28	ASP	R	Automotive Safety Package Following are the encoding for the different Safety features **Values:** ■ 0x0 (NONE): No Safety features selected ■ 0x1 (ECC_ONLY): Only "ECC protection for external memory" feature is selected ■ 0x2 (AS_NPPE): All the Automotive Safety features are selected without the "Parity Port Enable for external interface" feature ■ 0x3 (AS_PPE): All the Automotive Safety features are selected with the "Parity Port Enable for external interface" feature **Value After Reset:** ((DWC_EQOS_ASP_ALL&&DWC_EQOS_ASP_PPE)?\"3\":(DWC_EQOS_ASP_ALL&&DWC_EQOS_ASP_PPE==0)?\"2\" :(DWC_EQOS_ASP_ECC)?\"1\":\"0\") **Exists:** Always

Bits	Name	Memory Access	Description
27	TBSSEL	R	Time Based Scheduling Enable This bit is set to 1 when the Time Based Scheduling feature is selected. **Values:** ■ 0x0 (INACTIVE): Time Based Scheduling Enable feature is not selected ■ 0x1 (ACTIVE): Time Based Scheduling Enable feature is selected **Value After Reset:** ((DWC_EQOS_TBS)?\"1\":\"0\") **Exists:** Always
26	FPESEL	R	Frame Preemption Enable This bit is set to 1 when the Enable Frame preemption feature is selected. **Values:** ■ 0x0 (INACTIVE): Frame Preemption Enable feature is not selected ■ 0x1 (ACTIVE): Frame Preemption Enable feature is selected **Value After Reset:** ((DWC_EQOS_FPE)?\"1\":\"0\") **Exists:** Always
25:22	Reserved_25_22	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
21:20	ESTWID	R	Width of the Time Interval field in the Gate Control List This field indicates the width of the Configured Time Interval Field **Values:** ■ 0x0 (NOWIDTH): Width not configured ■ 0x1 (WIDTH16): 16 ■ 0x2 (WIDTH20): 20 ■ 0x3 (WIDTH24): 24 **Value After Reset:** ((DWC_EQOS_AV_EST==0)?\"0\":(DWC_EQOS_EST_WID==24)?\"3\":(DWC_EQOS_EST_WID==20)?\"2\":(DWC_EQOS_EST_WID==16)?\"1\":\"0\") **Exists:** Always

Bits	Name	Memory Access	Description
19:17	ESTDEP	R	Depth of the Gate Control List This field indicates the depth of Gate Control list expressed as Log2(DWC_EQOS_EST_DEP)-5 **Values:** ■ 0x0 (NODEPTH): No Depth configured ■ 0x1 (DEPTH64): 64 ■ 0x2 (DEPTH128): 128 ■ 0x3 (DEPTH256): 256 ■ 0x4 (DEPTH512): 512 ■ 0x5 (DEPTH1024): 1024 ■ 0x6 (RSVD): Reserved **Value After Reset:** ((DWC_EQOS_AV_EST==0)?\"0\":(DWC_EQOS_EST_DEP==1024)?\"5\":(DWC_EQOS_EST_DEP==512)?\"4\":(DWC_EQOS_EST_DEP==256)?\"3\":(DWC_EQOS_EST_DEP==128)?\"2\":(DWC_EQOS_EST_DEP==64)?\"1\":\"0\") **Exists:** Always
16	ESTSEL	R	Enhancements to Scheduling Traffic Enable This bit is set to 1 when the Enable Enhancements to Scheduling Traffic feature is selected. **Values:** ■ 0x0 (INACTIVE): Enable Enhancements to Scheduling Traffic feature is not selected ■ 0x1 (ACTIVE): Enable Enhancements to Scheduling Traffic feature is selected **Value After Reset:** ((DWC_EQOS_AV_EST)?\"1\":\"0\") **Exists:** Always
15	Reserved_15	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
14:13	FRPES	R	Flexible Receive Parser Table Entries size This field indicates the Max Number of Parser Entries supported by Flexible Receive Parser. **Values:** ■ 0x0 (64ENTR): 64 Entries ■ 0x1 (128ENTR): 128 Entries ■ 0x2 (256ENTR): 256 Entries ■ 0x3 (RSVD): Reserved **Value After Reset:** ((DWC_EQOS_FRP_EN==0)?\"0\":(DWC_EQOS_FRP_ENTRIES==256)?\"2\":(DWC_EQOS_FRP_ENTRIES==128)?\"1\":\"0\") **Exists:** Always

Bits	Name	Memory Access	Description
12:11	FRPBS	R	Flexible Receive Parser Buffer size This field indicates the supported Max Number of bytes of the packet data to be Parsed by Flexible Receive Parser. **Values:** ■ 0x0 (64BYTES): 64 Bytes ■ 0x1 (128BYTES): 128 Bytes ■ 0x2 (256BYTES): 256 Bytes ■ 0x3 (RSVD): Reserved **Value After Reset:** ((DWC_EQOS_FRP_EN==0)?\"0\":(DWC_EQOS_FRP_BUF_SIZE==256)?\"2\":(DWC_EQOS_FRP_BUF_SIZE==128)?\"1\":\"0\") **Exists:** Always
10	FRPSEL	R	Flexible Receive Parser Selected This bit is set to 1 when the Enable Flexible Programmable Receive Parser option is selected. **Values:** ■ 0x0 (INACTIVE): Flexible Receive Parser feature is not selected ■ 0x1 (ACTIVE): Flexible Receive Parser feature is selected **Value After Reset:** ((DWC_EQOS_FRP_EN)?\"1\":\"0\") **Exists:** Always
9	PDUPSEL	R	Broadcast/Multicast Packet Duplication This bit is set to 1 when the Broadcast/Multicast Packet Duplication feature is selected. **Values:** ■ 0x0 (INACTIVE): Broadcast/Multicast Packet Duplication feature is not selected ■ 0x1 (ACTIVE): Broadcast/Multicast Packet Duplication feature is selected **Value After Reset:** ((DWC_EQOS_PDUP)?\"1\":\"0\") **Exists:** Always
8:6	Reserved_7_6	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
5	DVLAN	R	Double VLAN Tag Processing Selected This bit is set to 1 when the Enable Double VLAN Processing Feature is selected. **Values:** ■ 0x0 (INACTIVE): Double VLAN option is not selected ■ 0x1 (ACTIVE): Double VLAN option is selected **Value After Reset:** ((DWC_EQOS_DOUBLE_VLAN_EN)?\"1\":\"0\") **Exists:** Always
4	CBTISEL	R	Queue/Channel based VLAN tag insertion on Tx Enable This bit is set to 1 when the Enable Queue/Channel based VLAN tag insertion on Tx Feature is selected. **Values:** ■ 0x0 (INACTIVE): Enable Queue/Channel based VLAN tag insertion on Tx feature is not selected ■ 0x1 (ACTIVE): Enable Queue/Channel based VLAN tag insertion on Tx feature is selected **Value After Reset:** ((DWC_EQOS_CBTI_EN)?\"1\":\"0\") **Exists:** Always
3	Reserved_3	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
2:0	NRVF	R	Number of Extended VLAN Tag Filters Enabled This field indicates the Number of Extended VLAN Tag Filters selected: **Values:** ■ 0x0 (NO_ERVLAN): No Extended Rx VLAN Filters ■ 0x1 (4_ERVLAN): 4 Extended Rx VLAN Filters ■ 0x2 (8_ERVLAN): 8 Extended Rx VLAN Filters ■ 0x3 (16_ERVLAN): 16 Extended Rx VLAN Filters ■ 0x4 (24_ERVLAN): 24 Extended Rx VLAN Filters ■ 0x5 (32_ERVLAN): 32 Extended Rx VLAN Filters ■ 0x6 (RSVD): Reserved **Value After Reset:** ((DWC_EQOS_NRVF==32)?\"5\":(DWC_EQOS_NRVF==24)?\"4\":(DWC_EQOS_NRVF==16)?\"3\":(DWC_EQOS_NRVF==8)?\"2\":(DWC_EQOS_NRVF==4)?\"1\":\"0\") **Exists:** Always

17.1.57 MAC_DPP_FSM_Interrupt_Status

- **Description:** This register contains the status of Automotive Safety related Data Path Parity Errors, Interface Timeout Errors, FSM State Parity Errors and FSM State Timeout Errors.
- **Size:** 32 bits
- **Offset:** 0x140
- **Exists:** (DWC_EQOS_ASP_DPP||DWC_EQOS_ASP_FSM||DWC_EQOS_ASP_ACT)

31:25	24	23:18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved_31_25	FSMPES	Reserved_23_18	SLVTES	MSTTES	RVCTES	R125ES	T125ES	PTES	ATES	CTES	RTES	TTES	ASRPES	CWPES	ARPES	MTSPES	MPES	RDPES	TPES	ATPES

Table 17-61 Fields for Register: MAC_DPP_FSM_Interrupt_Status

Bits	Name	Memory Access	Description
31:25	Reserved_31_25	R	**Value After Reset:** 0x0 **Exists:** Always
24	FSMPES	R/W	FSM State Parity Error Status This field when set indicates one of the FSMs State registers has a parity error detected. **Values:** ■ 0x0 (INACTIVE): FSM State Parity Error Status not detected ■ 0x1 (ACTIVE): FSM State Parity Error Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_ASP_ALL **Testable:** untestable
23:18	Reserved_23_18	R	**Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
17	SLVTES	R/W	Slave Read/Write Timeout Error Status This field when set indicates that an Application/CSR Timeout has occurred on the AXI slave interface. **Values:** ■ 0x0 (INACTIVE): Slave Read/Write Timeout Error Status not detected ■ 0x1 (ACTIVE): Slave Read/Write Timeout Error Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_ASP_ACT&&(DWC_EQOS_AXI_SLAVE||DWC_EQOS_AHB_SLAVE) **Testable:** untestable
16	MSTTES	R/W	Master Read/Write Timeout Error Status This field when set indicates that an Application/CSR Timeout has occurred on the master (AXI/AHB/ARI/ATI) interface. **Values:** ■ 0x0 (INACTIVE): Master Read/Write Timeout Error Status not detected ■ 0x1 (ACTIVE): Master Read/Write Timeout Error Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_ASP_ACT&&(DWC_EQOS_AXI_SUBSYS||DWC_EQOS_AHB_SUBSYS||DWC_EQOS_MTL_SUBSYS) **Testable:** untestable
15	RVCTES	R/W	REV MDC FSM Timeout Error Status This field when set indicates that one of the Rev Mdc FSM Timeout has occurred. **Values:** ■ 0x0 (INACTIVE): REV MDC FSM Timeout Error Status not detected ■ 0x1 (ACTIVE): REV MDC FSM Timeout Error Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_ASP_FSM&&(DWC_EQOS_REVMII_EN) **Testable:** untestable

Bits	Name	Memory Access	Description
14	R125ES	R/W	Rx125 FSM Timeout Error Status This field when set indicates that one of the Rx125 FSM Timeout has occurred. **Values:** ■ 0x0 (INACTIVE): Rx125 FSM Timeout Error Status not detected ■ 0x1 (ACTIVE): Rx125 FSM Timeout Error Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_ASP_FSM&&(DWC_EQOS_PCS_EN||DWC_EQOS_SSSMII_EN) **Testable:** untestable
13	T125ES	R/W	Tx125 FSM Timeout Error Status This field when set indicates that one of the Tx125 FSM Timeout has occurred. **Values:** ■ 0x0 (INACTIVE): Tx125 FSM Timeout Error Status not detected ■ 0x1 (ACTIVE): Tx125 FSM Timeout Error Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_ASP_FSM&&(DWC_EQOS_PCS_EN||DWC_EQOS_SMII_EN) **Testable:** untestable
12	PTES	R/W	PTP FSM Timeout Error Status This field when set indicates that one of the PTP FSM Timeout has occurred. **Values:** ■ 0x0 (INACTIVE): PTP FSM Timeout Error Status not detected ■ 0x1 (ACTIVE): PTP FSM Timeout Error Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_ASP_FSM&&(DWC_EQOS_AV_EST||DWC_EQOS_FLEXI_PPS_OUT_EN) **Testable:** untestable

Bits	Name	Memory Access	Description
11	ATES	R/W	APP FSM Timeout Error Status This field when set indicates that one of the APP FSM Timeout has occurred. **Values:** ■ 0x0 (INACTIVE): APP FSM Timeout Error Status not detected ■ 0x1 (ACTIVE): APP FSM Timeout Error Status detected **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_ASP_FSM&&(DWC_EQOS_SYS>0||((DWC_EQOS_EEE_EN||DWC_EQOS_SMA_EN)&&!(DWC_EQOS_CORE||DWC_EQOS_CSR_SLV_CLK))))||(DWC_EQOS_ASP_ACT&&((DWC_EQOS_AXI_SUBSYS||DWC_EQOS_AHB_SUBSYS)&&!DWC_EQOS_CSR_SLV_CLK)) **Testable:** untestable
10	CTES	R/W	CSR FSM Timeout Error Status This field when set indicates that one of the CSR FSM Timeout has occurred. **Values:** ■ 0x0 (INACTIVE): CSR FSM Timeout Error Status not detected ■ 0x1 (ACTIVE): CSR FSM Timeout Error Status detected **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_ASP_FSM&&(((DWC_EQOS_EEE_EN||DWC_EQOS_SMA_EN)&&(DWC_EQOS_CORE||DWC_EQOS_CSR_SLV_CLK))||((DWC_EQOS_AXI_SLAVE||DWC_EQOS_AHB_SLAVE)&&DWC_EQOS_CSR_SLV_CLK)))||(DWC_EQOS_ASP_ACT&&(DWC_EQOS_AXI_SLAVE||DWC_EQOS_AHB_SLAVE)&&DWC_EQOS_CSR_SLV_CLK) **Testable:** untestable
9	RTES	R/W	Rx FSM Timeout Error Status This field when set indicates that one of the Rx FSM Timeout has occurred. **Values:** ■ 0x0 (INACTIVE): Rx FSM Timeout Error Status not detected ■ 0x1 (ACTIVE): Rx FSM Timeout Error Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_ASP_ALL **Testable:** untestable

Bits	Name	Memory Access	Description
8	TTES	R/W	Tx FSM Timeout Error Status This field when set indicates that one of the Tx FSM Timeout has occurred. **Values:** ■ 0x0 (INACTIVE): Tx FSM Timeout Error Status not detected ■ 0x1 (ACTIVE): Tx FSM Timeout Error Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_ASP_ALL **Testable:** untestable
7	ASRPES	R/W	AXI Slave Read data path Parity checker Error Status This bit when set indicates that parity error is detected at the AXI Slave read data interface on "rdata_s_o" or at PC9 checker as shown in the Fig.AXI slave Interface Data path parity protection. **Values:** ■ 0x0 (INACTIVE): AXI Slave Read data path Parity checker Error Status not detected ■ 0x1 (ACTIVE): AXI Slave Read data path Parity checker Error Status detected **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_ASP_DPP&&DWC_EQOS_AXI_SLAVE) **Testable:** untestable
6	CWPES	R/W	CSR Write data path Parity checker Error Status This bit when set indicates that parity error is detected at the CSR write data interface on mci_wdata_i (or at PC8 checker as shown in Fig.AXI slave Interface Data path parity protection). When EPSI bit of MTL_DPP_Control register is set and if any parity mis-match is detected on the input slave parity ports (or at PC7 checker as shown in the Fig.AXI slave Interface Data path parity protection) will set this bit to one. **Values:** ■ 0x0 (INACTIVE): CSR Write data path Parity checker Error Status not detected ■ 0x1 (ACTIVE): CSR Write data path Parity checker Error Status detected **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_ASP_DPP&&DWC_EQOS_AXI_SLAVE) **Testable:** untestable

Bits	Name	Memory Access	Description
5	ARPES	R/W	Application Receive interface data path Parity Error Status This bit when set indicates that a parity error is detected at following checkers based on the system configuration as described below ■ In MTL configuration (DWC_EQOS_SYS=1), parity checker (PC6 as shown in Fig.Receive Data path Parity protection) at ARI interface. ■ In DMA configuration (DWC_EQOS_SYS=2), parity checker (PC6 as shown in Fig.Receive Data path Parity protection) at DMA application interface. ■ In AHB configuration (DWC_EQOS_SYS=3), parity checker (PC6 as shown in Fig.Receive Data path Parity protection) at AHB master interface. ■ In AXI configuration (DWC_EQOS_SYS=4), parity checker (PC6 as shown in Fig.Receive Data path Parity protection) at AXI master interface. **Values:** ■ 0x0 (INACTIVE): Application Receive interface data path Parity Error Status not detected ■ 0x1 (ACTIVE): Application Receive interface data path Parity Error Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_ASP_ALL **Testable:** untestable
4	MTSPES	R/W	MTL TX Status data path Parity checker Error Status This filed when set indicates that, parity error is detected on the MTL TX Status data on ati interface (or at PC5 as shown in Fig.Transmit data path parity protection). **Values:** ■ 0x0 (INACTIVE): MTL TX Status data path Parity checker Error Status not detected ■ 0x1 (ACTIVE): MTL TX Status data path Parity checker Error Status detected **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_ASP_DPP&&DWC_EQOS_SYS>1&&DWC_EQOS_TX_STS_DROP==0) **Testable:** untestable

Bits	Name	Memory Access	Description
3	MPES	R/W	MTL data path Parity checker Error Status This bit when set indicates that a parity error is detected at the MTL transmit write controller parity checker (or at PC4 as shown in Fig.Transmit data path parity protection). **Values:** ■ 0x0 (INACTIVE): MTL data path Parity checker Error Status not detected ■ 0x1 (ACTIVE): MTL data path Parity checker Error Status detected **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_ASP_DPP&&DWC_EQOS_SYS>0) **Testable:** untestable
2	RDPES	R/W	Read Descriptor Parity checker Error Status This bit when set indicates that a parity error is detected at the DMA Read descriptor parity checker (or at PC3 as shown in Fig.Transmit data path parity protection). **Values:** ■ 0x0 (INACTIVE): Read Descriptor Parity checker Error Status not detected ■ 0x1 (ACTIVE): Read Descriptor Parity checker Error Status detected **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_ASP_DPP&&DWC_EQOS_SYS>1) **Testable:** untestable
1	TPES	R/W	TSO data path Parity checker Error Status This bit when set indicates that a parity error is detected at the DMA TSO parity checker (or at PC2 as shown in Fig.Transmit data path parity protection). **Values:** ■ 0x0 (INACTIVE): TSO data path Parity checker Error Status not detected ■ 0x1 (ACTIVE): TSO data path Parity checker Error Status detected **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_ASP_DPP&&DWC_EQOS_TSO_EN) **Testable:** untestable

Bits	Name	Memory Access	Description
0	ATPES	R/W	Application Transmit Interface Parity checker Error Status This bit when set indicates that a parity error is detected on the AXI/AHB Master read data parity checker. This bit when set indicates that a parity error is detected on the interface port parity checker. Following are the checkers located based on the system configuration, ■ In MTL configuration (DWC_EQOS_SYS=1), parity checker (PC1 as shown in Fig.Transmit data path parity protection) at ATI interface. ■ In DMA configuration (DWC_EQOS_SYS=2), parity checker (PC1 as shown in Fig.Transmit data path parity protection) at DMA application interface. ■ In AHB configuration (DWC_EQOS_SYS=3), parity checker (PC1 as shown in Fig.Transmit data path parity protection) at AHB master interface. ■ In AXI configuration (DWC_EQOS_SYS=4), parity checker (PC1 as shown in Fig.Transmit data path parity protection) at AXI master interface. **Values:** ■ 0x0 (INACTIVE): Application Transmit Interface Parity checker Error Status not detected ■ 0x1 (ACTIVE): Application Transmit Interface Parity checker Error Status detected **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_ASP_DPP&&DWC_EQOS_ASP_PPE) **Testable:** untestable

17.1.58 MAC_AXI_SLV_DPE_Addr_Status

- **Description:** This register indicates the CSR address corresponding to the CSR write data on which parity error occured.
- **Size:** 32 bits
- **Offset:** 0x144
- **Exists:** (DWC_EQOS_ASP_DPP&&DWC_EQOS_AXI_SLAVE)

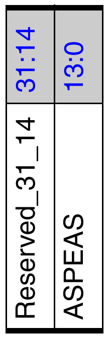


Table 17-62 Fields for Register: MAC_AXI_SLV_DPE_Addr_Status

Bits	Name	Memory Access	Description
31:14	Reserved_31_14	R	**Value After Reset:** 0x0 **Exists:** Always
13:0	ASPEAS	R	AXI Slave data path Parity Error Address Status This field holds the CSR address for which parity error is detected on the CSR write data. This field holds the first address for which parity error is detected on the write data and will be cleared on read. **Value After Reset:** 0x0 **Exists:** Always

17.1.59 MAC_FSM_Control

- **Description:** This register is used to control the FSM State parity and timeout error injection in Debug mode.
- **Size:** 32 bits
- **Offset:** 0x148
- **Exists:** (DWC_EQOS_ASP_FSM||DWC_EQOS_ASP_ACT)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7:2	1	0
RVCLGRNML	R125LGRNML	T125LGRNML	PLGRNML	ALGRNML	CLGRNML	RLGRNML	TLGRNML	RVCPEIN	R125PEIN	T125PEIN	PPEIN	APEIN	CPEIN	RPEIN	TPEIN	RVCTEIN	R125TEIN	T125TEIN	PTEIN	ATEIN	CTEIN	RTEIN	TTEIN	Reserved_7_2	PRTYEN	TMOUTEN

Table 17-63 Fields for Register: MAC_FSM_Control

Bits	Name	Memory Access	Description
31	RVCLGRNML	R/W	REV MDC Large/Normal Mode Select This field when set indicates that large mode tic generation is used for RevMII MDC domain, else normal mode tic generation is used. **Values:** ■ 0x0 (DISABLE): normal mode tic generation is used for REVMII MDC domain ■ 0x1 (ENABLE): large mode tic generation is used for REVMII MDC domain **Value After Reset:** 0x0 **Exists:** DWC_EQOS_ASP_FSM&&(DWC_EQOS_REVMII_EN)
30	R125LGRNML	R/W	Rx125 Large/Normal Mode Select This field when set indicates that large mode tic generation is used for Rx125 domain, else normal mode tic generation is used. **Values:** ■ 0x0 (DISABLE): normal mode tic generation is used for Rx125 domain ■ 0x1 (ENABLE): large mode tic generation is used for Rx125 domain **Value After Reset:** 0x0 **Exists:** DWC_EQOS_ASP_FSM&&(DWC_EQOS_PCS_EN||DWC_EQOS_SSSMII_EN)

Bits	Name	Memory Access	Description
29	T125LGRNML	R/W	Tx125 Large/Normal Mode Select This field when set indicates that large mode tic generation is used for Tx125 domain, else normal mode tic generation is used. **Values:** ■ 0x0 (DISABLE): normal mode tic generation is used for Tx125 domain ■ 0x1 (ENABLE): large mode tic generation is used for Tx125 domain **Value After Reset:** 0x0 **Exists:** DWC_EQOS_ASP_FSM&&(DWC_EQOS_PCS_EN||DWC_EQOS_SMII_EN)
28	PLGRNML	R/W	PTP Large/Normal Mode Select This field when set indicates that large mode tic generation is used for PTP domain, else normal mode tic generation is used. **Values:** ■ 0x0 (DISABLE): normal mode tic generation is used for PTP domain ■ 0x1 (ENABLE): large mode tic generation is used for PTP domain **Value After Reset:** 0x0 **Exists:** DWC_EQOS_ASP_FSM&&(DWC_EQOS_AV_EST||DWC_EQOS_FLEXI_PPS_OUT_EN)
27	ALGRNML	R/W	APP Large/Normal Mode Select This field when set indicates that large mode tic generation is used for APP domain, else normal mode tic generation is used. **Values:** ■ 0x0 (DISABLE): normal mode tic generation is used for APP domain ■ 0x1 (ENABLE): large mode tic generation is used for APP domain **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_ASP_FSM&&(DWC_EQOS_SYS>0||((DWC_EQOS_EEE_EN||DWC_EQOS_SMA_EN)&&!(DWC_EQOS_CORE||DWC_EQOS_CSR_SLV_CLK))))||(DWC_EQOS_ASP_ACT&&((DWC_EQOS_AXI_SUBSYS||DWC_EQOS_AHB_SUBSYS)&&!DWC_EQOS_CSR_SLV_CLK))

Bits	Name	Memory Access	Description
26	CLGRNML	R/W	CSR Large/Normal Mode Select This field when set indicates that large mode tic generation is used for CSR domain, else normal mode tic generation is used. **Values:** ■ 0x0 (DISABLE): normal mode tic generation is used for CSR domain ■ 0x1 (ENABLE): large mode tic generation is used for CSR domain **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_ASP_FSM&&(((DWC_EQOS_EEE_EN||DWC_EQOS_SMA_EN)&&(DWC_EQOS_CORE||DWC_EQOS_CSR_SLV_CLK))||((DWC_EQOS_AXI_SLAVE||DWC_EQOS_AHB_SLAVE)&&DWC_EQOS_CSR_SLV_CLK)))||(DWC_EQOS_ASP_ACT&&(DWC_EQOS_AXI_SLAVE||DWC_EQOS_AHB_SLAVE)&&DWC_EQOS_CSR_SLV_CLK)
25	RLGRNML	R/W	Rx Large/Normal Mode Select This field when set indicates that large mode tic generation is used for Rx domain, else normal mode tic generation is used. **Values:** ■ 0x0 (DISABLE): normal mode tic generation is used for Rx domain ■ 0x1 (ENABLE): large mode tic generation is used for Rx domain **Value After Reset:** 0x0 **Exists:** DWC_EQOS_ASP_ALL
24	TLGRNML	R/W	Tx Large/Normal Mode Select This field when set indicates that large mode tic generation is used for Tx domain, else normal mode tic generation is used. **Values:** ■ 0x0 (DISABLE): normal mode tic generation is used for Tx domain ■ 0x1 (ENABLE): large mode tic generation is used for Tx domain **Value After Reset:** 0x0 **Exists:** DWC_EQOS_ASP_ALL

Bits	Name	Memory Access	Description
23	RVCPEIN	R/W	REV MDC FSM Parity Error Injection This field when set indicates that Error Injection for REVMII MDC FSM Parity is enabled. **Values:** ■ 0x0 (DISABLE): REV MDC FSM Parity Error Injection is disabled ■ 0x1 (ENABLE): REV MDC FSM Parity Error Injection is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_ASP_FSM&&(DWC_EQOS_REVMII_EN)
22	R125PEIN	R/W	Rx125 FSM Parity Error Injection This field when set indicates that Error Injection for RX125 FSM Parity is enabled. **Values:** ■ 0x0 (DISABLE): Rx125 FSM Parity Error Injection is disabled ■ 0x1 (ENABLE): Rx125 FSM Parity Error Injection is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_ASP_FSM&&(DWC_EQOS_PCS_EN||DWC_EQOS_SSSMII_EN)
21	T125PEIN	R/W	Tx125 FSM Parity Error Injection This field when set indicates that Error Injection for TX125 FSM Parity is enabled. **Values:** ■ 0x0 (DISABLE): Tx125 FSM Parity Error Injection is disabled ■ 0x1 (ENABLE): Tx125 FSM Parity Error Injection is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_ASP_FSM&&(DWC_EQOS_PCS_EN||DWC_EQOS_SMII_EN)
20	PPEIN	R/W	PTP FSM Parity Error Injection This field when set indicates that Error Injection for PTP FSM Parity is enabled. **Values:** ■ 0x0 (DISABLE): PTP FSM Parity Error Injection is disabled ■ 0x1 (ENABLE): PTP FSM Parity Error Injection is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_ASP_FSM&&(DWC_EQOS_AV_EST||DWC_EQOS_FLEXI_PPS_OUT_EN)

Bits	Name	Memory Access	Description
19	APEIN	R/W	APP FSM Parity Error Injection This field when set indicates that Error Injection for APP FSM Parity is enabled. **Values:** ■ 0x0 (DISABLE): APP FSM Parity Error Injection is disabled ■ 0x1 (ENABLE): APP FSM Parity Error Injection is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_ASP_FSM&&(DWC_EQOS_SYS>0||((DWC_EQOS_EEE_EN||DWC_EQOS_SMA_EN)&&!(DWC_EQOS_CORE||DWC_EQOS_CSR_SLV_CLK)))
18	CPEIN	R/W	CSR FSM Parity Error Injection This field when set indicates that Error Injection for CSR Parity is enabled. **Values:** ■ 0x0 (DISABLE): CSR FSM Parity Error Injection is disabled ■ 0x1 (ENABLE): CSR FSM Parity Error Injection is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_ASP_FSM&&(((DWC_EQOS_EEE_EN||DWC_EQOS_SMA_EN)&&(DWC_EQOS_CORE||DWC_EQOS_CSR_SLV_CLK))||((DWC_EQOS_AXI_SLAVE||DWC_EQOS_AHB_SLAVE)&&DWC_EQOS_CSR_SLV_CLK))
17	RPEIN	R/W	Rx FSM Parity Error Injection This field when set indicates that Error Injection for RX FSM Parity is enabled. **Values:** ■ 0x0 (DISABLE): Rx FSM Parity Error Injection is disabled ■ 0x1 (ENABLE): Rx FSM Parity Error Injection is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_ASP_ALL
16	TPEIN	R/W	Tx FSM Parity Error Injection This field when set indicates that Error Injection for TX FSM Parity is enabled. **Values:** ■ 0x0 (DISABLE): Tx FSM Parity Error Injection is disabled ■ 0x1 (ENABLE): Tx FSM Parity Error Injection is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_ASP_ALL

Bits	Name	Memory Access	Description
15	RVCTEIN	R/W	REV MDC FSM Timeout Error Injection This field when set indicates that Error Injection for REVMII MDC FSM timeout is enabled. **Values:** ■ 0x0 (DISABLE): REV MDC FSM Timeout Error Injection is disabled ■ 0x1 (ENABLE): REV MDC FSM Timeout Error Injection is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_ASP_FSM&&(DWC_EQOS_REVMII_EN)
14	R125TEIN	R/W	Rx125 FSM Timeout Error Injection This field when set indicates that Error Injection for RX125 FSM timeout is enabled. **Values:** ■ 0x0 (DISABLE): Rx125 FSM Timeout Error Injection is disabled ■ 0x1 (ENABLE): Rx125 FSM Timeout Error Injection is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_ASP_FSM&&(DWC_EQOS_PCS_EN||DWC_EQOS_SSSMII_EN)
13	T125TEIN	R/W	Tx125 FSM Timeout Error Injection This field when set indicates that Error Injection for TX125 FSM timeout is enabled. **Values:** ■ 0x0 (DISABLE): Tx125 FSM Timeout Error Injection is disabled ■ 0x1 (ENABLE): Tx125 FSM Timeout Error Injection is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_ASP_FSM&&(DWC_EQOS_PCS_EN||DWC_EQOS_SMII_EN)

Bits	Name	Memory Access	Description
12	PTEIN	R/W	PTP FSM Timeout Error Injection This field when set indicates that Error Injection for PTP FSM timeout is enabled. **Values:** ■ 0x0 (DISABLE): PTP FSM Timeout Error Injection is disabled ■ 0x1 (ENABLE): PTP FSM Timeout Error Injection is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_ASP_FSM&&(DWC_EQOS_AV_EST||DWC_EQOS_FLEXI_PPS_OUT_EN)
11	ATEIN	R/W	APP FSM Timeout Error Injection This field when set indicates that Error Injection for APP FSM timeout is enabled. **Values:** ■ 0x0 (DISABLE): APP FSM Timeout Error Injection is disabled ■ 0x1 (ENABLE): APP FSM Timeout Error Injection is enabled **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_ASP_FSM&&(DWC_EQOS_SYS>0||((DWC_EQOS_EEE_EN||DWC_EQOS_SMA_EN)&&!(DWC_EQOS_CORE||DWC_EQOS_CSR_SLV_CLK))))||(DWC_EQOS_ASP_ACT&&((DWC_EQOS_AXI_SUBSYS||DWC_EQOS_AHB_SUBSYS)&&!DWC_EQOS_CSR_SLV_CLK))
10	CTEIN	R/W	CSR FSM Timeout Error Injection This field when set indicates that Error Injection for CSR timeout is enabled. **Values:** ■ 0x0 (DISABLE): CSR FSM Timeout Error Injection is disabled ■ 0x1 (ENABLE): CSR FSM Timeout Error Injection is enabled **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_ASP_FSM&&(((DWC_EQOS_EEE_EN||DWC_EQOS_SMA_EN)&&(DWC_EQOS_CORE||DWC_EQOS_CSR_SLV_CLK))||((DWC_EQOS_AXI_SLAVE||DWC_EQOS_AHB_SLAVE)&&DWC_EQOS_CSR_SLV_CLK)))||(DWC_EQOS_ASP_ACT&&(DWC_EQOS_AXI_SLAVE||DWC_EQOS_AHB_SLAVE)&&DWC_EQOS_CSR_SLV_CLK)

Bits	Name	Memory Access	Description
9	RTEIN	R/W	Rx FSM Timeout Error Injection This field when set indicates that Error Injection for RX FSM timeout is enabled. **Values:** ■ 0x0 (DISABLE): Rx FSM Timeout Error Injection is disabled ■ 0x1 (ENABLE): Rx FSM Timeout Error Injection is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_ASP_ALL
8	TTEIN	R/W	Tx FSM Timeout Error Injection This field when set indicates that Error Injection for TX FSM timeout is enabled. **Values:** ■ 0x0 (DISABLE): Tx FSM Timeout Error Injection is disabled ■ 0x1 (ENABLE): Tx FSM Timeout Error Injection is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_ASP_ALL
7:2	Reserved_7_2	R	**Value After Reset:** 0x0 **Exists:** Always
1	PRTYEN	R/W	This bit when set indicates that the FSM parity feature is enabled. **Values:** ■ 0x0 (DISABLE): FSM Parity feature is disabled ■ 0x1 (ENABLE): FSM Parity feature is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_ASP_ALL
0	TMOUTEN	R/W	This bit when set indicates that the FSM timeout feature is enabled. **Values:** ■ 0x0 (DISABLE): FSM timeout feature is disabled ■ 0x1 (ENABLE): FSM timeout feature is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_ASP_ACT||DWC_EQOS_ASP_FSM

17.1.60 MAC_FSM_ACT_Timer

- **Description:** This register is used to select the FSM and Interface Timeout values.
- **Size:** 32 bits
- **Offset:** 0x14c
- **Exists:** (DWC_EQOS_ASP_FSM || DWC_EQOS_ASP_ACT)

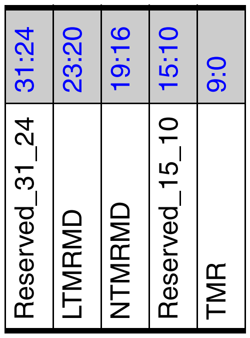


Table 17-64 Fields for Register: MAC_FSM_ACT_Timer

Bits	Name	Memory Access	Description
31:24	Reserved_31_24	R	**Value After Reset:** 0x0 **Exists:** Always
23:20	LTMRMD	R/W	This field provides the mode value to be used for large mode FSM and other interface time outs. The timeout duration based on the mode value is given below **Values:** ■ 0x0 (DISABLE): Timer disabled ■ 0x1 (1MICRO_SEC): 1us ■ 0x2 (4MILLI_SEC): 1.024ms (~4ms) ■ 0x3 (16MILLI_SEC): 16.384ms (~16ms) ■ 0x4 (64MILLI_SEC): 65.536ms (~64ms) ■ 0x5 (256MILLI_SEC): 262.144ms (~256ms) ■ 0x6 (1SEC): 1.048sec (~1sec) ■ 0x7 (4SEC): 4.194sec (~4sec) ■ 0x8 (16SEC): 16.777sec (~16sec) ■ 0x9 (32SEC): 33.554sec (~32sec) ■ 0xa (64SEC): 67.108sec (~64sec) ■ 0xb (RSVD): Reserved **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
19:16	NTMRMD	R/W	This field provides the value to be used for normal mode FSM and other interface time outs. The timeout duration based on the mode value is given below **Values:** ■ 0x0 (DISABLE): Timer disabled ■ 0x1 (1MICRO_SEC): 1us ■ 0x2 (4MILLI_SEC): 1.024ms (~4ms) ■ 0x3 (16MILLI_SEC): 16.384ms (~16ms) ■ 0x4 (64MILLI_SEC): 65.536ms (~64ms) ■ 0x5 (256MILLI_SEC): 262.144ms (~256ms) ■ 0x6 (1SEC): 1.048sec (~1sec) ■ 0x7 (4SEC): 4.194sec (~4sec) ■ 0x8 (16SEC): 16.777sec (~16sec) ■ 0x9 (32SEC): 33.554sec (~32sec) ■ 0xa (64SEC): 67.108sec (~64sec) ■ 0xb (RSVD): Reserved **Value After Reset:** 0x0 **Exists:** Always
15:10	Reserved_15_10	R	**Value After Reset:** 0x0 **Exists:** Always
9:0	TMR	R/W	This field indicates the number of CSR clocks required to generate 1us tic. **Value After Reset:** 0x0 **Exists:** Always

17.1.61 SNPS_SCS_REG1

- **Description:** Synopsys Reserved Register
- **Size:** 32 bits
- **Offset:** 0x150
- **Exists:** DWC_EQOS_ASP_ALL

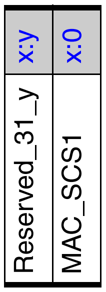


Table 17-65 Fields for Register: SNPS_SCS_REG1

Bits	Name	Memory Access	Description
x:y	Reserved_31_y	R	**Value After Reset:** 0x0 **Exists:** DWC_EQOS_SYS<2 **Range Variable[x]:** DWC_EQOS_SYS>0?8:16 + DWC_EQOS_SYS>0?24:16 - 1 **Range Variable[y]:** DWC_EQOS_SYS>0?24:16
x:0	MAC_SCS1	R/W	Synopsys Reserved, All the bits must be set to "0". This field is reserved for Synopsys Internal use, and must always be set to "0" unless instructed by Synopsys. Setting any bit to "1" might cause unexpected behavior in the IP. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[x]:** DWC_EQOS_SYS>1?32:DWC_EQOS_SYS>0?24:16 - 1

17.1.62 MAC_MDIO_Address

- **Description:** The MDIO Address register controls the management cycles to external PHY through a management interface.
- **Size:** 32 bits
- **Offset:** 0x200
- **Exists:** (DWC_EQOS_SMA_EN | | DWC_EQOS_REVMII_EN)

31:28	27	26	25:21	20:16	15	14:12	11:8	7:5	4	3	2	1	0
Reserved_31_28	PSE	BTB	PA	RDA	Reserved_15	NTC	CR	Reserved_7_5	SKAP	GOC_1	GOC_0	C45E	GB

Table 17-66 Fields for Register: MAC_MDIO_Address

Bits	Name	Memory Access	Description
31:28	Reserved_31_28	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
27	PSE	R/W	Preamble Suppression Enable When this bit is set, the SMA suppresses the 32-bit preamble and transmits MDIO frames with only 1 preamble bit. When this bit is 0, the MDIO frame always has 32 bits of preamble as defined in the IEEE specifications. **Values:** ■ 0x0 (DISABLE): Preamble Suppression disabled ■ 0x1 (ENABLE): Preamble Suppression enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_SMA_EN

Bits	Name	Memory Access	Description
26	BTB	R/W	Back to Back transactions When this bit is set and the NTC has value greater than 0, then the MAC informs the completion of a read or write command at the end of frame transfer (before the trailing clocks are transmitted). The software can thus initiate the next command which is executed immediately irrespective of the number trailing clocks generated for the previous frame. When this bit is reset, then the read/write command completion (GB is cleared) only after the trailing clocks are generated. In this mode, it is ensured that the NTC is always generated after each frame. This bit must not be set when NTC=0. **Values:** ■ 0x0 (DISABLE): Back to Back transactions disabled ■ 0x1 (ENABLE): Back to Back transactions enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_SMA_EN
25:21	PA	R/W	Physical Layer Address This field indicates which Clause 22 PHY devices (out of 32 devices) the MAC is accessing. For RevMII, this field gives the PHY Address of the RevMII module. This field indicates which Clause 45 capable PHYs (out of 32 PHYs) the MAC is accessing. **Value After Reset:** 0x0 **Exists:** Always **Testable:** untestable
20:16	RDA	R/W	Register/Device Address These bits select the PHY register in selected Clause 22 PHY device. For RevMII, these bits select the CSR register in the RevMII Registers set. These bits select the Device (MMD) in selected Clause 45 capable PHY. **Value After Reset:** 0x0 **Exists:** Always
15	Reserved_15	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
14:12	NTC	R/W	Number of Trailing Clocks This field controls the number of trailing clock cycles generated on gmii_mdc_o (MDC) after the end of transmission of MDIO frame. The valid values can be from 0 to 7. Programming the value to 3'h3 indicates that there are additional three clock cycles on the MDC line after the end of MDIO frame transfer. **Value After Reset:** 0x0 **Exists:** DWC_EQOS_SMA_EN

Bits	Name	Memory Access	Description
11:8	CR	R/W	CSR Clock Range The CSR Clock Range selection determines the frequency of the MDC clock according to the CSR clock frequency used in your design: ■ 0000: CSR clock = 60-100 MHz; MDC clock = CSR clock/42 ■ 0001: CSR clock = 100-150 MHz; MDC clock = CSR clock/62 ■ 0010: CSR clock = 20-35 MHz; MDC clock = CSR clock/16 ■ 0011: CSR clock = 35-60 MHz; MDC clock = CSR clock/26 ■ 0100: CSR clock = 150-250 MHz; MDC clock = CSR clock/102 ■ 0101: CSR clock = 250-300 MHz; MDC clock = CSR clock/124 ■ 0110: CSR clock = 300-500 MHz; MDC clock = CSR clock/204 ■ 0111: CSR clock = 500-800 MHz; MDC clock = CSR clock/324 The suggested range of CSR clock frequency applicable for each value (when Bit 11 = 0) ensures that the MDC clock is approximately between 1.0 MHz to 2.5 MHz freqency range. When Bit 11 is set, you can achieve a higher frequency of the MDC clock than the frequency limit of 2.5 MHz (specified in the IEEE 802.3) and program a clock divider of lower value. For example, when CSR clock is of 100 MHz frequency and you program these bits as 1010, the resultant MDC clock is of 12.5 MHz which is above the range specified in IEEE 802.3. Program the following values only if the interfacing chips support faster MDC clocks: ■ 1000: CSR clock/4 ■ 1001: CSR clock/6 ■ 1010: CSR clock/8 ■ 1011: CSR clock/10 ■ 1100: CSR clock/12 ■ 1101: CSR clock/14 ■ 1110: CSR clock/16 ■ 1111: CSR clock/18 These bits are not used for accessing RevMII. These bits are read-only if the RevMII interface is selected as single PHY interface. **Value After Reset:** 0x0 **Exists:** DWC_EQOS_SMA_EN
7:5	Reserved_7_5	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
4	SKAP	R/W	Skip Address Packet When this bit is set, the SMA does not send the address packets before read, write, or post-read increment address packets. This bit is valid only when C45E is set. **Values:** ■ 0x0 (DISABLE): Skip Address Packet is disabled ■ 0x1 (ENABLE): Skip Address Packet is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_SMA_EN
3	GOC_1	R/W	GMII Operation Command 1 This bit is higher bit of the operation command to the PHY or RevMII, GOC_1 and GOC_O is encoded as follows: ■ 00: Reserved ■ 01: Write ■ 10: Post Read Increment Address for Clause 45 PHY ■ 11: Read When Clause 22 PHY or RevMII is enabled, only Write and Read commands are valid. **Values:** ■ 0x0 (DISABLE): GMII Operation Command 1 is disabled ■ 0x1 (ENABLE): GMII Operation Command 1 is enabled **Value After Reset:** 0x0 **Exists:** Always **Testable:** untestable
2	GOC_0	R/W	GMII Operation Command 0 This is the lower bit of the operation command to the PHY or RevMII. When in SMA mode (MDIO master) this bit along with GOC_1 determines the operation to be performed to the PHY. When only RevMII is selected in configuration this bit is read-only and tied to 1. **Values:** ■ 0x0 (DISABLE): GMII Operation Command 0 is disabled ■ 0x1 (ENABLE): GMII Operation Command 0 is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_SMA_EN

Bits	Name	Memory Access	Description
1	C45E	R/W	Clause 45 PHY Enable When this bit is set, Clause 45 capable PHY is connected to MDIO. When this bit is reset, Clause 22 capable PHY is connected to MDIO. **Values:** ■ 0x0 (DISABLE): Clause 45 PHY is disabled ■ 0x1 (ENABLE): Clause 45 PHY is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_SMA_EN
0	GB	R/W	GMII Busy The application sets this bit to instruct the SMA to initiate a Read or Write access to the MDIO slave. The MAC clears this bit after the MDIO frame transfer is completed. Hence the software must not write or change any of the fields in MAC_MDIO_Address and MAC_MDIO_Data registers as long as this bit is set. For write transfers, the application must first write 16-bit data in the GDI field (and also RA field when C45E is set) in MAC_MDIO_Data register before setting this bit. When C45E is set, it should also write into the RA field of MAC_MDIO_Data register before initiating a read transfer. When a read transfer is completed (GB=0), the data read from the PHY register is valid in the GD field of the MAC_MDIO_Data register. **Note**: Even if the addressed PHY is not present, there is no change in the functionality of this bit. Access restriction applies. Setting 1 sets. Self-cleared. Setting 0 has no effect. **Values:** ■ 0x0 (DISABLE): GMII Busy is disabled ■ 0x1 (ENABLE): GMII Busy is enabled **Value After Reset:** 0x0 **Exists:** Always **Testable:** untestable

17.1.63 MAC_MDIO_Data

- **Description:** The MDIO Data register stores the Write data to be written to the PHY register located at the address specified in MAC_MDIO_Address. This register also stores the Read data from the PHY register located at the address specified by MDIO Address register.
- **Size:** 32 bits
- **Offset:** 0x204
- **Exists:** (DWC_EQOS_SMA_EN||DWC_EQOS_REVMII_EN)

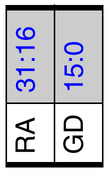


Table 17-67 Fields for Register: MAC_MDIO_Data

Bits	Name	Memory Access	Description
31:16	RA	R/W	Register Address This field is valid only when C45E is set. It contains the Register Address in the PHY to which the MDIO frame is intended for. **Value After Reset:** 0x0 **Exists:** Always **Testable:** untestable
15:0	GD	R/W	GMII Data This field contains the 16-bit data value read from the PHY or RevMII after a Management Read operation or the 16-bit data value to be written to the PHY or RevMII before a Management Write operation. **Value After Reset:** 0x0 **Exists:** Always **Testable:** untestable

17.1.64 MAC_GPIO_Control

- **Description:** The GPIO Control register controls the GPIO.
- **Size:** 32 bits
- **Offset:** 0x208
- **Exists:** (DWC_EQOS_GPIO_EN)

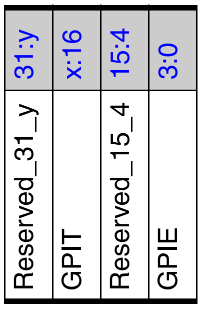


Table 17-68 Fields for Register: MAC_GPIO_Control

Bits	Name	Memory Access	Description
31:y	Reserved_31_y	R	Reserved. **Value After Reset:** 0x0 **Exists:** DWC_EQOS_GIW<16
x:16	GPIT	R/W	GPI Type When this bit is set, it indicates that the corresponding GPIS bit in MAC_GPIO_Status register is of latched-low (LL) type. When this bit is reset, it indicates that the corresponding GPIS is of latched-high (LH) type. The number of bits available in this field depends on the GP Input Signal Width option. Other bits are not used (reserved and always reset). **Value After Reset:** 0x0 **Exists:** Always **Range Variable[x]:** DWC_EQOS_GIW + 15
15:4	Reserved_15_4	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
3:0	GPIE	R/W	GPI Interrupt Enable When this bit is set and the programmed event (LL or LH) occurs on the corresponding GPIS bit of MAC_GPIO_Status register, the GPIIS bit of DMA_Interrupt_Status register is set and an interrupt is generated on the mci_intr_o or sbd_intr_o signal. The GPIIS bit is cleared when the application reads the Bits[7:0] of MAC_GPIO_Status register. When this bit is reset, the GPIIS bit is not set when any event occurs on the corresponding GPIS bits. **Value After Reset:** 0x0 **Exists:** Always

17.1.65 MAC_GPIO_Status

- **Description:** The General Purpose IO register provides the control to drive the following: up to 16 bits of output ports (GPO) and status of up to 16 input ports (GPIS).
- **Size:** 32 bits
- **Offset:** 0x20c
- **Exists:** (DWC_EQOS_GPIO_EN)

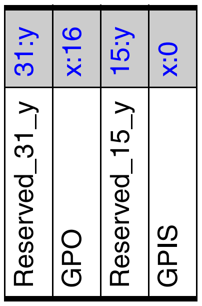


Table 17-69 Fields for Register: MAC_GPIO_Status

Bits	Name	Memory Access	Description
31:y	Reserved_31_y	R	Reserved. **Value After Reset:** 0x0 **Exists:** DWC_EQOS_GOW<16
x:16	GPO	R/W	General Purpose Output When this bit is set, it directly drives the gpo_o output ports. When this bit is reset, it does not directly drive the gpo_o output ports. The number of bits available in this field depends on the GP Input Signal Width option. Other bits are not used (reserved and always reset). **Value After Reset:** 0x0 **Exists:** Always **Range Variable[x]:** DWC_EQOS_GOW + 15
15:y	Reserved_15_y	R	Reserved. **Value After Reset:** 0x0 **Exists:** DWC_EQOS_GIW<16 **Range Variable[y]:** DWC_EQOS_GIW

Bits	Name	Memory Access	Description
x:0	GPIS	R/W	General Purpose Input Status This field gives the status of the signals connected to the gpi_i port. This field is of the following types based on the setting of the corresponding GPIT field of MAC_GPIO_Control register: ■ Latched-low (LL): This field is cleared when the corresponding gpi_i input becomes low. This field remains low until the application reads this field after which this field reflects the current value of gpi_i input. ■ Latched-high (LH): This field is set when the corresponding gpi_i input becomes high. This field remains high until the application reads this field after which this field reflects the current value of gpi_i input. The number of bits available in this field depends on the GP Input Signal Width option. Other bits are not used (reserved and always reset). 0000H **Value After Reset:** 0x0 **Exists:** Always **Testable:** untestable **Range Variable[x]:** DWC_EQOS_GIW - 1

17.1.66 MAC_ARP_Address

- **Description:** The ARP Address register contains the IPv4 Destination Address of the MAC.
- **Size:** 32 bits
- **Offset:** 0x210
- **Exists:** (DWC_EQOS_ARP_EN)

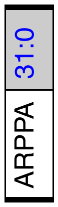


Table 17-70 Fields for Register: MAC_ARP_Address

Bits	Name	Memory Access	Description
31:0	ARPPA	R/W	ARP Protocol Address This field contains the IPv4 Destination Address of the MAC. This address is used for perfect match with the Protocol Address of Target field in the received ARP packet. This field is available only when the Enable IPv4 ARP Offload option is selected. **Value After Reset:** 0x0 **Exists:** Always

17.1.67 MAC_CSR_SW_Ctrl

- **Description:** This register contains SW programmable controls for changing the CSR access response and status bits clearing.
- **Size:** 32 bits
- **Offset:** 0x230
- **Exists:** Always

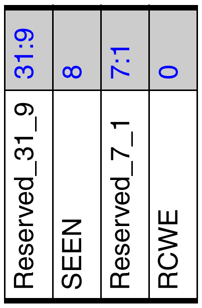


Table 17-71 Fields for Register: MAC_CSR_SW_Ctrl

Bits	Name	Memory Access	Description
31:9	Reserved_31_9	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
8	SEEN	R/W	Slave Error Response Enable When this bit is set, the MAC responds with Slave Error for accesses to reserved registers in CSR space. When this bit is reset, the MAC responds with Okay response to any register accessed from CSR space. **Values:** ■ 0x0 (DISABLE): Slave Error Response is disabled ■ 0x1 (ENABLE): Slave Error Response is enabled **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_CSR_PORT>1)
7:1	Reserved_7_1	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
0	RCWE	R/W	Register Clear on Write 1 Enable When this bit is set, the access mode of some register fields changes to Clear on Write 1, the application needs to set that respective bit to 1 to clear it. When this bit is reset, the access mode of these register fields remain as Clear on Read. **Values:** ■ 0x0 (DISABLE): Register Clear on Write 1 is disabled ■ 0x1 (ENABLE): Register Clear on Write 1 is enabled **Value After Reset:** 0x0 **Exists:** Always

17.1.68 MAC_FPE_CTRL_STS

- **Description:** This register controls the operation of Frame Preemption.
- **Size:** 32 bits
- **Offset:** 0x234
- **Exists:** (DWC_EQOS_FPE)

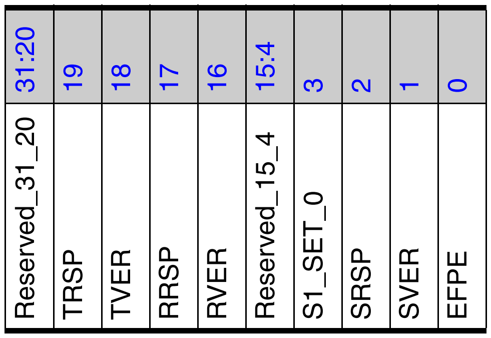


Table 17-72 Fields for Register: MAC_FPE_CTRL_STS

Bits	Name	Memory Access	Description
31:20	Reserved_31_20	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
19	TRSP	R/W	Transmitted Respond Frame Set when a Respond mPacket is transmitted (triggered by setting SRSP field). An interrupt can be generated for this event if FPEIE bit of MAC_Interrupt_Enable is set. Access restriction applies. Clears on read (or write of 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): Not transmitted Respond Frame ■ 0x1 (ACTIVE): transmitted Respond Frame **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
18	TVER	R/W	Transmitted Verify Frame Set when a Verify mPacket is transmitted (triggered by setting SVER field). An interrupt can be generated for this event if FPEIE bit of MAC_Interrupt_Enable is set. Access restriction applies. Clears on read (or write of 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): Not transmitted Verify Frame ■ 0x1 (ACTIVE): transmitted Verify Frame **Value After Reset:** 0x0 **Exists:** Always
17	RRSP	R/W	Received Respond Frame Set when a Respond mPacket is received. An interrupt can be generated for this event if FPEIE bit of MAC_Interrupt_Enable is set. Access restriction applies. Clears on read (or write of 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): Not received Respond Frame ■ 0x1 (ACTIVE): Received Respond Frame **Value After Reset:** 0x0 **Exists:** Always
16	RVER	R/W	Received Verify Frame Set when a Verify mPacket is received. An interrupt can be generated for this event if FPEIE bit of MAC_Interrupt_Enable is set. Access restriction applies. Clears on read (or write of 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): Not received Verify Frame ■ 0x1 (ACTIVE): Received Verify Frame **Value After Reset:** 0x0 **Exists:** Always
15:4	Reserved_15_4	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
3	S1_SET_0	R/W	Synopsys Reserved, Must be set to "0". This field is reserved for Synopsys Internal use, and must always be set to "0" unless instructed by Synopsys. Setting to "1" might cause unexpected behavior in the IP. **Value After Reset:** 0x0 **Exists:** Always
2	SRSP	R/W	Send Respond mPacket When set indicates hardware to send a Respond mPacket. Reset by hardware after sending the Respond mPacket. Access restriction applies. Setting 1 sets. Self-cleared. Setting 0 has no effect. **Values:** ■ 0x0 (DISABLE): Send Respond mPacket is disabled ■ 0x1 (ENABLE): Send Respond mPacket is enabled **Value After Reset:** 0x0 **Exists:** Always
1	SVER	R/W	Send Verify mPacket When set indicates hardware to send a verify mPacket. Reset by hardware after sending the Verify mPacket. Access restriction applies. Setting 1 sets. Self-cleared. Setting 0 has no effect. **Values:** ■ 0x0 (DISABLE): Send Verify mPacket is disabled ■ 0x1 (ENABLE): Send Verify mPacket is enabled **Value After Reset:** 0x0 **Exists:** Always
0	EFPE	R/W	Enable Tx Frame Preemption When set Frame Preemption Tx functionality is enabled. **Values:** ■ 0x0 (DISABLE): Tx Frame Preemption is disabled ■ 0x1 (ENABLE): Tx Frame Preemption is enabled **Value After Reset:** 0x0 **Exists:** Always

17.1.69 MAC_Ext_Cfg1

- **Description:** This register contains Split mode control field and offset field for Split Header feature.
- **Size:** 32 bits
- **Offset:** 0x238
- **Exists:** (DWC_EQOS_SPLIT_HDR_EN)

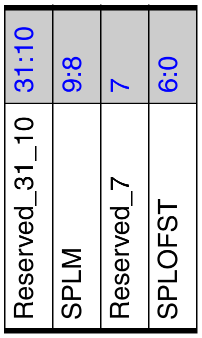


Table 17-73 Fields for Register: MAC_Ext_Cfg1

Bits	Name	Memory Access	Description
31:10	Reserved_31_10	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
9:8	SPLM	R/W	Split Mode These bits indicate the mode of splitting the incoming Rx packets. They are **Values:** ■ 0x0 (L3L4): Split at L3/L4 header ■ 0x1 (L2OFST): Split at L2 header with an offset. Always Split at SPLOFST bytes from the beginning of Length/Type field of the Frame ■ 0x2 (COMBN): Combination mode: Split similar to SPLM=00 for IP packets that are untagged or tagged and VLAN stripped ■ 0x3 (RSVD): Reserved **Value After Reset:** 0x0 **Exists:** Always
7	Reserved_7	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
6:0	SPLOFST	R/W	Split Offset These bits indicate the value of offset from the beginning of Length/Type field at which header split should take place when the appropriate SPLM is selected. The reset value of this field is 2 bytes indicating a split at L2 header. Value is in terms of bytes. **Value After Reset:** 0x2 **Exists:** Always

17.1.70 MAC_Presn_Time_ns

- **Description:** This register contains the 32-bit binary rollover equivalent time of the PTP System Time in ns
  Exists when DWC_EQOS_FLEXI_PPS_OUT_EN is configured
- **Size:** 32 bits
- **Offset:** 0x240
- **Exists:** DWC_EQOS_FLEXI_PPS_OUT_EN

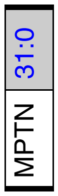


Table 17-74 Fields for Register: MAC_Presn_Time_ns

Bits	Name	Memory Access	Description
31:0	MPTN	R	MAC 1722 Presentation Time in ns These bits indicate the value of the 32-bit binary rollover equivalent time of the PTP System Time in ns **Value After Reset:** 0x0 **Exists:** Always

17.1.71 MAC_Presn_Time_Updt

- **Description:** This field holds the 32-bit value of MAC 1722 Presentation Time in ns, that should be added to the Current Presentation Time Counter value. Init happens when TSINIT is set, and update happens when the TSUPDT bit is set (TSINIT and TSINIT defined in MAC_Timestamp_ Control register).
  Exists when DWC_EQOS_FLEXI_PPS_OUT_EN is configured
- **Size:** 32 bits
- **Offset:** 0x244
- **Exists:** DWC_EQOS_FLEXI_PPS_OUT_EN

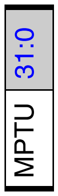


Table 17-75 Fields for Register: MAC_Presn_Time_Updt

Bits	Name	Memory Access	Description
31:0	MPTU	R/W	MAC 1722 Presentation Time Update This field holds the init value or the update value for the presentation time. When used for update, this field holds the 32-bit value in ns, that should be added to the Current Presentation Time Counter value. Init happens when TSINIT is set, and update happens when the TSUPDT bit is set (TSINIT and TSINIT defined in MAC_Timestamp_ Control register). When ADDSUB field of MAC_System_Time_Nanoseconds_Update is set, this value is directly used for subtraction **Value After Reset:** 0x0 **Exists:** Always

17.1.72 MAC_Address0_High

- **Description:** The MAC Address0 High register holds the upper 16 bits of the first 6-byte MAC address of the station. The first DA byte that is received on the (G)MII interface corresponds to the LS byte (Bits [7:0]) of the MAC Address Low register. For example, if 0x112233445566 is received (0x11 in lane 0 of the first column) on the (G)MII as the destination address, then the MacAddress0 Register [47:0] is compared with 0x665544332211.
  If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the MAC Address0 Low Register are written. For proper synchronization updates, the consecutive writes to this Address Low Register should be performed after at least four clock cycles in the destination clock domain.
- **Size:** 32 bits
- **Offset:** 0x300
- **Exists:** Always

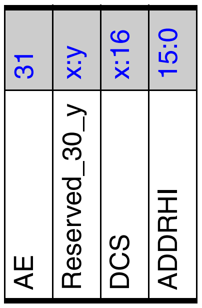


Table 17-76 Fields for Register: MAC_Address0_High

Bits	Name	Memory Access	Description
31	AE	R	Address Enable This bit is always set to 1. **Values:** ■ 0x0 (DISABLE): INVALID : This bit must be always set to 1 ■ 0x1 (ENABLE): This bit is always set to 1 **Value After Reset:** 0x1 **Exists:** Always

Bits	Name	Memory Access	Description
x:y	Reserved_30_y	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[x]:** (((DWC_EQOS_SYS>1)&&(DWC_EQOS_NUM_DMA_RX_CH>1)&&(!DWC_EQOS_PDUP))?15-DWC_EQOS_NUM_DMA_RX_CHW:15-DWC_EQOS_NUM_DMA_RX_CH) + (((DWC_EQOS_SYS>1)&&(DWC_EQOS_NUM_DMA_RX_CH>1)&&(!DWC_EQOS_PDUP))?16+DWC_EQOS_NUM_DMA_RX_CHW:16+DWC_EQOS_NUM_DMA_RX_CH) - 1 **Range Variable[y]:** (((DWC_EQOS_SYS>1)&&(DWC_EQOS_NUM_DMA_RX_CH>1)&&(!DWC_EQOS_PDUP))?16+DWC_EQOS_NUM_DMA_RX_CHW:16+DWC_EQOS_NUM_DMA_RX_CH)
x:16	DCS	R/W	DMA Channel Select If the PDC bit of MAC_Ext_Configuration register is not set: This field contains the binary representation of the DMA Channel number to which an Rx packet whose DA matches the MAC Address0 content is routed. If the PDC bit of MAC_Ext_Configuration register is set: This field contains the one-hot representation of one or more DMA Channel numbers to which an Rx packet whose DA matches the MAC Address0 content is routed. **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_SYS>1)&&(DWC_EQOS_NUM_DMA_RX_CH>1) **Range Variable[x]:** ((DWC_EQOS_PDUP)?\"DWC_EQOS_NUM_DMA_RX_CH\":\"DWC_EQOS_NUM_DMA_RX_CHW\") + 15
15:0	ADDRHI	R/W	MAC Address0[47:32] This field contains the upper 16 bits [47:32] of the first 6-byte MAC address. The MAC uses this field for filtering the received packets and inserting the MAC address in the Transmit Flow Control (Pause) Packets. **Value After Reset:** 0xffff **Exists:** Always

17.1.73 MAC_Address0_Low

- **Description:** The MAC Address0 Low register holds the lower 32 bits of the 6-byte first MAC address of the station.
- **Size:** 32 bits
- **Offset:** 0x304
- **Exists:** Always

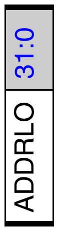


Table 17-77 Fields for Register: MAC_Address0_Low

Bits	Name	Memory Access	Description
31:0	ADDRLO	R/W	MAC Address0[31:0] This field contains the lower 32 bits of the first 6-byte MAC address. The MAC uses this field for filtering the received packets and inserting the MAC address in the Transmit Flow Control (Pause) Packets. **Value After Reset:** 0xffffffff **Exists:** Always

17.1.74 MAC_Address(#i)_High (for i = 1; i <= DWC_EQOS_ADD_MAC_ADDR_REG-1)

- **Description:** The MAC Address1 High register holds the upper 16 bits of the second 6-byte MAC address of the station.
  If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the MAC Address1 Low Register are written. For proper synchronization updates, the consecutive writes to this Address Low Register should be performed after at least four clock cycles in the destination clock domain.
- **Size:** 32 bits
- **Offset:** (0x0008*i)+0x0300
- **Exists:** (DWC_EQOS_ADD_MAC_ADDR_REG>=1)

31	30	29:24	x:y	x:16	15:0
AE	SA	MBC	Reserved_23_y	DCS	ADDRHI

Table 17-78 Fields for Register: MAC_Address(#i)_High (for i = 1; i <= DWC_EQOS_ADD_MAC_ADDR_REG-1)

Bits	Name	Memory Access	Description
31	AE	R/W	Address Enable When this bit is set, the address filter module uses the second MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Values:** ■ 0x0 (DISABLE): Address is ignored ■ 0x1 (ENABLE): Address is enabled **Value After Reset:** 0x0 **Exists:** Always
30	SA	R/W	Source Address When this bit is set, the MAC Address1[47:0] is used to compare with the SA fields of the received packet. When this bit is reset, the MAC Address1[47:0] is used to compare with the DA fields of the received packet. **Values:** ■ 0x0 (DA): Compare with Destination Address ■ 0x1 (SA): Compare with Source Address **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
29:24	MBC	R/W	Mask Byte Control These bits are mask control bits for comparing each of the MAC Address bytes. When set high, the MAC does not compare the corresponding byte of received DA or SA with the contents of MAC Address1 registers. Each bit controls the masking of the bytes as follows: ■ Bit 29: MAC_Address${i}_High[15:8] ■ Bit 28: MAC_Address${i}_High[7:0] ■ Bit 27: MAC_Address${i}_Low[31:24] ■ .. ■ Bit 24: MAC_Address${i}_Low[7:0] You can filter a group of addresses (known as group address filtering) by masking one or more bytes of the address. **Value After Reset:** 0x0 **Exists:** Always
x:y	Reserved_23_y	R	Reserved. **Value After Reset:** 0x0 **Exists:** (((DWC_EQOS_SYS>1)&&(DWC_EQOS_NUM_DMA_RX_CH>1)&&!(DWC_EQOS_PDUP))?(DWC_EQOS_NUM_DMA_RX_CHW<8):(DWC_EQOS_NUM_DMA_RX_CH<8)) **Range Variable[x]:** (((DWC_EQOS_SYS>1)&&(DWC_EQOS_NUM_DMA_RX_CH>1)&&(!DWC_EQOS_PDUP))?8-DWC_EQOS_NUM_DMA_RX_CHW:8-DWC_EQOS_NUM_DMA_RX_CH) + (((DWC_EQOS_SYS>1)&&(DWC_EQOS_NUM_DMA_RX_CH>1)&&(!DWC_EQOS_PDUP))?16+DWC_EQOS_NUM_DMA_RX_CHW:16+DWC_EQOS_NUM_DMA_RX_CH) - 1 **Range Variable[y]:** (((DWC_EQOS_SYS>1)&&(DWC_EQOS_NUM_DMA_RX_CH>1)&&(!DWC_EQOS_PDUP))?16+DWC_EQOS_NUM_DMA_RX_CHW:16+DWC_EQOS_NUM_DMA_RX_CH)

Bits	Name	Memory Access	Description
x:16	DCS	R/W	DMA Channel Select If the PDC bit of MAC_Ext_Configuration register is not set: This field contains the binary representation of the DMA Channel number to which an Rx packet whose DA matches the MAC Address(#i) content is routed. If the PDC bit of MAC_Ext_Configuration register is set: This field contains the one-hot representation of one or more DMA Channel numbers to which an Rx packet whose DA matches the MAC Address(#i) content is routed. **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_SYS>1)&&(DWC_EQOS_NUM_DMA_RX_CH>1) **Range Variable[x]:** ((DWC_EQOS_PDUP)?\"DWC_EQOS_NUM_DMA_RX_CH\":\"DWC_EQOS_NUM_DMA_RX_CHW\") + 15
15:0	ADDRHI	R/W	MAC Address1 [47:32] This field contains the upper 16 bits[47:32] of the second 6-byte MAC address. **Value After Reset:** 0xffff **Exists:** Always

17.1.75 MAC_Address(#i)_Low (for i = 1; i <= DWC_EQOS_ADD_MAC_ADDR_REG-1)

- **Description:** The MAC Address1 Low register holds the lower 32 bits of the second 6-byte MAC address of the station.
- **Size:** 32 bits
- **Offset:** (0x0008*i)+0x0304
- **Exists:** (DWC_EQOS_ADD_MAC_ADDR_REG>=1)

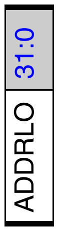


Table 17-79 Fields for Register: MAC_Address(#i)_Low (for i = 1; i <= DWC_EQOS_ADD_MAC_ADDR_REG-1)

Bits	Name	Memory Access	Description
31:0	ADDRLO	R/W	MAC Address1 [31:0] This field contains the lower 32 bits of second 6-byte MAC address. The content of this field is undefined until loaded by the application after the initialization process. **Value After Reset:** 0xffffffff **Exists:** Always

17.1.76 MAC_Address(#i)_High (for i = 32; i <= 63)

- **Description:** The MAC Address32 High register holds the upper 16 bits of the 33rd 6-byte MAC address of the station.
  If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the MAC Address Low Register are written. For proper synchronization updates, you should perform the consecutive writes to the MAC Address Low Register after at least four clock cycles of the destination clock.
- **Size:** 32 bits
- **Offset:** (0x0008*i)+0x0300
- **Exists:** (DWC_EQOS_ADD32_MAC_ADDR_REG)

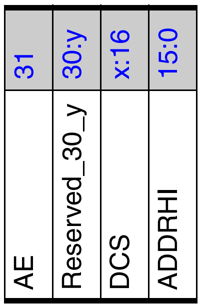


Table 17-80 Fields for Register: MAC_Address(#i)_High (for i = 32; i <= 63)

Bits	Name	Memory Access	Description
31	AE	R/W	Address Enable When this bit is set, the Address filter module uses the 33rd MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Values:** ■ 0x0 (DISABLE): Address is ignored ■ 0x1 (ENABLE): Address is enabled **Value After Reset:** 0x0 **Exists:** Always
30:y	Reserved_30_y	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[y]:** DWC_EQOS_NUM_DMA_RX_CHW + 16

Bits	Name	Memory Access	Description
x:16	DCS	R/W	DMA Channel Select This field contains the DMA Channel number to which an Rx packet whose DA matches the MAC Address32 content is routed. **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_SYS>1)&&(DWC_EQOS_NUM_DMA_RX_CH>1) **Range Variable[x]:** DWC_EQOS_NUM_DMA_RX_CHW + 15
15:0	ADDRHI	R/W	MAC Address32 [47:32] This field contains the upper 16 bits (47:32) of the 33rd 6-byte MAC address. **Value After Reset:** 0xffff **Exists:** Always

17.1.77 MAC_Address(#i)_Low (for i = 32; i <= 63)

- **Description:** The MAC Address32 Low register holds the lower 16 bits of the 33rd 6-byte MAC address of the station.
- **Size:** 32 bits
- **Offset:** (0x0008*i)+0x0304
- **Exists:** (DWC_EQOS_ADD32_MAC_ADDR_REG)

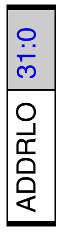


Table 17-81 Fields for Register: MAC_Address(#i)_Low (for i = 32; i <= 63)

Bits	Name	Memory Access	Description
31:0	ADDRLO	R/W	MAC Address32 [31:0] This field contains the lower 32 bits of the 33rd 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process. **Value After Reset:** 0xffffffff **Exists:** Always

17.1.78 MAC_Address(#i)_High (for i = 64; i <= 127)

- **Description:** The MAC Address32 High register holds the upper 16 bits of the 33rd 6-byte MAC address of the station.
  If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the MAC Address Low Register are written. For proper synchronization updates, you should perform the consecutive writes to the MAC Address Low Register after at least four clock cycles of the destination clock.
- **Size:** 32 bits
- **Offset:** (0x0008*i)+0x0300
- **Exists:** (DWC_EQOS_ADD64_MAC_ADDR_REG)

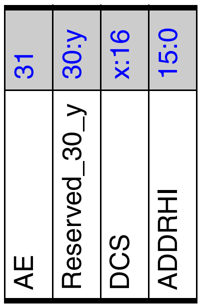


Table 17-82 Fields for Register: MAC_Address(#i)_High (for i = 64; i <= 127)

Bits	Name	Memory Access	Description
31	AE	R/W	Address Enable When this bit is set, the Address filter module uses the 33rd MAC address for perfect filtering. When this bit is reset, the address filter module ignores the address for filtering. **Values:** ■ 0x0 (DISABLE): Address is ignored ■ 0x1 (ENABLE): Address is enabled **Value After Reset:** 0x0 **Exists:** Always
30:y	Reserved_30_y	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[y]:** DWC_EQOS_NUM_DMA_RX_CHW + 16

Bits	Name	Memory Access	Description
x:16	DCS	R/W	DMA Channel Select This field contains the DMA Channel number to which an Rx packet whose DA matches the MAC Address32 content is routed. **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_SYS>1)&&(DWC_EQOS_NUM_DMA_RX_CH>1) **Range Variable[x]:** DWC_EQOS_NUM_DMA_RX_CHW + 15
15:0	ADDRHI	R/W	MAC Address32 [47:32] This field contains the upper 16 bits (47:32) of the 33rd 6-byte MAC address. **Value After Reset:** 0xffff **Exists:** Always

17.1.79 MAC_Address(#i)_Low (for i = 64; i <= 127)

- **Description:** The MAC Address32 Low register holds the lower 16 bits of the 33rd 6-byte MAC address of the station.
- **Size:** 32 bits
- **Offset:** (0x0008*i)+0x0304
- **Exists:** (DWC_EQOS_ADD64_MAC_ADDR_REG)

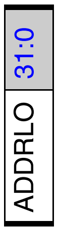


Table 17-83 Fields for Register: MAC_Address(#i)_Low (for i = 64; i <= 127)

Bits	Name	Memory Access	Description
31:0	ADDRLO	R/W	MAC Address32 [31:0] This field contains the lower 32 bits of the 33rd 6-byte MAC address. The content of this field is undefined until loaded by the Application after the initialization process. **Value After Reset:** 0xffffffff **Exists:** Always

17.1.80 MMC_Control

- **Description:** This register establishes the operating mode of MMC.
- **Size:** 32 bits
- **Offset:** 0x700
- **Exists:** (DWC_EQOS_MMC_EN)

31:9	8	7:6	5	4	3	2	1	0
Reserved_31_9	UCDBC	Reserved_7_6	CNTPRSTLVL	CNTPRST	CNTFREEZ	RSTONRD	CNTSTOPRO	CNTRST

Table 17-84 Fields for Register: MMC_Control

Bits	Name	Memory Access	Description
31:9	Reserved_31_9	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
8	UCDBC	R/W	Update MMC Counters for Dropped Broadcast Packets Note: *The CNTRST bit has a higher priority than the CNTPRST bit. Therefore, when the software tries to set both bits in the same write cycle, all counters are cleared and the CNTPRST bit is not set.* When set, the MAC updates all related MMC Counters for Broadcast packets that are dropped because of the setting of the DBF bit of MAC_Packet_Filter register. When reset, the MMC Counters are not updated for dropped Broadcast packets. **Values:** ■ 0x0 (DISABLE): Update MMC Counters for Dropped Broadcast Packets is disabled ■ 0x1 (ENABLE): Update MMC Counters for Dropped Broadcast Packets is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RX_EN_ATLEAST_ONE||DWC_EQOS_MMC_IPC_RX_EN_ATLEAST_ONE
7:6	Reserved_7_6	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
5	CNTPRSTLVL	R/W	Full-Half Preset When this bit is low and the CNTPRST bit is set, all MMC counters get preset to almost-half value. All octet counters get preset to 0x7FFF_F800 (Half 2KBytes) and all packet-counters gets preset to 0x7FFF_FFF0 (Half 16). When this bit is high and the CNTPRST bit is set, all MMC counters get preset to almost-full value. All octet counters get preset to 0xFFFF_F800 (Full 2KBytes) and all packet-counters gets preset to 0xFFFF_FFF0 (Full 16). For 16-bit counters, the almost-half preset values are 0x7800 and 0x7FF0 for the respective octet and packet counters. Similarly, the almost-full preset values for the 16-bit counters are 0xF800 and 0xFFF0. **Values:** ■ 0x0 (DISABLE): Full-Half Preset is disabled ■ 0x1 (ENABLE): Full-Half Preset is enabled **Value After Reset:** 0x0 **Exists:** Always
4	CNTPRST	R/W	Counters Preset When this bit is set, all counters are initialized or preset to almost full or almost half according to the CNTPRSTLVL bit. This bit is cleared automatically after 1 clock cycle. This bit, along with the CNTPRSTLVL bit, is useful for debugging and testing the assertion of interrupts because of MMC counter becoming half-full or full. Access restriction applies. Self-cleared. Setting 0 clears. Setting 1 sets. **Values:** ■ 0x0 (DISABLE): Counters Preset is disabled ■ 0x1 (ENABLE): Counters Preset is enabled **Value After Reset:** 0x0 **Exists:** Always **Testable:** untestable
3	CNTFREEZ	R/W	MMC Counter Freeze When this bit is set, it freezes all MMC counters to their current value. Until this bit is reset to 0, no MMC counter is updated because of any transmitted or received packet. If any MMC counter is read with the Reset on Read bit set, then that counter is also cleared in this mode. **Values:** ■ 0x0 (DISABLE): MMC Counter Freeze is disabled ■ 0x1 (ENABLE): MMC Counter Freeze is enabled **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
2	RSTONRD	R/W	Reset on Read When this bit is set, the MMC counters are reset to zero after Read (self-clearing after reset). The counters are cleared when the least significant byte lane (Bits[7:0]) is read. **Values:** ■ 0x0 (DISABLE): Reset on Read is disabled ■ 0x1 (ENABLE): Reset on Read is enabled **Value After Reset:** 0x0 **Exists:** Always
1	CNTSTOPRO	R/W	Counter Stop Rollover When this bit is set, the counter does not roll over to zero after reaching the maximum value. **Values:** ■ 0x0 (DISABLE): Counter Stop Rollover is disabled ■ 0x1 (ENABLE): Counter Stop Rollover is enabled **Value After Reset:** 0x0 **Exists:** Always
0	CNTRST	R/W	Counters Reset When this bit is set, all counters are reset. This bit is cleared automatically after 1 clock cycle. Access restriction applies. Self-cleared. Setting 0 clears. Setting 1 sets. **Values:** ■ 0x0 (DISABLE): Counters are not reset ■ 0x1 (ENABLE): All counters are reset **Value After Reset:** 0x0 **Exists:** Always **Testable:** untestable

17.1.81 MMC_Rx_Interrupt

- **Description:** This register maintains the interrupts generated from all Receive statistics counters. The MMC Receive Interrupt register maintains the interrupts that are generated when the following occur:
 - Receive statistic counters reach half of their maximum values (0x8000_0000 for 32 bit counter and 0x8000 for 16 bit counter).
 - Receive statistic counters cross their maximum values (0xFFFF_FFFF for 32 bit counter and 0xFFFF for 16 bit counter).
 When the Counter Stop Rollover is set, interrupts are set but the counter remains at all-ones. The MMC Receive Interrupt register is a 32 bit register. An interrupt bit is cleared when the respective MMC counter that caused the interrupt is read. The least significant byte lane (Bits[7:0]) of the respective counter must be read to clear the interrupt bit.
 Note: *R_SS_RC means that this register bit is set internally, and it is cleared when the Counter register is read.*
- **Size:** 32 bits
- **Offset:** 0x704
- **Exists:** (DWC_EQOS_MMC_RX_EN_ATLEAST_ONE||DWC_EQOS_MMC_RX_LPI_ATLEAST_ONE)

31:28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved_31_28	RXLPITRCIS	RXLPIUSCIS	RXCTRLPIS	RXRCVERRPIS	RXWDOGPIS	RXVLANGBPIS	RXFOVPIS	RXPAUSPIS	RXORANGEPIS	RXLENERPIS	RXUCGPIS	RX1024TMAXOCTGBPIS	RX512T1023OCTGBPIS	RX256T511OCTGBPIS	RX128T255OCTGBPIS	RX65T127OCTGBPIS	RX64OCTGBPIS	RXOSIZEGPIS	RXUSIZEGPIS	RXJABERPIS	RXRUNTPIS	RXALGNERPIS	RXCRCERPIS	RXMCGPIS	RXBCGPIS	RXGOCTIS	RXGBOCTIS	RXGBPKTIS

Table 17-85 Fields for Register: MMC_Rx_Interrupt

Bits	Name	Memory Access	Description
31:28	Reserved_31_28	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
27	RXLPITRCIS	R	MMC Receive LPI transition counter interrupt status This bit is set when the Rx_LPI_Tran_Cntr counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): MMC Receive LPI transition Counter Interrupt Status not detected ■ 0x1 (ACTIVE): MMC Receive LPI transition Counter Interrupt Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RX_LPI_TRANSITION_CNT_EN
26	RXLPIUSCIS	R	MMC Receive LPI microsecond counter interrupt status This bit is set when the Rx_LPI_USEC_Cntr counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): MMC Receive LPI microsecond Counter Interrupt Status not detected ■ 0x1 (ACTIVE): MMC Receive LPI microsecond Counter Interrupt Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RX_LPI_MICROSEC_TIMER_EN
25	RXCTRLPIS	R	MMC Receive Control Packet Counter Interrupt Status This bit is set when the rxctrlpackets_g counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): MMC Receive Control Packet Counter Interrupt Status not detected ■ 0x1 (ACTIVE): MMC Receive Control Packet Counter Interrupt Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RXCTRLPKT_CNT_EN

Bits	Name	Memory Access	Description
24	RXRCVERRPIS	R	MMC Receive Error Packet Counter Interrupt Status This bit is set when the rxrcverror counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): MMC Receive Error Packet Counter Interrupt Status not detected ■ 0x1 (ACTIVE): MMC Receive Error Packet Counter Interrupt Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RXRCVERR_CNT_EN
23	RXWDOGPIS	R	MMC Receive Watchdog Error Packet Counter Interrupt Status This bit is set when the rxwatchdog error counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): MMC Receive Watchdog Error Packet Counter Interrupt Status not detected ■ 0x1 (ACTIVE): MMC Receive Watchdog Error Packet Counter Interrupt Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RXWDGERR_CNT_EN
22	RXVLANGBPIS	R	MMC Receive VLAN Good Bad Packet Counter Interrupt Status This bit is set when the rxvlanpackets_gb counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): MMC Receive VLAN Good Bad Packet Counter Interrupt Status not detected ■ 0x1 (ACTIVE): MMC Receive VLAN Good Bad Packet Counter Interrupt Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RXVLANPKTGB_CNT_EN

Bits	Name	Memory Access	Description
21	RXFOVPIS	R	MMC Receive FIFO Overflow Packet Counter Interrupt Status This bit is set when the rxfifooverflow counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): MMC Receive FIFO Overflow Packet Counter Interrupt Status not detected ■ 0x1 (ACTIVE): MMC Receive FIFO Overflow Packet Counter Interrupt Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RXFIFOOVFL_CNT_EN
20	RXPAUSPIS	R	MMC Receive Pause Packet Counter Interrupt Status This bit is set when the rxpausepackets counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): MMC Receive Pause Packet Counter Interrupt Status not detected ■ 0x1 (ACTIVE): MMC Receive Pause Packet Counter Interrupt Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RXPAUSEPKT_CNT_EN
19	RXORANGEPIS	R	MMC Receive Out Of Range Error Packet Counter Interrupt Status. This bit is set when the rxoutofrangetype counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): MMC Receive Out Of Range Error Packet Counter Interrupt Status not detected ■ 0x1 (ACTIVE): MMC Receive Out Of Range Error Packet Counter Interrupt Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RXOUTOFRNG_TYP_CNT_EN

Bits	Name	Memory Access	Description
18	RXLENERPIS	R	MMC Receive Length Error Packet Counter Interrupt Status This bit is set when the rxlengtherror counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): MMC Receive Length Error Packet Counter Interrupt Status not detected ■ 0x1 (ACTIVE): MMC Receive Length Error Packet Counter Interrupt Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RXLENERR_CNT_EN
17	RXUCGPIS	R	MMC Receive Unicast Good Packet Counter Interrupt Status This bit is set when the rxunicastpackets_g counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): MMC Receive Unicast Good Packet Counter Interrupt Status not detected ■ 0x1 (ACTIVE): MMC Receive Unicast Good Packet Counter Interrupt Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RXUCASTG_CNT_EN
16	RX1024TMAXOCTGBPIS	R	MMC Receive 1024 to Maximum Octet Good Bad Packet Counter Interrupt Status This bit is set when the rx1024tomaxoctets_gb counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): MMC Receive 1024 to Maximum Octet Good Bad Packet Counter Interrupt Status not detected ■ 0x1 (ACTIVE): MMC Receive 1024 to Maximum Octet Good Bad Packet Counter Interrupt Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RXADDR_RNG_CNT_EN

Bits	Name	Memory Access	Description
15	RX512T1023OCTGBPIS	R	MMC Receive 512 to 1023 Octet Good Bad Packet Counter Interrupt Status This bit is set when the rx512to1023octets_gb counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): MMC Receive 512 to 1023 Octet Good Bad Packet Counter Interrupt Status not detected ■ 0x1 (ACTIVE): MMC Receive 512 to 1023 Octet Good Bad Packet Counter Interrupt Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RXADDR_RNG_CNT_EN
14	RX256T511OCTGBPIS	R	MMC Receive 256 to 511 Octet Good Bad Packet Counter Interrupt Status This bit is set when the rx256to511octets_gb counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): MMC Receive 256 to 511 Octet Good Bad Packet Counter Interrupt Status not detected ■ 0x1 (ACTIVE): MMC Receive 256 to 511 Octet Good Bad Packet Counter Interrupt Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RXADDR_RNG_CNT_EN
13	RX128T255OCTGBPIS	R	MMC Receive 128 to 255 Octet Good Bad Packet Counter Interrupt Status This bit is set when the rx128to255octets_gb counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): MMC Receive 128 to 255 Octet Good Bad Packet Counter Interrupt Status not detected ■ 0x1 (ACTIVE): MMC Receive 128 to 255 Octet Good Bad Packet Counter Interrupt Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RXADDR_RNG_CNT_EN

Bits	Name	Memory Access	Description
12	RX65T127OCTGBPIS	R	MMC Receive 65 to 127 Octet Good Bad Packet Counter Interrupt Status This bit is set when the rx65to127octets_gb counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): MMC Receive 65 to 127 Octet Good Bad Packet Counter Interrupt Status not detected ■ 0x1 (ACTIVE): MMC Receive 65 to 127 Octet Good Bad Packet Counter Interrupt Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RXADDR_RNG_CNT_EN
11	RX64OCTGBPIS	R	MMC Receive 64 Octet Good Bad Packet Counter Interrupt Status This bit is set when the rx64octets_gb counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): MMC Receive 64 Octet Good Bad Packet Counter Interrupt Status not detected ■ 0x1 (ACTIVE): MMC Receive 64 Octet Good Bad Packet Counter Interrupt Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RXADDR_RNG_CNT_EN
10	RXOSIZEGPIS	R	MMC Receive Oversize Good Packet Counter Interrupt Status This bit is set when the rxoversize_g counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): MMC Receive Oversize Good Packet Counter Interrupt Status not detected ■ 0x1 (ACTIVE): MMC Receive Oversize Good Packet Counter Interrupt Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RXOVERSZG_CNT_EN

Bits	Name	Memory Access	Description
9	RXUSIZEGPIS	R	MMC Receive Undersize Good Packet Counter Interrupt Status This bit is set when the rxundersize_g counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): MMC Receive Undersize Good Packet Counter Interrupt Status not detected ■ 0x1 (ACTIVE): MMC Receive Undersize Good Packet Counter Interrupt Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RXUNDERSZG_CNT_EN
8	RXJABERPIS	R	MMC Receive Jabber Error Packet Counter Interrupt Status This bit is set when the rxjabbererror counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): MMC Receive Jabber Error Packet Counter Interrupt Status not detected ■ 0x1 (ACTIVE): MMC Receive Jabber Error Packet Counter Interrupt Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RXJABERR_CNT_EN
7	RXRUNTPIS	R	MMC Receive Runt Packet Counter Interrupt Status This bit is set when the rxrunterror counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): MMC Receive Runt Packet Counter Interrupt Status not detected ■ 0x1 (ACTIVE): MMC Receive Runt Packet Counter Interrupt Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RXRUNTERR_CNT_EN

Bits	Name	Memory Access	Description
6	RXALGNERPIS	R	MMC Receive Alignment Error Packet Counter Interrupt Status This bit is set when the rxalignmenterror counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): MMC Receive Alignment Error Packet Counter Interrupt Status not detected ■ 0x1 (ACTIVE): MMC Receive Alignment Error Packet Counter Interrupt Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RXALGNERR_CNT_EN
5	RXCRCERPIS	R	MMC Receive CRC Error Packet Counter Interrupt Status This bit is set when the rxcrcerror counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): MMC Receive CRC Error Packet Counter Interrupt Status not detected ■ 0x1 (ACTIVE): MMC Receive CRC Error Packet Counter Interrupt Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RXCRCERR_CNT_EN
4	RXMCGPIS	R	MMC Receive Multicast Good Packet Counter Interrupt Status This bit is set when the rxmulticastpackets_g counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): MMC Receive Multicast Good Packet Counter Interrupt Status not detected ■ 0x1 (ACTIVE): MMC Receive Multicast Good Packet Counter Interrupt Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RXMCASTG_CNT_EN

Bits	Name	Memory Access	Description
3	RXBCGPIS	R	MMC Receive Broadcast Good Packet Counter Interrupt Status This bit is set when the rxbroadcastpackets_g counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): MMC Receive Broadcast Good Packet Counter Interrupt Status not detected ■ 0x1 (ACTIVE): MMC Receive Broadcast Good Packet Counter Interrupt Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RXBCASTG_CNT_EN
2	RXGOCTIS	R	MMC Receive Good Octet Counter Interrupt Status This bit is set when the rxoctetcount_g counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): MMC Receive Good Octet Counter Interrupt Status not detected ■ 0x1 (ACTIVE): MMC Receive Good Octet Counter Interrupt Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RXOCTG_CNT_EN
1	RXGBOCTIS	R	MMC Receive Good Bad Octet Counter Interrupt Status This bit is set when the rxoctetcount_gb counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): MMC Receive Good Bad Octet Counter Interrupt Status not detected ■ 0x1 (ACTIVE): MMC Receive Good Bad Octet Counter Interrupt Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RXOCTGB_CNT_EN

Bits	Name	Memory Access	Description
0	RXGBPKTIS	R	MMC Receive Good Bad Packet Counter Interrupt Status This bit is set when the rxpacketcount_gb counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): MMC Receive Good Bad Packet Counter Interrupt Status not detected ■ 0x1 (ACTIVE): MMC Receive Good Bad Packet Counter Interrupt Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RXPKTGB_CNT_EN

17.1.82 MMC_Tx_Interrupt

- **Description:** This register maintains the interrupts generated from all Transmit statistics counters.
  The MMC Transmit Interrupt register maintains the interrupts generated when transmit statistic counters reach half their maximum values
  (0x8000_0000 for 32 bit counter and 0x8000 for 16 bit counter), and when they cross their maximum values (0xFFFF_FFFF for 32-bit counter and 0xFFFF for 16-bit counter).
  When Counter Stop Rollover is set, the interrupts are set but the counter remains at all-ones.
  The MMC Transmit Interrupt register is a 32 bit register. An interrupt bit is cleared when the respective MMC counter that caused the interrupt is read.
  The least significant byte lane (Bits[7:0]) of the respective counter must be read to clear the interrupt bit.
- **Size:** 32 bits
- **Offset:** 0x708
- **Exists:** (DWC_EQOS_MMC_TX_EN_ATLEAST_ONE||DWC_EQOS_MMC_TX_LPI_ATLEAST_ONE)

31:28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved_31_28	TXLPITRCIS	TXLPIUSCIS	TXOSIZEGPIS	TXVLANGPIS	TXPAUSPIS	TXEXDEFFPIS	TXGPKTIS	TXGOCTIS	TXCARERPIS	TXEXCOLPIS	TXLATCOLPIS	TXDEFPIS	TXMCOLGPIS	TXSCOLGPIS	TXUFLOWERPIS	TXBCGBPIS	TXMCGBPIS	TXUCGBPIS	TX1024TMAXOCTGBPIS	TX512T1023OCTGBPIS	TX256T511OCTGBPIS	TX128T255OCTGBPIS	TX65T127OCTGBPIS	TX64OCTGBPIS	TXMCGPIS	TXBCGPIS	TXGBPKTIS	TXGBOCTIS

Table 17-86 Fields for Register: MMC_Tx_Interrupt

Bits	Name	Memory Access	Description
31:28	Reserved_31_28	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
27	TXLPITRCIS	R	MMC Transmit LPI transition counter interrupt status This bit is set when the Tx_LPI_Tran_Cntr counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): MMC Transmit LPI transition Counter Interrupt Status not detected ■ 0x1 (ACTIVE): MMC Transmit LPI transition Counter Interrupt Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_TX_LPI_TRANSITION_CNT_EN
26	TXLPIUSCIS	R	MMC Transmit LPI microsecond counter interrupt status This bit is set when the Tx_LPI_USEC_Cntr counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): MMC Transmit LPI microsecond Counter Interrupt Status not detected ■ 0x1 (ACTIVE): MMC Transmit LPI microsecond Counter Interrupt Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_TX_LPI_MICROSEC_TIMER_EN
25	TXOSIZEGPIS	R	MMC Transmit Oversize Good Packet Counter Interrupt Status This bit is set when the txoversize_g counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): MMC Transmit Oversize Good Packet Counter Interrupt Status not detected ■ 0x1 (ACTIVE): MMC Transmit Oversize Good Packet Counter Interrupt Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_TXOVERSZG_CNT_EN

Bits	Name	Memory Access	Description
24	TXVLANGPIS	R	MMC Transmit VLAN Good Packet Counter Interrupt Status This bit is set when the txvlanpackets_g counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): MMC Transmit VLAN Good Packet Counter Interrupt Status not detected ■ 0x1 (ACTIVE): MMC Transmit VLAN Good Packet Counter Interrupt Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_TXVLAN_CNT_EN
23	TXPAUSPIS	R	MMC Transmit Pause Packet Counter Interrupt Status This bit is set when the txpausepacketserror counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): MMC Transmit Pause Packet Counter Interrupt Status not detected ■ 0x1 (ACTIVE): MMC Transmit Pause Packet Counter Interrupt Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_TXPAUSE_CNT_EN
22	TXEXDEFPIS	R	MMC Transmit Excessive Deferral Packet Counter Interrupt Status This bit is set when the txexcessdef counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): MMC Transmit Excessive Deferral Packet Counter Interrupt Status not detected ■ 0x1 (ACTIVE): MMC Transmit Excessive Deferral Packet Counter Interrupt Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_TXEXSDEF_CNT_EN

Bits	Name	Memory Access	Description
21	TXGPKTIS	R	MMC Transmit Good Packet Counter Interrupt Status This bit is set when the txpacketcount_g counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): MMC Transmit Good Packet Counter Interrupt Status not detected ■ 0x1 (ACTIVE): MMC Transmit Good Packet Counter Interrupt Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_TXPKTG_CNT_EN
20	TXGOCTIS	R	MMC Transmit Good Octet Counter Interrupt Status This bit is set when the txoctetcount_g counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): MMC Transmit Good Octet Counter Interrupt Status not detected ■ 0x1 (ACTIVE): MMC Transmit Good Octet Counter Interrupt Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_TXOCTG_CNT_EN
19	TXCARERPIS	R	MMC Transmit Carrier Error Packet Counter Interrupt Status This bit is set when the txcarriererror counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): MMC Transmit Carrier Error Packet Counter Interrupt Status not detected ■ 0x1 (ACTIVE): MMC Transmit Carrier Error Packet Counter Interrupt Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_TXCARR_CNT_EN

Bits	Name	Memory Access	Description
18	TXEXCOLPIS	R	MMC Transmit Excessive Collision Packet Counter Interrupt Status This bit is set when the txexesscol counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): MMC Transmit Excessive Collision Packet Counter Interrupt Status not detected ■ 0x1 (ACTIVE): MMC Transmit Excessive Collision Packet Counter Interrupt Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_TXEXSCOL_CNT_EN
17	TXLATCOLPIS	R	MMC Transmit Late Collision Packet Counter Interrupt Status This bit is set when the txlatecol counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): MMC Transmit Late Collision Packet Counter Interrupt Status not detected ■ 0x1 (ACTIVE): MMC Transmit Late Collision Packet Counter Interrupt Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_TXLATECOL_CNT_EN
16	TXDEFPIS	R	MMC Transmit Deferred Packet Counter Interrupt Status This bit is set when the txdeferred counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): MMC Transmit Deferred Packet Counter Interrupt Status not detected ■ 0x1 (ACTIVE): MMC Transmit Deferred Packet Counter Interrupt Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_TXDEFRD_CNT_EN

Bits	Name	Memory Access	Description
15	TXMCOLGPIS	R	MMC Transmit Multiple Collision Good Packet Counter Interrupt Status This bit is set when the txmulticol_g counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): MMC Transmit Multiple Collision Good Packet Counter Interrupt Status not detected ■ 0x1 (ACTIVE): MMC Transmit Multiple Collision Good Packet Counter Interrupt Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_TXMULTCOLG_CNT_EN
14	TXSCOLGPIS	R	MMC Transmit Single Collision Good Packet Counter Interrupt Status This bit is set when the txsinglecol_g counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): MMC Transmit Single Collision Good Packet Counter Interrupt Status not detected ■ 0x1 (ACTIVE): MMC Transmit Single Collision Good Packet Counter Interrupt Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_TXSNGLCOLG_CNT_EN
13	TXUFLOWERPIS	R	MMC Transmit Underflow Error Packet Counter Interrupt Status This bit is set when the txunderflowerror counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): MMC Transmit Underflow Error Packet Counter Interrupt Status not detected ■ 0x1 (ACTIVE): MMC Transmit Underflow Error Packet Counter Interrupt Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_TXUNDRFLW_CNT_EN

Bits	Name	Memory Access	Description
12	TXBCGBPIS	R	MMC Transmit Broadcast Good Bad Packet Counter Interrupt Status This bit is set when the txbroadcastpackets_gb counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): MMC Transmit Broadcast Good Bad Packet Counter Interrupt Status not detected ■ 0x1 (ACTIVE): MMC Transmit Broadcast Good Bad Packet Counter Interrupt Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_TXBCASTGB_CNT_EN
11	TXMCGBPIS	R	MMC Transmit Multicast Good Bad Packet Counter Interrupt Status The bit is set when the txmulticastpackets_gb counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): MMC Transmit Multicast Good Bad Packet Counter Interrupt Status not detected ■ 0x1 (ACTIVE): MMC Transmit Multicast Good Bad Packet Counter Interrupt Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_TXMCASTGB_CNT_EN
10	TXUCGBPIS	R	MMC Transmit Unicast Good Bad Packet Counter Interrupt Status This bit is set when the txunicastpackets_gb counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): MMC Transmit Unicast Good Bad Packet Counter Interrupt Status not detected ■ 0x1 (ACTIVE): MMC Transmit Unicast Good Bad Packet Counter Interrupt Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_TXUCASTGB_CNT_EN

Bits	Name	Memory Access	Description
9	TX1024TMAXOCTGBPIS	R	MMC Transmit 1024 to Maximum Octet Good Bad Packet Counter Interrupt Status This bit is set when the tx1024tomaxoctets_gb counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): MMC Transmit 1024 to Maximum Octet Good Bad Packet Counter Interrupt Status not detected ■ 0x1 (ACTIVE): MMC Transmit 1024 to Maximum Octet Good Bad Packet Counter Interrupt Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_TXADDR_RNG_CNT_EN
8	TX512T1023OCTGBPIS	R	MMC Transmit 512 to 1023 Octet Good Bad Packet Counter Interrupt Status This bit is set when the tx512to1023octets_gb counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): MMC Transmit 512 to 1023 Octet Good Bad Packet Counter Interrupt Status not detected ■ 0x1 (ACTIVE): MMC Transmit 512 to 1023 Octet Good Bad Packet Counter Interrupt Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_TXADDR_RNG_CNT_EN
7	TX256T511OCTGBPIS	R	MMC Transmit 256 to 511 Octet Good Bad Packet Counter Interrupt Status This bit is set when the tx256to511octets_gb counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): MMC Transmit 256 to 511 Octet Good Bad Packet Counter Interrupt Status not detected ■ 0x1 (ACTIVE): MMC Transmit 256 to 511 Octet Good Bad Packet Counter Interrupt Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_TXADDR_RNG_CNT_EN

Bits	Name	Memory Access	Description
6	TX128T255OCTGBPIS	R	MMC Transmit 128 to 255 Octet Good Bad Packet Counter Interrupt Status This bit is set when the tx128to255octets_gb counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): MMC Transmit 128 to 255 Octet Good Bad Packet Counter Interrupt Status not detected ■ 0x1 (ACTIVE): MMC Transmit 128 to 255 Octet Good Bad Packet Counter Interrupt Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_TXADDR_RNG_CNT_EN
5	TX65T127OCTGBPIS	R	MMC Transmit 65 to 127 Octet Good Bad Packet Counter Interrupt Status This bit is set when the tx65to127octets_gb counter reaches half the maximum value, and also when it reaches the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): MMC Transmit 65 to 127 Octet Good Bad Packet Counter Interrupt Status not detected ■ 0x1 (ACTIVE): MMC Transmit 65 to 127 Octet Good Bad Packet Counter Interrupt Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_TXADDR_RNG_CNT_EN
4	TX64OCTGBPIS	R	MMC Transmit 64 Octet Good Bad Packet Counter Interrupt Status This bit is set when the tx64octets_gb counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): MMC Transmit 64 Octet Good Bad Packet Counter Interrupt Status not detected ■ 0x1 (ACTIVE): MMC Transmit 64 Octet Good Bad Packet Counter Interrupt Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_TXADDR_RNG_CNT_EN

Bits	Name	Memory Access	Description
3	TXMCGPIS	R	MMC Transmit Multicast Good Packet Counter Interrupt Status This bit is set when the txmulticastpackets_g counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): MMC Transmit Multicast Good Packet Counter Interrupt Status not detected ■ 0x1 (ACTIVE): MMC Transmit Multicast Good Packet Counter Interrupt Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_TXMCASTG_CNT_EN
2	TXBCGPIS	R	MMC Transmit Broadcast Good Packet Counter Interrupt Status This bit is set when the txbroadcastpackets_g counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): MMC Transmit Broadcast Good Packet Counter Interrupt Status not detected ■ 0x1 (ACTIVE): MMC Transmit Broadcast Good Packet Counter Interrupt Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_TXBCASTG_CNT_EN
1	TXGBPKTIS	R	MMC Transmit Good Bad Packet Counter Interrupt Status This bit is set when the txpacketcount_gb counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): MMC Transmit Good Bad Packet Counter Interrupt Status not detected ■ 0x1 (ACTIVE): MMC Transmit Good Bad Packet Counter Interrupt Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_TXPKTGB_CNT_EN

Bits	Name	Memory Access	Description
0	TXGBOCTIS	R	MMC Transmit Good Bad Octet Counter Interrupt Status This bit is set when the txoctetcount_gb counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): MMC Transmit Good Bad Octet Counter Interrupt Status not detected ■ 0x1 (ACTIVE): MMC Transmit Good Bad Octet Counter Interrupt Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_TXOCTGB_CNT_EN

17.1.83 MMC_Rx_Interrupt_Mask

- **Description:** This register maintains the masks for interrupts generated from all Receive statistics counters.
  The MMC Receive Interrupt Mask register maintains the masks for the interrupts generated when receive statistic counters reach half of their maximum value or the maximum values.
  This register is 32 bit wide.
- **Size:** 32 bits
- **Offset:** 0x70c
- **Exists:** (DWC_EQOS_MMC_RX_EN_ATLEAST_ONE||DWC_EQOS_MMC_RX_LPI_AT-LEAST_ONE)

31:28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved_31_28	RXLPITRCIM	RXLPIUSCIM	RXCTRLPIM	RXRCVERRPIM	RXWDOGPIM	RXVLANGBPIM	RXFOVPIM	RXPAUSPIM	RXORANGEPIM	RXLENERPIM	RXUCGPIM	RX1024TMAXOCTGBPIM	RX512T1023OCTGBPIM	RX256T511OCTGBPIM	RX128T255OCTGBPIM	RX65T127OCTGBPIM	RX64OCTGBPIM	RXOSIZEGPIM	RXUSIZEGPIM	RXJABERPIM	RXRUNTPIM	RXALGNERPIM	RXCRCERPIM	RXMCGPIM	RXBCGPIM	RXGOCTIM	RXGBOCTIM	RXGBPKTIM

Table 17-87 Fields for Register: MMC_Rx_Interrupt_Mask

Bits	Name	Memory Access	Description
31:28	Reserved_31_28	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
27	RXLPITRCIM	R/W	MMC Receive LPI transition counter interrupt Mask Setting this bit masks the interrupt when the Rx_LPI_Tran_Cntr counter reaches half of the maximum value or the maximum value. **Values:** ■ 0x0 (DISABLE): MMC Receive LPI transition counter interrupt Mask is disabled ■ 0x1 (ENABLE): MMC Receive LPI transition counter interrupt Mask is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RX_LPI_TRANSITION_CNT_EN

Bits	Name	Memory Access	Description
26	RXLPIUSCIM	R/W	MMC Receive LPI microsecond counter interrupt Mask Setting this bit masks the interrupt when the Rx_LPI_USEC_Cntr counter reaches half of the maximum value or the maximum value. **Values:** ■ 0x0 (DISABLE): MMC Receive LPI microsecond counter interrupt Mask is disabled ■ 0x1 (ENABLE): MMC Receive LPI microsecond counter interrupt Mask is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RX_LPI_MICROSEC_TIMER_EN
25	RXCTRLPIM	R/W	MMC Receive Control Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rxctrlpackets_g counter reaches half of the maximum value or the maximum value. **Values:** ■ 0x0 (DISABLE): MMC Receive Control Packet Counter Interrupt Mask is disabled ■ 0x1 (ENABLE): MMC Receive Control Packet Counter Interrupt Mask is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RXCTRLPKT_CNT_EN
24	RXRCVERRPIM	R/W	MMC Receive Error Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rxrcverror counter reaches half of the maximum value or the maximum value. **Values:** ■ 0x0 (DISABLE): MMC Receive Error Packet Counter Interrupt Mask is disabled ■ 0x1 (ENABLE): MMC Receive Error Packet Counter Interrupt Mask is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RXRCVERR_CNT_EN
23	RXWDOGPIM	R/W	MMC Receive Watchdog Error Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rxwatchdog counter reaches half of the maximum value or the maximum value. **Values:** ■ 0x0 (DISABLE): MMC Receive Watchdog Error Packet Counter Interrupt Mask is disabled ■ 0x1 (ENABLE): MMC Receive Watchdog Error Packet Counter Interrupt Mask is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RXWDGERR_CNT_EN

Bits	Name	Memory Access	Description
22	RXVLANGBPIM	R/W	MMC Receive VLAN Good Bad Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rxvlanpackets_gb counter reaches half of the maximum value or the maximum value. **Values:** ■ 0x0 (DISABLE): MMC Receive VLAN Good Bad Packet Counter Interrupt Mask is disabled ■ 0x1 (ENABLE): MMC Receive VLAN Good Bad Packet Counter Interrupt Mask is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RXVLANPKTGB_CNT_EN
21	RXFOVPIM	R/W	MMC Receive FIFO Overflow Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rxfifooverflow counter reaches half of the maximum value or the maximum value. **Values:** ■ 0x0 (DISABLE): MMC Receive FIFO Overflow Packet Counter Interrupt Mask is disabled ■ 0x1 (ENABLE): MMC Receive FIFO Overflow Packet Counter Interrupt Mask is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RXFIFOOVFL_CNT_EN
20	RXPAUSPIM	R/W	MMC Receive Pause Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rxpausepackets counter reaches half of the maximum value or the maximum value. **Values:** ■ 0x0 (DISABLE): MMC Receive Pause Packet Counter Interrupt Mask is disabled ■ 0x1 (ENABLE): MMC Receive Pause Packet Counter Interrupt Mask is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RXPAUSEPKT_CNT_EN
19	RXORANGEPIM	R/W	MMC Receive Out Of Range Error Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rxoutofrangetype counter reaches half of the maximum value or the maximum value. **Values:** ■ 0x0 (DISABLE): MMC Receive Out Of Range Error Packet Counter Interrupt Mask is disabled ■ 0x1 (ENABLE): MMC Receive Out Of Range Error Packet Counter Interrupt Mask is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RXOUTOFRNG_TYP_CNT_EN

Bits	Name	Memory Access	Description
18	RXLENERPIM	R/W	MMC Receive Length Error Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rxlengtherror counter reaches half of the maximum value or the maximum value. **Values:** ■ 0x0 (DISABLE): MMC Receive Length Error Packet Counter Interrupt Mask is disabled ■ 0x1 (ENABLE): MMC Receive Length Error Packet Counter Interrupt Mask is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RXLENERR_CNT_EN
17	RXUCGPIM	R/W	MMC Receive Unicast Good Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rxunicastpackets_g counter reaches half of the maximum value or the maximum value. **Values:** ■ 0x0 (DISABLE): MMC Receive Unicast Good Packet Counter Interrupt Mask is disabled ■ 0x1 (ENABLE): MMC Receive Unicast Good Packet Counter Interrupt Mask is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RXUCASTG_CNT_EN
16	RX1024TMAXOCTGBPIM	R/W	MMC Receive 1024 to Maximum Octet Good Bad Packet Counter Interrupt Mask. Setting this bit masks the interrupt when the rx1024tomaxoctets_gb counter reaches half of the maximum value or the maximum value. **Values:** ■ 0x0 (DISABLE): MMC Receive 1024 to Maximum Octet Good Bad Packet Counter Interrupt Mask is disabled ■ 0x1 (ENABLE): MMC Receive 1024 to Maximum Octet Good Bad Packet Counter Interrupt Mask is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RXADDR_RNG_CNT_EN

Bits	Name	Memory Access	Description
15	RX512T1023OCTGBPIM	R/W	MMC Receive 512 to 1023 Octet Good Bad Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rx512to1023octets_gb counter reaches half of the maximum value or the maximum value. **Values:** ■ 0x0 (DISABLE): MMC Receive 512 to 1023 Octet Good Bad Packet Counter Interrupt Mask is disabled ■ 0x1 (ENABLE): MMC Receive 512 to 1023 Octet Good Bad Packet Counter Interrupt Mask is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RXADDR_RNG_CNT_EN
14	RX256T511OCTGBPIM	R/W	MMC Receive 256 to 511 Octet Good Bad Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rx256to511octets_gb counter reaches half of the maximum value or the maximum value. **Values:** ■ 0x0 (DISABLE): MMC Receive 256 to 511 Octet Good Bad Packet Counter Interrupt Mask is disabled ■ 0x1 (ENABLE): MMC Receive 256 to 511 Octet Good Bad Packet Counter Interrupt Mask is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RXADDR_RNG_CNT_EN
13	RX128T255OCTGBPIM	R/W	MMC Receive 128 to 255 Octet Good Bad Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rx128to255octets_gb counter reaches half of the maximum value or the maximum value. **Values:** ■ 0x0 (DISABLE): MMC Receive 128 to 255 Octet Good Bad Packet Counter Interrupt Mask is disabled ■ 0x1 (ENABLE): MMC Receive 128 to 255 Octet Good Bad Packet Counter Interrupt Mask is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RXADDR_RNG_CNT_EN

Bits	Name	Memory Access	Description
12	RX65T127OCTGBPIM	R/W	MMC Receive 65 to 127 Octet Good Bad Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rx65to127octets_gb counter reaches half of the maximum value or the maximum value. **Values:** ■ 0x0 (DISABLE): MMC Receive 65 to 127 Octet Good Bad Packet Counter Interrupt Mask is disabled ■ 0x1 (ENABLE): MMC Receive 65 to 127 Octet Good Bad Packet Counter Interrupt Mask is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RXADDR_RNG_CNT_EN
11	RX64OCTGBPIM	R/W	MMC Receive 64 Octet Good Bad Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rx64octets_gb counter reaches half of the maximum value or the maximum value. **Values:** ■ 0x0 (DISABLE): MMC Receive 64 Octet Good Bad Packet Counter Interrupt Mask is disabled ■ 0x1 (ENABLE): MMC Receive 64 Octet Good Bad Packet Counter Interrupt Mask is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RXADDR_RNG_CNT_EN
10	RXOSIZEGPIM	R/W	MMC Receive Oversize Good Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rxoversize_g counter reaches half of the maximum value or the maximum value. **Values:** ■ 0x0 (DISABLE): MMC Receive Oversize Good Packet Counter Interrupt Mask is disabled ■ 0x1 (ENABLE): MMC Receive Oversize Good Packet Counter Interrupt Mask is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RXOVERSZG_CNT_EN

Bits	Name	Memory Access	Description
9	RXUSIZEGPIM	R/W	MMC Receive Undersize Good Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rxundersize_g counter reaches half of the maximum value or the maximum value. **Values:** ■ 0x0 (DISABLE): MMC Receive Undersize Good Packet Counter Interrupt Mask is disabled ■ 0x1 (ENABLE): MMC Receive Undersize Good Packet Counter Interrupt Mask is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RXUNDERSZG_CNT_EN
8	RXJABERPIM	R/W	MMC Receive Jabber Error Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rxjabbererror counter reaches half of the maximum value or the maximum value. **Values:** ■ 0x0 (DISABLE): MMC Receive Jabber Error Packet Counter Interrupt Mask is disabled ■ 0x1 (ENABLE): MMC Receive Jabber Error Packet Counter Interrupt Mask is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RXJABERR_CNT_EN
7	RXRUNTPIM	R/W	MMC Receive Runt Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rxrunterror counter reaches half of the maximum value or the maximum value. **Values:** ■ 0x0 (DISABLE): MMC Receive Runt Packet Counter Interrupt Mask is disabled ■ 0x1 (ENABLE): MMC Receive Runt Packet Counter Interrupt Mask is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RXRUNTERR_CNT_EN
6	RXALGNERPIM	R/W	MMC Receive Alignment Error Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rxalignmenterror counter reaches half of the maximum value or the maximum value. **Values:** ■ 0x0 (DISABLE): MMC Receive Alignment Error Packet Counter Interrupt Mask is disabled ■ 0x1 (ENABLE): MMC Receive Alignment Error Packet Counter Interrupt Mask is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RXALGNERR_CNT_EN

Bits	Name	Memory Access	Description
5	RXCRCERPIM	R/W	MMC Receive CRC Error Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rxcrcerror counter reaches half of the maximum value or the maximum value. **Values:** ■ 0x0 (DISABLE): MMC Receive CRC Error Packet Counter Interrupt Mask is disabled ■ 0x1 (ENABLE): MMC Receive CRC Error Packet Counter Interrupt Mask is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RXCRCERR_CNT_EN
4	RXMCGPIM	R/W	MMC Receive Multicast Good Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rxmulticastpackets_g counter reaches half of the maximum value or the maximum value. **Values:** ■ 0x0 (DISABLE): MMC Receive Multicast Good Packet Counter Interrupt Mask is disabled ■ 0x1 (ENABLE): MMC Receive Multicast Good Packet Counter Interrupt Mask is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RXMCASTG_CNT_EN
3	RXBCGPIM	R/W	MMC Receive Broadcast Good Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rxbroadcastpackets_g counter reaches half of the maximum value or the maximum value. **Values:** ■ 0x0 (DISABLE): MMC Receive Broadcast Good Packet Counter Interrupt Mask is disabled ■ 0x1 (ENABLE): MMC Receive Broadcast Good Packet Counter Interrupt Mask is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RXBCASTG_CNT_EN
2	RXGOCTIM	R/W	MMC Receive Good Octet Counter Interrupt Mask Setting this bit masks the interrupt when the rxoctetcount_g counter reaches half of the maximum value or the maximum value. **Values:** ■ 0x0 (DISABLE): MMC Receive Good Octet Counter Interrupt Mask is disabled ■ 0x1 (ENABLE): MMC Receive Good Octet Counter Interrupt Mask is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RXOCTG_CNT_EN

Bits	Name	Memory Access	Description
1	RXGBOCTIM	R/W	MMC Receive Good Bad Octet Counter Interrupt Mask Setting this bit masks the interrupt when the rxoctetcount_gb counter reaches half of the maximum value or the maximum value. **Values:** ■ 0x0 (DISABLE): MMC Receive Good Bad Octet Counter Interrupt Mask is disabled ■ 0x1 (ENABLE): MMC Receive Good Bad Octet Counter Interrupt Mask is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RXOCTGB_CNT_EN
0	RXGBPKTIM	R/W	MMC Receive Good Bad Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rxpacketcount_gb counter reaches half of the maximum value or the maximum value. **Values:** ■ 0x0 (DISABLE): MMC Receive Good Bad Packet Counter Interrupt Mask is disabled ■ 0x1 (ENABLE): MMC Receive Good Bad Packet Counter Interrupt Mask is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RXPKTGB_CNT_EN

17.1.84 MMC_Tx_Interrupt_Mask

- **Description:** This register maintains the masks for interrupts generated from all Transmit statistics counters.
  The MMC Transmit Interrupt Mask register maintains the masks for the interrupts generated when the transmit statistic counters reach half of their maximum value or the maximum values. This register is 32 bit wide.
- **Size:** 32 bits
- **Offset:** 0x710
- **Exists:** (DWC_EQOS_MMC_TX_EN_ATLEAST_ONE||DWC_EQOS_MMC_TX_LPI_ATLEAST_ONE)

31:28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved_31_28	TXLPITRCIM	TXLPIUSCIM	TXOSIZEGPIM	TXVLANGPIM	TXPAUSPIM	TXEXDEFPIM	TXGPKTIM	TXGOCTIM	TXCARERPIM	TXEXCOLPIM	TXLATCOLPIM	TXDEFPIM	TXMCOLGPIM	TXSCOLGPIM	TXUFLOWERPIM	TXBCGBPIM	TXMCGBPIM	TXUCGBPIM	TX1024TMAXOCTGBPIM	TX512T1023OCTGBPIM	TX256T511OCTGBPIM	TX128T255OCTGBPIM	TX65T127OCTGBPIM	TX64OCTGBPIM	TXMCGPIM	TXBCGPIM	TXGBPKTIM	TXGBOCTIM

Table 17-88 Fields for Register: MMC_Tx_Interrupt_Mask

Bits	Name	Memory Access	Description
31:28	Reserved_31_28	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
27	TXLPITRCIM	R/W	MMC Transmit LPI transition counter interrupt Mask Setting this bit masks the interrupt when the Tx_LPI_Tran_Cntr counter reaches half of the maximum value or the maximum value. **Values:** ■ 0x0 (DISABLE): MMC Transmit LPI transition counter interrupt Mask is disabled ■ 0x1 (ENABLE): MMC Transmit LPI transition counter interrupt Mask is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_TX_LPI_TRANSITION_CNT_EN

Bits	Name	Memory Access	Description
26	TXLPIUSCIM	R/W	MMC Transmit LPI microsecond counter interrupt Mask Setting this bit masks the interrupt when the Tx_LPI_USEC_Cntr counter reaches half of the maximum value or the maximum value. **Values:** ■ 0x0 (DISABLE): MMC Transmit LPI microsecond counter interrupt Mask is disabled ■ 0x1 (ENABLE): MMC Transmit LPI microsecond counter interrupt Mask is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_TX_LPI_MICROSEC_TIMER_EN
25	TXOSIZEGPIM	R/W	MMC Transmit Oversize Good Packet Counter Interrupt Mask Setting this bit masks the interrupt when the txoversize_g counter reaches half of the maximum value or the maximum value. **Values:** ■ 0x0 (DISABLE): MMC Transmit Oversize Good Packet Counter Interrupt Mask is disabled ■ 0x1 (ENABLE): MMC Transmit Oversize Good Packet Counter Interrupt Mask is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_TXOVERSZG_CNT_EN
24	TXVLANGPIM	R/W	MMC Transmit VLAN Good Packet Counter Interrupt Mask Setting this bit masks the interrupt when the txvlanpackets_g counter reaches half of the maximum value or the maximum value. **Values:** ■ 0x0 (DISABLE): MMC Transmit VLAN Good Packet Counter Interrupt Mask is disabled ■ 0x1 (ENABLE): MMC Transmit VLAN Good Packet Counter Interrupt Mask is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_TXVLAN_CNT_EN
23	TXPAUSPIM	R/W	MMC Transmit Pause Packet Counter Interrupt Mask Setting this bit masks the interrupt when the txpausepackets counter reaches half of the maximum value or the maximum value. **Values:** ■ 0x0 (DISABLE): MMC Transmit Pause Packet Counter Interrupt Mask is disabled ■ 0x1 (ENABLE): MMC Transmit Pause Packet Counter Interrupt Mask is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_TXPAUSE_CNT_EN

Bits	Name	Memory Access	Description
22	TXEXDEFPIM	R/W	MMC Transmit Excessive Deferral Packet Counter Interrupt Mask Setting this bit masks the interrupt when the txexcessdef counter reaches half of the maximum value or the maximum value. **Values:** ■ 0x0 (DISABLE): MMC Transmit Excessive Deferral Packet Counter Interrupt Mask is disabled ■ 0x1 (ENABLE): MMC Transmit Excessive Deferral Packet Counter Interrupt Mask is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_TXEXSDEF_CNT_EN
21	TXGPKTIM	R/W	MMC Transmit Good Packet Counter Interrupt Mask Setting this bit masks the interrupt when the txpacketcount_g counter reaches half of the maximum value or the maximum value. **Values:** ■ 0x0 (DISABLE): MMC Transmit Good Packet Counter Interrupt Mask is disabled ■ 0x1 (ENABLE): MMC Transmit Good Packet Counter Interrupt Mask is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_TXPKTG_CNT_EN
20	TXGOCTIM	R/W	MMC Transmit Good Octet Counter Interrupt Mask Setting this bit masks the interrupt when the txoctetcount_g counter reaches half of the maximum value or the maximum value. **Values:** ■ 0x0 (DISABLE): MMC Transmit Good Octet Counter Inter-rupt Mask is disabled ■ 0x1 (ENABLE): MMC Transmit Good Octet Counter Inter-rupt Mask is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_TXOCTG_CNT_EN
19	TXCARERPIM	R/W	MMC Transmit Carrier Error Packet Counter Interrupt Mask Setting this bit masks the interrupt when the txcarriererror counter reaches half of the maximum value or the maximum value. **Values:** ■ 0x0 (DISABLE): MMC Transmit Carrier Error Packet Counter Interrupt Mask is disabled ■ 0x1 (ENABLE): MMC Transmit Carrier Error Packet Counter Interrupt Mask is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_TXCARR_CNT_EN

Bits	Name	Memory Access	Description
18	TXEXCOLPIM	R/W	MMC Transmit Excessive Collision Packet Counter Interrupt Mask Setting this bit masks the interrupt when the txexcesscol counter reaches half of the maximum value or the maximum value. **Values:** ■ 0x0 (DISABLE): MMC Transmit Excessive Collision Packet Counter Interrupt Mask is disabled ■ 0x1 (ENABLE): MMC Transmit Excessive Collision Packet Counter Interrupt Mask is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_TXEXSCOL_CNT_EN
17	TXLATCOLPIM	R/W	MMC Transmit Late Collision Packet Counter Interrupt Mask Setting this bit masks the interrupt when the txlatecol counter reaches half of the maximum value or the maximum value. **Values:** ■ 0x0 (DISABLE): MMC Transmit Late Collision Packet Counter Interrupt Mask is disabled ■ 0x1 (ENABLE): MMC Transmit Late Collision Packet Counter Interrupt Mask is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_TXLATECOL_CNT_EN
16	TXDEFPIM	R/W	MMC Transmit Deferred Packet Counter Interrupt Mask Setting this bit masks the interrupt when the txdeferred counter reaches half of the maximum value or the maximum value. **Values:** ■ 0x0 (DISABLE): MMC Transmit Deferred Packet Counter Interrupt Mask is disabled ■ 0x1 (ENABLE): MMC Transmit Deferred Packet Counter Interrupt Mask is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_TXDEFRD_CNT_EN
15	TXMCOLGPIM	R/W	MMC Transmit Multiple Collision Good Packet Counter Interrupt Mask Setting this bit masks the interrupt when the txmulticol_g counter reaches half of the maximum value or the maximum value. **Values:** ■ 0x0 (DISABLE): MMC Transmit Multiple Collision Good Packet Counter Interrupt Mask is disabled ■ 0x1 (ENABLE): MMC Transmit Multiple Collision Good Packet Counter Interrupt Mask is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_TXMULTCOLG_CNT_EN

Bits	Name	Memory Access	Description
14	TXSCOLGPIM	R/W	MMC Transmit Single Collision Good Packet Counter Interrupt Mask Setting this bit masks the interrupt when the txsinglecol_g counter reaches half of the maximum value or the maximum value. **Values:** ■ 0x0 (DISABLE): MMC Transmit Single Collision Good Packet Counter Interrupt Mask is disabled ■ 0x1 (ENABLE): MMC Transmit Single Collision Good Packet Counter Interrupt Mask is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_TXSNGLCOLG_CNT_EN
13	TXUFLOWERPIM	R/W	MMC Transmit Underflow Error Packet Counter Interrupt Mask Setting this bit masks the interrupt when the txunderflowerror counter reaches half of the maximum value or the maximum value. **Values:** ■ 0x0 (DISABLE): MMC Transmit Underflow Error Packet Counter Interrupt Mask is disabled ■ 0x1 (ENABLE): MMC Transmit Underflow Error Packet Counter Interrupt Mask is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_TXUNDRFLW_CNT_EN
12	TXBCGBPIM	R/W	MMC Transmit Broadcast Good Bad Packet Counter Interrupt Mask Setting this bit masks the interrupt when the txbroadcastpackets_gb counter reaches half of the maximum value or the maximum value. **Values:** ■ 0x0 (DISABLE): MMC Transmit Broadcast Good Bad Packet Counter Interrupt Mask is disabled ■ 0x1 (ENABLE): MMC Transmit Broadcast Good Bad Packet Counter Interrupt Mask is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_TXBCASTGB_CNT_EN

Bits	Name	Memory Access	Description
11	TXMCGBPIM	R/W	MMC Transmit Multicast Good Bad Packet Counter Interrupt Mask Setting this bit masks the interrupt when the txmulticastpackets_gb counter reaches half of the maximum value or the maximum value. **Values:** ■ 0x0 (DISABLE): MMC Transmit Multicast Good Bad Packet Counter Interrupt Mask is disabled ■ 0x1 (ENABLE): MMC Transmit Multicast Good Bad Packet Counter Interrupt Mask is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_TXMCASTGB_CNT_EN
10	TXUCGBPIM	R/W	MMC Transmit Unicast Good Bad Packet Counter Interrupt Mask Setting this bit masks the interrupt when the txunicastpackets_gb counter reaches half of the maximum value or the maximum value. **Values:** ■ 0x0 (DISABLE): MMC Transmit Unicast Good Bad Packet Counter Interrupt Mask is disabled ■ 0x1 (ENABLE): MMC Transmit Unicast Good Bad Packet Counter Interrupt Mask is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_TXUCASTGB_CNT_EN
9	TX1024TMAXOCTGBPIM	R/W	MMC Transmit 1024 to Maximum Octet Good Bad Packet Counter Interrupt Mask Setting this bit masks the interrupt when the tx1024tomaxoctets_gb counter reaches half of the maximum value or the maximum value. **Values:** ■ 0x0 (DISABLE): MMC Transmit 1024 to Maximum Octet Good Bad Packet Counter Interrupt Mask is disabled ■ 0x1 (ENABLE): MMC Transmit 1024 to Maximum Octet Good Bad Packet Counter Interrupt Mask is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_TXADDR_RNG_CNT_EN

Bits	Name	Memory Access	Description
8	TX512T1023OCTGBPIM	R/W	MMC Transmit 512 to 1023 Octet Good Bad Packet Counter Interrupt Mask Setting this bit masks the interrupt when the tx512to1023octets_gb counter reaches half of the maximum value or the maximum value. **Values:** ■ 0x0 (DISABLE): MMC Transmit 512 to 1023 Octet Good Bad Packet Counter Interrupt Mask is disabled ■ 0x1 (ENABLE): MMC Transmit 512 to 1023 Octet Good Bad Packet Counter Interrupt Mask is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_TXADDR_RNG_CNT_EN
7	TX256T511OCTGBPIM	R/W	MMC Transmit 256 to 511 Octet Good Bad Packet Counter Interrupt Mask Setting this bit masks the interrupt when the tx256to511octets_gb counter reaches half of the maximum value or the maximum value. **Values:** ■ 0x0 (DISABLE): MMC Transmit 256 to 511 Octet Good Bad Packet Counter Interrupt Mask is disabled ■ 0x1 (ENABLE): MMC Transmit 256 to 511 Octet Good Bad Packet Counter Interrupt Mask is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_TXADDR_RNG_CNT_EN
6	TX128T255OCTGBPIM	R/W	MMC Transmit 128 to 255 Octet Good Bad Packet Counter Interrupt Mask Setting this bit masks the interrupt when the tx128to255octets_gb counter reaches half of the maximum value or the maximum value. **Values:** ■ 0x0 (DISABLE): MMC Transmit 128 to 255 Octet Good Bad Packet Counter Interrupt Mask is disabled ■ 0x1 (ENABLE): MMC Transmit 128 to 255 Octet Good Bad Packet Counter Interrupt Mask is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_TXADDR_RNG_CNT_EN

Bits	Name	Memory Access	Description
5	TX65T127OCTGBPIM	R/W	MMC Transmit 65 to 127 Octet Good Bad Packet Counter Interrupt Mask Setting this bit masks the interrupt when the tx65to127octets_gb counter reaches half of the maximum value or the maximum value. **Values:** ■ 0x0 (DISABLE): MMC Transmit 65 to 127 Octet Good Bad Packet Counter Interrupt Mask is disabled ■ 0x1 (ENABLE): MMC Transmit 65 to 127 Octet Good Bad Packet Counter Interrupt Mask is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_TXADDR_RNG_CNT_EN
4	TX64OCTGBPIM	R/W	MMC Transmit 64 Octet Good Bad Packet Counter Interrupt Mask Setting this bit masks the interrupt when the tx64octets_gb counter reaches half of the maximum value or the maximum value. **Values:** ■ 0x0 (DISABLE): MMC Transmit 64 Octet Good Bad Packet Counter Interrupt Mask is disabled ■ 0x1 (ENABLE): MMC Transmit 64 Octet Good Bad Packet Counter Interrupt Mask is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_TXADDR_RNG_CNT_EN
3	TXMCGPIM	R/W	MMC Transmit Multicast Good Packet Counter Interrupt Mask Setting this bit masks the interrupt when the txmulticastpackets_g counter reaches half of the maximum value or the maximum value. **Values:** ■ 0x0 (DISABLE): MMC Transmit Multicast Good Packet Counter Interrupt Mask is disabled ■ 0x1 (ENABLE): MMC Transmit Multicast Good Packet Counter Interrupt Mask is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_TXMCASTG_CNT_EN

Bits	Name	Memory Access	Description
2	TXBCGPIM	R/W	MMC Transmit Broadcast Good Packet Counter Interrupt Mask Setting this bit masks the interrupt when the txbroadcastpackets_g counter reaches half of the maximum value or the maximum value. **Values:** ■ 0x0 (DISABLE): MMC Transmit Broadcast Good Packet Counter Interrupt Mask is disabled ■ 0x1 (ENABLE): MMC Transmit Broadcast Good Packet Counter Interrupt Mask is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_TXBCASTG_CNT_EN
1	TXGBPKTIM	R/W	MMC Transmit Good Bad Packet Counter Interrupt Mask Setting this bit masks the interrupt when the txpacketcount_gb counter reaches half of the maximum value or the maximum value. **Values:** ■ 0x0 (DISABLE): MMC Transmit Good Bad Packet Counter Interrupt Mask is disabled ■ 0x1 (ENABLE): MMC Transmit Good Bad Packet Counter Interrupt Mask is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_TXPKTGB_CNT_EN
0	TXGBOCTIM	R/W	MMC Transmit Good Bad Octet Counter Interrupt Mask Setting this bit masks the interrupt when the txoctetcount_gb counter reaches half of the maximum value or the maximum value. **Values:** ■ 0x0 (DISABLE): MMC Transmit Good Bad Octet Counter Interrupt Mask is disabled ■ 0x1 (ENABLE): MMC Transmit Good Bad Octet Counter Interrupt Mask is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_TXOCTGB_CNT_EN

17.1.85 Tx_Octet_Count_Good_Bad

- **Description:** This register provides the number of bytes transmitted by the DWC_ether_qos, exclusive of preamble and retried bytes, in good and bad packets.
- **Size:** 32 bits
- **Offset:** 0x714
- **Exists:** (DWC_EQOS_MMC_TXOCTGB_CNT_EN)

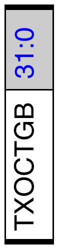


Table 17-89 Fields for Register: Tx_Octet_Count_Good_Bad

Bits	Name	Memory Access	Description
31:0	TXOCTGB	R	Tx Octet Count Good Bad This field indicates the number of bytes transmitted, exclusive of preamble and retried bytes, in good and bad packets. **Value After Reset:** 0x0 **Exists:** Always

17.1.86 Tx_Packet_Count_Good_Bad

- **Description:** This register provides the number of good and bad packets transmitted by DWC_ether_qos, exclusive of retried packets.
- **Size:** 32 bits
- **Offset:** 0x718
- **Exists:** (DWC_EQOS_MMC_TXPKTGB_CNT_EN)

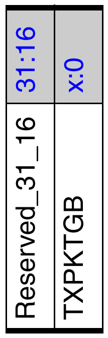


Table 17-90 Fields for Register: Tx_Packet_Count_Good_Bad

Bits	Name	Memory Access	Description
31:16	Reserved_31_16	R	Reserved. **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_MMC_TXPKTGB_CNT_16BIT_EN)
x:0	TXPKTGB	R	Tx Packet Count Good Bad This field indicates the number of good and bad packets transmitted, exclusive of retried packets. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[x]:** DWC_EQOS_MMC_TXPKTGB_CNT_16BIT_EN?\"16\":\"32\" - 1

17.1.87 Tx_Broadcast_Packets_Good

- **Description:** This register provides the number of good broadcast packets transmitted by DWC_ether_qos.
- **Size:** 32 bits
- **Offset:** 0x71c
- **Exists:** (DWC_EQOS_MMC_TXBCASTG_CNT_EN)

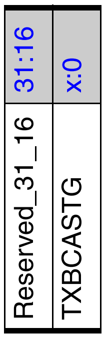


Table 17-91 Fields for Register: Tx_Broadcast_Packets_Good

Bits	Name	Memory Access	Description
31:16	Reserved_31_16	R	Reserved. **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_MMC_TXBCASTG_CNT_16BIT_EN)
x:0	TXBCASTG	R	Tx Broadcast Packets Good This field indicates the number of good broadcast packets transmitted. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[x]:** DWC_EQOS_MMC_TXBCASTG_CNT_16BIT_EN?\"16\":\"32\" - 1

17.1.88 Tx_Multicast_Packets_Good

- **Description:** This register provides the number of good multicast packets transmitted by DWC_ether_qos.
- **Size:** 32 bits
- **Offset:** 0x720
- **Exists:** (DWC_EQOS_MMC_TXMCASTG_CNT_EN)

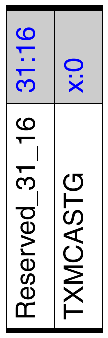


Table 17-92 Fields for Register: Tx_Multicast_Packets_Good

Bits	Name	Memory Access	Description
31:16	Reserved_31_16	R	Reserved. **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_MMC_TXMCASTG_CNT_16BIT_EN)
x:0	TXMCASTG	R	Tx Multicast Packets Good This field indicates the number of good multicast packets transmitted. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[x]:** DWC_EQOS_MMC_TXMCASTG_CNT_16BIT_EN?\"16\":\"32\" - 1

17.1.89 Tx_64Octets_Packets_Good_Bad

- **Description:** This register provides the number of good and bad packets transmitted by DWC_ether_qos with length 64 bytes, exclusive of preamble and retried packets.
- **Size:** 32 bits
- **Offset:** 0x724
- **Exists:** (DWC_EQOS_MMC_TXADDR_RNG_CNT_EN)

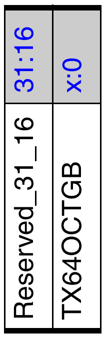


Table 17-93 Fields for Register: Tx_64Octets_Packets_Good_Bad

Bits	Name	Memory Access	Description
31:16	Reserved_31_16	R	Reserved. **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_MMC_TXADDR_RNG_CNT_16BIT_EN)
x:0	TX64OCTGB	R	Tx 64Octets Packets Good_Bad This field indicates the number of good and bad packets transmitted with length 64 bytes, exclusive of preamble and retried packets. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[x]:** DWC_EQOS_MMC_TXADDR_RNG_CNT_16BIT_EN?\"16\":\"32\" - 1

17.1.90 Tx_65To127Octets_Packets_Good_Bad

- **Description:** This register provides the number of good and bad packets transmitted by DWC_ether_qos with length between 65 and 127 (inclusive) bytes, exclusive of preamble and retried packets.
- **Size:** 32 bits
- **Offset:** 0x728
- **Exists:** (DWC_EQOS_MMC_TXADDR_RNG_CNT_EN)

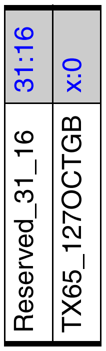


Table 17-94 Fields for Register: Tx_65To127Octets_Packets_Good_Bad

Bits	Name	Memory Access	Description
31:16	Reserved_31_16	R	Reserved. **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_MMC_TXADDR_RNG_CNT_16BIT_EN)
x:0	TX65_127OCTGB	R	Tx 65To127Octets Packets Good Bad This field indicates the number of good and bad packets transmitted with length between 65 and 127 (inclusive) bytes, exclusive of preamble and retried packets. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[x]:** DWC_EQOS_MMC_TXADDR_RNG_CNT_16BIT_EN?\"16\":\"32\" - 1

17.1.91 Tx_128To255Octets_Packets_Good_Bad

- **Description:** This register provides the number of good and bad packets transmitted by DWC_ether_qos with length between 128 to 255 (inclusive) bytes, exclusive of preamble and retried packets.
- **Size:** 32 bits
- **Offset:** 0x72c
- **Exists:** (DWC_EQOS_MMC_TXADDR_RNG_CNT_EN)

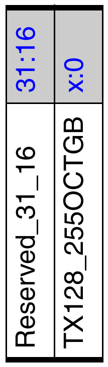


Table 17-95 Fields for Register: Tx_128To255Octets_Packets_Good_Bad

Bits	Name	Memory Access	Description
31:16	Reserved_31_16	R	Reserved. **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_MMC_TXADDR_RNG_CNT_16BIT_EN)
x:0	TX128_255OCTGB	R	Tx 128To255Octets Packets Good Bad This field indicates the number of good and bad packets transmitted with length between 128 and 255 (inclusive) bytes, exclusive of preamble and retried packets. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[x]:** DWC_EQOS_MMC_TXADDR_RNG_CNT_16BIT_EN?\"16\":\"32\" - 1

17.1.92 Tx_256To511Octets_Packets_Good_Bad

- **Description:** This register provides the number of good and bad packets transmitted by DWC_ether_qos with length between 256 to 511 (inclusive) bytes, exclusive of preamble and retried packets.
- **Size:** 32 bits
- **Offset:** 0x730
- **Exists:** (DWC_EQOS_MMC_TXADDR_RNG_CNT_EN)

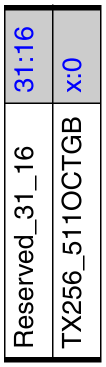


Table 17-96 Fields for Register: Tx_256To511Octets_Packets_Good_Bad

Bits	Name	Memory Access	Description
31:16	Reserved_31_16	R	Reserved. **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_MMC_TXADDR_RNG_CNT_16BIT_EN)
x:0	TX256_511OCTGB	R	Tx 256To511Octets Packets Good Bad This field indicates the number of good and bad packets transmitted with length between 256 and 511 (inclusive) bytes, exclusive of preamble and retried packets. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[x]:** DWC_EQOS_MMC_TXADDR_RNG_CNT_16BIT_EN?\"16\":\"32\" - 1

17.1.93 Tx_512To1023Octets_Packets_Good_Bad

- **Description:** This register provides the number of good and bad packets transmitted by DWC_ether_qos with length 512 to 1023 (inclusive) bytes, exclusive of preamble and retried packets.
- **Size:** 32 bits
- **Offset:** 0x734
- **Exists:** (DWC_EQOS_MMC_TXADDR_RNG_CNT_EN)

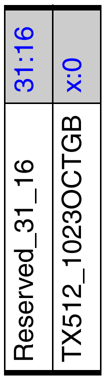


Table 17-97 Fields for Register: Tx_512To1023Octets_Packets_Good_Bad

Bits	Name	Memory Access	Description
31:16	Reserved_31_16	R	Reserved. **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_MMC_TXADDR_RNG_CNT_16BIT_EN)
x:0	TX512_1023OCTGB	R	Tx 512To1023Octets Packets Good Bad This field indicates the number of good and bad packets transmitted with length between 512 and 1023 (inclusive) bytes, exclusive of preamble and retried packets. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[x]:** DWC_EQOS_MMC_TXADDR_RNG_CNT_16BIT_EN?\"16\":\"32\" - 1

17.1.94 Tx_1024ToMaxOctets_Packets_Good_Bad

- **Description:** This register provides the number of good and bad packets transmitted by DWC_ether_qos with length 1024 to maxsize (inclusive) bytes, exclusive of preamble and retried packets.
- **Size:** 32 bits
- **Offset:** 0x738
- **Exists:** (DWC_EQOS_MMC_TXADDR_RNG_CNT_EN)

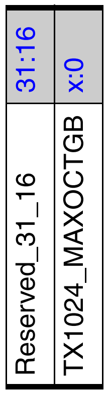


Table 17-98 Fields for Register: Tx_1024ToMaxOctets_Packets_Good_Bad

Bits	Name	Memory Access	Description
31:16	Reserved_31_16	R	Reserved. **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_MMC_TXADDR_RNG_CNT_16BIT_EN)
x:0	TX1024_MAXOCTGB	R	Tx 1024ToMaxOctets Packets Good Bad This field indicates the number of good and bad packets transmitted with length between 1024 and maxsize (inclusive) bytes, exclusive of preamble and retried packets. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[x]:** DWC_EQOS_MMC_TXADDR_RNG_CNT_16BIT_EN?\"16\":\"32\" - 1

17.1.95 Tx_Unicast_Packets_Good_Bad

- **Description:** This register provides the number of good and bad unicast packets transmitted by DWC_ether_qos.
- **Size:** 32 bits
- **Offset:** 0x73c
- **Exists:** (DWC_EQOS_MMC_TXUCASTGB_CNT_EN)

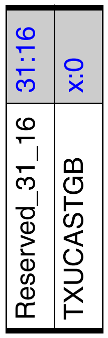


Table 17-99 Fields for Register: Tx_Unicast_Packets_Good_Bad

Bits	Name	Memory Access	Description
31:16	Reserved_31_16	R	Reserved. **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_MMC_TXUCASTGB_CNT_16BIT_EN)
x:0	TXUCASTGB	R	Tx Unicast Packets Good Bad This field indicates the number of good and bad unicast packets transmitted. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[x]:** DWC_EQOS_MMC_TXUCASTGB_CNT_16BIT_EN?\"16\":\" 32\" - 1

17.1.96 Tx_Multicast_Packets_Good_Bad

- **Description:** This register provides the number of good and bad multicast packets transmitted by DWC_ether_qos.
- **Size:** 32 bits
- **Offset:** 0x740
- **Exists:** (DWC_EQOS_MMC_TXMCASTGB_CNT_EN)

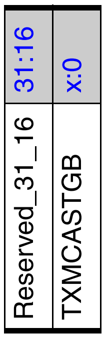


Table 17-100 Fields for Register: Tx_Multicast_Packets_Good_Bad

Bits	Name	Memory Access	Description
31:16	Reserved_31_16	R	Reserved. **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_MMC_TXMCASTGB_CNT_16BIT_EN)
x:0	TXMCASTGB	R	Tx Multicast Packets Good Bad This field indicates the number of good and bad multicast packets transmitted. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[x]:** DWC_EQOS_MMC_TXMCASTGB_CNT_16BIT_EN?\"16\":\"32\" - 1

17.1.97 Tx_Broadcast_Packets_Good_Bad

- **Description:** This register provides the number of good and bad broadcast packets transmitted by DWC_ether_qos.
- **Size:** 32 bits
- **Offset:** 0x744
- **Exists:** (DWC_EQOS_MMC_TXBCASTGB_CNT_EN)

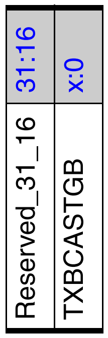


Table 17-101 Fields for Register: Tx_Broadcast_Packets_Good_Bad

Bits	Name	Memory Access	Description
31:16	Reserved_31_16	R	Reserved. **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_MMC_TXBCASTGB_CNT_16BIT_EN)
x:0	TXBCASTGB	R	Tx Broadcast Packets Good Bad This field indicates the number of good and bad broadcast packets transmitted. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[x]:** DWC_EQOS_MMC_TXBCASTGB_CNT_16BIT_EN?\"16\":\"32\" - 1

17.1.98 Tx_Underflow_Error_Packets

- **Description:** This register provides the number of packets aborted by DWC_ether_qos because of packets underflow error.
- **Size:** 32 bits
- **Offset:** 0x748
- **Exists:** (DWC_EQOS_MMC_TXUNDRFLW_CNT_EN)

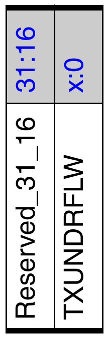


Table 17-102 Fields for Register: Tx_Underflow_Error_Packets

Bits	Name	Memory Access	Description
31:16	Reserved_31_16	R	Reserved. **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_MMC_TXUNDRFLW_CNT_16BIT_EN)
x:0	TXUNDRFLW	R	Tx Underflow Error Packets This field indicates the number of packets aborted because of packets underflow error. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[x]:** DWC_EQOS_MMC_TXUNDRFLW_CNT_16BIT_EN?\"16\":\" 32\" - 1

17.1.99 Tx_Single_Collision_Good_Packets

- **Description:** This register provides the number of successfully transmitted packets by DWC_ether_qos after a single collision in the half-duplex mode.
- **Size:** 32 bits
- **Offset:** 0x74c
- **Exists:** (DWC_EQOS_MMC_TXSNGLCOLG_CNT_EN)

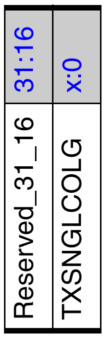


Table 17-103 Fields for Register: Tx_Single_Collision_Good_Packets

Bits	Name	Memory Access	Description
31:16	Reserved_31_16	R	Reserved. **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_MMC_TXSNGLCOLG_CNT_16BIT_EN)
x:0	TXSNGLCOLG	R	Tx Single Collision Good Packets This field indicates the number of successfully transmitted packets after a single collision in the half-duplex mode. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[x]:** DWC_EQOS_MMC_TXSNGLCOLG_CNT_16BIT_EN?\"16\":\ "32\" - 1

17.1.100 Tx_Multiple_Collision_Good_Packets

- **Description:** This register provides the number of successfully transmitted packets by DWC_ether_qos after multiple collisions in the half-duplex mode.
- **Size:** 32 bits
- **Offset:** 0x750
- **Exists:** (DWC_EQOS_MMC_TXMULTCOLG_CNT_EN)

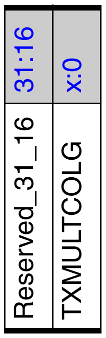


Table 17-104 Fields for Register: Tx_Multiple_Collision_Good_Packets

Bits	Name	Memory Access	Description
31:16	Reserved_31_16	R	Reserved. **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_MMC_TXMULTCOLG_CNT_16BIT_EN)
x:0	TXMULTCOLG	R	Tx Multiple Collision Good Packets This field indicates the number of successfully transmitted packets after multiple collisions in the half-duplex mode. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[x]:** DWC_EQOS_MMC_TXMULTCOLG_CNT_16BIT_EN?\"16\":\"32\" - 1

17.1.101 Tx_Deferred_Packets

- **Description:** This register provides the number of successfully transmitted by DWC_ether_qos after a deferral in the half-duplex mode.
- **Size:** 32 bits
- **Offset:** 0x754
- **Exists:** (DWC_EQOS_MMC_TXDEFRD_CNT_EN)

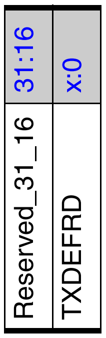


Table 17-105 Fields for Register: Tx_Deferred_Packets

Bits	Name	Memory Access	Description
31:16	Reserved_31_16	R	Reserved. **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_MMC_TXDEFRD_CNT_16BIT_EN)
x:0	TXDEFRD	R	Tx Deferred Packets This field indicates the number of successfully transmitted after a deferral in the half-duplex mode. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[x]:** DWC_EQOS_MMC_TXDEFRD_CNT_16BIT_EN?\"16\":\"32\" - 1

17.1.102 Tx_Late_Collision_Packets

- **Description:** This register provides the number of packets aborted by DWC_ether_qos because of late collision error.
- **Size:** 32 bits
- **Offset:** 0x758
- **Exists:** (DWC_EQOS_MMC_TXLATECOL_CNT_EN)

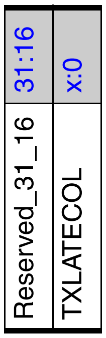


Table 17-106 Fields for Register: Tx_Late_Collision_Packets

Bits	Name	Memory Access	Description
31:16	Reserved_31_16	R	Reserved. **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_MMC_TXLATECOL_CNT_16BIT_EN)
x:0	TXLATECOL	R	Tx Late Collision Packets This field indicates the number of packets aborted because of late collision error. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[x]:** DWC_EQOS_MMC_TXLATECOL_CNT_16BIT_EN?\"16\":\"32\" - 1

17.1.103 Tx_Excessive_Collision_Packets

- **Description:** This register provides the number of packets aborted by DWC_ether_qos because of excessive (16) collision errors.
- **Size:** 32 bits
- **Offset:** 0x75c
- **Exists:** (DWC_EQOS_MMC_TXEXSCOL_CNT_EN)

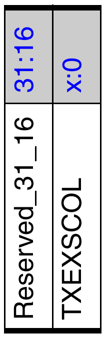


Table 17-107 Fields for Register: Tx_Excessive_Collision_Packets

Bits	Name	Memory Access	Description
31:16	Reserved_31_16	R	Reserved. **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_MMC_TXEXSCOL_CNT_16BIT_EN)
x:0	TXEXSCOL	R	Tx Excessive Collision Packets This field indicates the number of packets aborted because of excessive (16) collision errors. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[x]:** DWC_EQOS_MMC_TXEXSCOL_CNT_16BIT_EN?\"16\":\"32\" - 1

17.1.104 Tx_Carrier_Error_Packets

- **Description:** This register provides the number of packets aborted by DWC_ether_qos because of carrier sense error (no carrier or loss of carrier).
- **Size:** 32 bits
- **Offset:** 0x760
- **Exists:** (DWC_EQOS_MMC_TXCARR_CNT_EN)

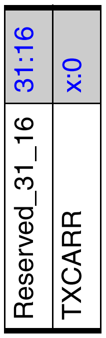


Table 17-108 Fields for Register: Tx_Carrier_Error_Packets

Bits	Name	Memory Access	Description
31:16	Reserved_31_16	R	Reserved. **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_MMC_TXCARR_CNT_16BIT_EN)
x:0	TXCARR	R	Tx Carrier Error Packets This field indicates the number of packets aborted because of carrier sense error (no carrier or loss of carrier). **Value After Reset:** 0x0 **Exists:** Always **Range Variable[x]:** DWC_EQOS_MMC_TXCARR_CNT_16BIT_EN?\"16\":\"32\" - 1

17.1.105 Tx_Octet_Count_Good

- **Description:** This register provides the number of bytes transmitted by DWC_ether_qos, exclusive of preamble, only in good packets.
- **Size:** 32 bits
- **Offset:** 0x764
- **Exists:** (DWC_EQOS_MMC_TXOCTG_CNT_EN)

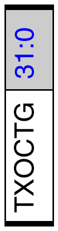


Table 17-109 Fields for Register: Tx_Octet_Count_Good

Bits	Name	Memory Access	Description
31:0	TXOCTG	R	Tx Octet Count Good This field indicates the number of bytes transmitted, exclusive of preamble, only in good packets. **Value After Reset:** 0x0 **Exists:** Always

17.1.106 Tx_Packet_Count_Good

- **Description:** This register provides the number of good packets transmitted by DWC_ether_qos.
- **Size:** 32 bits
- **Offset:** 0x768
- **Exists:** (DWC_EQOS_MMC_TXPKTG_CNT_EN)

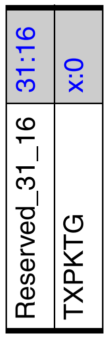


Table 17-110 Fields for Register: Tx_Packet_Count_Good

Bits	Name	Memory Access	Description
31:16	Reserved_31_16	R	Reserved. **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_MMC_TXPKTG_CNT_16BIT_EN)
x:0	TXPKTG	R	Tx Packet Count Good This field indicates the number of good packets transmitted. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[x]:** DWC_EQOS_MMC_TXPKTG_CNT_16BIT_EN?\"16\":\"32\" - 1

17.1.107 Tx_Excessive_Deferral_Error

- **Description:** This register provides the number of packets aborted by DWC_ether_qos because of excessive deferral error (deferred for more than two max-sized packet times).
- **Size:** 32 bits
- **Offset:** 0x76c
- **Exists:** (DWC_EQOS_MMC_TXEXSDEF_CNT_EN)

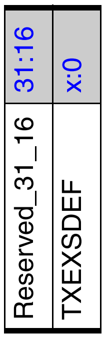


Table 17-111 Fields for Register: Tx_Excessive_Deferral_Error

Bits	Name	Memory Access	Description
31:16	Reserved_31_16	R	Reserved. **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_MMC_TXEXSDEF_CNT_16BIT_EN)
x:0	TXEXSDEF	R	Tx Excessive Deferral Error This field indicates the number of packets aborted because of excessive deferral error (deferred for more than two max-sized packet times). **Value After Reset:** 0x0 **Exists:** Always **Range Variable[x]:** DWC_EQOS_MMC_TXEXSDEF_CNT_16BIT_EN?\"16\":\"32 \" - 1

17.1.108 Tx_Pause_Packets

- **Description:** This register provides the number of good Pause packets transmitted by DWC_ether_qos.
- **Size:** 32 bits
- **Offset:** 0x770
- **Exists:** (DWC_EQOS_MMC_TXPAUSE_CNT_EN)

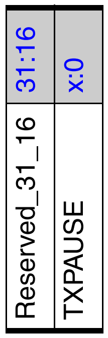


Table 17-112 Fields for Register: Tx_Pause_Packets

Bits	Name	Memory Access	Description
31:16	Reserved_31_16	R	Reserved. **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_MMC_TXPAUSE_CNT_16BIT_EN)
x:0	TXPAUSE	R	Tx Pause Packets This field indicates the number of good Pause packets transmitted. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[x]:** DWC_EQOS_MMC_TXPAUSE_CNT_16BIT_EN?\"16\":\"32\" - 1

17.1.109 Tx_VLAN_Packets_Good

- **Description:** This register provides the number of good VLAN packets transmitted by DWC_ether_qos.
- **Size:** 32 bits
- **Offset:** 0x774
- **Exists:** (DWC_EQOS_MMC_TXVLAN_CNT_EN)

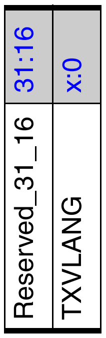


Table 17-113 Fields for Register: Tx_VLAN_Packets_Good

Bits	Name	Memory Access	Description
31:16	Reserved_31_16	R	Reserved. **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_MMC_TXVLAN_CNT_16BIT_EN)
x:0	TXVLANG	R	Tx VLAN Packets Good This field provides the number of good VLAN packets transmitted. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[x]:** DWC_EQOS_MMC_TXVLAN_CNT_16BIT_EN?\"16\":\"32\" - 1

17.1.110 Tx_OSize_Packets_Good

- **Description:** This register provides the number of packets transmitted by DWC_ether_qos without errors and with length greater than the maxsize (1,518 or 1,522 bytes for VLAN tagged packets; 2000 bytes if enabled in S2KP bit of the MAC_Configuration register).
- **Size:** 32 bits
- **Offset:** 0x778
- **Exists:** (DWC_EQOS_MMC_TXOVERSZG_CNT_EN)

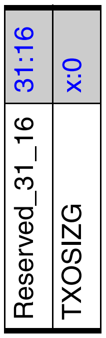


Table 17-114 Fields for Register: Tx_OSize_Packets_Good

Bits	Name	Memory Access	Description
31:16	Reserved_31_16	R	Reserved. **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_MMC_TXOVERSZG_CNT_16BIT_EN)
x:0	TXOSIZG	R	Tx OSize Packets Good This field indicates the number of packets transmitted without errors and with length greater than the maxsize (1,518 or 1,522 bytes for VLAN tagged packets; 2000 bytes if enabled in S2KP bit of the MAC_Configuration register). **Value After Reset:** 0x0 **Exists:** Always **Range Variable[x]:** DWC_EQOS_MMC_TXOVERSZG_CNT_16BIT_EN?\"16\":\" 32\" - 1

17.1.111 Rx_Packets_Count_Good_Bad

- **Description:** This register provides the number of good and bad packets received by DWC_ether_qos.
- **Size:** 32 bits
- **Offset:** 0x780
- **Exists:** (DWC_EQOS_MMC_RXPKTGB_CNT_EN)

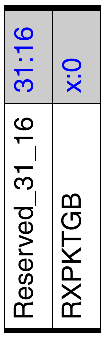


Table 17-115 Fields for Register: Rx_Packets_Count_Good_Bad

Bits	Name	Memory Access	Description
31:16	Reserved_31_16	R	Reserved. **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_MMC_RXPKTGB_CNT_16BIT_EN)
x:0	RXPKTGB	R	Rx Packets Count Good Bad This field indicates the number of good and bad packets received. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[x]:** DWC_EQOS_MMC_RXPKTGB_CNT_16BIT_EN?\"16\":\"32\" - 1

17.1.112 Rx_Octet_Count_Good_Bad

- **Description:** This register provides the number of bytes received by DWC_ther_qos, exclusive of preamble, in good and bad packets.
- **Size:** 32 bits
- **Offset:** 0x784
- **Exists:** (DWC_EQOS_MMC_RXOCTGB_CNT_EN)

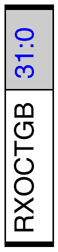


Table 17-116 Fields for Register: Rx_Octet_Count_Good_Bad

Bits	Name	Memory Access	Description
31:0	RXOCTGB	R	Rx Octet Count Good Bad This field indicates the number of bytes received, exclusive of preamble, in good and bad packets. **Value After Reset:** 0x0 **Exists:** Always

17.1.113 Rx_Octet_Count_Good

- **Description:** This register provides the number of bytes received by DWC_ether_qos, exclusive of preamble, only in good packets.
- **Size:** 32 bits
- **Offset:** 0x788
- **Exists:** (DWC_EQOS_MMC_RXOCTG_CNT_EN)

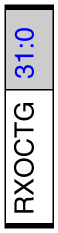


Table 17-117 Fields for Register: Rx_Octet_Count_Good

Bits	Name	Memory Access	Description
31:0	RXOCTG	R	Rx Octet Count Good This field indicates the number of bytes received, exclusive of preamble, only in good packets. **Value After Reset:** 0x0 **Exists:** Always

17.1.114 Rx_Broadcast_Packets_Good

- **Description:** This register provides the number of good broadcast packets received by DWC_ether_qos.
- **Size:** 32 bits
- **Offset:** 0x78c
- **Exists:** (DWC_EQOS_MMC_RXBCASTG_CNT_EN)

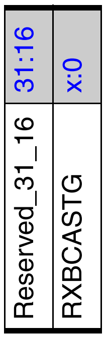


Table 17-118 Fields for Register: Rx_Broadcast_Packets_Good

Bits	Name	Memory Access	Description
31:16	Reserved_31_16	R	Reserved. **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_MMC_RXBCASTG_CNT_16BIT_EN)
x:0	RXBCASTG	R	Rx Broadcast Packets Good This field indicates the number of good broadcast packets received. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[x]:** DWC_EQOS_MMC_RXBCASTG_CNT_16BIT_EN?\"16\":\"32\" - 1

17.1.115 Rx_Multicast_Packets_Good

- **Description:** This register provides the number of good multicast packets received by DWC_ether_qos.
- **Size:** 32 bits
- **Offset:** 0x790
- **Exists:** (DWC_EQOS_MMC_RXMCASTG_CNT_EN)

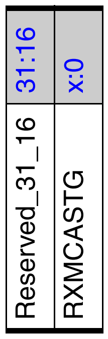


Table 17-119 Fields for Register: Rx_Multicast_Packets_Good

Bits	Name	Memory Access	Description
31:16	Reserved_31_16	R	Reserved. **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_MMC_RXMCASTG_CNT_16BIT_EN)
x:0	RXMCASTG	R	Rx Multicast Packets Good This field indicates the number of good multicast packets received. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[x]:** DWC_EQOS_MMC_RXMCASTG_CNT_16BIT_EN?\"16\":\"32\" - 1

17.1.116 Rx_CRC_Error_Packets

- **Description:** This register provides the number of packets received by DWC_ether_qos with CRC error.
- **Size:** 32 bits
- **Offset:** 0x794
- **Exists:** (DWC_EQOS_MMC_RXCRCERR_CNT_EN)

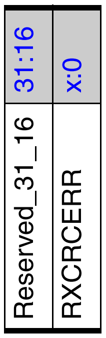


Table 17-120 Fields for Register: Rx_CRC_Error_Packets

Bits	Name	Memory Access	Description
31:16	Reserved_31_16	R	Reserved. **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_MMC_RXCRCERR_CNT_16BIT_EN)
x:0	RXCRCERR	R	Rx CRC Error Packets This field indicates the number of packets received with CRC error. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[x]:** DWC_EQOS_MMC_RXCRCERR_CNT_16BIT_EN?\"16\":\"32\" - 1

17.1.117 Rx_Alignment_Error_Packets

- **Description:** This register provides the number of packets received by DWC_ether_qos with alignment (dribble) error. It is valid only in 10/100 mode.
- **Size:** 32 bits
- **Offset:** 0x798
- **Exists:** (DWC_EQOS_MMC_RXALGNERR_CNT_EN)

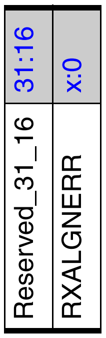


Table 17-121 Fields for Register: Rx_Alignment_Error_Packets

Bits	Name	Memory Access	Description
31:16	Reserved_31_16	R	Reserved. **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_MMC_RXALGNERR_CNT_16BIT_EN)
x:0	RXALGNERR	R	Rx Alignment Error Packets This field indicates the number of packets received with alignment (dribble) error. It is valid only in 10/100 mode. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[x]:** DWC_EQOS_MMC_RXALGNERR_CNT_16BIT_EN?\"16\":\" 32\" - 1

17.1.118 Rx_Runt_Error_Packets

- **Description:** This register provides the number of packets received by DWC_ether_qos with runt (length less than 64 bytes and CRC error) error.
- **Size:** 32 bits
- **Offset:** 0x79c
- **Exists:** (DWC_EQOS_MMC_RXRUNTERR_CNT_EN)

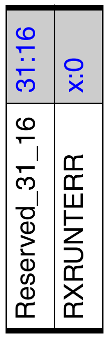


Table 17-122 Fields for Register: Rx_Runt_Error_Packets

Bits	Name	Memory Access	Description
31:16	Reserved_31_16	R	Reserved. **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_MMC_RXRUNTERR_CNT_16BIT_EN)
x:0	RXRUNTERR	R	Rx Runt Error Packets This field indicates the number of packets received with runt (length less than 64 bytes and CRC error) error. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[x]:** DWC_EQOS_MMC_RXRUNTERR_CNT_16BIT_EN?\"16\":\" 32\" - 1

17.1.119 Rx_Jabber_Error_Packets

- **Description:** This register provides the number of giant packets received by DWC_ether_qos with length (including CRC) greater than 1,518 bytes (1,522 bytes for VLAN tagged) and with CRC error. If Jumbo Packet mode is enabled, packets of length greater than 9,018 bytes (9,022 bytes for VLAN tagged) are considered as giant packets.
- **Size:** 32 bits
- **Offset:** 0x7a0
- **Exists:** (DWC_EQOS_MMC_RXJABERR_CNT_EN)

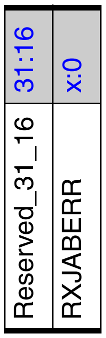


Table 17-123 Fields for Register: Rx_Jabber_Error_Packets

Bits	Name	Memory Access	Description
31:16	Reserved_31_16	R	Reserved. **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_MMC_RXJABERR_CNT_16BIT_EN)
x:0	RXJABERR	R	Rx Jabber Error Packets This field indicates the number of giant packets received with length (including CRC) greater than 1,518 bytes (1,522 bytes for VLAN tagged) and with CRC error. If Jumbo Packet mode is enabled, packets of length greater than 9,018 bytes (9,022 bytes for VLAN tagged) are considered as giant packets. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[x]:** DWC_EQOS_MMC_RXJABERR_CNT_16BIT_EN?\"16\":\"32 \" - 1

17.1.120 Rx_Undersize_Packets_Good

- **Description:** This register provides the number of packets received by DWC_ether_qos with length less than 64 bytes, without any errors.
- **Size:** 32 bits
- **Offset:** 0x7a4
- **Exists:** (DWC_EQOS_MMC_RXUNDERSZG_CNT_EN)

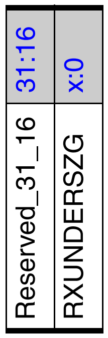


Table 17-124 Fields for Register: Rx_Undersize_Packets_Good

Bits	Name	Memory Access	Description
31:16	Reserved_31_16	R	Reserved. **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_MMC_RXUNDERSZG_CNT_16BIT_EN)
x:0	RXUNDERSZG	R	Rx Undersize Packets Good This field indicates the number of packets received with length less than 64 bytes, without any errors. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[x]:** DWC_EQOS_MMC_RXUNDERSZG_CNT_16BIT_EN?\"16\": \"32\" - 1

17.1.121 Rx_Oversize_Packets_Good

- **Description:** This register provides the number of packets received by DWC_ether_qos without errors, with length greater than the maxsize (1,518 bytes or 1,522 bytes for VLAN tagged packets; 2000 bytes if enabled in the S2KP bit of the MAC_Configuration register).
- **Size:** 32 bits
- **Offset:** 0x7a8
- **Exists:** (DWC_EQOS_MMC_RXOVERSZG_CNT_EN)

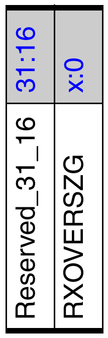


Table 17-125 Fields for Register: Rx_Oversize_Packets_Good

Bits	Name	Memory Access	Description
31:16	Reserved_31_16	R	Reserved. **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_MMC_RXOVERSZG_CNT_16BIT_EN)
x:0	RXOVERSZG	R	Rx Oversize Packets Good This field indicates the number of packets received without errors, with length greater than the maxsize (1,518 bytes or 1,522 bytes for VLAN tagged packets; 2000 bytes if enabled in the S2KP bit of the MAC_Configuration register). **Value After Reset:** 0x0 **Exists:** Always **Range Variable[x]:** DWC_EQOS_MMC_RXOVERSZG_CNT_16BIT_EN?\"16\":\" 32\" - 1

17.1.122 Rx_64Octets_Packets_Good_Bad

- **Description:** This register provides the number of good and bad packets received by DWC_ether_qos with length 64 bytes, exclusive of the preamble.
- **Size:** 32 bits
- **Offset:** 0x7ac
- **Exists:** (DWC_EQOS_MMC_RXADDR_RNG_CNT_EN)

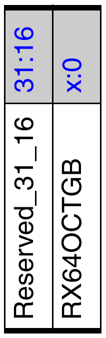


Table 17-126 Fields for Register: Rx_64Octets_Packets_Good_Bad

Bits	Name	Memory Access	Description
31:16	Reserved_31_16	R	Reserved. **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_MMC_RXADDR_RNG_CNT_16BIT_EN)
x:0	RX64OCTGB	R	Rx 64 Octets Packets Good Bad This field indicates the number of good and bad packets received with length 64 bytes, exclusive of the preamble. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[x]:** DWC_EQOS_MMC_RXADDR_RNG_CNT_16BIT_EN?\"16\": \"32\" - 1

17.1.123 Rx_65To127Octets_Packets_Good_Bad

- **Description:** This register provides the number of good and bad packets received by DWC_ether_qos with length between 65 and 127 (inclusive) bytes, exclusive of the preamble.
- **Size:** 32 bits
- **Offset:** 0x7b0
- **Exists:** (DWC_EQOS_MMC_RXADDR_RNG_CNT_EN)

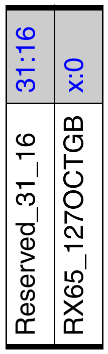


Table 17-127 Fields for Register: Rx_65To127Octets_Packets_Good_Bad

Bits	Name	Memory Access	Description
31:16	Reserved_31_16	R	Reserved. **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_MMC_RXADDR_RNG_CNT_16BIT_EN)
x:0	RX65_127OCTGB	R	Rx 65-127 Octets Packets Good Bad This field indicates the number of good and bad packets received with length between 65 and 127 (inclusive) bytes, exclusive of the preamble. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[x]:** DWC_EQOS_MMC_RXADDR_RNG_CNT_16BIT_EN?\"16\": \"32\" - 1

17.1.124 Rx_128To255Octets_Packets_Good_Bad

- **Description:** This register provides the number of good and bad packets received by DWC_ether_qos with length between 128 and 255 (inclusive) bytes, exclusive of the preamble.
- **Size:** 32 bits
- **Offset:** 0x7b4
- **Exists:** (DWC_EQOS_MMC_RXADDR_RNG_CNT_EN)

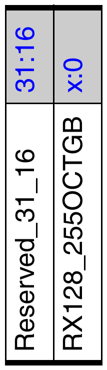


Table 17-128 Fields for Register: Rx_128To255Octets_Packets_Good_Bad

Bits	Name	Memory Access	Description
31:16	Reserved_31_16	R	Reserved. **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_MMC_RXADDR_RNG_CNT_16BIT_EN)
x:0	RX128_255OCTGB	R	Rx 128-255 Octets Packets Good Bad This field indicates the number of good and bad packets received with length between 128 and 255 (inclusive) bytes, exclusive of the preamble. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[x]:** DWC_EQOS_MMC_RXADDR_RNG_CNT_16BIT_EN?\"16\": \"32\" - 1

17.1.125 Rx_256To511Octets_Packets_Good_Bad

- **Description:** This register provides the number of good and bad packets received by DWC_ether_qos with length between 256 and 511 (inclusive) bytes, exclusive of the preamble.
- **Size:** 32 bits
- **Offset:** 0x7b8
- **Exists:** (DWC_EQOS_MMC_RXADDR_RNG_CNT_EN)

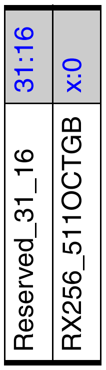


Table 17-129 Fields for Register: Rx_256To511Octets_Packets_Good_Bad

Bits	Name	Memory Access	Description
31:16	Reserved_31_16	R	Reserved. **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_MMC_RXADDR_RNG_CNT_16BIT_EN)
x:0	RX256_511OCTGB	R	Rx 256-511 Octets Packets Good Bad This field indicates the number of good and bad packets received with length between 256 and 511 (inclusive) bytes, exclusive of the preamble. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[x]:** DWC_EQOS_MMC_RXADDR_RNG_CNT_16BIT_EN?\"16\": \"32\" - 1

17.1.126 Rx_512To1023Octets_Packets_Good_Bad

- **Description:** This register provides the number of good and bad packets received by DWC_ether_qos with length between 512 and 1023 (inclusive) bytes, exclusive of the preamble.
- **Size:** 32 bits
- **Offset:** 0x7bc
- **Exists:** (DWC_EQOS_MMC_RXADDR_RNG_CNT_EN)

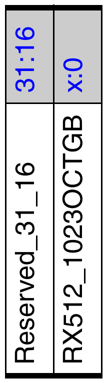


Table 17-130 Fields for Register: Rx_512To1023Octets_Packets_Good_Bad

Bits	Name	Memory Access	Description
31:16	Reserved_31_16	R	Reserved. **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_MMC_RXADDR_RNG_CNT_16BIT_EN)
x:0	RX512_1023OCTGB	R	RX 512-1023 Octets Packets Good Bad This field indicates the number of good and bad packets received with length between 512 and 1023 (inclusive) bytes, exclusive of the preamble. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[x]:** DWC_EQOS_MMC_RXADDR_RNG_CNT_16BIT_EN?\"16\": \"32\" - 1

17.1.127 Rx_1024ToMaxOctets_Packets_Good_Bad

- **Description:** This register provides the number of good and bad packets received by DWC_ether_qos with length between 1024 and maxsize (inclusive) bytes, exclusive of the preamble.
- **Size:** 32 bits
- **Offset:** 0x7c0
- **Exists:** (DWC_EQOS_MMC_RXADDR_RNG_CNT_EN)

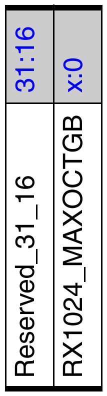


Table 17-131 Fields for Register: Rx_1024ToMaxOctets_Packets_Good_Bad

Bits	Name	Memory Access	Description
31:16	Reserved_31_16	R	Reserved. **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_MMC_RXADDR_RNG_CNT_16BIT_EN)
x:0	RX1024_MAXOCTGB	R	Rx 1024-Max Octets Good Bad This field indicates the number of good and bad packets received with length between 1024 and maxsize (inclusive) bytes, exclusive of the preamble. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[x]:** DWC_EQOS_MMC_RXADDR_RNG_CNT_16BIT_EN?\"16\": \"32\" - 1

17.1.128 Rx_Unicast_Packets_Good

- **Description:** This register provides the number of good unicast packets received by DWC_ether_qos.
- **Size:** 32 bits
- **Offset:** 0x7c4
- **Exists:** (DWC_EQOS_MMC_RXUCASTG_CNT_EN)

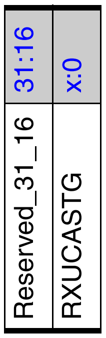


Table 17-132 Fields for Register: Rx_Unicast_Packets_Good

Bits	Name	Memory Access	Description
31:16	Reserved_31_16	R	Reserved. **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_MMC_RXUCASTG_CNT_16BIT_EN)
x:0	RXUCASTG	R	Rx Unicast Packets Good This field indicates the number of good unicast packets received. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[x]:** DWC_EQOS_MMC_RXUCASTG_CNT_16BIT_EN?\"16\":\"32\" - 1

17.1.129 Rx_Length_Error_Packets

- **Description:** This register provides the number of packets received by DWC_ether_qos with length error (Length Type field not equal to packet size), for all packets with valid length field.
- **Size:** 32 bits
- **Offset:** 0x7c8
- **Exists:** (DWC_EQOS_MMC_RXLENERR_CNT_EN)

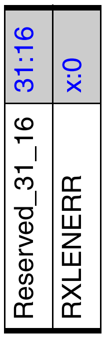


Table 17-133 Fields for Register: Rx_Length_Error_Packets

Bits	Name	Memory Access	Description
31:16	Reserved_31_16	R	Reserved. **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_MMC_RXLENERR_CNT_16BIT_EN)
x:0	RXLENERR	R	Rx Length Error Packets This field indicates the number of packets received with length error (Length Type field not equal to packet size), for all packets with valid length field. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[x]:** DWC_EQOS_MMC_RXLENERR_CNT_16BIT_EN?\"16\":\"32 \" - 1

17.1.130 Rx_Out_Of_Range_Type_Packets

- **Description:** This register provides the number of packets received by DWC_ether_qos with length field not equal to the valid packet size (greater than 1,500 but less than 1,536).
- **Size:** 32 bits
- **Offset:** 0x7cc
- **Exists:** (DWC_EQOS_MMC_RXOUTOFRNG_TYP_CNT_EN)

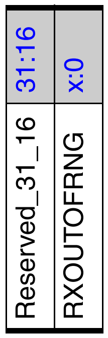


Table 17-134 Fields for Register: Rx_Out_Of_Range_Type_Packets

Bits	Name	Memory Access	Description
31:16	Reserved_31_16	R	Reserved. **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_MMC_RXOUTOFRNG_TYP_CNT_16BIT_EN)
x:0	RXOUTOFRNG	R	Rx Out of Range Type Packet This field indicates the number of packets received with length field not equal to the valid packet size (greater than 1,500 but less than 1,536). **Value After Reset:** 0x0 **Exists:** Always **Range Variable[x]:** DWC_EQOS_MMC_RXOUTOFRNG_TYP_CNT_16BIT_EN?\ "16\":\"32\" - 1

17.1.131 Rx_Pause_Packets

- **Description:** This register provides the number of good and valid Pause packets received by DWC_ether_qos.
- **Size:** 32 bits
- **Offset:** 0x7d0
- **Exists:** (DWC_EQOS_MMC_RXPAUSEPKT_CNT_EN)

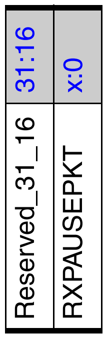


Table 17-135 Fields for Register: Rx_Pause_Packets

Bits	Name	Memory Access	Description
31:16	Reserved_31_16	R	Reserved. **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_MMC_RXPAUSEPKT_CNT_16BIT_EN)
x:0	RXPAUSEPKT	R	Rx Pause Packets This field indicates the number of good and valid Pause packets received. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[x]:** DWC_EQOS_MMC_RXPAUSEPKT_CNT_16BIT_EN?\"16\":\" 32\" - 1

17.1.132 Rx_FIFO_Overflow_Packets

- **Description:** This register provides the number of missed received packets because of FIFO overflow in DWC_ether_qos.
- **Size:** 32 bits
- **Offset:** 0x7d4
- **Exists:** (DWC_EQOS_MMC_RXFIFOOVFL_CNT_EN)

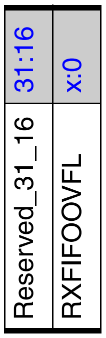


Table 17-136 Fields for Register: Rx_FIFO_Overflow_Packets

Bits	Name	Memory Access	Description
31:16	Reserved_31_16	R	Reserved. **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_MMC_RXFIFOOVFL_CNT_16BIT_EN)
x:0	RXFIFOOVFL	R	Rx FIFO Overflow Packets This field indicates the number of missed received packets because of FIFO overflow. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[x]:** DWC_EQOS_MMC_RXFIFOOVFL_CNT_16BIT_EN?\"16\":\" 32\" - 1

17.1.133 Rx_VLAN_Packets_Good_Bad

- **Description:** This register provides the number of good and bad VLAN packets received by DWC_ether_qos.
- **Size:** 32 bits
- **Offset:** 0x7d8
- **Exists:** (DWC_EQOS_MMC_RXVLANPKTGB_CNT_EN)

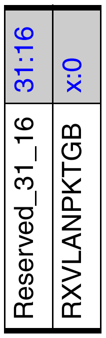


Table 17-137 Fields for Register: Rx_VLAN_Packets_Good_Bad

Bits	Name	Memory Access	Description
31:16	Reserved_31_16	R	Reserved. **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_MMC_RXVLANPKTGB_CNT_16BIT_EN)
x:0	RXVLANPKTGB	R	Rx VLAN Packets Good Bad This field indicates the number of good and bad VLAN packets received. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[x]:** DWC_EQOS_MMC_RXVLANPKTGB_CNT_16BIT_EN?\"16\" :\"32\" - 1

17.1.134 Rx_Watchdog_Error_Packets

- **Description:** This register provides the number of packets received by DWC_ether_qos with error because of watchdog timeout error (packets with a data load larger than 2,048 bytes (when JE and WD bits are reset in MAC_Configuration register), 10,240 bytes (when JE bit is set and WD bit is reset in MAC_Configuration register), 16,384 bytes (when WD bit is set in MAC_Configuration register) or the value programmed in the MAC_Watchdog_Timeout register).
- **Size:** 32 bits
- **Offset:** 0x7dc
- **Exists:** (DWC_EQOS_MMC_RXWDGERR_CNT_EN)

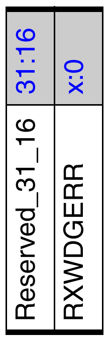


Table 17-138 Fields for Register: Rx_Watchdog_Error_Packets

Bits	Name	Memory Access	Description
31:16	Reserved_31_16	R	Reserved. **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_MMC_RXWDGERR_CNT_16BIT_EN)
x:0	RXWDGERR	R	Rx Watchdog Error Packets This field indicates the number of packets received with error because of watchdog timeout error (packets with a data load larger than 2,048 bytes (when JE and WD bits are reset in MAC_Configuration register), 10,240 bytes (when JE bit is set and WD bit is reset in MAC_Configuration register), 16,384 bytes (when WD bit is set in MAC_Configuration register) or the value programmed in the MAC_Watchdog_Timeout register). **Value After Reset:** 0x0 **Exists:** Always **Range Variable[x]:** DWC_EQOS_MMC_RXWDGERR_CNT_16BIT_EN?\"16\":\"32\" - 1

17.1.135 Rx_Receive_Error_Packets

- **Description:** This register provides the number of packets received by DWC_ether_qos with Receive error or Packet Extension error on the GMII or MII interface.
- **Size:** 32 bits
- **Offset:** 0x7e0
- **Exists:** (DWC_EQOS_MMC_RXRCVERR_CNT_EN)

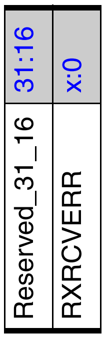


Table 17-139 Fields for Register: Rx_Receive_Error_Packets

Bits	Name	Memory Access	Description
31:16	Reserved_31_16	R	Reserved. **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_MMC_RXRCVERR_CNT_16BIT_EN)
x:0	RXRCVERR	R	Rx Receive Error Packets This field indicates the number of packets received with Receive error or Packet Extension error on the GMII or MII interface. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[x]:** DWC_EQOS_MMC_RXRCVERR_CNT_16BIT_EN?\"16\":\"32\" - 1

17.1.136 Rx_Control_Packets_Good

- **Description:** This register provides the number of good control packets received by DWC_ether_qos.
- **Size:** 32 bits
- **Offset:** 0x7e4
- **Exists:** (DWC_EQOS_MMC_RXCTRLPKT_CNT_EN)

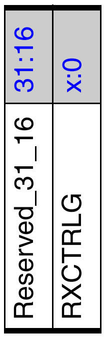


Table 17-140 Fields for Register: Rx_Control_Packets_Good

Bits	Name	Memory Access	Description
31:16	Reserved_31_16	R	Reserved. **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_MMC_RXCTRLPKT_CNT_16BIT_EN)
x:0	RXCTRLG	R	Rx Control Packets Good This field indicates the number of good control packets received. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[x]:** DWC_EQOS_MMC_RXCTRLPKT_CNT_16BIT_EN?\"16\":\"32\" - 1

17.1.137 Tx_LPI_USEC_Cntr

- **Description:** This register provides the number of microseconds Tx LPI is asserted by DWC_ether_qos.
- **Size:** 32 bits
- **Offset:** 0x7ec
- **Exists:** (DWC_EQOS_MMC_TX_LPI_MICROSEC_TIMER_EN)

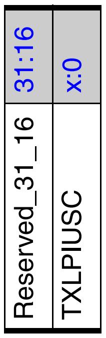


Table 17-141 Fields for Register: Tx_LPI_USEC_Cntr

Bits	Name	Memory Access	Description
31:16	Reserved_31_16	R	Reserved. **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_MMC_TX_LPI_MICROSEC_TIMER_16BIT_EN)
x:0	TXLPIUSC	R	Tx LPI Microseconds Counter This field indicates the number of microseconds Tx LPI is asserted. For every Tx LPI Entry and Exit, the Timer value can have an error of +/- 1 microsecond. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[x]:** DWC_EQOS_MMC_TX_LPI_MICROSEC_TIMER_16BIT_EN ?\"16\":\"32\" - 1

17.1.138 Tx_LPI_Tran_Cntr

- **Description:** This register provides the number of times DWC_ether_qos has entered Tx LPI.
- **Size:** 32 bits
- **Offset:** 0x7f0
- **Exists:** (DWC_EQOS_MMC_TX_LPI_TRANSITION_CNT_EN)

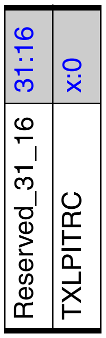


Table 17-142 Fields for Register: Tx_LPI_Tran_Cntr

Bits	Name	Memory Access	Description
31:16	Reserved_31_16	R	Reserved. **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_MMC_TX_LPI_MICROSEC_TIMER_16BIT_EN)
x:0	TXLPITRC	R	Tx LPI Transition counter This field indicates the number of times Tx LPI Entry has occurred. Even if Tx LPI Entry occurs in Automate Mode (because of LPITXA bit set in the LPI Control and Status register), the counter will increment. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[x]:** DWC_EQOS_MMC_TX_LPI_TRANSITION_CNT_16BIT_EN ?\"16\":\"32\" - 1

17.1.139 Rx_LPI_USEC_Cntr

- **Description:** This register provides the number of microseconds Rx LPI is sampled by DWC_ether_qos.
- **Size:** 32 bits
- **Offset:** 0x7f4
- **Exists:** (DWC_EQOS_MMC_RX_LPI_MICROSEC_TIMER_EN)

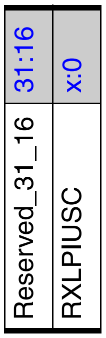


Table 17-143 Fields for Register: Rx_LPI_USEC_Cntr

Bits	Name	Memory Access	Description
31:16	Reserved_31_16	R	Reserved. **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_MMC_RX_LPI_MICROSEC_TIMER_16BIT_EN)
x:0	RXLPIUSC	R	Rx LPI Microseconds Counter This field indicates the number of microseconds Rx LPI is asserted. For every Rx LPI Entry and Exit, the Timer value can have an error of +/- 1 microsecond. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[x]:** DWC_EQOS_MMC_RX_LPI_MICROSEC_TIMER_16BIT_EN ?\"16\":\"32\" - 1

17.1.140 Rx_LPI_Tran_Cntr

- **Description:** This register provides the number of times DWC_ether_qos has entered Rx LPI.
- **Size:** 32 bits
- **Offset:** 0x7f8
- **Exists:** (DWC_EQOS_MMC_RX_LPI_TRANSITION_CNT_EN)

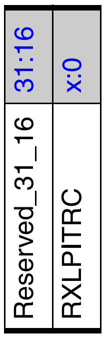


Table 17-144 Fields for Register: Rx_LPI_Tran_Cntr

Bits	Name	Memory Access	Description
31:16	Reserved_31_16	R	Reserved. **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_MMC_RX_LPI_MICROSEC_TIMER_16BIT_EN)
x:0	RXLPITRC	R	Rx LPI Transition counter This field indicates the number of times Rx LPI Entry has occurred. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[x]:** DWC_EQOS_MMC_RX_LPI_TRANSITION_CNT_16BIT_EN ?\"16\":\"32\" - 1

17.1.141 MMC_IPC_Rx_Interrupt_Mask

- **Description:** This register maintains the mask for the interrupt generated from the receive IPC statistic counters.
  The MMC Receive Checksum Off load Interrupt Mask register maintains the masks for the interrupts generated when the receive IPC (Checksum Off load) statistic counters reach half their maximum value, and when they reach their maximum values. This register is 32 bits wide.
- **Size:** 32 bits
- **Offset:** 0x800
- **Exists:** ((DWC_EQOS_MMC_IPC_RX_PKTS_ATLEAST_ONE||DWC_EQOS_MMC_IPC_RX_DGRAM_BCNT_ATLEAST_ONE||DWC_EQOS_MMC_IPC_RX_PAYLD_BCNT_ATLEAST_ONE))

31:30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15:14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved_31_30	RXICMPEROIM	RXICMPGOIM	RXTCPEROIM	RXTCPGOIM	RXUDPEROIM	RXUDPGOIM	RXIPV6NOPAYOIM	RXIPV6HEROIM	RXIPV6GOIM	RXIPV4UDSBLOIM	RXIPV4FRAGOIM	RXIPV4NOPAYOIM	RXIPV4HEROIM	RXIPV4GOIM	Reserved_15_14	RXICMPERPIM	RXICMPGPIM	RXTCPERPIM	RXTCPGPIM	RXUDPERPIM	RXUDPGPIM	RXIPV6NOPAYPIM	RXIPV6HERPIM	RXIPV6GPIM	RXIPV4UDSBLPIM	RXIPV4FRAGPIM	RXIPV4NOPAYPIM	RXIPV4HERPIM	RXIPV4GPIM

Table 17-145 Fields for Register: MMC_IPC_Rx_Interrupt_Mask

Bits	Name	Memory Access	Description
31:30	Reserved_31_30	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
29	RXICMPEROIM	R/W	MMC Receive ICMP Error Octet Counter Interrupt Mask Setting this bit masks the interrupt when the rxicmp_err_octets counter reaches half of the maximum value or the maximum value. **Values:** ■ 0x0 (DISABLE): MMC Receive ICMP Error Octet Counter Interrupt Mask is disabled ■ 0x1 (ENABLE): MMC Receive ICMP Error Octet Counter Interrupt Mask is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RX_ICMP_ERR_OCTET_CNT_EN

Bits	Name	Memory Access	Description
28	RXICMPGOIM	R/W	MMC Receive ICMP Good Octet Counter Interrupt Mask Setting this bit masks the interrupt when the rxicmp_gd_octets counter reaches half of the maximum value or the maximum value. **Values:** ■ 0x0 (DISABLE): MMC Receive ICMP Good Octet Counter Interrupt Mask is disabled ■ 0x1 (ENABLE): MMC Receive ICMP Good Octet Counter Interrupt Mask is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RX_ICMP_GD_OCTET_CNT_EN
27	RXTCPEROIM	R/W	MMC Receive TCP Error Octet Counter Interrupt Mask Setting this bit masks the interrupt when the rxtcp_err_octets counter reaches half of the maximum value or the maximum value. **Values:** ■ 0x0 (DISABLE): MMC Receive TCP Error Octet Counter Interrupt Mask is disabled ■ 0x1 (ENABLE): MMC Receive TCP Error Octet Counter Interrupt Mask is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RX_TCP_ERR_OCTET_CNT_EN
26	RXTCPGOIM	R/W	MMC Receive TCP Good Octet Counter Interrupt Mask Setting this bit masks the interrupt when the rxtcp_gd_octets counter reaches half of the maximum value or the maximum value. **Values:** ■ 0x0 (DISABLE): MMC Receive TCP Good Octet Counter Interrupt Mask is disabled ■ 0x1 (ENABLE): MMC Receive TCP Good Octet Counter Interrupt Mask is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RX_TCP_GD_OCTET_CNT_EN

Bits	Name	Memory Access	Description
25	RXUDPEROIM	R/W	MMC Receive UDP Good Octet Counter Interrupt Mask Setting this bit masks the interrupt when the rxudp_err_octets counter reaches half of the maximum value or the maximum value. **Values:** ■ 0x0 (DISABLE): MMC Receive UDP Good Octet Counter Interrupt Mask is disabled ■ 0x1 (ENABLE): MMC Receive UDP Good Octet Counter Interrupt Mask is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RX_UDP_ERR_OCTET_CNT_EN
24	RXUDPGOIM	R/W	MMC Receive IPV6 No Payload Octet Counter Interrupt Mask Setting this bit masks the interrupt when the rxudp_gd_octets counter reaches half of the maximum value or the maximum value. **Values:** ■ 0x0 (DISABLE): MMC Receive IPV6 No Payload Octet Counter Interrupt Mask is disabled ■ 0x1 (ENABLE): MMC Receive IPV6 No Payload Octet Counter Interrupt Mask is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RX_UDP_GD_OCTET_CNT_EN
23	RXIPV6NOPAYOIM	R/W	MMC Receive IPV6 Header Error Octet Counter Interrupt Mask Setting this bit masks the interrupt when the rxipv6_nopay_octets counter reaches half of the maximum value or the maximum value. **Values:** ■ 0x0 (DISABLE): MMC Receive IPV6 Header Error Octet Counter Interrupt Mask is disabled ■ 0x1 (ENABLE): MMC Receive IPV6 Header Error Octet Counter Interrupt Mask is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RX_IPV6_NOPAY_OCTET_CNT_EN

Bits	Name	Memory Access	Description
22	RXIPV6HEROIM	R/W	MMC Receive IPV6 Good Octet Counter Interrupt Mask Setting this bit masks the interrupt when the rxipv6_hdrerr_octets counter reaches half of the maximum value or the maximum value. **Values:** ■ 0x0 (DISABLE): MMC Receive IPV6 Good Octet Counter Interrupt Mask is disabled ■ 0x1 (ENABLE): MMC Receive IPV6 Good Octet Counter Interrupt Mask is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RX_IPV6_HDRERR_OCTET_CNT_EN
21	RXIPV6GOIM	R/W	MMC Receive IPV6 Good Octet Counter Interrupt Mask Setting this bit masks the interrupt when the rxipv6_gd_octets counter reaches half of the maximum value or the maximum value. **Values:** ■ 0x0 (DISABLE): MMC Receive IPV6 Good Octet Counter Interrupt Mask is disabled ■ 0x1 (ENABLE): MMC Receive IPV6 Good Octet Counter Interrupt Mask is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RX_IPV6_GD_OCTET_CNT_EN
20	RXIPV4UDSBLOIM	R/W	MMC Receive IPV4 UDP Checksum Disabled Octet Counter Interrupt Mask Setting this bit masks the interrupt when the rxipv4_udsbl_octets counter reaches half of the maximum value or the maximum value. **Values:** ■ 0x0 (DISABLE): MMC Receive IPV4 UDP Checksum Disabled Octet Counter Interrupt Mask is disabled ■ 0x1 (ENABLE): MMC Receive IPV4 UDP Checksum Disabled Octet Counter Interrupt Mask is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RX_IPV4_UDSBL_OCTET_CNT_EN

Bits	Name	Memory Access	Description
19	RXIPV4FRAGOIM	R/W	MMC Receive IPV4 Fragmented Octet Counter Interrupt Mask Setting this bit masks the interrupt when the rxipv4_frag_octets counter reaches half of the maximum value or the maximum value. **Values:** ■ 0x0 (DISABLE): MMC Receive IPV4 Fragmented Octet Counter Interrupt Mask is disabled ■ 0x1 (ENABLE): MMC Receive IPV4 Fragmented Octet Counter Interrupt Mask is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RX_IPV4_FRAG_OCTET_CNT_EN
18	RXIPV4NOPAYOIM	R/W	MMC Receive IPV4 No Payload Octet Counter Interrupt Mask Setting this bit masks the interrupt when the rxipv4_nopay_octets counter reaches half of the maximum value or the maximum value. **Values:** ■ 0x0 (DISABLE): MMC Receive IPV4 No Payload Octet Counter Interrupt Mask is disabled ■ 0x1 (ENABLE): MMC Receive IPV4 No Payload Octet Counter Interrupt Mask is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RX_IPV4_NOPAY_OCTET_CNT_EN
17	RXIPV4HEROIM	R/W	MMC Receive IPV4 Header Error Octet Counter Interrupt Mask Setting this bit masks the interrupt when the rxipv4_hdrerr_octets counter reaches half of the maximum value or the maximum value. **Values:** ■ 0x0 (DISABLE): MMC Receive IPV4 Header Error Octet Counter Interrupt Mask is disabled ■ 0x1 (ENABLE): MMC Receive IPV4 Header Error Octet Counter Interrupt Mask is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RX_IPV4_HDRERR_OCTET_CNT_EN

Bits	Name	Memory Access	Description
16	RXIPV4GOIM	R/W	MMC Receive IPV4 Good Octet Counter Interrupt Mask Setting this bit masks the interrupt when the rxipv4_gd_octets counter reaches half of the maximum value or the maximum value. **Values:** ■ 0x0 (DISABLE): MMC Receive IPV4 Good Octet Counter Interrupt Mask is disabled ■ 0x1 (ENABLE): MMC Receive IPV4 Good Octet Counter Interrupt Mask is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RX_IPV4_GD_OCTET_CNT_EN
15:14	Reserved_15_14	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
13	RXICMPERPIM	R/W	MMC Receive ICMP Error Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rxicmp_err_pkts counter reaches half of the maximum value or the maximum value. **Values:** ■ 0x0 (DISABLE): MMC Receive ICMP Error Packet Counter Interrupt Mask is disabled ■ 0x1 (ENABLE): MMC Receive ICMP Error Packet Counter Interrupt Mask is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RX_ICMP_ERR_PKTS_CNT_EN
12	RXICMPGPIM	R/W	MMC Receive ICMP Good Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rxicmp_gd_pkts counter reaches half of the maximum value or the maximum value. **Values:** ■ 0x0 (DISABLE): MMC Receive ICMP Good Packet Counter Interrupt Mask is disabled ■ 0x1 (ENABLE): MMC Receive ICMP Good Packet Counter Interrupt Mask is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RX_ICMP_GD_PKTS_CNT_EN

Bits	Name	Memory Access	Description
11	RXTCPERPIM	R/W	MMC Receive TCP Error Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rxtcp_err_pkts counter reaches half of the maximum value or the maximum value. **Values:** ■ 0x0 (DISABLE): MMC Receive TCP Error Packet Counter Interrupt Mask is disabled ■ 0x1 (ENABLE): MMC Receive TCP Error Packet Counter Interrupt Mask is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RX_TCP_ERR_PKTS_CNT_EN
10	RXTCPGPIM	R/W	MMC Receive TCP Good Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rxtcp_gd_pkts counter reaches half of the maximum value or the maximum value. **Values:** ■ 0x0 (DISABLE): MMC Receive TCP Good Packet Counter Interrupt Mask is disabled ■ 0x1 (ENABLE): MMC Receive TCP Good Packet Counter Interrupt Mask is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RX_TCP_GD_PKTS_CNT_EN
9	RXUDPERPIM	R/W	MMC Receive UDP Error Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rxudp_err_pkts counter reaches half of the maximum value or the maximum value. **Values:** ■ 0x0 (DISABLE): MMC Receive UDP Error Packet Counter Interrupt Mask is disabled ■ 0x1 (ENABLE): MMC Receive UDP Error Packet Counter Interrupt Mask is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RX_UDP_ERR_PKTS_CNT_EN
8	RXUDPGPIM	R/W	MMC Receive UDP Good Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rxudp_gd_pkts counter reaches half of the maximum value or the maximum value. **Values:** ■ 0x0 (DISABLE): MMC Receive UDP Good Packet Counter Interrupt Mask is disabled ■ 0x1 (ENABLE): MMC Receive UDP Good Packet Counter Interrupt Mask is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RX_UDP_GD_PKTS_CNT_EN

Bits	Name	Memory Access	Description
7	RXIPV6NOPAYPIM	R/W	MMC Receive IPV6 No Payload Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rxipv6_nopay_pkts counter reaches half of the maximum value or the maximum value. **Values:** ■ 0x0 (DISABLE): MMC Receive IPV6 No Payload Packet Counter Interrupt Mask is disabled ■ 0x1 (ENABLE): MMC Receive IPV6 No Payload Packet Counter Interrupt Mask is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RX_IPV6_NOPAY_PKTS_CNT_EN
6	RXIPV6HERPIM	R/W	MMC Receive IPV6 Header Error Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rxipv6_hdrerr_pkts counter reaches half of the maximum value or the maximum value. **Values:** ■ 0x0 (DISABLE): MMC Receive IPV6 Header Error Packet Counter Interrupt Mask is disabled ■ 0x1 (ENABLE): MMC Receive IPV6 Header Error Packet Counter Interrupt Mask is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RX_IPV6_HDRERR_PKTS_CNT_EN
5	RXIPV6GPIM	R/W	MMC Receive IPV6 Good Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rxipv6_gd_pkts counter reaches half of the maximum value or the maximum value. **Values:** ■ 0x0 (DISABLE): MMC Receive IPV6 Good Packet Counter Interrupt Mask is disabled ■ 0x1 (ENABLE): MMC Receive IPV6 Good Packet Counter Interrupt Mask is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RX_IPV6_GD_PKTS_CNT_EN

Bits	Name	Memory Access	Description
4	RXIPV4UDSBLPIM	R/W	MMC Receive IPV4 UDP Checksum Disabled Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rxipv4_udsbl_pkts counter reaches half of the maximum value or the maximum value. **Values:** ■ 0x0 (DISABLE): MMC Receive IPV4 UDP Checksum Disabled Packet Counter Interrupt Mask is disabled ■ 0x1 (ENABLE): MMC Receive IPV4 UDP Checksum Disabled Packet Counter Interrupt Mask is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RX_IPV4_UDSBL_PKTS_CNT_EN
3	RXIPV4FRAGPIM	R/W	MMC Receive IPV4 Fragmented Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rxipv4_frag_pkts counter reaches half of the maximum value or the maximum value. **Values:** ■ 0x0 (DISABLE): MMC Receive IPV4 Fragmented Packet Counter Interrupt Mask is disabled ■ 0x1 (ENABLE): MMC Receive IPV4 Fragmented Packet Counter Interrupt Mask is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RX_IPV4_FRAG_PKTS_CNT_EN
2	RXIPV4NOPAYPIM	R/W	MMC Receive IPV4 No Payload Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rxipv4_nopay_pkts counter reaches half of the maximum value or the maximum value. **Values:** ■ 0x0 (DISABLE): MMC Receive IPV4 No Payload Packet Counter Interrupt Mask is disabled ■ 0x1 (ENABLE): MMC Receive IPV4 No Payload Packet Counter Interrupt Mask is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RX_IPV4_NOPAY_PKTS_CNT_EN

Bits	Name	Memory Access	Description
1	RXIPV4HERPIM	R/W	MMC Receive IPV4 Header Error Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rxipv4_hdrerr_pkts counter reaches half of the maximum value or the maximum value. **Values:** ■ 0x0 (DISABLE): MMC Receive IPV4 Header Error Packet Counter Interrupt Mask is disabled ■ 0x1 (ENABLE): MMC Receive IPV4 Header Error Packet Counter Interrupt Mask is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RX_IPV4_HDRERR_PKTS_CNT_EN
0	RXIPV4GPIM	R/W	MMC Receive IPV4 Good Packet Counter Interrupt Mask Setting this bit masks the interrupt when the rxipv4_gd_pkts counter reaches half of the maximum value or the maximum value. **Values:** ■ 0x0 (DISABLE): MMC Receive IPV4 Good Packet Counter Interrupt Mask is disabled ■ 0x1 (ENABLE): MMC Receive IPV4 Good Packet Counter Interrupt Mask is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RX_IPV4_GD_PKTS_CNT_EN

17.1.142 MMC_IPC_Rx_Interrupt

- **Description:** This register maintains the interrupt that the receive IPC statistic counters generate. The MMC Receive Checksum Offload Interrupt register maintains the interrupts generated when receive IPC statistic counters reach half their maximum values (0x8000_0000 for 32 bit counter and 0x8000 for 16 bit counter), and when they cross their maximum values (0xFFFF_FFFF for 32 bit counter and 0xFFFF for 16 bit counter). When Counter Stop Rollover is set, the interrupts are set but the counter remains at all-ones.
 The MMC Receive Checksum Offload Interrupt register is 32 bit wide. When the MMC IPC counter that caused the interrupt is read, its corresponding interrupt bit is cleared. The counter's least-significant byte lane (Bits[7:0]) must be read to clear the interrupt bit.
- **Size:** 32 bits
- **Offset:** 0x808
- **Exists:** ((DWC_EQOS_MMC_IPC_RX_PKTS_ATLEAST_ONE||DWC_EQOS_MMC_IPC_RX_DGRAM_BCNT_ATLEAST_ONE||DWC_EQOS_MMC_IPC_RX_PAYLD_BCNT_ATLEAST_ONE))

31:30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15:14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved_31_30	RXICMPEROIS	RXICMPGOIS	RXTCPEROIS	RXTCPGOIS	RXUDPEROIS	RXUDPGOIS	RXIPV6NOPAYOIS	RXIPV6HEROIS	RXIPV6GOIS	RXIPV4UDSBLOIS	RXIPV4FRAGOIS	RXIPV4NOPAYOIS	RXIPV4HEROIS	RXIPV4GOIS	Reserved_15_14	RXICMPERPIS	RXICMPGPIS	RXTCPERPIS	RXTCPGPIS	RXUDPERPIS	RXUDPGPIS	RXIPV6NOPAYPIS	RXIPV6HERPIS	RXIPV6GPIS	RXIPV4UDSBLPIS	RXIPV4FRAGPIS	RXIPV4NOPAYPIS	RXIPV4HERPIS	RXIPV4GPIS

Table 17-146 Fields for Register: MMC_IPC_Rx_Interrupt

Bits	Name	Memory Access	Description
31:30	Reserved_31_30	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
29	RXICMPEROIS	R	MMC Receive ICMP Error Octet Counter Interrupt Status This bit is set when the rxicmp_err_octets counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): MMC Receive ICMP Error Octet Counter Interrupt Status not detected ■ 0x1 (ACTIVE): MMC Receive ICMP Error Octet Counter Interrupt Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RX_ICMP_ERR_OCTET_CNT_EN

Bits	Name	Memory Access	Description
28	RXICMPGOIS	R	MMC Receive ICMP Good Octet Counter Interrupt Status This bit is set when the rxicmp_gd_octets counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): MMC Receive ICMP Good Octet Counter Interrupt Status not detected ■ 0x1 (ACTIVE): MMC Receive ICMP Good Octet Counter Interrupt Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RX_ICMP_GD_OCTET_CNT_EN
27	RXTCPEROIS	R	MMC Receive TCP Error Octet Counter Interrupt Status This bit is set when the rxtcp_err_octets counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): MMC Receive TCP Error Octet Counter Interrupt Status not detected ■ 0x1 (ACTIVE): MMC Receive TCP Error Octet Counter Interrupt Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RX_TCP_ERR_OCTET_CNT_EN
26	RXTCPGOIS	R	MMC Receive TCP Good Octet Counter Interrupt Status This bit is set when the rxtcp_gd_octets counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): MMC Receive TCP Good Octet Counter Interrupt Status not detected ■ 0x1 (ACTIVE): MMC Receive TCP Good Octet Counter Interrupt Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RX_TCP_GD_OCTET_CNT_EN

Bits	Name	Memory Access	Description
25	RXUDPEROIS	R	MMC Receive UDP Error Octet Counter Interrupt Status This bit is set when the rxudp_err_octets counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): MMC Receive UDP Error Octet Counter Interrupt Status not detected ■ 0x1 (ACTIVE): MMC Receive UDP Error Octet Counter Interrupt Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RX_UDP_ERR_OCTET_CNT_EN
24	RXUDPGOIS	R	MMC Receive UDP Good Octet Counter Interrupt Status This bit is set when the rxudp_gd_octets counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): MMC Receive UDP Good Octet Counter Interrupt Status not detected ■ 0x1 (ACTIVE): MMC Receive UDP Good Octet Counter Interrupt Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RX_UDP_GD_OCTET_CNT_EN
23	RXIPV6NOPAYOIS	R	MMC Receive IPV6 No Payload Octet Counter Interrupt Status This bit is set when the rxipv6_nopay_octets counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): MMC Receive IPV6 No Payload Octet Counter Interrupt Status not detected ■ 0x1 (ACTIVE): MMC Receive IPV6 No Payload Octet Counter Interrupt Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RX_IPV6_NOPAY_OCTET_CNT_EN

Bits	Name	Memory Access	Description
22	RXIPV6HEROIS	R	MMC Receive IPV6 Header Error Octet Counter Interrupt Status This bit is set when the rxipv6_hdrerr_octets counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): MMC Receive IPV6 Header Error Octet Counter Interrupt Status not detected ■ 0x1 (ACTIVE): MMC Receive IPV6 Header Error Octet Counter Interrupt Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RX_IPV6_HDRERR_OCTET_CNT_EN
21	RXIPV6GOIS	R	MMC Receive IPV6 Good Octet Counter Interrupt Status This bit is set when the rxipv6_gd_octets counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): MMC Receive IPV6 Good Octet Counter Interrupt Status not detected ■ 0x1 (ACTIVE): MMC Receive IPV6 Good Octet Counter Interrupt Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RX_IPV6_GD_OCTET_CNT_EN
20	RXIPV4UDSBLOIS	R	MMC Receive IPV4 UDP Checksum Disabled Octet Counter Interrupt Status This bit is set when the rxipv4_udsbl_octets counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): MMC Receive IPV4 UDP Checksum Disabled Octet Counter Interrupt Status not detected ■ 0x1 (ACTIVE): MMC Receive IPV4 UDP Checksum Disabled Octet Counter Interrupt Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RX_IPV4_UDSBL_OCTET_CNT_EN

Bits	Name	Memory Access	Description
19	RXIPV4FRAGOIS	R	MMC Receive IPV4 Fragmented Octet Counter Interrupt Status This bit is set when the rxipv4_frag_octets counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): MMC Receive IPV4 Fragmented Octet Counter Interrupt Status not detected ■ 0x1 (ACTIVE): MMC Receive IPV4 Fragmented Octet Counter Interrupt Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RX_IPV4_FRAG_OCTET_CNT_EN
18	RXIPV4NOPAYOIS	R	MMC Receive IPV4 No Payload Octet Counter Interrupt Status This bit is set when the rxipv4_nopay_octets counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): MMC Receive IPV4 No Payload Octet Counter Interrupt Status not detected ■ 0x1 (ACTIVE): MMC Receive IPV4 No Payload Octet Counter Interrupt Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RX_IPV4_NOPAY_OCTET_CNT_EN
17	RXIPV4HEROIS	R	MMC Receive IPV4 Header Error Octet Counter Interrupt Status This bit is set when the rxipv4_hdrerr_octets counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): MMC Receive IPV4 Header Error Octet Counter Interrupt Status not detected ■ 0x1 (ACTIVE): MMC Receive IPV4 Header Error Octet Counter Interrupt Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RX_IPV4_HDRERR_OCTET_CNT_EN

Bits	Name	Memory Access	Description
16	RXIPV4GOIS	R	MMC Receive IPV4 Good Octet Counter Interrupt Status This bit is set when the rxipv4_gd_octets counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): MMC Receive IPV4 Good Octet Counter Interrupt Status not detected ■ 0x1 (ACTIVE): MMC Receive IPV4 Good Octet Counter Interrupt Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RX_IPV4_GD_OCTET_CNT_EN
15:14	Reserved_15_14	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
13	RXICMPERPIS	R	MMC Receive ICMP Error Packet Counter Interrupt Status This bit is set when the rxicmp_err_pkts counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): MMC Receive ICMP Error Packet Counter Interrupt Status not detected ■ 0x1 (ACTIVE): MMC Receive ICMP Error Packet Counter Interrupt Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RX_ICMP_ERR_PKTS_CNT_EN
12	RXICMPGPIS	R	MMC Receive ICMP Good Packet Counter Interrupt Status This bit is set when the rxicmp_gd_pkts counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): MMC Receive ICMP Good Packet Counter Interrupt Status not detected ■ 0x1 (ACTIVE): MMC Receive ICMP Good Packet Counter Interrupt Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RX_ICMP_GD_PKTS_CNT_EN

Bits	Name	Memory Access	Description
11	RXTCPERPIS	R	MMC Receive TCP Error Packet Counter Interrupt Status This bit is set when the rxtcp_err_pkts counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): MMC Receive TCP Error Packet Counter Interrupt Status not detected ■ 0x1 (ACTIVE): MMC Receive TCP Error Packet Counter Interrupt Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RX_TCP_ERR_PKTS_CNT_EN
10	RXTCPGPIS	R	MMC Receive TCP Good Packet Counter Interrupt Status This bit is set when the rxtcp_gd_pkts counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): MMC Receive TCP Good Packet Counter Interrupt Status not detected ■ 0x1 (ACTIVE): MMC Receive TCP Good Packet Counter Interrupt Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RX_TCP_GD_PKTS_CNT_EN
9	RXUDPERPIS	R	MMC Receive UDP Error Packet Counter Interrupt Status This bit is set when the rxudp_err_pkts counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): MMC Receive UDP Error Packet Counter Interrupt Status not detected ■ 0x1 (ACTIVE): MMC Receive UDP Error Packet Counter Interrupt Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RX_UDP_ERR_PKTS_CNT_EN

Bits	Name	Memory Access	Description
8	RXUDPGPIS	R	MC Receive UDP Good Packet Counter Interrupt Status This bit is set when the rxudp_gd_pkts counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): MMC Receive UDP Good Packet Counter Interrupt Status not detected ■ 0x1 (ACTIVE): MMC Receive UDP Good Packet Counter Interrupt Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RX_UDP_GD_PKTS_CNT_EN
7	RXIPV6NOPAYPIS	R	MMC Receive IPV6 No Payload Packet Counter Interrupt Status This bit is set when the rxipv6_nopay_pkts counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): MMC Receive IPV6 No Payload Packet Counter Interrupt Status not detected ■ 0x1 (ACTIVE): MMC Receive IPV6 No Payload Packet Counter Interrupt Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RX_IPV6_NOPAY_PKTS_CNT_EN
6	RXIPV6HERPIS	R	MMC Receive IPV6 Header Error Packet Counter Interrupt Status This bit is set when the rxipv6_hdrerr_pkts counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): MMC Receive IPV6 Header Error Packet Counter Interrupt Status not detected ■ 0x1 (ACTIVE): MMC Receive IPV6 Header Error Packet Counter Interrupt Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RX_IPV6_HDRERR_PKTS_CNT_EN

Bits	Name	Memory Access	Description
5	RXIPV6GPIS	R	MMC Receive IPV6 Good Packet Counter Interrupt Status This bit is set when the rxipv6_gd_pkts counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): MMC Receive IPV6 Good Packet Counter Interrupt Status not detected ■ 0x1 (ACTIVE): MMC Receive IPV6 Good Packet Counter Interrupt Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RX_IPV6_GD_PKTS_CNT_EN
4	RXIPV4UDSBLPIS	R	MMC Receive IPV4 UDP Checksum Disabled Packet Counter Interrupt Status This bit is set when the rxipv4_udsbl_pkts counter reaches half of the maximum value or the maximum value. **Values:** ■ 0x0 (INACTIVE): MMC Receive IPV4 UDP Checksum Disabled Packet Counter Interrupt Status not detected ■ 0x1 (ACTIVE): MMC Receive IPV4 UDP Checksum Disabled Packet Counter Interrupt Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RX_IPV4_UDSBL_PKTS_CNT_EN
3	RXIPV4FRAGPIS	R	MMC Receive IPV4 Fragmented Packet Counter Interrupt Status This bit is set when the rxipv4_frag_pkts counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): MMC Receive IPV4 Fragmented Packet Counter Interrupt Status not detected ■ 0x1 (ACTIVE): MMC Receive IPV4 Fragmented Packet Counter Interrupt Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RX_IPV4_FRAG_PKTS_CNT_EN

Bits	Name	Memory Access	Description
2	RXIPV4NOPAYPIS	R	MMC Receive IPV4 No Payload Packet Counter Interrupt Status This bit is set when the rxipv4_nopay_pkts counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): MMC Receive IPV4 No Payload Packet Counter Interrupt Status not detected ■ 0x1 (ACTIVE): MMC Receive IPV4 No Payload Packet Counter Interrupt Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RX_IPV4_NOPAY_PKTS_CNT_EN
1	RXIPV4HERPIS	R	MMC Receive IPV4 Header Error Packet Counter Interrupt Status This bit is set when the rxipv4_hdrerr_pkts counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): MMC Receive IPV4 Header Error Packet Counter Interrupt Status not detected ■ 0x1 (ACTIVE): MMC Receive IPV4 Header Error Packet Counter Interrupt Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RX_IPV4_HDRERR_PKTS_CNT_EN
0	RXIPV4GPIS	R	MMC Receive IPV4 Good Packet Counter Interrupt Status This bit is set when the rxipv4_gd_pkts counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): MMC Receive IPV4 Good Packet Counter Interrupt Status not detected ■ 0x1 (ACTIVE): MMC Receive IPV4 Good Packet Counter Interrupt Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_MMC_RX_IPV4_GD_PKTS_CNT_EN

17.1.143 RxIPv4_Good_Packets

- **Description:** This register provides the number of good IPv4 datagrams received by DWC_ether_qos with the TCP, UDP, or ICMP payload.
- **Size:** 32 bits
- **Offset:** 0x810
- **Exists:** (DWC_EQOS_MMC_RX_IPV4_GD_PKTS_CNT_EN)

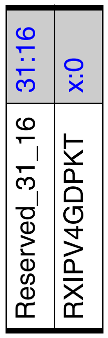


Table 17-147 Fields for Register: RxIPv4_Good_Packets

Bits	Name	Memory Access	Description
31:16	Reserved_31_16	R	Reserved. **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_MMC_RX_IPV4_GD_PKTS_CNT_16BIT_EN)
x:0	RXIPV4GDPKT	R	RxIPv4 Good Packets This field indicates the number of good IPv4 datagrams received with the TCP, UDP, or ICMP payload. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[x]:** DWC_EQOS_MMC_RX_IPV4_GD_PKTS_CNT_16BIT_EN?\ "16\":\"32\" - 1

17.1.144 RxIPv4_Header_Error_Packets

- **Description:** RxIPv4 Header Error Packets
  This register provides the number of IPv4 datagrams received by DWC_ether_qos with header (checksum, length, or version mismatch) errors.
- **Size:** 32 bits
- **Offset:** 0x814
- **Exists:** (DWC_EQOS_MMC_RX_IPV4_HDRERR_PKTS_CNT_EN)

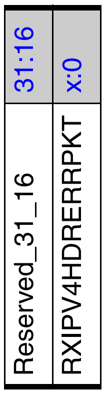


Table 17-148 Fields for Register: RxIPv4_Header_Error_Packets

Bits	Name	Memory Access	Description
31:16	Reserved_31_16	R	Reserved. **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_MMC_RX_IPV4_HDRERR_PKTS_CNT_16BIT_EN)
x:0	RXIPV4HDRERRPKT	R	RxIPv4 Header Error Packets This field indicates the number of IPv4 datagrams received with header (checksum, length, or version mismatch) errors. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[x]:** DWC_EQOS_MMC_RX_IPV4_HDRERR_PKTS_CNT_16BIT_EN?\"16\":\"32\" - 1

17.1.145 RxIPv4_No_Payload_Packets

- **Description:** This register provides the number of IPv4 datagram packets received by DWC_ether_qos that did not have a TCP, UDP, or ICMP payload.
- **Size:** 32 bits
- **Offset:** 0x818
- **Exists:** (DWC_EQOS_MMC_RX_IPV4_NOPAY_PKTS_CNT_EN)

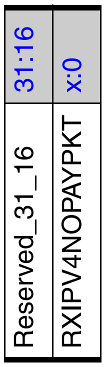


Table 17-149 Fields for Register: RxIPv4_No_Payload_Packets

Bits	Name	Memory Access	Description
31:16	Reserved_31_16	R	Reserved. **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_MMC_RX_IPV4_NOPAY_PKTS_CNT_16BIT_EN)
x:0	RXIPV4NOPAYPKT	R	RxIPv4 Payload Packets This field indicates the number of IPv4 datagram packets received that did not have a TCP, UDP, or ICMP payload. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[x]:** DWC_EQOS_MMC_RX_IPV4_NOPAY_PKTS_CNT_16BIT_EN?\"16\":\"32\" - 1

17.1.146 RxIPv4_Fragmented_Packets

- **Description:** This register provides the number of good IPv4 datagrams received by DWC_ether_qos with fragmentation.
- **Size:** 32 bits
- **Offset:** 0x81c
- **Exists:** (DWC_EQOS_MMC_RX_IPV4_FRAG_PKTS_CNT_EN)

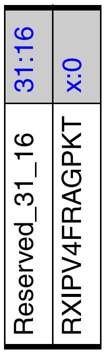


Table 17-150 Fields for Register: RxIPv4_Fragmented_Packets

Bits	Name	Memory Access	Description
31:16	Reserved_31_16	R	Reserved. **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_MMC_RX_IPV4_FRAG_PKTS_CNT_16BIT_EN)
x:0	RXIPV4FRAGPKT	R	RxIPv4 Fragmented Packets This field indicates the number of good IPv4 datagrams received with fragmentation. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[x]:** DWC_EQOS_MMC_RX_IPV4_FRAG_PKTS_CNT_16BIT_EN?\"16\":\"32\" - 1

17.1.147 RxIPv4_UDP_Checksum_Disabled_Packets

- **Description:** This register provides the number of good IPv4 datagrams received by DWC_ether_qos that had a UDP payload with checksum disabled.
- **Size:** 32 bits
- **Offset:** 0x820
- **Exists:** (DWC_EQOS_MMC_RX_IPV4_UDSBL_PKTS_CNT_EN)

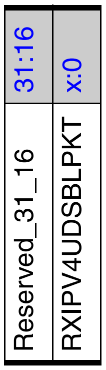


Table 17-151 Fields for Register: RxIPv4_UDP_Checksum_Disabled_Packets

Bits	Name	Memory Access	Description
31:16	Reserved_31_16	R	Reserved. **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_MMC_RX_IPV4_UDSBL_PKTS_CNT_16BIT_EN)
x:0	RXIPV4UDSBLPKT	R	RxIPv4 UDP Checksum Disabled Packets This field indicates the number of good IPv4 datagrams received that had a UDP payload with checksum disabled. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[x]:** DWC_EQOS_MMC_RX_IPV4_UDSBL_PKTS_CNT_16BIT_EN?\"16\":\"32\" - 1

17.1.148 RxIPv6_Good_Packets

- **Description:** This register provides the number of good IPv6 datagrams received by DWC_ether_qos with the TCP, UDP, or ICMP payload.
- **Size:** 32 bits
- **Offset:** 0x824
- **Exists:** (DWC_EQOS_MMC_RX_IPV6_GD_PKTS_CNT_EN)

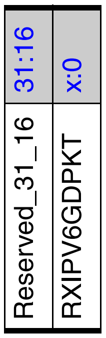


Table 17-152 Fields for Register: RxIPv6_Good_Packets

Bits	Name	Memory Access	Description
31:16	Reserved_31_16	R	Reserved. **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_MMC_RX_IPV6_GD_PKTS_CNT_16BIT_EN)
x:0	RXIPV6GDPKT	R	RxIPv6 Good Packets This field indicates the number of good IPv6 datagrams received with the TCP, UDP, or ICMP payload. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[x]:** DWC_EQOS_MMC_RX_IPV6_GD_PKTS_CNT_16BIT_EN?\"16\":\"32\" - 1

17.1.149 RxIPv6_Header_Error_Packets

- **Description:** This register provides the number of IPv6 datagrams received by DWC_ether_qos with header (length or version mismatch) errors.
- **Size:** 32 bits
- **Offset:** 0x828
- **Exists:** (DWC_EQOS_MMC_RX_IPV6_HDRERR_PKTS_CNT_EN)

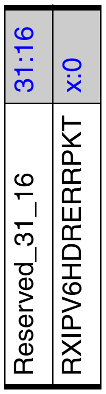


Table 17-153 Fields for Register: RxIPv6_Header_Error_Packets

Bits	Name	Memory Access	Description
31:16	Reserved_31_16	R	Reserved. **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_MMC_RX_IPV6_HDRERR_PKTS_CNT_16BIT_EN)
x:0	RXIPV6HDRERRPKT	R	RxIPv6 Header Error Packets This field indicates the number of IPv6 datagrams received with header (length or version mismatch) errors. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[x]:** DWC_EQOS_MMC_RX_IPV6_HDRERR_PKTS_CNT_16BIT_EN?\"16\":\"32\" - 1

17.1.150 RxIPv6_No_Payload_Packets

- **Description:** This register provides the number of IPv6 datagram packets received by DWC_ether_qos that did not have a TCP, UDP, or ICMP payload. This includes all IPv6 datagrams with fragmentation or security extension headers.
- **Size:** 32 bits
- **Offset:** 0x82c
- **Exists:** (DWC_EQOS_MMC_RX_IPV6_NOPAY_PKTS_CNT_EN)

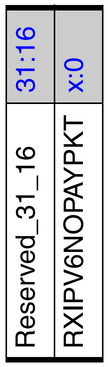


Table 17-154 Fields for Register: RxIPv6_No_Payload_Packets

Bits	Name	Memory Access	Description
31:16	Reserved_31_16	R	Reserved. **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_MMC_RX_IPV6_NOPAY_PKTS_CNT_16BIT_EN)
x:0	RXIPV6NOPAYPKT	R	RxIPv6 Payload Packets This field indicates the number of IPv6 datagram packets received that did not have a TCP, UDP, or ICMP payload. This includes all IPv6 datagrams with fragmentation or security extension headers. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[x]:** DWC_EQOS_MMC_RX_IPV6_NOPAY_PKTS_CNT_16BIT_EN?\"16\":\"32\" - 1

17.1.151 RxUDP_Good_Packets

- **Description:** This register provides the number of good IP datagrams received by DWC_ether_qos with a good UDP payload. This counter is not updated when the RxIPv4_UDP_Checksum_Disabled_Packets counter is incremented.
- **Size:** 32 bits
- **Offset:** 0x830
- **Exists:** (DWC_EQOS_MMC_RX_UDP_GD_PKTS_CNT_EN)

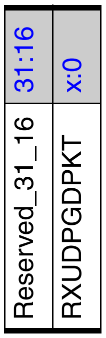


Table 17-155 Fields for Register: RxUDP_Good_Packets

Bits	Name	Memory Access	Description
31:16	Reserved_31_16	R	Reserved. **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_MMC_RX_UDP_GD_PKTS_CNT_16BIT_EN)
x:0	RXUDPGDPKT	R	RxUDP Good Packets This field indicates the number of good IP datagrams received with a good UDP payload. This counter is not updated when the RxIPv4_UDP_Checksum_Disabled_Packets counter is incremented. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[x]:** DWC_EQOS_MMC_RX_UDP_GD_PKTS_CNT_16BIT_EN?\"16\":\"32\" - 1

17.1.152 RxUDP_Error_Packets

- **Description:** This register provides the number of good IP datagrams received by DWC_ether_qos whose UDP payload has a checksum error.
- **Size:** 32 bits
- **Offset:** 0x834
- **Exists:** (DWC_EQOS_MMC_RX_UDP_ERR_PKTS_CNT_EN)

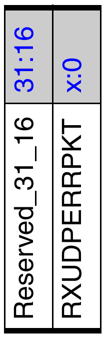


Table 17-156 Fields for Register: RxUDP_Error_Packets

Bits	Name	Memory Access	Description
31:16	Reserved_31_16	R	Reserved. **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_MMC_RX_UDP_ERR_PKTS_CNT_16BIT_EN)
x:0	RXUDPERRPKT	R	RxUDP Error Packets This field indicates the number of good IP datagrams received whose UDP payload has a checksum error. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[x]:** DWC_EQOS_MMC_RX_UDP_ERR_PKTS_CNT_16BIT_EN? \"16\":\"32\" - 1

17.1.153 RxTCP_Good_Packets

- **Description:** This register provides the number of good IP datagrams received by DWC_ether_qos with a good TCP payload.
- **Size:** 32 bits
- **Offset:** 0x838
- **Exists:** (DWC_EQOS_MMC_RX_TCP_GD_PKTS_CNT_EN)

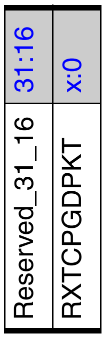


Table 17-157 Fields for Register: RxTCP_Good_Packets

Bits	Name	Memory Access	Description
31:16	Reserved_31_16	R	Reserved. **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_MMC_RX_TCP_GD_PKTS_CNT_16BIT_EN)
x:0	RXTCPGDPKT	R	RxTCP Good Packets This field indicates the number of good IP datagrams received with a good TCP payload. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[x]:** DWC_EQOS_MMC_RX_TCP_GD_PKTS_CNT_16BIT_EN?\" 16\":\"32\" - 1

17.1.154 RxTCP_Error_Packets

- **Description:** This register provides the number of good IP datagrams received by DWC_ether_qos whose TCP payload has a checksum error.
- **Size:** 32 bits
- **Offset:** 0x83c
- **Exists:** (DWC_EQOS_MMC_RX_TCP_ERR_PKTS_CNT_EN)

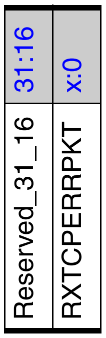


Table 17-158 Fields for Register: RxTCP_Error_Packets

Bits	Name	Memory Access	Description
31:16	Reserved_31_16	R	Reserved. **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_MMC_RX_TCP_ERR_PKTS_CNT_16BIT_EN)
x:0	RXTCPERRPKT	R	RxTCP Error Packets This field indicates the number of good IP datagrams received whose TCP payload has a checksum error. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[x]:** DWC_EQOS_MMC_RX_TCP_ERR_PKTS_CNT_16BIT_EN?\"16\":\"32\" - 1

17.1.155 RxICMP_Good_Packets

- **Description:** This register provides the number of good IP datagrams received by DWC_ether_qos with a good ICMP payload.
- **Size:** 32 bits
- **Offset:** 0x840
- **Exists:** (DWC_EQOS_MMC_RX_ICMP_GD_PKTS_CNT_EN)

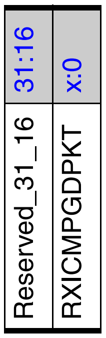


Table 17-159 Fields for Register: RxICMP_Good_Packets

Bits	Name	Memory Access	Description
31:16	Reserved_31_16	R	Reserved. **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_MMC_RX_ICMP_GD_PKTS_CNT_16BIT_EN)
x:0	RXICMPGDPKT	R	RxICMP Good Packets This field indicates the number of good IP datagrams received with a good ICMP payload. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[x]:** DWC_EQOS_MMC_RX_ICMP_GD_PKTS_CNT_16BIT_EN? \"16\":\"32\" - 1

17.1.156 RxICMP_Error_Packets

- **Description:** This register provides the number of good IP datagrams received by DWC_ether_qos whose ICMP payload has a checksum error.
- **Size:** 32 bits
- **Offset:** 0x844
- **Exists:** (DWC_EQOS_MMC_RX_ICMP_ERR_PKTS_CNT_EN)

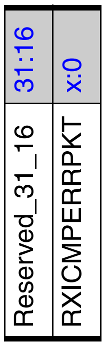


Table 17-160 Fields for Register: RxICMP_Error_Packets

Bits	Name	Memory Access	Description
31:16	Reserved_31_16	R	Reserved. **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_MMC_RX_ICMP_ERR_PKTS_CNT_16BIT_EN)
x:0	RXICMPERRPKT	R	RxICMP Error Packets This field indicates the number of good IP datagrams received whose ICMP payload has a checksum error. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[x]:** DWC_EQOS_MMC_RX_ICMP_ERR_PKTS_CNT_16BIT_EN ?\"16\":\"32\" - 1

17.1.157 RxIPv4_Good_Octets

- **Description:** This register provides the number of bytes received by DWC_ether_qos in good IPv4 datagrams encapsulating TCP, UDP, or ICMP data. (Ethernet header, FCS, pad, or IP pad bytes are not included in this counter.
- **Size:** 32 bits
- **Offset:** 0x850
- **Exists:** (DWC_EQOS_MMC_RX_IPV4_GD_OCTET_CNT_EN)

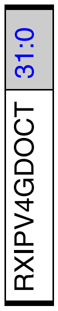


Table 17-161 Fields for Register: RxIPv4_Good_Octets

Bits	Name	Memory Access	Description
31:0	RXIPV4GDOCT	R	RxIPv4 Good Octets This field indicates the number of bytes received in good IPv4 datagrams encapsulating TCP, UDP, or ICMP data. (Ethernet header, FCS, pad, or IP pad bytes are not included in this counter. **Value After Reset:** 0x0 **Exists:** Always

17.1.158 RxIPv4_Header_Error_Octets

- **Description:** This register provides the number of bytes received by DWC_ether_qos in IPv4 datagrams with header errors (checksum, length, version mismatch). The value in the Length field of IPv4 header is used to update this counter. (Ethernet header, FCS, pad, or IP pad bytes are not included in this counter.
- **Size:** 32 bits
- **Offset:** 0x854
- **Exists:** (DWC_EQOS_MMC_RX_IPV4_HDRERR_OCTET_CNT_EN)

Table 17-162 Fields for Register: RxIPv4_Header_Error_Octets

Bits	Name	Memory Access	Description
31:0	RXIPV4HDRERROCT	R	RxIPv4 Header Error Octets This field indicates the number of bytes received in IPv4 datagrams with header errors (checksum, length, version mismatch). The value in the Length field of IPv4 header is used to update this counter. (Ethernet header, FCS, pad, or IP pad bytes are not included in this counter. **Value After Reset:** 0x0 **Exists:** Always

17.1.159 RxIPv4_No_Payload_Octets

- **Description:** This register provides the number of bytes received by DWC_ether_qos in IPv4 datagrams that did not have a TCP, UDP, or ICMP payload. The value in the Length field of IPv4 header is used to update this counter. (Ethernet header, FCS, pad, or IP pad bytes are not included in this counter.
- **Size:** 32 bits
- **Offset:** 0x858
- **Exists:** (DWC_EQOS_MMC_RX_IPV4_NOPAY_OCTET_CNT_EN)

Table 17-163 Fields for Register: RxIPv4_No_Payload_Octets

Bits	Name	Memory Access	Description
31:0	RXIPV4NOPAYOCT	R	RxIPv4 Payload Octets This field indicates the number of bytes received in IPv4 datagrams that did not have a TCP, UDP, or ICMP payload. The value in the Length field of IPv4 header is used to update this counter. (Ethernet header, FCS, pad, or IP pad bytes are not included in this counter. **Value After Reset:** 0x0 **Exists:** Always

17.1.160 RxIPv4_Fragmented_Octets

- **Description:** This register provides the number of bytes received by DWC_ether_qos in fragmented IPv4 datagrams. The value in the Length field of IPv4 header is used to update this counter. (Ethernet header, FCS, pad, or IP pad bytes are not included in this counter.
- **Size:** 32 bits
- **Offset:** 0x85c
- **Exists:** (DWC_EQOS_MMC_RX_IPV4_FRAG_OCTET_CNT_EN)

Table 17-164 Fields for Register: RxIPv4_Fragmented_Octets

Bits	Name	Memory Access	Description
31:0	RXIPV4FRAGOCT	R	RxIPv4 Fragmented Octets This field indicates the number of bytes received in fragmented IPv4 datagrams. The value in the Length field of IPv4 header is used to update this counter. (Ethernet header, FCS, pad, or IP pad bytes are not included in this counter. **Value After Reset:** 0x0 **Exists:** Always

17.1.161 RxIPv4_UDP_Checksum_Disable_Octets

- **Description:** This register provides the number of bytes received by DWC_ether_qos in a UDP segment that had the UDP checksum disabled. This counter does not count IP Header bytes. (Ethernet header, FCS, pad, or IP pad bytes are not included in this counter.
- **Size:** 32 bits
- **Offset:** 0x860
- **Exists:** (DWC_EQOS_MMC_RX_IPV4_UDSBL_OCTET_CNT_EN)

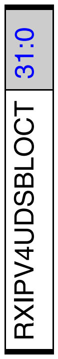


Table 17-165 Fields for Register: RxIPv4_UDP_Checksum_Disable_Octets

Bits	Name	Memory Access	Description
31:0	RXIPV4UDSBLOCT	R	RxIPv4 UDP Checksum Disable Octets This field indicates the number of bytes received in a UDP segment that had the UDP checksum disabled. This counter does not count IP Header bytes. (Ethernet header, FCS, pad, or IP pad bytes are not included in this counter. **Value After Reset:** 0x0 **Exists:** Always

17.1.162 RxIPv6_Good_Octets

- **Description:** This register provides the number of bytes received by DWC_ether_qos in good IPv6 datagrams encapsulating TCP, UDP, or ICMP data. (Ethernet header, FCS, pad, or IP pad bytes are not included in this counter.
- **Size:** 32 bits
- **Offset:** 0x864
- **Exists:** (DWC_EQOS_MMC_RX_IPV6_GD_OCTET_CNT_EN)

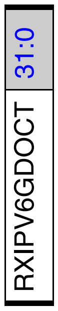


Table 17-166 Fields for Register: RxIPv6_Good_Octets

Bits	Name	Memory Access	Description
31:0	RXIPV6GDOCT	R	RxIPv6 Good Octets This field indicates the number of bytes received in good IPv6 datagrams encapsulating TCP, UDP, or ICMP data. (Ethernet header, FCS, pad, or IP pad bytes are not included in this counter. **Value After Reset:** 0x0 **Exists:** Always

17.1.163 RxIPv6_Header_Error_Octets

- **Description:** This register provides the number of bytes received by DWC_ether_qos in IPv6 datagrams with header errors (length, version mismatch). The value in the Length field of IPv6 header is used to update this counter. (Ethernet header, FCS, pad, or IP pad bytes are not included in this counter.
- **Size:** 32 bits
- **Offset:** 0x868
- **Exists:** (DWC_EQOS_MMC_RX_IPV6_HDRERR_OCTET_CNT_EN)

Table 17-167 Fields for Register: RxIPv6_Header_Error_Octets

Bits	Name	Memory Access	Description
31:0	RXIPV6HDRERROCT	R	RxIPv6 Header Error Octets This field indicates the number of bytes received in IPv6 datagrams with header errors (length, version mismatch). The value in the Length field of IPv6 header is used to update this counter. (Ethernet header, FCS, pad, or IP pad bytes are not included in this counter. **Value After Reset:** 0x0 **Exists:** Always

17.1.164 RxIPv6_No_Payload_Octets

- **Description:** This register provides the number of bytes received by DWC_ether_qos in IPv6 datagrams that did not have a TCP, UDP, or ICMP payload. The value in the Length field of IPv6 header is used to update this counter. (Ethernet header, FCS, pad, or IP pad bytes are not included in this counter.
- **Size:** 32 bits
- **Offset:** 0x86c
- **Exists:** (DWC_EQOS_MMC_RX_IPV6_NOPAY_OCTET_CNT_EN)

Table 17-168 Fields for Register: RxIPv6_No_Payload_Octets

Bits	Name	Memory Access	Description
31:0	RXIPV6NOPAYOCT	R	RxIPv6 Payload Octets This field indicates the number of bytes received in IPv6 datagrams that did not have a TCP, UDP, or ICMP payload. The value in the Length field of IPv6 header is used to update this counter. (Ethernet header, FCS, pad, or IP pad bytes are not included in this counter. **Value After Reset:** 0x0 **Exists:** Always

17.1.165 RxUDP_Good_Octets

- **Description:** This register provides the number of bytes received by DWC_ether_qos in a good UDP segment. This counter does not count IP header bytes.
- **Size:** 32 bits
- **Offset:** 0x870
- **Exists:** (DWC_EQOS_MMC_RX_UDP_GD_OCTET_CNT_EN)

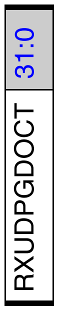


Table 17-169 Fields for Register: RxUDP_Good_Octets

Bits	Name	Memory Access	Description
31:0	RXUDPGDOCT	R	RxUDP Good Octets This field indicates the number of bytes received in a good UDP segment. This counter does not count IP header bytes. **Value After Reset:** 0x0 **Exists:** Always

17.1.166 RxUDP_Error_Octets

- **Description:** This register provides the number of bytes received by DWC_ether_qos in a UDP segment that had checksum errors. This counter does not count IP header bytes.
- **Size:** 32 bits
- **Offset:** 0x874
- **Exists:** (DWC_EQOS_MMC_RX_UDP_ERR_OCTET_CNT_EN)

Table 17-170 Fields for Register: RxUDP_Error_Octets

Bits	Name	Memory Access	Description
31:0	RXUDPERROCT	R	RxUDP Error Octets This field indicates the number of bytes received in a UDP segment that had checksum errors. This counter does not count IP header bytes. **Value After Reset:** 0x0 **Exists:** Always

17.1.167 RxTCP_Good_Octets

- **Description:** This register provides the number of bytes received by DWC_ether_qos in a good TCP segment. This counter does not count IP header bytes.
- **Size:** 32 bits
- **Offset:** 0x878
- **Exists:** (DWC_EQOS_MMC_RX_TCP_GD_OCTET_CNT_EN)

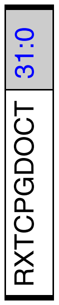


Table 17-171 Fields for Register: RxTCP_Good_Octets

Bits	Name	Memory Access	Description
31:0	RXTCPGDOCT	R	RxTCP Good Octets This field indicates the number of bytes received in a good TCP segment. This counter does not count IP header bytes. **Value After Reset:** 0x0 **Exists:** Always

17.1.168 RxTCP_Error_Octets

- **Description:** This register provides the number of bytes received by DWC_ether_qos in a TCP segment that had checksum errors. This counter does not count IP header bytes.
- **Size:** 32 bits
- **Offset:** 0x87c
- **Exists:** (DWC_EQOS_MMC_RX_TCP_ERR_OCTET_CNT_EN)

Table 17-172 Fields for Register: RxTCP_Error_Octets

Bits	Name	Memory Access	Description
31:0	RXTCPERROCT	R	RxTCP Error Octets This field indicates the number of bytes received in a TCP segment that had checksum errors. This counter does not count IP header bytes. **Value After Reset:** 0x0 **Exists:** Always

17.1.169 RxICMP_Good_Octets

- **Description:** This register provides the number of bytes received by DWC_ether_qos in a good ICMP segment. This counter does not count IP header bytes.
- **Size:** 32 bits
- **Offset:** 0x880
- **Exists:** (DWC_EQOS_MMC_RX_ICMP_GD_OCTET_CNT_EN)

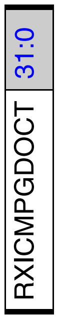


Table 17-173 Fields for Register: RxICMP_Good_Octets

Bits	Name	Memory Access	Description
31:0	RXICMPGDOCT	R	RxICMP Good Octets This field indicates the number of bytes received in a good ICMP segment. This counter does not count IP header bytes. **Value After Reset:** 0x0 **Exists:** Always

17.1.170 RxICMP_Error_Octets

- **Description:** This register provides the number of bytes received by DWC_ether_qos in a ICMP segment that had checksum errors. This counter does not count IP header bytes.
- **Size:** 32 bits
- **Offset:** 0x884
- **Exists:** (DWC_EQOS_MMC_RX_ICMP_ERR_OCTET_CNT_EN)

Table 17-174 Fields for Register: RxICMP_Error_Octets

Bits	Name	Memory Access	Description
31:0	RXICMPERROCT	R	RxICMP Error Octets This field indicates the number of bytes received in a ICMP segment that had checksum errors. This counter does not count IP header bytes. **Value After Reset:** 0x0 **Exists:** Always

17.1.171 MMC_FPE_Tx_Interrupt

- **Description:** This register maintains the interrupts generated from all FPE related Transmit statistics counters. The MMC FPE Transmit Interrupt register maintains the interrupts generated when transmit statistic counters reach half their maximum values (0x8000_0000 for 32 bit counter and 0x8000 for 16 bit counter), and when they cross their maximum values (0xFFFF_FFFF for 32-bit counter and 0xFFFF for 16-bit counter). When Counter Stop Rollover is set, the interrupts are set but the counter remains at all-ones. The MMC FPE Transmit Interrupt register is a 32 bit register. An interrupt bit is cleared when the respective MMC counter that caused the interrupt is read. The least significant byte lane (Bits[7:0]) of the respective counter must be read to clear the interrupt bit.
- **Size:** 32 bits
- **Offset:** 0x8a0
- **Exists:** (DWC_EQOS_FPE&&DWC_EQOS_MMC_FPE_EN)

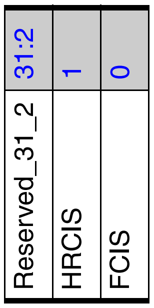


Table 17-175 Fields for Register: MMC_FPE_Tx_Interrupt

Bits	Name	Memory Access	Description
31:2	Reserved_31_2	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
1	HRCIS	R	MMC Tx Hold Request Counter Interrupt Status This bit is set when the Tx_Hold_Req_Cntr counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. Exists when any one of the RX/TX MMC counters are enabled during FPE with AV_EST Enabled configuration. **Values:** ■ 0x0 (INACTIVE): MMC Tx Hold Request Counter Interrupt Status not detected ■ 0x1 (ACTIVE): MMC Tx Hold Request Counter Interrupt Status detected **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_AV_EST)

Bits	Name	Memory Access	Description
0	FCIS	R	MMC Tx FPE Fragment Counter Interrupt status This bit is set when the Tx_FPE_Fragment_Cntr counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. Exists when any one of the RX/TX MMC counters are enabled during FPE Enabled configuration. **Values:** ■ 0x0 (INACTIVE): MMC Tx FPE Fragment Counter Interrupt status not detected ■ 0x1 (ACTIVE): MMC Tx FPE Fragment Counter Interrupt status detected **Value After Reset:** 0x0 **Exists:** Always

17.1.172 MMC_FPE_Tx_Interrupt_Mask

- **Description:** This register maintains the masks for interrupts generated from all FPE related Transmit statistics counters. The MMC Receive Interrupt Mask register maintains the masks for the interrupts generated when FPE related receive statistic counters reach half of their maximum value or the maximum values. This register is 32 bit wide.
- **Size:** 32 bits
- **Offset:** 0x8a4
- **Exists:** (DWC_EQOS_FPE&&DWC_EQOS_MMC_FPE_EN)

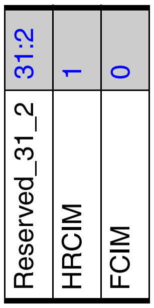


Table 17-176 Fields for Register: MMC_FPE_Tx_Interrupt_Mask

Bits	Name	Memory Access	Description
31:2	Reserved_31_2	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
1	HRCIM	R/W	MMC Transmit Hold Request Counter Interrupt Mask Setting this bit masks the interrupt when the Tx_Hold_Req_Cntr counter reaches half of the maximum value or the maximum value. Exists when any one of the RX/TX MMC counters are enabled during FPE with AV_EST Enabled configuration. **Values:** ■ 0x0 (DISABLE): MMC Transmit Hold Request Counter Interrupt Mask is disabled ■ 0x1 (ENABLE): MMC Transmit Hold Request Counter Interrupt Mask is enabled **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_AV_EST)

Bits	Name	Memory Access	Description
0	FCIM	R/W	MMC Transmit Fragment Counter Interrupt Mask Setting this bit masks the interrupt when the Tx_FPE_Fragment_Cntr counter reaches half of the maximum value or the maximum value. Exists when any one of the RX/TX MMC counters are enabled during FPE Enabled configuration. **Values:** ■ 0x0 (DISABLE): MMC Transmit Fragment Counter Interrupt Mask is disabled ■ 0x1 (ENABLE): MMC Transmit Fragment Counter Interrupt Mask is enabled **Value After Reset:** 0x0 **Exists:** Always

17.1.173 MMC_Tx_FPE_Fragment_Cntr

- **Description:** This register provides the number of additional mPackets transmitted due to preemption.
- **Size:** 32 bits
- **Offset:** 0x8a8
- **Exists:** (DWC_EQOS_FPE&&DWC_EQOS_MMC_FPE_EN)

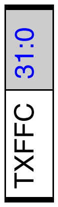


Table 17-177 Fields for Register: MMC_Tx_FPE_Fragment_Cntr

Bits	Name	Memory Access	Description
31:0	TXFFC	R	Tx FPE Fragment counter This field indicates the number of additional mPackets that has been transmitted due to preemption Exists when any one of the RX/TX MMC counters are enabled during FPE Enabled configuration. **Value After Reset:** 0x0 **Exists:** Always

17.1.174 MMC_Tx_Hold_Req_Cntr

- **Description:** This register provides the count of number of times a hold request is given to MAC
- **Size:** 32 bits
- **Offset:** 0x8ac
- **Exists:** (DWC_EQOS_FPE&&DWC_EQOS_MMC_FPE_EN)&&DWC_EQOS_AV_EST

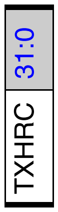


Table 17-178 **Fields for Register: MMC_Tx_Hold_Req_Cntr**

Bits	Name	Memory Access	Description
31:0	TXHRC	R	Tx Hold Request Counter This field indicates count of number of a hold request is given to MAC. Exists when any one of the RX/TX MMC counters are enabled during FPE with AV_EST Enabled configuration. **Value After Reset:** 0x0 **Exists:** Always

17.1.175 MMC_FPE_Rx_Interrupt

- **Description:** This register maintains the interrupts generated from all FPE related Receive statistics counters. The MMC FPE Receive Interrupt register maintains the interrupts generated when transmit statistic counters reach half their maximum values (0x8000_0000 for 32 bit counter and 0x8000 for 16 bit counter), and when they cross their maximum values (0xFFFF_FFFF for 32-bit counter and 0xFFFF for 16-bit counter). When Counter Stop Rollover is set, the interrupts are set but the counter remains at all-ones. The MMC FPE Receive Interrupt register is a 32 bit register. An interrupt bit is cleared when the respective MMC counter that caused the interrupt is read. The least significant byte lane (Bits[7:0]) of the respective counter must be read to clear the interrupt bit.
- **Size:** 32 bits
- **Offset:** 0x8c0
- **Exists:** (DWC_EQOS_FPE&&DWC_EQOS_MMC_FPE_EN)

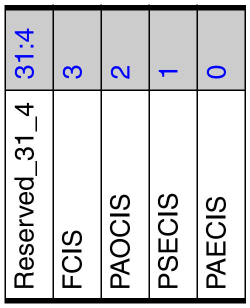


Table 17-179 **Fields for Register: MMC_FPE_Rx_Interrupt**

Bits	Name	Memory Access	Description
31:4	Reserved_31_4	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
3	FCIS	R	MMC Rx FPE Fragment Counter Interrupt Status This bit is set when the Rx_FPE_Fragment_Cntr counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. Exists when any one of the RX/TX MMC counters are enabled during FPE Enabled configuration. **Values:** ■ 0x0 (INACTIVE): MMC Rx FPE Fragment Counter Interrupt Status not detected ■ 0x1 (ACTIVE): MMC Rx FPE Fragment Counter Interrupt Status detected **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
2	PAOCIS	R	MMC Rx Packet Assembly OK Counter Interrupt Status This bit is set when the Rx_Packet_Assemble_Ok_Cntr counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. Exists when any one of the RX/TX MMC counters are enabled during FPE Enabled configuration. **Values:** ■ 0x0 (INACTIVE): MMC Rx Packet Assembly OK Counter Interrupt Status not detected ■ 0x1 (ACTIVE): MMC Rx Packet Assembly OK Counter Interrupt Status detected **Value After Reset:** 0x0 **Exists:** Always
1	PSECIS	R	MMC Rx Packet SMD Error Counter Interrupt Status This bit is set when the Rx_Packet_SMD_Err_Cntr counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. Exists when any one of the RX/TX MMC counters are enabled during FPE Enabled configuration. **Values:** ■ 0x0 (INACTIVE): MMC Rx Packet SMD Error Counter Interrupt Status not detected ■ 0x1 (ACTIVE): MMC Rx Packet SMD Error Counter Interrupt Status detected **Value After Reset:** 0x0 **Exists:** Always
0	PAECIS	R	MMC Rx Packet Assembly Error Counter Interrupt Status This bit is set when the Rx_Packet_Assemble_Err_Cntr counter reaches half of the maximum value or the maximum value. Access restriction applies. Clears on read. Self-set to 1 on internal event. Exists when any one of the RX/TX MMC counters are enabled during FPE Enabled configuration. **Values:** ■ 0x0 (INACTIVE): MMC Rx Packet Assembly Error Counter Interrupt Status not detected ■ 0x1 (ACTIVE): MMC Rx Packet Assembly Error Counter Interrupt Status detected **Value After Reset:** 0x0 **Exists:** Always

17.1.176 MMC_FPE_Rx_Interrupt_Mask

- **Description:** This register maintains the masks for interrupts generated from all FPE related Receive statistics counters. The MMC Receive Interrupt Mask register maintains the masks for the interrupts generated when FPE related receive statistic counters reach half of their maximum value or the maximum values. This register is 32 bit wide.
- **Size:** 32 bits
- **Offset:** 0x8c4
- **Exists:** (DWC_EQOS_FPE&&DWC_EQOS_MMC_FPE_EN)

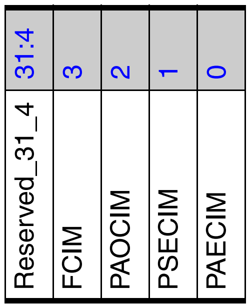


Table 17-180 Fields for Register: MMC_FPE_Rx_Interrupt_Mask

Bits	Name	Memory Access	Description
31:4	Reserved_31_4	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
3	FCIM	R/W	MMC Rx FPE Fragment Counter Interrupt Mask Setting this bit masks the interrupt when the Tx_FPE_Fragment_Cntr counter reaches half of the maximum value or the maximum value. Exists when any one of the RX/TX MMC counters are enabled during FPE Enabled configuration. **Values:** ■ 0x0 (DISABLE): MMC Rx FPE Fragment Counter Interrupt Mask is disabled ■ 0x1 (ENABLE): MMC Rx FPE Fragment Counter Interrupt Mask is enabled **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
2	PAOCIM	R/W	MMC Rx Packet Assembly OK Counter Interrupt Mask Setting this bit masks the interrupt when the Rx_Packet_Assemble_Ok_Cntr counter reaches half of the maximum value or the maximum value. Exists when any one of the RX/TX MMC counters are enabled during FPE Enabled configuration. **Values:** ■ 0x0 (DISABLE): MMC Rx Packet Assembly OK Counter Interrupt Mask is disabled ■ 0x1 (ENABLE): MMC Rx Packet Assembly OK Counter Interrupt Mask is enabled **Value After Reset:** 0x0 **Exists:** Always
1	PSECIM	R/W	MMC Rx Packet SMD Error Counter Interrupt Mask Setting this bit masks the interrupt when the R Rx_Packet_SMD_Err_Cntr counter reaches half of the maximum value or the maximum value. Exists when any one of the RX/TX MMC counters are enabled during FPE Enabled configuration. **Values:** ■ 0x0 (DISABLE): MMC Rx Packet SMD Error Counter Interrupt Mask is disabled ■ 0x1 (ENABLE): MMC Rx Packet SMD Error Counter Interrupt Mask is enabled **Value After Reset:** 0x0 **Exists:** Always
0	PAECIM	R/W	MMC Rx Packet Assembly Error Counter Interrupt Mask Setting this bit masks the interrupt when the R Rx_Packet_Assemble_Err_Cntr counter reaches half of the maximum value or the maximum value. Exists when any one of the RX/TX MMC counters are enabled during FPE Enabled configuration. **Values:** ■ 0x0 (DISABLE): MMC Rx Packet Assembly Error Counter Interrupt Mask is disabled ■ 0x1 (ENABLE): MMC Rx Packet Assembly Error Counter Interrupt Mask is enabled **Value After Reset:** 0x0 **Exists:** Always

17.1.177 MMC_Rx_Packet_Assembly_Err_Cntr

- **Description:** This register provides the number of MAC frames with reassembly errors on the Receiver, due to mismatch in the Fragment Count value.
- **Size:** 32 bits
- **Offset:** 0x8c8
- **Exists:** (DWC_EQOS_FPE&&DWC_EQOS_MMC_FPE_EN)

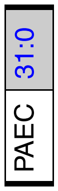


Table 17-181 Fields for Register: MMC_Rx_Packet_Assembly_Err_Cntr

Bits	Name	Memory Access	Description
31:0	PAEC	R	Rx Packet Assembly Error Counter This field indicates the number of MAC frames with reassembly errors on the Receiver, due to mismatch in the Fragment Count value. Exists when any one of the RX/TX MMC counters are enabled during FPE Enabled configuration. **Value After Reset:** 0x0 **Exists:** Always

17.1.178 MMC_Rx_Packet_SMD_Err_Cntr

- **Description:** This register provides the number of received MAC frames rejected due to unknown SMD value and MAC frame fragments rejected due to arriving with an SMD-C when there was no preceding preempted frame.
- **Size:** 32 bits
- **Offset:** 0x8cc
- **Exists:** (DWC_EQOS_FPE&&DWC_EQOS_MMC_FPE_EN)

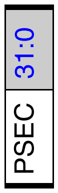


Table 17-182 Fields for Register: MMC_Rx_Packet_SMD_Err_Cntr

Bits	Name	Memory Access	Description
31:0	PSEC	R	Rx Packet SMD Error Counter This field indicates the number of MAC frames rejected due to unknown SMD value and MAC frame fragments rejected due to arriving with an SMD-C when there was no preceding preempted frame. Exists when at least one of the RX/TX MMC counters are enabled during FPE Enabled configuration. **Value After Reset:** 0x0 **Exists:** Always

17.1.179 MMC_Rx_Packet_Assembly_OK_Cntr

- **Description:** This register provides the number of MAC frames that were successfully reassembled and delivered to MAC.
- **Size:** 32 bits
- **Offset:** 0x8d0
- **Exists:** (DWC_EQOS_FPE&&DWC_EQOS_MMC_FPE_EN)

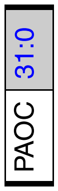


Table 17-183 Fields for Register: MMC_Rx_Packet_Assembly_OK_Cntr

Bits	Name	Memory Access	Description
31:0	PAOC	R	Rx Packet Assembly OK Counter This field indicates the number of MAC frames that were successfully reassembled and delivered to MAC. Exists when at least one of the RX/TX MMC counters are enabled during FPE Enabled configuration. **Value After Reset:** 0x0 **Exists:** Always

17.1.180 MMC_Rx_FPE_Fragment_Cntr

- **Description:** This register provides the number of additional mPackets received due to preemption.
- **Size:** 32 bits
- **Offset:** 0x8d4
- **Exists:** (DWC_EQOS_FPE&&DWC_EQOS_MMC_FPE_EN)

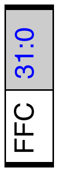


Table 17-184 Fields for Register: MMC_Rx_FPE_Fragment_Cntr

Bits	Name	Memory Access	Description
31:0	FFC	R	Rx FPE Fragment Counter This field indicates the number of additional mPackets received due to preemption Exists when at least one of the RX/TX MMC counters are enabled during FPE Enabled configuration. **Value After Reset:** 0x0 **Exists:** Always

17.1.181 MAC_L3_L4_Control(#i) (for i = 0; i <= DWC_EQOS_L3_L4_FILTER_NUM-1)

- **Description:** The Layer 3 and Layer 4 Control register controls the operations of filter 0 of Layer 3 and Layer 4.
- **Size:** 32 bits
- **Offset:** (0x0030*i)+0x0900
- **Exists:** (DWC_EQOS_L3_L4_FILTER_NUM>0)

31:29	28	x:y	x:24	23:22	21	20	19	18	17	16	15:11	10:6	5	4	3	2	1	0
Reserved_31_29	DMCHEN0	Reserved_27_y	DMCHN0	Reserved_23_22	L4DPIM0	L4DPM0	L4SPIM0	L4SPM0	Reserved_17	L4PEN0	L3HDBM0	L3HSBM0	L3DAIM0	L3DAM0	L3SAIM0	L3SAM0	Reserved_1	L3PEN0

Table 17-185 Fields for Register: MAC_L3_L4_Control(#i) (for i = 0; i <= DWC_EQOS_L3_L4_FILTER_NUM-1)

Bits	Name	Memory Access	Description
31:29	Reserved_31_29	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
28	DMCHEN0	R/W	DMA Channel Select Enable When set, this bit enables the selection of the DMA channel number for the packet that is passed by this L3_L4 filter. The DMA channel is indicated by the DMCHN bits. When this bit is reset, the DMA channel is not decided by this filter. **Values:** ■ 0x0 (DISABLE): DMA Channel Select is disabled ■ 0x1 (ENABLE): DMA Channel Select is enabled **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_SYS>1)&&(DWC_EQOS_NUM_DMA_RX_CH>1)

Bits	Name	Memory Access	Description
x:y	Reserved_27_y	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[x]:** DWC_EQOS_SYS>1&&DWC_EQOS_NUM_DMA_RX_CH>1 ?4-DWC_EQOS_NUM_DMA_RX_CHW:\"0x1\" + DWC_EQOS_SYS>1&&DWC_EQOS_NUM_DMA_RX_CH>1 ?24+DWC_EQOS_NUM_DMA_RX_CHW:\"27\" - 1 **Range Variable[y]:** DWC_EQOS_SYS>1&&DWC_EQOS_NUM_DMA_RX_CH>1 ?24+DWC_EQOS_NUM_DMA_RX_CHW:\"27\"
x:24	DMCHN0	R/W	DMA Channel Number When DMCHEN is set high, this field selects the DMA Channel number to which the packet passed by this filter is routed. The width of this field depends on the number of the DMA channels present in your configuration. **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_SYS>1)&&(DWC_EQOS_NUM_DMA_RX_CH >1) **Range Variable[x]:** DWC_EQOS_SYS>1&&DWC_EQOS_NUM_DMA_RX_CH>1 ?DWC_EQOS_NUM_DMA_RX_CHW:\"0x3\" + 23
23:22	Reserved_23_22	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
21	L4DPIM0	R/W	Layer 4 Destination Port Inverse Match Enable When this bit is set, the Layer 4 Destination Port number field is enabled for inverse matching. When this bit is reset, the Layer 4 Destination Port number field is enabled for perfect matching. This bit is valid and applicable only when the L4DPM0 bit is set high. **Values:** ■ 0x0 (DISABLE): Layer 4 Destination Port Inverse Match is disabled ■ 0x1 (ENABLE): Layer 4 Destination Port Inverse Match is enabled **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
20	L4DPM0	R/W	Layer 4 Destination Port Match Enable When this bit is set, the Layer 4 Destination Port number field is enabled for matching. When this bit is reset, the MAC ignores the Layer 4 Destination Port number field for matching. **Values:** ■ 0x0 (DISABLE): Layer 4 Destination Port Match is disabled ■ 0x1 (ENABLE): Layer 4 Destination Port Match is enabled **Value After Reset:** 0x0 **Exists:** Always
19	L4SPIM0	R/W	Layer 4 Source Port Inverse Match Enable When this bit is set, the Layer 4 Source Port number field is enabled for inverse matching. When this bit is reset, the Layer 4 Source Port number field is enabled for perfect matching. This bit is valid and applicable only when the L4SPM0 bit is set high. **Values:** ■ 0x0 (DISABLE): Layer 4 Source Port Inverse Match is disabled ■ 0x1 (ENABLE): Layer 4 Source Port Inverse Match is enabled **Value After Reset:** 0x0 **Exists:** Always
18	L4SPM0	R/W	Layer 4 Source Port Match Enable When this bit is set, the Layer 4 Source Port number field is enabled for matching. When this bit is reset, the MAC ignores the Layer 4 Source Port number field for matching. **Values:** ■ 0x0 (DISABLE): Layer 4 Source Port Match is disabled ■ 0x1 (ENABLE): Layer 4 Source Port Match is enabled **Value After Reset:** 0x0 **Exists:** Always
17	Reserved_17	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
16	L4PEN0	R/W	Layer 4 Protocol Enable When this bit is set, the Source and Destination Port number fields of UDP packets are used for matching. When this bit is reset, the Source and Destination Port number fields of TCP packets are used for matching. The Layer 4 matching is done only when the L4SPM0 or L4DPM0 bit is set. **Values:** ■ 0x0 (DISABLE): Layer 4 Protocol is disabled ■ 0x1 (ENABLE): Layer 4 Protocol is enabled **Value After Reset:** 0x0 **Exists:** Always
15:11	L3HDBM0	R/W	Layer 3 IP DA Higher Bits Match **IPv4 Packets:** This field contains the number of higher bits of IP Destination Address that are matched in the IPv4 packets. The following list describes the values of this field: ■ 0: No bits are masked. ■ 1: LSb[0] is masked ■ 2: Two LSbs [1:0] are masked ■ .. ■ 31: All bits except MSb are masked. **IPv6 Packets:** Bits[12:11] of this field correspond to Bits[6:5] of L3HSBM0 which indicate the number of lower bits of IP Source or Destination Address that are masked in the IPv6 packets. The following list describes the concatenated values of the L3HDBM0[1:0] and L3HSBM0 bits: ■ 0: No bits are masked. ■ 1: LSb[0] is masked. ■ 2: Two LSbs [1:0] are masked ■ .. ■ 127: All bits except MSb are masked. This field is valid and applicable only when the L3DAM0 or L3SAM0 bit is set. **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
10:6	L3HSBM0	R/W	Layer 3 IP SA Higher Bits Match **IPv4 Packets:** This field contains the number of lower bits of IP Source Address that are masked for matching in the IPv4 packets. The following list describes the values of this field: ■ 0: No bits are masked. ■ 1: LSb[0] is masked ■ 2: Two LSbs [1:0] are masked ■ .. ■ 31: All bits except MSb are masked. **IPv6 Packets:** This field contains Bits[4:0] of L3HSBM0. These bits indicate the number of higher bits of IP Source or Destination Address matched in the IPv6 packets. This field is valid and applicable only when the L3DAM0 or L3SAM0 bit is set high. **Value After Reset:** 0x0 **Exists:** Always
5	L3DAIM0	R/W	Layer 3 IP DA Inverse Match Enable When this bit is set, the Layer 3 IP Destination Address field is enabled for inverse matching. When this bit is reset, the Layer 3 IP Destination Address field is enabled for perfect matching. This bit is valid and applicable only when the L3DAM0 bit is set high. **Values:** ■ 0x0 (DISABLE): Layer 3 IP DA Inverse Match is disabled ■ 0x1 (ENABLE): Layer 3 IP DA Inverse Match is enabled **Value After Reset:** 0x0 **Exists:** Always
4	L3DAM0	R/W	Layer 3 IP DA Match Enable When this bit is set, the Layer 3 IP Destination Address field is enabled for matching. When this bit is reset, the MAC ignores the Layer 3 IP Destination Address field for matching. **Note**: When the L3PEN0 bit is set, you should set either this bit or the L3SAM0 bit because either IPv6 DA or SA can be checked for filtering. **Values:** ■ 0x0 (DISABLE): Layer 3 IP DA Match is disabled ■ 0x1 (ENABLE): Layer 3 IP DA Match is enabled **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
3	L3SAIM0	R/W	Layer 3 IP SA Inverse Match Enable When this bit is set, the Layer 3 IP Source Address field is enabled for inverse matching. When this bit reset, the Layer 3 IP Source Address field is enabled for perfect matching. This bit is valid and applicable only when the L3SAM0 bit is set. **Values:** ■ 0x0 (DISABLE): Layer 3 IP SA Inverse Match is disabled ■ 0x1 (ENABLE): Layer 3 IP SA Inverse Match is enabled **Value After Reset:** 0x0 **Exists:** Always
2	L3SAM0	R/W	Layer 3 IP SA Match Enable When this bit is set, the Layer 3 IP Source Address field is enabled for matching. When this bit is reset, the MAC ignores the Layer 3 IP Source Address field for matching. **Note**: When the L3PEN0 bit is set, you should set either this bit or the L3DAM0 bit because either IPv6 SA or DA can be checked for filtering. **Values:** ■ 0x0 (DISABLE): Layer 3 IP SA Match is disabled ■ 0x1 (ENABLE): Layer 3 IP SA Match is enabled **Value After Reset:** 0x0 **Exists:** Always
1	Reserved_1	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
0	L3PEN0	R/W	Layer 3 Protocol Enable When this bit is set, the Layer 3 IP Source or Destination Address matching is enabled for IPv6 packets. When this bit is reset, the Layer 3 IP Source or Destination Address matching is enabled for IPv4 packets. The Layer 3 matching is done only when the L3SAM0 or L3DAM0 bit is set. **Values:** ■ 0x0 (DISABLE): Layer 3 Protocol is disabled ■ 0x1 (ENABLE): Layer 3 Protocol is enabled **Value After Reset:** 0x0 **Exists:** Always

17.1.182 MAC_Layer4_Address(#i) (for i = 0; i <= DWC_EQOS_L3_L4_FILTER_NUM-1)

- **Description:** The MAC_Layer4_Address(#i), MAC_L3_L4_Control(#i), MAC_Layer3_Addr0_Reg(#i), MAC_Layer3_Addr1_Reg(#i), MAC_Layer3_Addr2_Reg(#i) and MAC_Layer3_Addr3_Reg(#i) registers are reserved (RO with default value) if Enable Layer 3 and Layer 4 Packet Filter option is not selected while configuring the core.
  You can configure the Layer 3 and Layer 4 Address Registers to be double-synchronized by selecting the Synchronize Layer 3 and Layer 4 Address Registers to Rx Clock Domain option while configuring the core. When you select this option, the synchronization is triggered only when Bits[31:24] (in little-endian mode) or Bits[7:0] (in big-endian mode) of the Layer 3 and Layer 4 Address Registers are written. For proper synchronization updates, you should perform consecutive writes to same Layer 3 and Layer 4 Address Registers after at least four clock cycles delay of the destination clock.
- **Size:** 32 bits
- **Offset:** (0x0030*i)+0x0904
- **Exists:** (DWC_EQOS_L3_L4_FILTER_NUM>0)

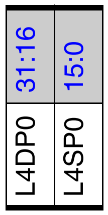


Table 17-186 Fields for Register: MAC_Layer4_Address(#i) (for i = 0; i <= DWC_EQOS_L3_L4_FILTER_NUM-1)

Bits	Name	Memory Access	Description
31:16	L4DP0	R/W	Layer 4 Destination Port Number Field When the L4PEN0 bit is reset and the L4DPM0 bit is set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with the TCP Destination Port Number field in the IPv4 or IPv6 packets. When the L4PEN0 and L4DPM0 bits are set in MAC_L3_L4_Control0 register, this field contains the value to be matched with the UDP Destination Port Number field in the IPv4 or IPv6 packets. **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
15:0	L4SP0	R/W	Layer 4 Source Port Number Field When the L4PEN0 bit is reset and the L4SPM0 bit is set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with the TCP Source Port Number field in the IPv4 or IPv6 packets. When the L4PEN0 and L4SPM0 bits are set in MAC_L3_L4_Control0 register, this field contains the value to be matched with the UDP Source Port Number field in the IPv4 or IPv6 packets. **Value After Reset:** 0x0 **Exists:** Always

17.1.183 MAC_Layer3_Addr0_Reg(#i) (for i = 0; i <= DWC_EQOS_L3_L4_FILTER_NUM-1)

- **Description:** For IPv4 packets, the Layer 3 Address 0 Register 0 register contains the 32-bit IP Source Address field. For IPv6 packets, it contains Bits[31:0] of the 128-bit IP Source Address or Destination Address field.
- **Size:** 32 bits
- **Offset:** (0x0030*i)+0x0910
- **Exists:** (DWC_EQOS_L3_L4_FILTER_NUM>0)

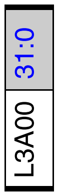


Table 17-187 Fields for Register: MAC_Layer3_Addr0_Reg(#i) (for i = 0; i <= DWC_EQOS_L3_L4_FILTER_NUM-1)

Bits	Name	Memory Access	Description
31:0	L3A00	R/W	Layer 3 Address 0 Field When the L3PEN0 and L3SAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[31:0] of the IP Source Address field in the IPv6 packets. When the L3PEN0 and L3DAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[31:0] of the IP Destination Address field in the IPv6 packets. When the L3PEN0 bit is reset and the L3SAM0 bit is set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with the IP Source Address field in the IPv4 packets. **Value After Reset:** 0x0 **Exists:** Always

17.1.184 MAC_Layer3_Addr1_Reg(#i) (for i = 0; i <= DWC_EQOS_L3_L4_FILTER_NUM-1)

- **Description:** For IPv4 packets, the Layer 3 Address 1 Register 0 register contains the 32-bit IP Destination Address field. For IPv6 packets, it contains Bits[63:32] of the 128-bit IP Source Address or Destination Address field.
- **Size:** 32 bits
- **Offset:** (0x0030*i)+0x0914
- **Exists:** (DWC_EQOS_L3_L4_FILTER_NUM>0)

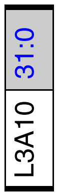


Table 17-188 Fields for Register: MAC_Layer3_Addr1_Reg(#i) (for i = 0; i <= DWC_EQOS_L3_L4_FILTER_NUM-1)

Bits	Name	Memory Access	Description
31:0	L3A10	R/W	Layer 3 Address 1 Field When the L3PEN0 and L3SAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[63:32] of the IP Source Address field in the IPv6 packets. When the L3PEN0 and L3DAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[63:32] of the IP Destination Address field in the IPv6 packets. When the L3PEN0 bit is reset and the L3SAM0 bit is set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with the IP Destination Address field in the IPv4 packets. **Value After Reset:** 0x0 **Exists:** Always

17.1.185 MAC_Layer3_Addr2_Reg(#i) (for i = 0; i <= DWC_EQOS_L3_L4_FILTER_NUM-1)

- **Description:** The Layer 3 Address 2 Register 0 register is reserved for IPv4 packets. For IPv6 packets, it contains Bits[95:64] of 128-bit IP Source Address or Destination Address field.
- **Size:** 32 bits
- **Offset:** (0x0030*i)+0x0918
- **Exists:** (DWC_EQOS_L3_L4_FILTER_NUM>0)

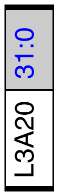


Table 17-189 Fields for Register: MAC_Layer3_Addr2_Reg(#i) (for i = 0; i <= DWC_EQOS_L3_L4_FILTER_NUM-1)

Bits	Name	Memory Access	Description
31:0	L3A20	R/W	Layer 3 Address 2 Field When the L3PEN0 and L3SAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[95:64] of the IP Source Address field in the IPv6 packets. When the L3PEN0 and L3DAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[95:64] of the IP Destination Address field in the IPv6 packets. When the L3PEN0 bit is reset in the MAC_L3_L4_Control0 register, this field is not used. **Value After Reset:** 0x0 **Exists:** Always

17.1.186 MAC_Layer3_Addr3_Reg(#i) (for i = 0; i <= DWC_EQOS_L3_L4_FILTER_NUM-1)

- **Description:** The Layer 3 Address 3 Register 0 register is reserved for IPv4 packets. For IPv6 packets, it contains Bits[127:96] of 128-bit IP Source Address or Destination Address field.
- **Size:** 32 bits
- **Offset:** (0x0030*i)+0x091C
- **Exists:** (DWC_EQOS_L3_L4_FILTER_NUM>0)

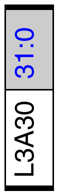


Table 17-190 Fields for Register: MAC_Layer3_Addr3_Reg(#i) (for i = 0; i <= DWC_EQOS_L3_L4_FILTER_NUM-1)

Bits	Name	Memory Access	Description
31:0	L3A30	R/W	Layer 3 Address 3 Field When the L3PEN0 and L3SAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[127:96] of the IP Source Address field in the IPv6 packets. When the L3PEN0 and L3DAM0 bits are set in the MAC_L3_L4_Control0 register, this field contains the value to be matched with Bits[127:96] of the IP Destination Address field in the IPv6 packets. When the L3PEN0 bit is reset in the MAC_L3_L4_Control0 register, this field is not used. **Value After Reset:** 0x0 **Exists:** Always

17.1.187 MAC_Timestamp_Control

- **Description:** This register controls the operation of the System Time generator and processing of PTP packets for timestamping in the Receiver.
- **Size:** 32 bits
- **Offset:** 0xb00
- **Exists:** (DWC_EQOS_TIME_STAMPING)

31:29	28	27:25	24	23:21	20	19	18	17:16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved_31_29	AV8021ASMEN	Reserved_27_25	TXTSSTSM	Reserved_23_21	ESTI	CSC	TSENMACADDR	SNAPTYPSEL	TSMSTRENA	TSEVNTENA	TSIPV4ENA	TSIPV6ENA	TSIPENA	TSVER2ENA	TSCTRLSSR	TSENALL	Reserved_7	PTGE	TSADDREG	TSTRIG	TSUPDT	TSINIT	TSCFUPDT	TSENA

Table 17-191 Fields for Register: MAC_Timestamp_Control

Bits	Name	Memory Access	Description
31:29	Reserved_31_29	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
28	AV8021ASMEN	R/W	AV 802.1AS Mode Enable When this bit is set, the MAC processes only untagged PTP over Ethernet packets for providing PTP status and capturing timestamp snapshots, that is, IEEE 802.1AS mode of operation. When PTP offload feature is enabled, for the purpose of PTP offload, the transport specific field in the PTP header is generated and checked based on the value of this bit. **Values:** ■ 0x0 (DISABLE): AV 802.1AS Mode is disabled ■ 0x1 (ENABLE): AV 802.1AS Mode is enabled **Value After Reset:** 0x0 **Exists:** Always
27:25	Reserved_27_25	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
24	TXTSSTSM	R/W	Transmit Timestamp Status Mode When this bit is set, the MAC overwrites the earlier transmit timestamp status even if it is not read by the software. The MAC indicates this by setting the TXTSSMIS bit of the MAC_Tx_Timestamp_Status_Nanoseconds register. When this bit is reset, the MAC ignores the timestamp status of current packet if the timestamp status of previous packet is not read by the software. The MAC indicates this by setting the TXTSSMIS bit of the MAC_Tx_Timestamp_Status_Nanoseconds register. **Values:** ■ 0x0 (DISABLE): Transmit Timestamp Status Mode is disabled ■ 0x1 (ENABLE): Transmit Timestamp Status Mode is enabled **Value After Reset:** 0x0 **Exists:** ((!DWC_EQOS_CORE)||DWC_EQOS_PTO_EN)
23:21	Reserved_23_21	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
20	ESTI	R/W	External System Time Input When this bit is set, the MAC uses the external 64-bit reference System Time input for the following: ■ To take the timestamp provided as status ■ To insert the timestamp in transmit PTP packets when One-step Timestamp or Timestamp Offload feature is enabled. When this bit is reset, the MAC uses the internal reference System Time. **Values:** ■ 0x0 (DISABLE): External System Time Input is disabled ■ 0x1 (ENABLE): External System Time Input is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_SYSTIME_SOURCE==2

Bits	Name	Memory Access	Description
19	CSC	R/W	Enable checksum correction during OST for PTP over UDP/IPv4 packets When this bit is set, the last two bytes of PTP message sent over UDP/IPv4 is updated to keep the UDP checksum correct, for changes made to origin timestamp and/or correction field as part of one step timestamp operation. The application shall form the packet with these two dummy bytes. When reset, no updates are done to keep the UDP checksum correct. The application shall form the packet with UDP checksum set to 0. **Values:** ■ 0x0 (DISABLE): checksum correction during OST for PTP over UDP/IPv4 packets is disabled ■ 0x1 (ENABLE): checksum correction during OST for PTP over UDP/IPv4 packets is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_POU_OST_EN
18	TSENMACADDR	R/W	Enable MAC Address for PTP Packet Filtering When this bit is set, the DA MAC address (that matches any MAC Address register) is used to filter the PTP packets when PTP is directly sent over Ethernet. When this bit is set, received PTP packets with DA containing a special multicast or unicast address that matches the one programmed in MAC address registers are considered for processing as indicated below, when PTP is directly sent over Ethernet. For normal time stamping operation, MAC address registers 0 to 31 is considered for unicast destination address matching. For PTP offload, only MAC address register 0 is considered for unicast destination address matching. **Values:** ■ 0x0 (DISABLE): MAC Address for PTP Packet Filtering is disabled ■ 0x1 (ENABLE): MAC Address for PTP Packet Filtering is enabled **Value After Reset:** 0x0 **Exists:** Always
17:16	SNAPTYPSEL	R/W	Select PTP packets for Taking Snapshots These bits, along with Bits 15 and 14, decide the set of PTP packet types for which snapshot needs to be taken. The encoding is given in Timestamp Snapshot Dependency on Register Bits Table. **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
15	TSMSTRENA	R/W	Enable Snapshot for Messages Relevant to Master When this bit is set, the snapshot is taken only for the messages that are relevant to the master node. Otherwise, the snapshot is taken for the messages relevant to the slave node. **Values:** ■ 0x0 (DISABLE): Snapshot for Messages Relevant to Master is disabled ■ 0x1 (ENABLE): Snapshot for Messages Relevant to Master is enabled **Value After Reset:** 0x0 **Exists:** Always
14	TSEVNTENA	R/W	Enable Timestamp Snapshot for Event Messages When this bit is set, the timestamp snapshot is taken only for event messages (SYNC, Delay_Req, Pdelay_Req, or Pdelay_Resp). When this bit is reset, the snapshot is taken for all messages except Announce, Management, and Signaling. For more information about the timestamp snapshots, see Timestamp Snapshot Dependency on Register Bits Table. **Values:** ■ 0x0 (DISABLE): Timestamp Snapshot for Event Messages is disabled ■ 0x1 (ENABLE): Timestamp Snapshot for Event Messages is enabled **Value After Reset:** 0x0 **Exists:** Always
13	TSIPV4ENA	R/W	Enable Processing of PTP Packets Sent over IPv4-UDP When this bit is set, the MAC receiver processes the PTP packets encapsulated in IPv4-UDP packets. When this bit is reset, the MAC ignores the PTP transported over IPv4-UDP packets. This bit is set by default. **Values:** ■ 0x0 (DISABLE): Processing of PTP Packets Sent over IPv4-UDP is disabled ■ 0x1 (ENABLE): Processing of PTP Packets Sent over IPv4-UDP is enabled **Value After Reset:** 0x1 **Exists:** Always

Bits	Name	Memory Access	Description
12	TSIPV6ENA	R/W	Enable Processing of PTP Packets Sent over IPv6-UDP When this bit is set, the MAC receiver processes the PTP packets encapsulated in IPv6-UDP packets. When this bit is clear, the MAC ignores the PTP transported over IPv6-UDP packets. **Values:** ■ 0x0 (DISABLE): Processing of PTP Packets Sent over IPv6-UDP is disabled ■ 0x1 (ENABLE): Processing of PTP Packets Sent over IPv6-UDP is enabled **Value After Reset:** 0x0 **Exists:** Always
11	TSIPENA	R/W	Enable Processing of PTP over Ethernet Packets When this bit is set, the MAC receiver processes the PTP packets encapsulated directly in the Ethernet packets. When this bit is reset, the MAC ignores the PTP over Ethernet packets. **Values:** ■ 0x0 (DISABLE): Processing of PTP over Ethernet Packets is disabled ■ 0x1 (ENABLE): Processing of PTP over Ethernet Packets is enabled **Value After Reset:** 0x0 **Exists:** Always
10	TSVER2ENA	R/W	Enable PTP Packet Processing for Version 2 Format When this bit is set, the IEEE 1588 version 2 format is used to process the PTP packets. When this bit is reset, the IEEE 1588 version 1 format is used to process the PTP packets. The IEEE 1588 formats are described in 'PTP Processing and Control'. **Values:** ■ 0x0 (DISABLE): PTP Packet Processing for Version 2 Format is disabled ■ 0x1 (ENABLE): PTP Packet Processing for Version 2 Format is enabled **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
9	TSCTRLSSR	R/W	Timestamp Digital or Binary Rollover Control When this bit is set, the Timestamp Low register rolls over after 0x3B9A_C9FF value (that is, 1 nanosecond accuracy) and increments the timestamp (High) seconds. When this bit is reset, the rollover value of sub-second register is 0x7FFF_FFFF. The sub-second increment must be programmed correctly depending on the PTP reference clock frequency and the value of this bit. **Values:** ■ 0x0 (DISABLE): Timestamp Digital or Binary Rollover Control is disabled ■ 0x1 (ENABLE): Timestamp Digital or Binary Rollover Control is enabled **Value After Reset:** 0x0 **Exists:** Always
8	TSENALL	R/W	Enable Timestamp for All Packets When this bit is set, the timestamp snapshot is enabled for all packets received by the MAC. **Values:** ■ 0x0 (DISABLE): Timestamp for All Packets disabled ■ 0x1 (ENABLE): Timestamp for All Packets enabled **Value After Reset:** 0x0 **Exists:** Always
7	Reserved_7	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
6	PTGE	R/W	Presentation Time Generation Enable When this bit is set the Presentation Time generation will be enabled. **Values:** ■ 0x0 (DISABLE): Presentation Time Generation is disabled ■ 0x1 (ENABLE): Presentation Time Generation is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_FLEXI_PPS_OUT_EN

Bits	Name	Memory Access	Description
5	TSADDREG	R/W	Update Addend Register When this bit is set, the content of the Timestamp Addend register is updated in the PTP block for fine correction. This bit is cleared when the update is complete. This bit should be zero before it is set. Access restriction applies. Setting 1 sets. Self-cleared. Setting 0 has no effect. **Values:** ■ 0x0 (DISABLE): Addend Register is not updated ■ 0x1 (ENABLE): Addend Register is updated **Value After Reset:** 0x0 **Exists:** DWC_EQOS_COARSE_FINE_CORRECTION!=0&&DWC_EQOS_SYSTIME_SOURCE!=1 **Testable:** untestable
4	TSTRIG	R/W	Enable Timestamp Interrupt Trigger When this bit is set, the timestamp interrupt is generated when the System Time becomes greater than the value written in the Target Time register. This bit is reset after the Timestamp Trigger Interrupt is generated. Access restriction applies. Setting 1 sets. Self-cleared. Setting 0 has no effect. **Values:** ■ 0x0 (DISABLE): Timestamp Interrupt Trigger is not enabled ■ 0x1 (ENABLE): Timestamp Interrupt Trigger is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_SYSTIME_SOURCE!=1&&!DWC_EQOS_FLEXI_PPS_OUT_EN **Testable:** untestable

Bits	Name	Memory Access	Description
3	TSUPDT	R/W	Update Timestamp When this bit is set, the system time is updated (added or subtracted) with the value specified in MAC_System_Time_Seconds_Update and MAC_System_Time_Nanoseconds_Update registers. This bit should be zero before updating it. This bit is reset when the update is complete in hardware. The Timestamp Higher Word register (if enabled during core configuration) is not updated. When Media Clock Generation and Recovery is configured (DWC_EQOS_FLEXI_PPS_OUT_EN) and enabled MAC_Presn_Time_Updt should also be updated before setting this field. Access restriction applies. Setting 1 sets. Self-cleared. Setting 0 has no effect. **Values:** ■ 0x0 (DISABLE): Timestamp is not updated ■ 0x1 (ENABLE): Timestamp is updated **Value After Reset:** 0x0 **Exists:** DWC_EQOS_SYSTIME_SOURCE!=1 **Testable:** untestable
2	TSINIT	R/W	Initialize Timestamp When this bit is set, the system time is initialized (overwritten) with the value specified in the MAC_System_Time_Seconds_Update and MAC_System_Time_Nanoseconds_Update registers. This bit should be zero before it is updated. This bit is reset when the initialization is complete. The Timestamp Higher Word register (if enabled during core configuration) can only be initialized. When Media Clock Generation and Recovery is configured (DWC_EQOS_FLEXI_PPS_OUT_EN) and enabled MAC_Presn_Time_Updt should also be updated before setting this field. Access restriction applies. Setting 1 sets. Self-cleared. Setting 0 has no effect. **Values:** ■ 0x0 (DISABLE): Timestamp is not initialized ■ 0x1 (ENABLE): Timestamp is initialized **Value After Reset:** 0x0 **Exists:** DWC_EQOS_SYSTIME_SOURCE!=1 **Testable:** untestable

Bits	Name	Memory Access	Description
1	TSCFUPDT	R/W	Fine or Coarse Timestamp Update When this bit is set, the Fine method is used to update system timestamp. When this bit is reset, Coarse method is used to update the system timestamp. **Values:** ■ 0x0 (COARSE): Coarse method is used to update system timestamp ■ 0x1 (FINE): Fine method is used to update system time-stamp **Value After Reset:** 0x0 **Exists:** DWC_EQOS_COARSE_FINE_CORRECTION!=0&&DWC_EQOS_SYSTIME_SOURCE!=1
0	TSENA	R/W	Enable Timestamp When this bit is set, the timestamp is added for Transmit and Receive packets. When disabled, timestamp is not added for transmit and receive packets and the Timestamp Generator is also suspended. You need to initialize the Timestamp (system time) after enabling this mode. On the Receive side, the MAC processes the 1588 packets only if this bit is set. **Values:** ■ 0x0 (DISABLE): Timestamp is disabled ■ 0x1 (ENABLE): Timestamp is enabled **Value After Reset:** 0x0 **Exists:** Always

17.1.188 MAC_Sub_Second_Increment

- **Description:** This register specifies the value to be added to the internal system time register every cycle of clk_ptp_ref_i clock.
- **Size:** 32 bits
- **Offset:** 0xb04
- **Exists:** (DWC_EQOS_TIME_STAMPING&&DWC_EQOS_SYSTIME_SOURCE!=1)

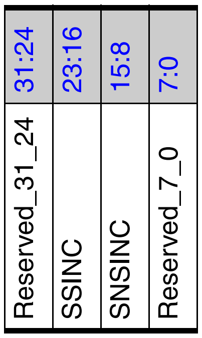


Table 17-192 Fields for Register: MAC_Sub_Second_Increment

Bits	Name	Memory Access	Description
31:24	Reserved_31_24	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
23:16	SSINC	R/W	Sub-second Increment Value The value programmed in this field is accumulated every clock cycle (of clk_ptp_i) with the contents of the sub-second register. For example, when the PTP clock is 50 MHz (period is 20 ns), you should program 20 (0x14) when the System Time Nanoseconds register has an accuracy of 1 ns [Bit 9 (TSCTRLSSR) is set in MAC_Timestamp_Control]. When TSCTRLSSR is clear, the Nanoseconds register has a resolution of ~0.465 ns. In this case, you should program a value of 43 (0x2B) which is derived by 20 ns/0.465. **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
15:8	SNSINC	R/W	Sub-nanosecond Increment Value This field contains the sub-nanosecond increment value, represented in nanoseconds multiplied by 2^8. This value is accumulated with the sub-nanoseconds field of the subsecond register. For example, when TSCTRLSSR field in the MAC_Timestamp_Control register is set. and if the required increment is 5.3ns, then SSINC should be 0x05 and SNSINC should be 0x4C. **Value After Reset:** 0x0 **Exists:** DWC_EQOS_ADV_SUBNSEC_SUPPORT
7:0	Reserved_7_0	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always

17.1.189 MAC_System_Time_Seconds

- **Description:** The System Time Seconds register, along with System Time Nanoseconds register, indicates the current value of the system time maintained by the MAC. Though it is updated on a continuous basis, there is some delay from the actual time because of clock domain transfer latencies (from clk_ptp_ref_i to CSR clock).
- **Size:** 32 bits
- **Offset:** 0xb08
- **Exists:** (DWC_EQOS_TIME_STAMPING&&DWC_EQOS_SYSTIME_SOURCE!=1)

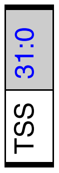


Table 17-193 Fields for Register: MAC_System_Time_Seconds

Bits	Name	Memory Access	Description
31:0	TSS	R	Timestamp Second The value in this field indicates the current value in seconds of the System Time maintained by the MAC. **Value After Reset:** 0x0 **Exists:** Always

17.1.190 MAC_System_Time_Nanoseconds

- **Description:** The System Time Nanoseconds register, along with System Time Seconds register, indicates the current value of the system time maintained by the MAC.
- **Size:** 32 bits
- **Offset:** 0xb0c
- **Exists:** (DWC_EQOS_TIME_STAMPING&&DWC_EQOS_SYSTIME_SOURCE!=1)

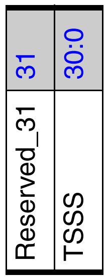


Table 17-194 Fields for Register: MAC_System_Time_Nanoseconds

Bits	Name	Memory Access	Description
31	Reserved_31	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
30:0	TSSS	R	Timestamp Sub Seconds The value in this field has the sub-second representation of time, with an accuracy of 0.46 ns. When Bit 9 is set in MAC_Timestamp_Control, each bit represents 1 ns. The maximum value is 0x3B9A_C9FF after which it rolls-over to zero. **Value After Reset:** 0x0 **Exists:** Always

17.1.191 MAC_System_Time_Seconds_Update

- **Description:** The System Time Seconds Update register, along with the System Time Nanoseconds Update register, initializes or updates the system time maintained by the MAC. You must write both registers before setting the TSINIT or TSUPDT bits in DWC_eqos_top_map/EQOS_MAC/MAC_-Timestamp_Control.
- **Size:** 32 bits
- **Offset:** 0xb10
- **Exists:** (DWC_EQOS_TIME_STAMPING&&DWC_EQOS_SYSTIME_SOURCE!=1)

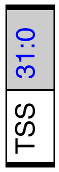


Table 17-195 Fields for Register: MAC_System_Time_Seconds_Update

Bits	Name	Memory Access	Description
31:0	TSS	R/W	Timestamp Seconds The value in this field is the seconds part of the update. When ADDSUB is reset, this field must be programmed with the seconds part of the update value. When ADDSUB is set, this field must be programmed with the complement of the seconds part of the update value. For example, if 2.000000001 seconds need to be subtracted from the system time, the TSS field in the MAC_Timestamp_Seconds_Update register must be 0xFFFF_FFFE (that is, 2^32 - 2). **Value After Reset:** 0x0 **Exists:** Always

17.1.192 MAC_System_Time_Nanoseconds_Update

- **Description:** MAC System Time Nanoseconds Update register.
- **Size:** 32 bits
- **Offset:** 0xb14
- **Exists:** (DWC_EQOS_TIME_STAMPING&&DWC_EQOS_SYSTIME_SOURCE!=1)

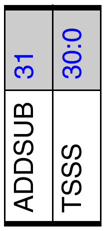


Table 17-196 Fields for Register: MAC_System_Time_Nanoseconds_Update

Bits	Name	Memory Access	Description
31	ADDSUB	R/W	Add or Subtract Time When this bit is set, the time value is subtracted with the contents of the update register. When this bit is reset, the time value is added with the contents of the update register. **Values:** ■ 0x0 (ADD): Add time ■ 0x1 (SUB): Subtract time **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
30:0	TSSS	R/W	Timestamp Sub Seconds The value in this field is the sub-seconds part of the update. When ADDSUB is reset, this field must be programmed with the sub-seconds part of the update value, with an accuracy based on the TSCTRLSSR bit of the MAC_Timestamp_Control register. When ADDSUB is set, this field must be programmed with the complement of the sub-seconds part of the update value as described below. When TSCTRLSSR bit in MAC_Timestamp_Control is set, the programmed value must be 10^9 - <sub-second value>. When TSCTRLSSR bit in MAC_Timestamp_Control is reset, the programmed value must be 2^31 - <sub-second_value>. When the TSCTRLSSR bit is reset in the MAC_Timestamp_Control register, each bit represents an accuracy of 0.46 ns. When the TSCTRLSSR bit is set in the MAC_Timestamp_Control register, each bit represents 1 ns and the programmed value should not exceed 0x3B9A_C9FF. For example, if 2.000000001 seconds need to be subtracted from the system time, then the TSSS field in the MAC_Timestamp_Nanoseconds_Update register must be 0x7FFF_FFFF (that is, 2^31 - 1), when TSCTRLSSR bit in MAC_Timestamp_Control is reset and 0x3B9A_C9FF (that is, 10^9 - 1), when TSCTRLSSR bit in MAC_Timestamp_Control is set. **Value After Reset:** 0x0 **Exists:** Always

17.1.193 MAC_Timestamp_Addend

- **Description:** Timestamp Addend register. This register value is used only when the system time is configured for Fine Update mode (TSCFUPDT bit in the MAC_Timestamp_Control register). The content of this register is added to a 32-bit accumulator in every clock cycle (of clk_ptp_ref_i) and the system time is updated whenever the accumulator overflows.
- **Size:** 32 bits
- **Offset:** 0xb18
- **Exists:** (DWC_EQOS_TIME_STAMPING&&DWC_EQOS_SYSTIME_SOURCE!=1)

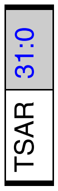


Table 17-197 Fields for Register: MAC_Timestamp_Addend

Bits	Name	Memory Access	Description
31:0	TSAR	R/W	Timestamp Addend Register This field indicates the 32-bit time value to be added to the Accumulator register to achieve time synchronization. **Value After Reset:** 0x0 **Exists:** Always

17.1.194 MAC_System_Time_Higher_Word_Seconds

- **Description:** System Time - Higher Word Seconds register.
- **Size:** 32 bits
- **Offset:** 0xb1c
- **Exists:** (DWC_EQOS_ADV_TIME_HIGH_WORD)

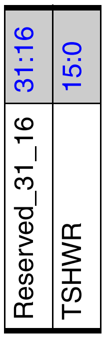


Table 17-198 Fields for Register: MAC_System_Time_Higher_Word_Seconds

Bits	Name	Memory Access	Description
31:16	Reserved_31_16	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
15:0	TSHWR	R/W	Timestamp Higher Word Register This field contains the most-significant 16-bits of timestamp seconds value. This register is optional. You can add this register by selecting the Add IEEE 1588 Higher Word Register option. This register is directly written to initialize the value and it is incremented when there is an overflow from 32-bits of the System Time Seconds register. Access restriction applies. Updated based on the event. Setting 1 sets. Setting 0 clears. **Value After Reset:** 0x0 **Exists:** Always

17.1.195 MAC_Timestamp_Status

- **Description:** Timestamp Status register. All bits except Bits[27:25] gets cleared when the application reads this register.
- **Size:** 32 bits
- **Offset:** 0xb20
- **Exists:** ((DWC_EQOS_TIME_STAMPING&&(DWC_EQOS_SYSTIME_-SOURCE!=1||!DWC_EQOS_CORE))||DWC_EQOS_ADV_TIME_AUX_SNAP||DWC_EQOS_P-TO_EN)

31:30	29:25	24	23:20	19:16	15	14:10	9	8	7	6	5	4	3	2	1	0
Reserved_31_30	ATSNS	ATSSTM	Reserved_23_20	ATSSTN	TXTSSIS	Reserved_14_10	TSTRGTERR3	TSTARGT3	TSTRGTERR2	TSTARGT2	TSTRGTERR1	TSTARGT1	TSTRGTERR0	AUXTSTRIG	TSTARGT0	TSSOVF

Table 17-199 Fields for Register: MAC_Timestamp_Status

Bits	Name	Memory Access	Description
31:30	Reserved_31_30	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
29:25	ATSNS	R	Number of Auxiliary Timestamp Snapshots This field indicates the number of Snapshots available in the FIFO. A value equal to the selected depth of FIFO (4, 8, or 16) indicates that the Auxiliary Snapshot FIFO is full. These bits are cleared (to 00000) when the Auxiliary snapshot FIFO clear bit is set. This bit is valid only if the Add IEEE 1588 Auxiliary Snapshot option is selected. **Value After Reset:** 0x0 **Exists:** DWC_EQOS_ADV_TIME_AUX_SNAP

Bits	Name	Memory Access	Description
24	ATSSTM	R	Auxiliary Timestamp Snapshot Trigger Missed This bit is set when the Auxiliary timestamp snapshot FIFO is full and external trigger was set. This indicates that the latest snapshot is not stored in the FIFO. This bit is valid only if the Add IEEE 1588 Auxiliary Snapshot option is selected. **Values:** ■ 0x0 (INACTIVE): Auxiliary Timestamp Snapshot Trigger Missed status not detected ■ 0x1 (ACTIVE): Auxiliary Timestamp Snapshot Trigger Missed status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_ADV_TIME_AUX_SNAP
23:20	Reserved_23_20	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
19:16	ATSSTN	R	Auxiliary Timestamp Snapshot Trigger Identifier These bits identify the Auxiliary trigger inputs for which the timestamp available in the Auxiliary Snapshot Register is applicable. When more than one bit is set at the same time, it means that corresponding auxiliary triggers were sampled at the same clock. These bits are applicable only if the number of Auxiliary snapshots is more than one. One bit is assigned for each trigger as shown in the following list: ■ Bit 16: Auxiliary trigger 0 ■ Bit 17: Auxiliary trigger 1 ■ Bit 18: Auxiliary trigger 2 ■ Bit 19: Auxiliary trigger 3 The software can read this register to find the triggers that are set when the timestamp is taken. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Value After Reset:** 0x0 **Exists:** DWC_EQOS_AUX_SNAP_IN_NUM>1

Bits	Name	Memory Access	Description
15	TXTSSIS	R	Tx Timestamp Status Interrupt Status In non-EQOS_CORE configurations when drop transmit status is enabled in MTL, this bit is set when the captured transmit timestamp is updated in the MAC_Tx_Timestamp_Status_Nanoseconds and MAC_Tx_Timestamp_Status_Seconds registers. When PTP offload feature is enabled, this bit is set when the captured transmit timestamp is updated in the MAC_Tx_Timestamp_Status_Nanoseconds and MAC_Tx_Timestamp_Status_Seconds registers, for PTO generated Delay Request and Pdelay request packets. This bit is cleared when the MAC_Tx_Timestamp_Status_Seconds register is read (or write to MAC_Tx_Timestamp_Status_Seconds register when RCWE bit of MAC_CSR_SW_Ctrl register is set). **Values:** ■ 0x0 (INACTIVE): Tx Timestamp Status Interrupt status not detected ■ 0x1 (ACTIVE): Tx Timestamp Status Interrupt status detected **Value After Reset:** 0x0 **Exists:** !DWC_EQOS_CORE||DWC_EQOS_PTO_EN
14:10	Reserved_14_10	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
9	TSTRGTERR3	R	Timestamp Target Time Error This bit is set when the latest target time programmed in the MAC_PPS3_Target_Time_Seconds and MAC_PPS3_Target_Time_Nanoseconds registers elapses. This bit is cleared when the application reads this bit. Access restriction applies. Clears on read (or this bit is written to 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): Timestamp Target Time Error status not detected ■ 0x1 (ACTIVE): Timestamp Target Time Error status detected **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_PPS_OUT_NUM>3)

Bits	Name	Memory Access	Description
8	TSTARGT3	R	Timestamp Target Time Reached for Target Time PPS3 When this bit is set, it indicates that the value of system time is greater than or equal to the value specified in the MAC_PPS3_Target_Time_Seconds and MAC_PPS3_Target_Time_Nanoseconds registers. Access restriction applies. Clears on read (or this bit is written to 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): Timestamp Target Time Reached for Target Time PPS3 status not detected ■ 0x1 (ACTIVE): Timestamp Target Time Reached for Target Time PPS3 status detected **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_PPS_OUT_NUM>3)
7	TSTRGTERR2	R	Timestamp Target Time Error This bit is set when the latest target time programmed in the MAC_PPS2_Target_Time_Seconds and MAC_PPS2_Target_Time_Nanoseconds registers elapses. This bit is cleared when the application reads this bit. Access restriction applies. Clears on read (or this bit is written to 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): Timestamp Target Time Error status not detected ■ 0x1 (ACTIVE): Timestamp Target Time Error status detected **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_PPS_OUT_NUM>2)
6	TSTARGT2	R	Timestamp Target Time Reached for Target Time PPS2 When set, this bit indicates that the value of system time is greater than or equal to the value specified in the MAC_PPS2_Target_Time_Seconds and MAC_PPS2_Target_Time_Nanoseconds registers. Access restriction applies. Clears on read (or this bit is written to 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): Timestamp Target Time Reached for Target Time PPS2 status not detected ■ 0x1 (ACTIVE): Timestamp Target Time Reached for Target Time PPS2 status detected **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_PPS_OUT_NUM>2)

Bits	Name	Memory Access	Description
5	TSTRGTERR1	R	Timestamp Target Time Error This bit is set when the latest target time programmed in the MAC_PPS1_Target_Time_Seconds and MAC_PPS1_Target_Time_Nanoseconds registers elapses. This bit is cleared when the application reads this bit. Access restriction applies. Clears on read (or this bit is written to 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): Timestamp Target Time Error status not detected ■ 0x1 (ACTIVE): Timestamp Target Time Error status detected **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_PPS_OUT_NUM>1)
4	TSTARGT1	R	Timestamp Target Time Reached for Target Time PPS1 When set, this bit indicates that the value of system time is greater than or equal to the value specified in the MAC_PPS1_Target_Time_Seconds and MAC_PPS1_Target_Time_Nanoseconds registers. Access restriction applies. Clears on read (or this bit is written to 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): Timestamp Target Time Reached for Target Time PPS1 status not detected ■ 0x1 (ACTIVE): Timestamp Target Time Reached for Target Time PPS1 status detected **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_PPS_OUT_NUM>1)
3	TSTRGTERR0	R	Timestamp Target Time Error This bit is set when the latest target time programmed in the MAC_PPS0_Target_Time_Seconds and MAC_PPS0_Target_Time_Nanoseconds registers elapses. This bit is cleared when the application reads this bit. Access restriction applies. Clears on read (or this bit is written to 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): Timestamp Target Time Error status not detected ■ 0x1 (ACTIVE): Timestamp Target Time Error status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_SYSTIME_SOURCE!=1

Bits	Name	Memory Access	Description
2	AUXTSTRIG	R	Auxiliary Timestamp Trigger Snapshot This bit is set high when the auxiliary snapshot is written to the FIFO. This bit is valid only if the Add IEEE 1588 Auxiliary Snapshot option is selected. Access restriction applies. Clears on read (or this bit is written to 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): Auxiliary Timestamp Trigger Snapshot status not detected ■ 0x1 (ACTIVE): Auxiliary Timestamp Trigger Snapshot status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_ADV_TIME_AUX_SNAP
1	TSTARGT0	R	Timestamp Target Time Reached When set, this bit indicates that the value of system time is greater than or equal to the value specified in the MAC_PPS0_Target_Time_Seconds and MAC_PPS0_Target_Time_Nanoseconds registers. Access restriction applies. Clears on read (or this bit is written to 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): Timestamp Target Time Reached status not detected ■ 0x1 (ACTIVE): Timestamp Target Time Reached status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_SYSTIME_SOURCE!=1
0	TSSOVF	R	Timestamp Seconds Overflow When this bit is set, it indicates that the seconds value of the timestamp (when supporting version 2 format) has overflowed beyond 32'hFFFF_FFFF. Access restriction applies. Clears on read (or this bit is written to 1 when RCWE bit in MAC_CSR_SW_Ctrl register is set). Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): Timestamp Seconds Overflow status not detected ■ 0x1 (ACTIVE): Timestamp Seconds Overflow status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_SYSTIME_SOURCE!=1

17.1.196 MAC_Tx_Timestamp_Status_Nanoseconds

- **Description:** This register contains the nanosecond part of timestamp captured for Transmit packets when Tx status is disabled.
  The MAC_Tx_Timestamp_Status_Nanoseconds register, along with MAC_Tx_Timestamp_Status_Seconds, gives the 64-bit timestamp captured for the PTP packet successfully transmitted by the MAC. This value is considered to be read by the application when the last byte of MAC_Tx_Timestamp_Status_Nanoseconds is read. In the little-endian mode, this means when bits[31:24] are read; in big-endian mode, bits[7:0] are read.
  If the application does not read these registers and timestamp of another packet is captured, then either the current timestamp is lost (overwritten) or the new timestamp is lost (dropped), depending on the setting of the TXTSSTSM bit of the MAC_Timestamp_Control register. The status bit TXTSC bit [15] in MAC_Timestamp_Status register is set whenever the MAC transmitter captures the timestamp.
- **Size:** 32 bits
- **Offset:** 0xb30
- **Exists:** ((DWC_EQOS_TIME_STAMPING&&!DWC_EQOS_CORE)||DWC_EQOS_PTO_EN)

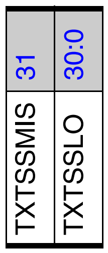


Table 17-200 Fields for Register: MAC_Tx_Timestamp_Status_Nanoseconds

Bits	Name	Memory Access	Description
31	TXTSSMIS	R	Transmit Timestamp Status Missed When this bit is set, it indicates one of the following: ■ The timestamp of the current packet is ignored if TXTSSTSM bit of the MAC_Timestamp_Control register is reset ■ The timestamp of the previous packet is overwritten with timestamp of the current packet if TXTSSTSM bit of the MAC_Timestamp_Control register is set. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): Transmit Timestamp Status Missed status not detected ■ 0x1 (ACTIVE): Transmit Timestamp Status Missed status detected **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
30:0	TXTSSLO	R	Transmit Timestamp Status Low This field contains the 31 bits of the Nanoseconds field of the Transmit packet's captured timestamp. **Value After Reset:** 0x0 **Exists:** Always

17.1.197 MAC_Tx_Timestamp_Status_Seconds

- **Description:** The register contains the higher 32 bits of the timestamp (in seconds) captured when a PTP packet is transmitted.
- **Size:** 32 bits
- **Offset:** 0xb34
- **Exists:** ((DWC_EQOS_TIME_STAMPING&&!DWC_EQOS_CORE) | |DWC_EQOS_PTO_EN)

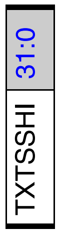


Table 17-201 Fields for Register: MAC_Tx_Timestamp_Status_Seconds

Bits	Name	Memory Access	Description
31:0	TXTSSHI	R	Transmit Timestamp Status High This field contains the lower 32 bits of the Seconds field of Transmit packet's captured timestamp. **Value After Reset:** 0x0 **Exists:** Always

17.1.198 MAC_Auxiliary_Control

- **Description:** The Auxiliary Timestamp Control register controls the Auxiliary Timestamp snapshot.
- **Size:** 32 bits
- **Offset:** 0xb40
- **Exists:** (DWC_EQOS_ADV_TIME_AUX_SNAP)

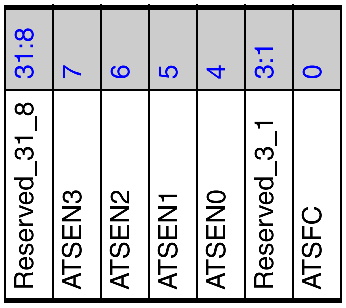


Table 17-202 Fields for Register: MAC_Auxiliary_Control

Bits	Name	Memory Access	Description
31:8	Reserved_31_8	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
7	ATSEN3	R/W	Auxiliary Snapshot 3 Enable This bit controls the capturing of Auxiliary Snapshot Trigger 3. When this bit is set, the auxiliary snapshot of the event on ptp_aux_trig_i[3] input is enabled. When this bit is reset, the events on this input are ignored. **Values:** ■ 0x0 (DISABLE): Auxiliary Snapshot $i is disabled ■ 0x1 (ENABLE): Auxiliary Snapshot $i is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_AUX_SNAP_IN_NUM>3
6	ATSEN2	R/W	Auxiliary Snapshot 2 Enable This bit controls the capturing of Auxiliary Snapshot Trigger 2. When this bit is set, the auxiliary snapshot of the event on ptp_aux_trig_i[2] input is enabled. When this bit is reset, the events on this input are ignored. **Values:** ■ 0x0 (DISABLE): Auxiliary Snapshot $i is disabled ■ 0x1 (ENABLE): Auxiliary Snapshot $i is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_AUX_SNAP_IN_NUM>2

Bits	Name	Memory Access	Description
5	ATSEN1	R/W	Auxiliary Snapshot 1 Enable This bit controls the capturing of Auxiliary Snapshot Trigger 1. When this bit is set, the auxiliary snapshot of the event on ptp_aux_trig_i[1] input is enabled. When this bit is reset, the events on this input are ignored. **Values:** ■ 0x0 (DISABLE): Auxiliary Snapshot $i is disabled ■ 0x1 (ENABLE): Auxiliary Snapshot $i is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_AUX_SNAP_IN_NUM>1
4	ATSEN0	R/W	Auxiliary Snapshot 0 Enable This bit controls the capturing of Auxiliary Snapshot Trigger 0. When this bit is set, the auxiliary snapshot of the event on ptp_aux_trig_i[0] input is enabled. When this bit is reset, the events on this input are ignored. **Values:** ■ 0x0 (DISABLE): Auxiliary Snapshot $i is disabled ■ 0x1 (ENABLE): Auxiliary Snapshot $i is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_AUX_SNAP_IN_NUM>0
3:1	Reserved_3_1	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
0	ATSFC	R/W	Auxiliary Snapshot FIFO Clear When set, this bit resets the pointers of the Auxiliary Snapshot FIFO. This bit is cleared when the pointers are reset and the FIFO is empty. When this bit is high, the auxiliary snapshots are stored in the FIFO. Access restriction applies. Setting 1 sets. Self-cleared. Setting 0 has no effect. **Values:** ■ 0x0 (DISABLE): Auxiliary Snapshot FIFO Clear is disabled ■ 0x1 (ENABLE): Auxiliary Snapshot FIFO Clear is enabled **Value After Reset:** 0x0 **Exists:** Always **Testable:** untestable

17.1.199 MAC_Auxiliary_Timestamp_Nanoseconds

- **Description:** The Auxiliary Timestamp Nanoseconds register, along with MAC_Auxiliary_Timestamp_Seconds, gives the 64-bit timestamp stored as auxiliary snapshot. These two registers form the read port of a 64-bit wide FIFO with a depth of 4, 8, or 16 as selected while configuring the core. You can store multiple snapshots in this FIFO. Bits[29:25] in MAC_Timestamp_Status indicate the fill-level of the FIFO. The top of the FIFO is removed only when the last byte of MAC_Auxiliary_Timestamp_Seconds register is read. In the little-endian mode, this means when Bits[31:24] are read and in big-endian mode, Bits[7:0] are read.
- **Size:** 32 bits
- **Offset:** 0xb48
- **Exists:** (DWC_EQOS_ADV_TIME_AUX_SNAP)

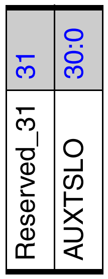


Table 17-203 Fields for Register: MAC_Auxiliary_Timestamp_Nanoseconds

Bits	Name	Memory Access	Description
31	Reserved_31	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
30:0	AUXTSLO	R	Auxiliary Timestamp Contains the lower 31 bits (nanoseconds field) of the auxiliary timestamp. **Value After Reset:** 0x0 **Exists:** Always

17.1.200 MAC_Auxiliary_Timestamp_Seconds

- **Description:** The Auxiliary Timestamp - Seconds register contains the lower 32 bits of the Seconds field of the auxiliary timestamp register.
- **Size:** 32 bits
- **Offset:** 0xb4c
- **Exists:** (DWC_EQOS_ADV_TIME_AUX_SNAP)

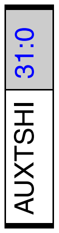


Table 17-204 Fields for Register: MAC_Auxiliary_Timestamp_Seconds

Bits	Name	Memory Access	Description
31:0	AUXTSHI	R	Auxiliary Timestamp Contains the lower 32 bits of the Seconds field of the auxiliary timestamp. **Value After Reset:** 0x0 **Exists:** Always

17.1.201 MAC_Timestamp_Ingress_Asym_Corr

- **Description:** The MAC Timestamp Ingress Asymmetry Correction register contains the Ingress Asymmetry Correction value to be used while updating correction field in PDelay_Resp PTP messages.
- **Size:** 32 bits
- **Offset:** 0xb50
- **Exists:** (DWC_EQOS_OST_EN)

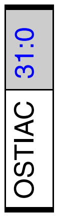


Table 17-205 Fields for Register: MAC_Timestamp_Ingress_Asym_Corr

Bits	Name	Memory Access	Description
31:0	OSTIAC	R/W	One-Step Timestamp Ingress Asymmetry Correction This field contains the ingress path asymmetry value to be added to correctionField of Pdelay_Resp PTP packet. The programmed value should be in units of nanoseconds and multiplied by 2^16. For example, 2.5 ns is represented as 0x00028000. The value can also be negative, which is represented in 2's complement form with bit 31 representing the sign bit. **Value After Reset:** 0x0 **Exists:** Always

17.1.202 MAC_Timestamp_Egress_Asym_Corr

- **Description:** The MAC Timestamp Egress Asymmetry Correction register contains the Egress Asymmetry Correction value to be used while updating the correction field in PDelay_Req PTP messages.
- **Size:** 32 bits
- **Offset:** 0xb54
- **Exists:** (DWC_EQOS_OST_EN)

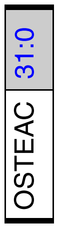


Table 17-206 Fields for Register: MAC_Timestamp_Egress_Asym_Corr

Bits	Name	Memory Access	Description
31:0	OSTEAC	R/W	One-Step Timestamp Egress Asymmetry Correction This field contains the egress path asymmetry value to be subtracted from correctionField of Pdelay_Resp PTP packet. The programmed value must be the negated value in units of nanoseconds multiplied by 2^16. For example, if the required correction is +2.5 ns, the programmed value must be 0xFFFD_8000, which is the 2's complement of 0x0002_8000(2.5 * 216). Similarly, if the required correction is -3.3 ns, the programmed value is 0x0003_4CCC (3.3 * 216). **Value After Reset:** 0x0 **Exists:** Always

17.1.203 MAC_Timestamp_Ingress_Corr_Nanosecond

- **Description:** This register contains the correction value in nanoseconds to be used with the captured timestamp value in the ingress path.
- **Size:** 32 bits
- **Offset:** 0xb58
- **Exists:** DWC_EQOS_TIME_STAMPING

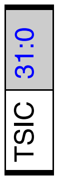


Table 17-207 Fields for Register: MAC_Timestamp_Ingress_Corr_Nanosecond

Bits	Name	Memory Access	Description
31:0	TSIC	R/W	Timestamp Ingress Correction This field contains the ingress path correction value as defined by the Ingress Correction expression. **Value After Reset:** 0x0 **Exists:** Always

17.1.204 MAC_Timestamp_Egress_Corr_Nanosecond

- **Description:** This register contains the correction value in nanoseconds to be used with the captured timestamp value in the egress path.
- **Size:** 32 bits
- **Offset:** 0xb5c
- **Exists:** DWC_EQOS_TIME_STAMPING

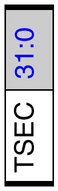


Table 17-208 Fields for Register: MAC_Timestamp_Egress_Corr_Nanosecond

Bits	Name	Memory Access	Description
31:0	TSEC	R/W	Timestamp Egress Correction This field contains the nanoseconds part of the egress path correction value as defined by the Egress Correction expression. **Value After Reset:** 0x0 **Exists:** Always

17.1.205 MAC_Timestamp_Ingress_Corr_Subnanosec

- **Description:** This register contains the sub-nanosecond part of the correction value to be used with the captured timestamp value, for ingress direction.
- **Size:** 32 bits
- **Offset:** 0xb60
- **Exists:** DWC_EQOS_ADV_SUBNSEC_SUPPORT

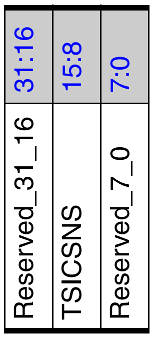


Table 17-209 Fields for Register: MAC_Timestamp_Ingress_Corr_Subnanosec

Bits	Name	Memory Access	Description
31:16	Reserved_31_16	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
15:8	TSICSNS	R/W	Timestamp Ingress Correction, sub-nanoseconds This field contains the sub-nanoseconds part of the ingress path correction value as defined by the "Ingress Correction" expression. **Value After Reset:** 0x0 **Exists:** Always
7:0	Reserved_7_0	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always

17.1.206 MAC_Timestamp_Egress_Corr_Subnanosec

- **Description:** This register contains the sub-nanosecond part of the correction value to be used with the captured timestamp value, for egress direction.
- **Size:** 32 bits
- **Offset:** 0xb64
- **Exists:** DWC_EQOS_ADV_SUBNSEC_SUPPORT

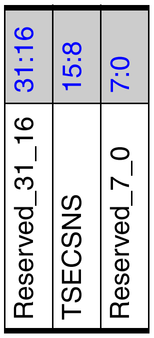


Table 17-210 Fields for Register: MAC_Timestamp_Egress_Corr_Subnanosec

Bits	Name	Memory Access	Description
31:16	Reserved_31_16	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
15:8	TSECSNS	R/W	Timestamp Egress Correction, sub-nanoseconds This field contains the sub-nanoseconds part of the egress path correction value as defined by the "Egress Correction" expression. **Value After Reset:** 0x0 **Exists:** Always
7:0	Reserved_7_0	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always

17.1.207 MAC_Timestamp_Ingress_Latency

- **Description:** This register holds the Ingress MAC latency.
- **Size:** 32 bits
- **Offset:** 0xb68
- **Exists:** DWC_EQOS_TIME_STAMPING

31:28	27:16	15:8	7:0
Reserved_31_28	ITLNS	ITLSNS	Reserved_7_0

Table 17-211 Fields for Register: MAC_Timestamp_Ingress_Latency

Bits	Name	Memory Access	Description
31:28	Reserved_31_28	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
27:16	ITLNS	R	Ingress Timestamp Latency, in sub-nanoseconds This register holds the average latency in sub-nanoseconds between the input ports (phy_rxd_i) of MAC and the actual point (GMII/MII) where the ingress timestamp is taken. Ingress correction value is computed as described in the section 7.1.2.4.1 of QoS Databook. **Value After Reset:** 0x0 **Exists:** Always **Testable:** untestable
15:8	ITLSNS	R	Ingress Timestamp Latency, in nanoseconds This register holds the average latency in nanoseconds between the input ports (phy_rxd_i) of MAC and the actual point (GMII/MII) where the ingress timestamp is taken. Ingress correction value is computed as described in the section 7.1.2.4.1 of QoS Databook. **Value After Reset:** 0x0 **Exists:** Always **Testable:** untestable
7:0	Reserved_7_0	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always

17.1.208 MAC_Timestamp_Egress_Latency

- **Description:** This register holds the Egress MAC latency.
- **Size:** 32 bits
- **Offset:** 0xb6c
- **Exists:** DWC_EQOS_TIME_STAMPING

31:28	27:16	15:8	7:0
Reserved_31_28	ETLNS	ETLSNS	Reserved_7_0

Table 17-212 Fields for Register: MAC_Timestamp_Egress_Latency

Bits	Name	Memory Access	Description
31:28	Reserved_31_28	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
27:16	ETLNS	R	Egress Timestamp Latency, in nanoseconds This register holds the average latency in nanoseconds between the actual point (GMII/MII) where the egress timestamp is taken and the output ports (phy_txd_o) of the MAC. Ingress correction value is computed as described in the section 7.1.2.4.2 of QoS Databook. **Value After Reset:** 0x0 **Exists:** Always **Testable:** untestable
15:8	ETLSNS	R	Egress Timestamp Latency, in sub-nanoseconds This register holds the average latency in sub-nanoseconds between the actual point (GMII/MII) where the egress timestamp is taken and the output ports (phy_txd_o) of the MAC. Ingress correction value is computed as described in the section 7.1.2.4.2 of QoS Databook. **Value After Reset:** 0x0 **Exists:** Always **Testable:** untestable
7:0	Reserved_7_0	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always

17.1.209 MAC_PPS_Control

- **Description:** PPS Control register.
  Bits[30:24] of this register are valid only when four Flexible PPS outputs are selected. Bits[22:16] are valid only when three or more Flexible PPS outputs are selected. Bits[14:8] are valid only when two or more Flexible PPS outputs are selected. Bits[6:4] are valid only when Flexible PPS feature is selected.
- **Size:** 32 bits
- **Offset:** 0xb70
- **Exists:** (DWC_EQOS_TIME_STAMPING&&DWC_EQOS_SYSTIME_SOURCE!=1)

31	30:29	28	27:24	23	22:21	20	19:16	15	14:13	12	11:8	7	6:5	4	3:0
MCGREN3	TRGTMODSEL3	Reserved_28	PPSCMD3	MCGREN2	TRGTMODSEL2	Reserved_20	PPSCMD2	MCGREN1	TRGTMODSEL1	Reserved_12	PPSCMD1	MCGREN0	TRGTMODSEL0	PPSEN0	PPSCTRL_PPSCMD

Table 17-213 Fields for Register: MAC_PPS_Control

Bits	Name	Memory Access	Description
31	MCGREN3	R/W	MCGR Mode Enable for PPS3 Output This field enables the 3rd PPS instance to operate in PPS or MCGR mode. When set it operates in MCGR mode and on reset it operates in PPS mode. **Value After Reset:** 0x0 **Exists:** DWC_EQOS_FLEXI_PPS_OUT_EN&&DWC_EQOS_PPS_OUT_3_EN

Bits	Name	Memory Access	Description
30:29	TRGTMODSEL3	R/W	Target Time Register Mode for PPS3 Output This field indicates the Target Time registers (MAC_PPS3_Target_Time_Seconds and MAC_PPS3_Target_Time_Nanoseconds) mode for PPS3 output signal. **Values:** ■ 0x0 (ONLY_INT): Target Time registers are programmed only for generating the interrupt event. The Flexible PPS function must not be enabled in this mode, otherwise spurious transitions may be observed on the corresponding ptp_pps_o output port ■ 0x1 (RSVD): Reserved ■ 0x2 (INT_ST): Target Time registers are programmed for generating the interrupt event and starting or stopping the PPS0 output signal generation ■ 0x3 (ONLY_ST): Target Time registers are programmed only for starting or stopping the PPS0 output signal generation. No interrupt is asserted **Value After Reset:** 0x0 **Exists:** DWC_EQOS_PPS_OUT_NUM==4
28	Reserved_28	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
27:24	PPSCMD3	R/W	Flexible PPS3 Output Control This field controls the flexible PPS3 output (ptp_pps_o[3]) signal. This field is similar to the PPSCMD0[2:0] field. If MCGREN3 is set, then PPSCMD3 indicated by these 4 bits [27:24] are taken as Presentation Time Control bits for media clock generation and recovery for comparator instance 3. This field is similar to the PPSCMD0 Presentation Time Control bits. If MCGREN3 is not set then only 3 bits from [26:24] is used as PPSCMD3 and the 4th bit is to be set as 0. Access restriction applies. Setting 1 sets. Self-cleared. Setting 0 has no effect. **Value After Reset:** 0x0 **Exists:** DWC_EQOS_PPS_OUT_NUM==4 **Testable:** untestable

Bits	Name	Memory Access	Description
23	MCGREN2	R/W	MCGR Mode Enable for PPS2 Output This field enables the 2nd PPS instance to operate in PPS or MCGR mode. When set it operates in MCGR mode and on reset it operates in PPS mode. **Values:** ■ 0x0 (DISABLE): 2nd PPS instance is disabled to operate in PPS or MCGR mode ■ 0x1 (ENABLE): 2nd PPS instance is enabled to operate in PPS or MCGR mode **Value After Reset:** 0x0 **Exists:** DWC_EQOS_FLEXI_PPS_OUT_EN&&DWC_EQOS_PPS_OUT_2_EN
22:21	TRGTMODSEL2	R/W	Target Time Register Mode for PPS2 Output This field indicates the Target Time registers (MAC_PPS2_Target_Time_Seconds and MAC_PPS2_Target_Time_Nanoseconds) mode for PPS2 output signal. **Values:** ■ 0x0 (ONLY_INT): Target Time registers are programmed only for generating the interrupt event. The Flexible PPS function must not be enabled in this mode, otherwise spurious transitions may be observed on the corresponding ptp_pps_o output port ■ 0x1 (RSVD): Reserved ■ 0x2 (INT_ST): Target Time registers are programmed for generating the interrupt event and starting or stopping the PPS0 output signal generation ■ 0x3 (ONLY_ST): Target Time registers are programmed only for starting or stopping the PPS0 output signal generation. No interrupt is asserted **Value After Reset:** 0x0 **Exists:** DWC_EQOS_PPS_OUT_NUM>=3
20	Reserved_20	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
19:16	PPSCMD2	R/W	Flexible PPS2 Output Control This field controls the flexible PPS2 output (ptp_pps_o[2]) signal. This field is similar to the PPSCMD0 field. If MCGREN2 is set, then PPSCMD2 indicated by these 4 bits [19:16] are taken as Presentation Time Control bits for media clock generation and recovery for comparator instance 2. This field is similar to the PPSCMD0 Presentation Time Control bits. If MCGREN2 is not set then only 3 bits from [18:16] is used as PPSCMD2 and the 4th bit is to be set as 0. Access restriction applies. Setting 1 sets. Self-cleared. Setting 0 has no effect. **Value After Reset:** 0x0 **Exists:** DWC_EQOS_PPS_OUT_NUM>=3 **Testable:** untestable
15	MCGREN1	R/W	MCGR Mode Enable for PPS1 Output This field enables the 1st PPS instance to operate in PPS or MCGR mode. When set it operates in MCGR mode and on reset it operates in PPS mode. **Values:** ■ 0x0 (DISABLE): 1st PPS instance is disabled to operate in PPS or MCGR mode ■ 0x1 (ENABLE): 1st PPS instance is enabled to operate in PPS or MCGR mode **Value After Reset:** 0x0 **Exists:** DWC_EQOS_FLEXI_PPS_OUT_EN&&DWC_EQOS_PPS_OUT_1_EN
14:13	TRGTMODSEL1	R/W	Target Time Register Mode for PPS1 Output This field indicates the Target Time registers (MAC_PPS1_Target_Time_Seconds and MAC_PPS1_Target_Time_Nanoseconds) mode for PPS1 output signal. **Values:** ■ 0x0 (ONLY_INT): Target Time registers are programmed only for generating the interrupt event. The Flexible PPS function must not be enabled in this mode, otherwise spurious transitions may be observed on the corresponding ptp_pps_o output port ■ 0x1 (RSVD): Reserved ■ 0x2 (INT_ST): Target Time registers are programmed for generating the interrupt event and starting or stopping the PPS0 output signal generation ■ 0x3 (ONLY_ST): Target Time registers are programmed only for starting or stopping the PPS0 output signal generation. No interrupt is asserted **Value After Reset:** 0x0 **Exists:** DWC_EQOS_PPS_OUT_NUM>=2

Bits	Name	Memory Access	Description
12	Reserved_12	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
11:8	PPSCMD1	R/W	Flexible PPS1 Output Control This field controls the flexible PPS1 output (ptp_pps_o[1]) signal. This field is similar to the PPSCMD0 field. If MCGREN1 is set, then PPSCMD1 indicated by these 4 bits [11:8] are taken as Presentation Time Control bits for media clock generation and recovery for comparator instance 1. This field is similar to the PPSCMD0 Presentation Time Control bits. If MCGREN1 is not set then only 3 bits from [10:8] is used as PPSCMD1 and the 4th bit is to be set as 0. Access restriction applies. Setting 1 sets. Self-cleared. Setting 0 has no effect. **Value After Reset:** 0x0 **Exists:** DWC_EQOS_PPS_OUT_NUM>=2 **Testable:** untestable
7	MCGREN0	R/W	MCGR Mode Enable for PPS0 Output This field enables the 0th PPS instance to operate in PPS or MCGR mode. When set it operates in MCGR mode and on reset it operates in PPS mode. **Values:** ■ 0x0 (PPS): 0th PPS instance is enabled to operate in PPS mode ■ 0x1 (MCGR): 0th PPS instance is enabled to operate in MCGR mode **Value After Reset:** 0x0 **Exists:** DWC_EQOS_FLEXI_PPS_OUT_EN

Bits	Name	Memory Access	Description
6:5	TRGTMODSEL0	R/W	Target Time Register Mode for PPS0 Output This field indicates the Target Time registers (MAC_PPS0_Target_Time_Seconds and MAC_PPS0_Target_Time_Nanoseconds) mode for PPS0 output signal: **Values:** ■ 0x0 (ONLY_INT): Target Time registers are programmed only for generating the interrupt event. The Flexible PPS function must not be enabled in this mode, otherwise spurious transitions may be observed on the corresponding ptp_pps_o output port ■ 0x1 (RSVD): Reserved ■ 0x2 (INT_ST): Target Time registers are programmed for generating the interrupt event and starting or stopping the PPS0 output signal generation ■ 0x3 (ONLY_ST): Target Time registers are programmed only for starting or stopping the PPS0 output signal generation. No interrupt is asserted **Value After Reset:** 0x0 **Exists:** DWC_EQOS_FLEXI_PPS_OUT_EN
4	PPSEN0	R/W	Flexible PPS Output Mode Enable When this bit is set, Bits[3:0] function as PPSCMD. When this bit is reset, Bits[3:0] function as PPSCTRL (Fixed PPS mode). **Values:** ■ 0x0 (DISABLE): Flexible PPS Output Mode is disabled ■ 0x1 (ENABLE): Flexible PPS Output Mode is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_FLEXI_PPS_OUT_EN

Bits	Name	Memory Access	Description
3:0	PPSCTRL_PPSCMD	R/W	**PPS Output Frequency Control** This field controls the frequency of the PPS0 output (ptp_pps_o[0]) signal. The default value of PPSCTRL is 0000, and the PPS output is 1 pulse (of width clk_ptp_i) every second. For other values of PPSCTRL, the PPS output becomes a generated clock of following frequencies: ■ 0001: The binary rollover is 2 Hz, and the digital rollover is 1 Hz. ■ 0010: The binary rollover is 4 Hz, and the digital rollover is 2 Hz. ■ 0011: The binary rollover is 8 Hz, and the digital rollover is 4 Hz. ■ 0100: The binary rollover is 16 Hz, and the digital rollover is 8 Hz. ■ .. ■ 1111: The binary rollover is 32.768 KHz and the digital roll-over is 16.384 KHz. **Note**: In the binary rollover mode, the PPS output (ptp_pps_o) has a duty cycle of 50 percent with these frequencies. In the digital rollover mode, the PPS output frequency is an average number. The actual clock is of different frequency that gets synchronized every second. For example: ■ When PPSCTRL = 0001, the PPS (1 Hz) has a low period of 537 ms and a high period of 463 ms ■ When PPSCTRL = 0010, the PPS (2 Hz) is a sequence of One clock of 50 percent duty cycle and 537 ms period Second clock of 463 ms period (268 ms low and 195 ms high) ■ When PPSCTRL = 0011, the PPS (4 Hz) is a sequence of Three clocks of 50 percent duty cycle and 268 ms period Fourth clock of 195 ms period (134 ms low and 61 ms high) This behavior is because of the non-linear toggling of bits in the digital rollover mode in the MAC_System_Time_Nanoseconds register. *or* **Flexible PPS Output (ptp_pps_o[0]) Control**

Bits	Name	Memory Access	Description
3:0...(cont.)	PPSCTRL_PPSCMD.	R/W	Programming these bits with a non-zero value instructs the MAC to initiate an event. When the command is transferred or synchronized to the PTP clock domain, these bits get cleared automatically. The software should ensure that these bits are programmed only when they are 'all-zero'. The following list describes the values of PPSCMD0: ■ 0000: No Command ■ 0001: START Single Pulse This command generates single pulse rising at the start point defined in MAC_PPS0_Target_Time_Seconds and MAC_PPS0_Target_Time_Nanoseconds register and of a duration defined in the PPS0 Width Register. ■ 0010: START Pulse Train This command generates the train of pulses rising at the start point defined in the Target Time Registers and of a duration defined in the PPS0 Width Register and repeated at interval defined in the PPS Interval Register. By default, the PPS pulse train is free-running unless stopped by the 'Stop Pulse train at time' or 'Stop Pulse Train immediately' commands. ■ 0011: Cancel START This command cancels the START Single Pulse and START Pulse Train commands if the system time has not crossed the programmed start time. ■ 0100: STOP Pulse train at time This command stops the train of pulses initiated by the START Pulse Train command (PPSCMD = 0010) after the time programmed in the Target Time registers elapses. ■ 0101: STOP Pulse Train immediately This command immediately stops the train of pulses initiated by the START Pulse Train command (PPSCMD = 0010). ■ 110: Cancel STOP Pulse train This command cancels the STOP pulse train at time command if the programmed stop time has not elapsed. The PPS pulse train becomes free-running on the successful execution of this command. ■ 0111-1111: Reserved *or* **Presentation Time Control**

Bits	Name	Memory Access	Description
3:0...(cont..)	PPSCTRL_PPSCMD..	R/W	If MCGREN0 is set then these bits are treated as Presentation time control bits. The following list describes the values of PPSCMD0: ■ 0000: No Operation related to MCGR will be carried out. If set to this value in the mid of clock recovery or generation all the processing inputs will be flushed. ■ 0001: Capture the Presentation time at rising edge of mcg_pst_trig_i[0] into the MAC_PPS0_Target_Time_Seconds register ■ 0010: Capture the Presentation time at falling edge of mcg_pst_trig_i[0] into the MAC_PPS0_Target_Time_Seconds register ■ 0011: Capture the Presentation time at both edges of mcg_pst_trig_i[0] into the MAC_PPS0_Target_Time_Seconds register ■ 0100-1000: Reserved ■ 1001: Toggle output on compare ■ 1010: Pulse output low on compare for one PTP-clock cycle ■ 1011: Pulse output high on compare for one PTP-clock cycle ■ 1100-1111: Reserved **Value After Reset:** 0x0 **Exists:** Always

17.1.210 MAC_PPS(#i)_Target_Time_Seconds (for i = 0; i <= DWC_EQOS_PPS_OUT_NUM-1)

- **Description:** The PPS Target Time Seconds register, along with PPS Target Time Nanoseconds register, is used to schedule an interrupt event [Bit 1 of MAC_Timestamp_Status] when the system time exceeds the value programmed in these registers.
- **Size:** 32 bits
- **Offset:** (0x0010*i)+0x0B80
- **Exists:** (DWC_EQOS_TIME_STAMPING&&DWC_EQOS_SYSTIME_SOURCE!=1)

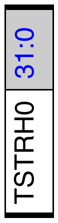


Table 17-214 Fields for Register: MAC_PPS(#i)_Target_Time_Seconds (for i = 0; i <= DWC_EQOS_PPS_OUT_NUM-1)

Bits	Name	Memory Access	Description
31:0	TSTRH0	R/W	PPS Target Time Seconds Register This field stores the time in seconds. When the timestamp value matches or exceeds both Target Timestamp registers, the MAC starts or stops the PPS signal output and generates an interrupt (if enabled) based on Target Time mode selected for the corresponding PPS output in the MAC_PPS_Control register. If DWC_EQOS_FLEXI_PPS_OUT_EN is enabled in the configuration and PTGE field of MAC_Timestamp_Control Register is set with Presentation time control set in recovery mode, then these bits indicate the TPT being programmed by the application and in generation mode it indicates the CPT generated at the sampled trigger. **Value After Reset:** 0x0 **Exists:** Always

17.1.211 MAC_PPS(#i)_Target_Time_Nanoseconds (for i = 0; i <= DWC_EQOS_PPS_OUT_NUM-1)

- **Description:** PPS0 Target Time Nanoseconds register.
- **Size:** 32 bits
- **Offset:** (0x0010*i)+0x0B84
- **Exists:** (DWC_EQOS_TIME_STAMPING&&DWC_EQOS_SYSTIME_SOURCE!=1)

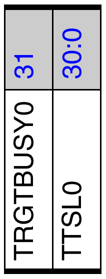


Table 17-215 Fields for Register: MAC_PPS(#i)_Target_Time_Nanoseconds (for i = 0; i <= DWC_EQOS_PPS_OUT_NUM-1)

Bits	Name	Memory Access	Description
31	TRGTBUSY0	R/W	PPS Target Time Register Busy The MAC sets this bit when the PPSCMD0 field in the MAC_PPS_Control register is programmed to 010 or 011. Programming the PPSCMD0 field to 010 or 011 instructs the MAC to synchronize the Target Time Registers to the PTP clock domain. The MAC clears this bit after synchronizing the Target Time Registers to the PTP clock domain The application must not update the Target Time Registers when this bit is read as 1. Otherwise, the synchronization of the previous programmed time gets corrupted. **Values:** ■ 0x0 (INACTIVE): PPS Target Time Register Busy status is not detected ■ 0x1 (ACTIVE): PPS Target Time Register Busy is detected **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_PPS_OUT_NUM>0) **Testable:** untestable

Bits	Name	Memory Access	Description
30:0	TTSL0	R/W	Target Time Low for PPS Register This register stores the time in (signed) nanoseconds. When the value of the timestamp matches the value in both Target Timestamp registers, the MAC starts or stops the PPS signal output and generates an interrupt (if enabled) based on the TRGTMODSEL0 field (Bits [6:5]) in MAC_PPS_Control. When the TSCTRLSSR bit is reset in the MAC_Timestamp_Control register, this value should be (time in ns / 0.465). The actual start or stop time of the PPS signal output may have an error margin up to one unit of sub-second increment value. When the TSCTRLSSR bit is set in the MAC_Timestamp_Control register, this value should not exceed 0x3B9A_C9FF. The actual start or stop time of the PPS signal output may have an error margin up to one unit of sub-second increment value. Access restriction applies. Setting 1 sets. Self-cleared. Setting 0 has no effect. **Value After Reset:** 0x0 **Exists:** Always

17.1.212 MAC_PPS(#i)_Interval (for i = 0; i <= DWC_EQOS_PPS_OUT_NUM-1)

- **Description:** The PPS0 Interval register contains the number of units of sub-second increment value between the rising edges of PPS0 signal output (ptp_pps_o[0]).
- **Size:** 32 bits
- **Offset:** (0x0010*i)+0x0B88
- **Exists:** (DWC_EQOS_PPS_OUT_NUM>0)

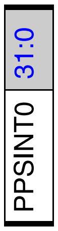


Table 17-216 Fields for Register: MAC_PPS(#i)_Interval (for i = 0; i <= DWC_EQOS_PPS_OUT_NUM-1)

Bits	Name	Memory Access	Description
31:0	PPSINT0	R/W	PPS Output Signal Interval These bits store the interval between the rising edges of PPS0 signal output. The interval is stored in terms of number of units of sub-second increment value. You need to program one value less than the required interval. For example, if the PTP reference clock is 50 MHz (period of 20 ns), and desired interval between the rising edges of PPS0 signal output is 100 ns (that is, 5 units of sub-second increment value), you should program value 4 (5-1) in this register. **Value After Reset:** 0x0 **Exists:** Always

17.1.213 MAC_PPS(#i)_Width (for i = 0; i <= DWC_EQOS_PPS_OUT_NUM-1)

- **Description:** The PPS0 Width register contains the number of units of sub-second increment value between the rising and corresponding falling edges of PPS0 signal output (ptp_pps_o[0]).
- **Size:** 32 bits
- **Offset:** (0x0010*i)+0x0B8C
- **Exists:** (DWC_EQOS_PPS_OUT_NUM>0)

Table 17-217 Fields for Register: MAC_PPS(#i)_Width (for i = 0; i <= DWC_EQOS_PPS_OUT_NUM-1)

Bits	Name	Memory Access	Description
31:0	PPSWIDTH0	R/W	PPS Output Signal Width These bits store the width between the rising edge and corresponding falling edge of PPS0 signal output. The width is stored in terms of number of units of sub-second increment value. You need to program one value less than the required interval. For example, if PTP reference clock is 50 MHz (period of 20 ns), and width between the rising and corresponding falling edges of PPS0 signal output is 80 ns (that is, four units of sub-second increment value), you should program value 3 (4-1) in this register. **Note**: The value programmed in this register must be lesser than the value programmed in MAC_PPS0_Interval. **Value After Reset:** 0x0 **Exists:** Always

17.1.214 MAC_PTO_Control

- **Description:** This register controls the PTP Offload Engine operation. This register is available only when the Enable PTP Timestamp Offload feature is selected.
- **Size:** 32 bits
- **Offset:** 0xbc0
- **Exists:** (DWC_EQOS_PTO_EN)

31:16	15:8	7	6	5	4	3	2	1	0
Reserved_31_16	DN	PDRDIS	DRRDIS	APDREQTRIG	ASYNCTRIG	Reserved_3	APDREQEN	ASYNCEN	PTOEN

Table 17-218 Fields for Register: MAC_PTO_Control

Bits	Name	Memory Access	Description
31:16	Reserved_31_16	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
15:8	DN	R/W	Domain Number This field indicates the domain Number in which the PTP node is operating. **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
7	PDRDIS	R/W	Disable Peer Delay Response response generation When this bit is set, the Peer Delay Response (Pdelay_Resp) response is not be generated for received Peer Delay Request (Pdelay_Req) request packet, as required by the programmed mode. Note: Setting this bit to 1 affects the normal PTP Offload operation and the time synchronization. So, this bit must be set only if there is problem with Pdelay_Resp generation in Hardware and/or Pdelay_Resp generation is handled by Software. **Values:** ■ 0x0 (ENABLE): Peer Delay Response response generation is enabled ■ 0x1 (DISABLE): Peer Delay Response response generation is disabled **Value After Reset:** 0x0 **Exists:** Always
6	DRRDIS	R/W	Disable PTO Delay Request/Response response generation When this bit is set, the Delay Request and Delay response is not generated for received SYNC and Delay request packet respectively, as required by the programmed mode. **Values:** ■ 0x0 (ENABLE): PTO Delay Request/Response response generation is enabled ■ 0x1 (DISABLE): PTO Delay Request/Response response generation is disabled **Value After Reset:** 0x0 **Exists:** Always
5	APDREQTRIG	R/W	Automatic PTP Pdelay_Req message Trigger When this bit is set, one PTP Pdelay_Req message is transmitted. This bit is automatically cleared after the PTP Pdelay_Req message is transmitted. The application should set the APDREQEN bit for this operation. Access restriction applies. Setting 1 sets. Self-cleared. Setting 0 has no effect. **Values:** ■ 0x0 (DISABLE): Automatic PTP Pdelay_Req message Trigger is disabled ■ 0x1 (ENABLE): Automatic PTP Pdelay_Req message Trigger is enabled **Value After Reset:** 0x0 **Exists:** Always **Testable:** untestable

Bits	Name	Memory Access	Description
4	ASYNCTRIG	R/W	Automatic PTP SYNC message Trigger When this bit is set, one PTP SYNC message is transmitted. This bit is automatically cleared after the PTP SYNC message is transmitted. The application should set the ASYNCEN bit for this operation. Access restriction applies. Setting 1 sets. Self-cleared. Setting 0 has no effect. **Values:** ■ 0x0 (DISABLE): Automatic PTP SYNC message Trigger is disabled ■ 0x1 (ENABLE): Automatic PTP SYNC message Trigger is enabled **Value After Reset:** 0x0 **Exists:** Always **Testable:** untestable
3	Reserved_3	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
2	APDREQEN	R/W	Automatic PTP Pdelay_Req message Enable When this bit is set, PTP Pdelay_Req message is generated periodically based on interval programmed or trigger from application, when the MAC is programmed to be in Peer-to-Peer Transparent mode. **Values:** ■ 0x0 (DISABLE): Automatic PTP Pdelay_Req message is disabled ■ 0x1 (ENABLE): Automatic PTP Pdelay_Req message is enabled **Value After Reset:** 0x0 **Exists:** Always
1	ASYNCEN	R/W	Automatic PTP SYNC message Enable When this bit is set, PTP SYNC message is generated periodically based on interval programmed or trigger from application, when the MAC is programmed to be in Clock Master mode. **Values:** ■ 0x0 (DISABLE): Automatic PTP SYNC message is disabled ■ 0x1 (ENABLE): Automatic PTP SYNC message is enabled **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
0	PTOEN	R/W	PTP Offload Enable When this bit is set, the PTP Offload feature is enabled. **Values:** ■ 0x0 (DISABLE): PTP Offload feature is disabled ■ 0x1 (ENABLE): PTP Offload feature is enabled **Value After Reset:** 0x0 **Exists:** Always

17.1.215 MAC_Source_Port_Identity0

- **Description:** This register contains Bits[31:0] of the 80-bit Source Port Identity of the PTP node. This register is available only when the Enable PTP Timestamp Offload feature is selected.
- **Size:** 32 bits
- **Offset:** 0xbc4
- **Exists:** (DWC_EQOS_PTO_EN)

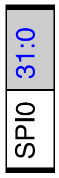


Table 17-219 Fields for Register: MAC_Source_Port_Identity0

Bits	Name	Memory Access	Description
31:0	SPI0	R/W	Source Port Identity 0 This field indicates bits [31:0] of sourcePortIdentity of PTP node. **Value After Reset:** 0x0 **Exists:** Always

17.1.216 MAC_Source_Port_Identity1

- **Description:** This register contains Bits[63:32] of the 80-bit Source Port Identity of the PTP node. This register is available only when the Enable PTP Timestamp Offload feature is selected.
- **Size:** 32 bits
- **Offset:** 0xbc8
- **Exists:** (DWC_EQOS_PTO_EN)

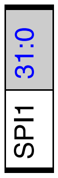


Table 17-220 Fields for Register: MAC_Source_Port_Identity1

Bits	Name	Memory Access	Description
31:0	SPI1	R/W	Source Port Identity 1 This field indicates bits [63:32] of sourcePortIdentity of PTP node. **Value After Reset:** 0x0 **Exists:** Always

17.1.217 MAC_Source_Port_Identity2

- **Description:** This register contains Bits[79:64] of the 80-bit Source Port Identity of the PTP node. This register is available only when the Enable PTP Timestamp Offload feature is selected.
- **Size:** 32 bits
- **Offset:** 0xbcc
- **Exists:** (DWC_EQOS_PTO_EN)

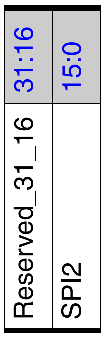


Table 17-221 Fields for Register: MAC_Source_Port_Identity2

Bits	Name	Memory Access	Description
31:16	Reserved_31_16	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
15:0	SPI2	R/W	Source Port Identity 2 This field indicates bits [79:64] of sourcePortIdentity of PTP node. **Value After Reset:** 0x0 **Exists:** Always

17.1.218 MAC_Log_Message_Interval

- **Description:** This register contains the periodic intervals for automatic PTP packet generation. This register is available only when the Enable PTP Timestamp Offload feature is selected.
- **Size:** 32 bits
- **Offset:** 0xbd0
- **Exists:** (DWC_EQOS_PTO_EN)

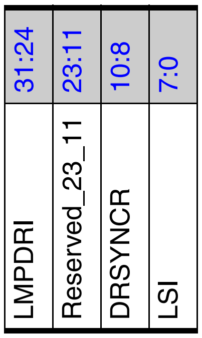


Table 17-222 Fields for Register: MAC_Log_Message_Interval

Bits	Name	Memory Access	Description
31:24	LMPDRI	R/W	Log Min Pdelay_Req Interval This field indicates logMinPdelayReqInterval of PTP node. This is used to schedule the periodic Pdelay request packet transmission. Allowed values are -15 to 15.Negative value must be represented in 2's-complement form. For example, if the required value is -1, the value programmed must be 0xFF. **Value After Reset:** 0x0 **Exists:** Always
23:11	Reserved_23_11	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
10:8	DRSYNCR	R/W	Delay_Req to SYNC Ratio In Slave mode, it is used for controlling frequency of Delay_Req messages transmitted. ■ 0: DelayReq generated for every received SYNC ■ 1: DelayReq generated every alternate reception of SYNC ■ 2: for every 4 SYNC messages ■ 3: for every 8 SYNC messages ■ 4: for every 16 SYNC messages ■ 5: for every 32 SYNC messages ■ 6-7: Reserved The master sends this information (logMinDelayReqInterval) in the DelayResp PTP messages to the slave. The DWC_ether_qos Receiver processes this value from the received DelayResp messages and updates this field accordingly. In the Slave mode, the host must not write/update this register unless it has to override the received value. In Master mode, the sum of this field and logSyncInterval (LSI) field is provided in the logMinDelayReqInterval field of the generated multicast Delay_Resp PTP message. Access restriction applies. Updated based on the event. Setting 1 sets. Setting 0 clears. **Values:** ■ 0x0 (SYNC1): DelayReq generated for every received SYNC ■ 0x1 (SYNC2): DelayReq generated every alternate reception of SYNC ■ 0x2 (SYNC4): for every 4 SYNC messages ■ 0x3 (SYNC8): for every 8 SYNC messages ■ 0x4 (SYNC16): for every 16 SYNC messages ■ 0x5 (SYNC32): for every 32 SYNC messages ■ 0x6 (RSVD): Reserved **Value After Reset:** 0x0 **Exists:** Always
7:0	LSI	R/W	Log Sync Interval This field indicates the periodicity of the automatically generated SYNC message when the PTP node is Master. Allowed values are -15 to 15. Negative value must be represented in 2's-complement form. For example, if the required value is -1, the value programmed must be 0xFF. **Value After Reset:** 0x0 **Exists:** Always

17.2 EQOS_MTL Registers

17.2.1 MTL_Operation_Mode

- **Description:** The Operation Mode register establishes the Transmit and Receive operating modes and commands.
- **Size:** 32 bits
- **Offset:** 0xc00
- **Exists:** (DWC_EQOS_SYS>0)

31:16	15	14:10	9	8	7	6:5	4:3	2	1	0
Reserved_31_16	FRPE	Reserved_14_10	CNTCLR	CNTPRST	Reserved_7	SCHALG	Reserved_4_3	RAA	DTXSTS	Reserved_0

Table 17-223 Fields for Register: MTL_Operation_Mode

Bits	Name	Memory Access	Description
31:16	Reserved_31_16	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
15	FRPE	R/W	Flexible Rx parser Enable When this bit is set to 1, the Programmable Rx Parser functionality is enabled. When the Rx parser is disabled and if the Rx parser is in the middle of the parsing then it gets disabled only after completing the current packet parsing. When the Rx parser is enabled from disabled state then the Rx parser gets activated for the next immediate packet. **Values:** ■ 0x0 (DISABLE): Flexible Rx parser is disabled ■ 0x1 (ENABLE): Flexible Rx parser is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_FRP_EN
14:10	Reserved_14_10	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
9	CNTCLR	R/W	Counters Reset When this bit is set, all counters are reset. This bit is cleared automatically after 1 clock cycle. If this bit is set along with CNT_PRESET bit, CNT_PRESET has precedence. Access restriction applies. Setting 1 sets. Self-cleared. Setting 0 has no effect. **Values:** ■ 0x0 (DISABLE): Counters are not reset ■ 0x1 (ENABLE): All counters are reset **Value After Reset:** 0x0 **Exists:** Always **Testable:** untestable
8	CNTPRST	R/W	Counters Preset When this bit is set, ■ MTL_TxQ[0-7]_Underflow register is initialized/preset to 12'h7F0. ■ Missed Packet and Overflow Packet counters in MTL_RxQ[0-7]_Missed_Packet_Overflow_Cnt register is initialized/preset to 12'h7F0. Access restriction applies. Setting 1 sets. Self-cleared. Setting 0 has no effect. **Values:** ■ 0x0 (DISABLE): Counters Preset is disabled ■ 0x1 (ENABLE): Counters Preset is enabled **Value After Reset:** 0x0 **Exists:** Always **Testable:** untestable
7	Reserved_7	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
6:5	SCHALG	R/W	Tx Scheduling Algorithm This field indicates the algorithm for Tx scheduling: **Values:** ■ 0x0 (WRR): WRR algorithm ■ 0x1 (WFQ): WFQ algorithm when DCB feature is selected.Otherwise, Reserved ■ 0x2 (DWRR): DWRR algorithm when DCB feature is selected.Otherwise, Reserved ■ 0x3 (SP): Strict priority algorithm **Value After Reset:** 0x0 **Exists:** DWC_EQOS_NUM_TXQ>1

Bits	Name	Memory Access	Description
4:3	Reserved_4_3	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
2	RAA	R/W	Receive Arbitration Algorithm This field is used to select the arbitration algorithm for the Rx side. ■ 0: Strict priority (SP) Queue 0 has the lowest priority and the last queue has the highest priority. ■ 1: Weighted Strict Priority (WSP) **Values:** ■ 0x0 (SP): Strict priority (SP) ■ 0x1 (WSP): Weighted Strict Priority (WSP) **Value After Reset:** 0x0 **Exists:** DWC_EQOS_NUM_RXQ>1
1	DTXSTS	R/W	Drop Transmit Status When this bit is set, the Tx packet status received from the MAC is dropped in the MTL. When this bit is reset, the Tx packet status received from the MAC is forwarded to the application. **Values:** ■ 0x0 (DISABLE): Drop Transmit Status is disabled ■ 0x1 (ENABLE): Drop Transmit Status is enabled **Value After Reset:** ((DWC_EQOS_TX_STS_DROP)?\"0x1\":\"0x0\") **Exists:** !DWC_EQOS_TX_STS_DROP
0	Reserved_0	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always

17.2.2 MTL_DBG_CTL

- **Description:** The FIFO Debug Access Control and Status register controls the operation mode of FIFO debug access.
- **Size:** 32 bits
- **Offset:** 0xc08
- **Exists:** (DWC_EQOS_DBG_FIFO_ACCESS)

31:19	18:17	16	15	14	13:12	11	10	9	8	7	6:5	4	3:2	1	0
Reserved_31_19	EIEC	EIEE	STSIE	PKTIE	FIFOSEL	FIFOWREN	FIFORDEN	RSTSEL	RSTALL	Reserved_7	PKTSTATE	Reserved_4	BYTEEN	DBGMOD	FDBGEN

Table 17-224 Fields for Register: MTL_DBG_CTL

Bits	Name	Memory Access	Description
31:19	Reserved_31_19	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
18:17	EIEC	R/W	ECC Inject Error Control for Tx, Rx and TSO memories When EIEE bit of this register is set, following are the errors inserted based on the value encoded in this field. **Values:** ■ 0x0 (1BIT): Insert 1 bit error ■ 0x1 (2BIT): Insert 2 bit errors ■ 0x2 (3BIT): Insert 3 bit errors ■ 0x3 (1BIT_ADDR): Insert 1 bit error in address field **Value After Reset:** 0x0 **Exists:** DWC_EQOS_ASP_ECC
16	EIEE	R/W	ECC Inject Error Enable for Tx, Rx and TSO memories When set, enables the ECC error injection feature. When reset, disables the ECC error injection feature. **Values:** ■ 0x0 (DISABLE): ECC Inject Error for Tx, Rx and TSO memories is disabled ■ 0x1 (ENABLE): ECC Inject Error for Tx, Rx and TSO memories is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_ASP_ECC

Bits	Name	Memory Access	Description
15	STSIE	R/W	Transmit Status Available Interrupt Status Enable When this bit is set, an interrupt is generated when Transmit status is available in slave mode. **Values:** ■ 0x0 (DISABLE): Transmit Packet Available Interrupt Status is disabled ■ 0x1 (ENABLE): Transmit Packet Available Interrupt Status is enabled **Value After Reset:** 0x0 **Exists:** Always
14	PKTIE	R/W	Receive Packet Available Interrupt Status Enable When this bit is set, an interrupt is generated when EOP of received packet is written to the Rx FIFO. **Values:** ■ 0x0 (DISABLE): Receive Packet Available Interrupt Status is disabled ■ 0x1 (ENABLE): Receive Packet Available Interrupt Status is enabled **Value After Reset:** 0x0 **Exists:** Always
13:12	FIFOSEL	R/W	FIFO Selected for Access This field indicates the FIFO selected for debug access: **Values:** ■ 0x0 (TXFIFO): Tx FIFO ■ 0x1 (TXSTSFIFO): Tx Status FIFO (only read access when SLVMOD is set) ■ 0x2 (TSOFIFO): TSO FIFO (cannot be accessed when SLVMOD is set) ■ 0x3 (RXFIFO): Rx FIFO **Value After Reset:** 0x0 **Exists:** Always
11	FIFOWREN	R/W	FIFO Write Enable When this bit is set, it enables the Write operation on selected FIFO when FIFO Debug Access is enabled. This bit must not be written to 1 when FIFO Debug Access is not enabled, that is FDBGEN bit is 0. Access restriction applies. Self-cleared. Setting 0 clears. Setting 1 sets. **Values:** ■ 0x0 (DISABLE): FIFO Write is disabled ■ 0x1 (ENABLE): FIFO Write is enabled **Value After Reset:** 0x0 **Exists:** Always **Testable:** untestable

Bits	Name	Memory Access	Description
10	FIFORDEN	R/W	FIFO Read Enable When this bit is set, it enables the Read operation on selected FIFO when FIFO Debug Access is enabled. This bit must not be written to 1 when FIFO Debug Access is not enabled, that is FDBGEN bit is 0. Access restriction applies. Self-cleared. Setting 0 clears. Setting 1 sets. **Values:** ■ 0x0 (DISABLE): FIFO Read is disabled ■ 0x1 (ENABLE): FIFO Read is enabled **Value After Reset:** 0x0 **Exists:** Always **Testable:** untestable
9	RSTSEL	R/W	Reset Pointers of Selected FIFO When this bit is set, the pointers of the currently-selected FIFO are reset when FIFO Debug Access is enabled. This bit must not be written to 1 when FIFO Debug Access is not enabled, that is FDBGEN bit is 0. Access restriction applies. Self-cleared. Setting 0 clears. Setting 1 sets. **Values:** ■ 0x0 (DISABLE): Reset Pointers of Selected FIFO is disabled ■ 0x1 (ENABLE): Reset Pointers of Selected FIFO is enabled **Value After Reset:** 0x0 **Exists:** Always **Testable:** untestable
8	RSTALL	R/W	Reset All Pointers When this bit is set, the pointers of all FIFOs are reset when FIFO Debug Access is enabled. This bit must not be written to 1 when FIFO Debug Access is not enabled, that is FDBGEN bit is 0. Access restriction applies. Self-cleared. Setting 0 clears. Setting 1 sets. **Values:** ■ 0x0 (DISABLE): Reset All Pointers is disabled ■ 0x1 (ENABLE): Reset All Pointers is enabled **Value After Reset:** 0x0 **Exists:** Always **Testable:** untestable
7	Reserved_7	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
6:5	PKTSTATE	R/W	Encoded Packet State This field is used to write the control information to the Tx FIFO or Rx FIFO. Tx FIFO: ■ 00: Packet Data ■ 01: Control Word ■ 10: SOP Data ■ 11: EOP Data Rx FIFO: ■ 00: Packet Data ■ 01: Normal Status ■ 10: Last Status ■ 11: EOP **Values:** ■ 0x0 (PKT_DATA): Packet Data ■ 0x1 (CW_NS): Control Word/Normal Status ■ 0x2 (SOP_LS): SOP Data/Last Status ■ 0x3 (EOP): EOP Data/EOP **Value After Reset:** 0x0 **Exists:** Always
4	Reserved_4	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
3:2	BYTEEN	R/W	Byte Enables This field indicates the number of data bytes valid in the data register during Write operation. This is valid only when PKTSTATE is 2'b10 (EOP) and Tx FIFO or Rx FIFO is selected. **Values:** ■ 0x0 (B0_VAL): Byte 0 valid ■ 0x1 (B01_VAL): Byte 0 and Byte 1 are valid ■ 0x2 (B012_VAL): Byte 0, Byte 1, and Byte 2 are valid ■ 0x3 (B0123_VAL): All four bytes are valid **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
1	DBGMOD	R/W	Debug Mode Access to FIFO When this bit is set, it indicates that the current access to the FIFO is read, write, and debug access. In this mode, the following access types are allowed: ■ Read and Write access to Tx FIFO, TSO FIFO, and Rx FIFO ■ Read access is allowed to Tx Status FIFO. When this bit is reset, it indicates that the current access to the FIFO is slave access bypassing the DMA. In this mode, the following access are allowed: ■ Write access to the Tx FIFO ■ Read access to the Rx FIFO and Tx Status FIFO **Values:** ■ 0x0 (DISABLE): Debug Mode Access to FIFO is disabled ■ 0x1 (ENABLE): Debug Mode Access to FIFO is enabled **Value After Reset:** 0x0 **Exists:** Always
0	FDBGEN	R/W	FIFO Debug Access Enable When this bit is set, it indicates that the debug mode access to the FIFO is enabled. When this bit is reset, it indicates that the FIFO can be accessed only through a master interface. **Values:** ■ 0x0 (DISABLE): FIFO Debug Access is disabled ■ 0x1 (ENABLE): FIFO Debug Access is enabled **Value After Reset:** 0x0 **Exists:** Always

17.2.3 MTL_DBG_STS

- **Description:** The FIFO Debug Status register contains the status of FIFO debug access.
- **Size:** 32 bits
- **Offset:** 0xc0c
- **Exists:** (DWC_EQOS_DBG_FIFO_ACCESS)

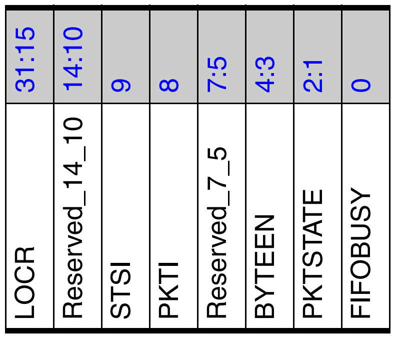


Table 17-225 Fields for Register: MTL_DBG_STS

Bits	Name	Memory Access	Description
31:15	LOCR	R	Remaining Locations in the FIFO Slave Access Mode: This field indicates the space available in selected FIFO. Debug Access Mode: This field contains the Write or Read pointer value of the selected FIFO during Write or Read operation, respectively. Reset: In single Tx Queue configurations, (DWC_EQOS_TXFIFO_SIZE/(DWC_EQOS_DATAWIDTH/8)), Otherwise 0000H **Value After Reset:** ((DWC_EQOS_NUM_TXQ==1)?(DWC_EQOS_TXFIFO_SIZE/(DWC_EQOS_DATAWIDTH/8)):\"0x0\") **Exists:** Always
14:10	Reserved_14_10	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
9	STSI	R/W	Transmit Status Available Interrupt Status When set, this bit indicates that the Slave mode Tx packet is transmitted, and the status is available in Tx Status FIFO. This bit is reset when 1 is written to this bit. **Values:** ■ 0x0 (INACTIVE): Transmit Status Available Interrupt Status not detected ■ 0x1 (ACTIVE): Transmit Status Available Interrupt Status detected **Value After Reset:** 0x0 **Exists:** Always **Testable:** untestable
8	PKTI	R/W	Receive Packet Available Interrupt Status When set, this bit indicates that MAC layer has written the EOP of received packet to the Rx FIFO. This bit is reset when 1 is written to this bit. **Values:** ■ 0x0 (INACTIVE): Receive Packet Available Interrupt Status not detected ■ 0x1 (ACTIVE): Receive Packet Available Interrupt Status detected **Value After Reset:** 0x0 **Exists:** Always **Testable:** untestable
7:5	Reserved_7_5	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
4:3	BYTEEN	R	Byte Enables This field indicates the number of data bytes valid in the data register during Read operation. This is valid only when PKTSTATE is 2'b10 (EOP) and Tx FIFO or Rx FIFO is selected. **Values:** ■ 0x0 (B0_VAL): Byte 0 valid ■ 0x1 (B01_VAL): Byte 0 and Byte 1 are valid ■ 0x2 (B012_VAL): Byte 0, Byte 1, and Byte 2 are valid ■ 0x3 (B0123_VAL): All four bytes are valid **Value After Reset:** 0x3 **Exists:** Always

Bits	Name	Memory Access	Description
2:1	PKTSTATE	R	Encoded Packet State This field is used to get the control or status information of the selected FIFO. Tx FIFO: ■ 00: Packet Data ■ 01: Control Word ■ 10: SOP Data ■ 11: EOP Data Rx FIFO: ■ 00: Packet Data ■ 01: Normal Status ■ 10: Last Status ■ 11: EOP This field is applicable only for Tx FIFO and Rx FIFO during Read operation. **Values:** ■ 0x0 (PKT_DATA): Packet Data ■ 0x1 (CW_NS): Control Word/Normal Status ■ 0x2 (SOP_LS): SOP Data/Last Status ■ 0x3 (EOP): EOP Data/EOP **Value After Reset:** 0x0 **Exists:** Always
0	FIFOBUSY	R	FIFO Busy When set, this bit indicates that a FIFO operation is in progress in the MAC and content of the following fields is not valid: ■ All other fields of this register ■ All fields of the MTL_FIFO_Debug_Data register **Values:** ■ 0x0 (INACTIVE): FIFO Busy not detected ■ 0x1 (ACTIVE): FIFO Busy detected **Value After Reset:** 0x0 **Exists:** Always

17.2.4 MTL_FIFO_Debug_Data

- **Description:** The FIFO Debug Data register contains the data to be written to or read from the FIFOs.
- **Size:** 32 bits
- **Offset:** 0xc10
- **Exists:** (DWC_EQOS_DBG_FIFO_ACCESS)

Table 17-226 Fields for Register: MTL_FIFO_Debug_Data

Bits	Name	Memory Access	Description
31:0	FDBGDATA	R/W	FIFO Debug Data During debug or slave access write operation, this field contains the data to be written to the Tx FIFO, Rx FIFO, or TSO FIFO. During debug or slave access read operation, this field contains the data read from the Tx FIFO, Rx FIFO, TSO FIFO, or Tx Status FIFO. **Value After Reset:** 0x0 **Exists:** Always **Testable:** untestable

17.2.5 MTL_Interrupt_Status

- **Description:** The software driver (application) reads this register during interrupt service routine or polling to determine the interrupt status of MTL queues and the MAC.
- **Size:** 32 bits
- **Offset:** 0xc20
- **Exists:** Always

31:24	23	22:19	18	17	16	15:8	7	6	5	4	3	2	1	0
Reserved_31_24	MTLPIS	Reserved_22_19	ESTIS	DBGIS	MACIS	Reserved_15_8	Q7IS	Q6IS	Q5IS	Q4IS	Q3IS	Q2IS	Q1IS	Q0IS

Table 17-227 Fields for Register: MTL_Interrupt_Status

Bits	Name	Memory Access	Description
31:24	Reserved_31_24	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
23	MTLPIS	R	MTL Rx Parser Interrupt Status This bit indicates that there is an interrupt from Rx Parser Block. To reset this bit, the application must read the MTL_ Rxp_Interrupt_Status register to get the exact cause of the interrupt and clear its source. **Values:** ■ 0x0 (INACTIVE): MTL Rx Parser Interrupt status not detected ■ 0x1 (ACTIVE): MTL Rx Parser Interrupt status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_FRP_EN
22:19	Reserved_22_19	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
18	ESTIS	R	EST (TAS- 802.1Qbv) Interrupt Status This bit indicates an interrupt event during the operation of 802.1Qbv. To reset this bit, the application must clear the error/event that has caused the Interrupt. **Values:** ■ 0x0 (INACTIVE): EST (TAS- 802.1Qbv) Interrupt status not detected ■ 0x1 (ACTIVE): EST (TAS- 802.1Qbv) Interrupt status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_AV_EST
17	DBGIS	R	Debug Interrupt status This bit indicates an interrupt event during the slave access. To reset this bit, the application must read the FIFO Debug Access Status register to get the exact cause of the interrupt and clear its source. **Values:** ■ 0x0 (INACTIVE): Debug Interrupt status not detected ■ 0x1 (ACTIVE): Debug Interrupt status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_DBG_FIFO_ACCESS
16	MACIS	R	MAC Interrupt Status This bit indicates an interrupt event in the MAC. To reset this bit, the application must read the corresponding register in the MAC to get the exact cause of the interrupt and clear its source. **Values:** ■ 0x0 (INACTIVE): MAC Interrupt status not detected ■ 0x1 (ACTIVE): MAC Interrupt status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_SYS==1
15:8	Reserved_15_8	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
7	Q7IS	R	Queue 7 Interrupt status This bit indicates that there is an interrupt from Queue 7. To reset this bit, the application must read the MTL_Q7_Interrupt_Control_Status Status register to get the exact cause of the interrupt and clear its source. **Values:** ■ 0x0 (INACTIVE): Queue 7 Interrupt status not detected ■ 0x1 (ACTIVE): Queue 7 Interrupt status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_NUM_TXQ>7||DWC_EQOS_NUM_RXQ>7
6	Q6IS	R	Queue 6 Interrupt status This bit indicates that there is an interrupt from Queue 6. To reset this bit, the application must read the MTL_Q6_Interrupt_Control_Status register to get the exact cause of the interrupt and clear its source. **Values:** ■ 0x0 (INACTIVE): Queue 6 Interrupt status not detected ■ 0x1 (ACTIVE): Queue 6 Interrupt status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_NUM_TXQ>6||DWC_EQOS_NUM_RXQ>6
5	Q5IS	R	Queue 5 Interrupt status This bit indicates that there is an interrupt from Queue 5. To reset this bit, the application must read the MTL_Q5_Interrupt_Control_Status register to get the exact cause of the interrupt and clear its source. **Values:** ■ 0x0 (INACTIVE): Queue 5 Interrupt status not detected ■ 0x1 (ACTIVE): Queue 5 Interrupt status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_NUM_TXQ>5||DWC_EQOS_NUM_RXQ>5
4	Q4IS	R	Queue 4 Interrupt status This bit indicates that there is an interrupt from Queue 4. To reset this bit, the application must read the MTL_Q4_Interrupt_Control_Status register to get the exact cause of the interrupt and clear its source. **Values:** ■ 0x0 (INACTIVE): Queue 4 Interrupt status not detected ■ 0x1 (ACTIVE): Queue 4 Interrupt status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_NUM_TXQ>4||DWC_EQOS_NUM_RXQ>4

Bits	Name	Memory Access	Description
3	Q3IS	R	Queue 3 Interrupt status This bit indicates that there is an interrupt from Queue 3. To reset this bit, the application must read the MTL_Q3_Interrupt_Control_Status register to get the exact cause of the interrupt and clear its source. **Values:** ■ 0x0 (INACTIVE): Queue 3 Interrupt status not detected ■ 0x1 (ACTIVE): Queue 3 Interrupt status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_NUM_TXQ>3||DWC_EQOS_NUM_RXQ>3
2	Q2IS	R	Queue 2 Interrupt status This bit indicates that there is an interrupt from Queue 2. To reset this bit, the application must read the MTL_Q2_Interrupt_Control_Status register to get the exact cause of the interrupt and clear its source. **Values:** ■ 0x0 (INACTIVE): Queue 2 Interrupt status not detected ■ 0x1 (ACTIVE): Queue 2 Interrupt status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_NUM_TXQ>2||DWC_EQOS_NUM_RXQ>2
1	Q1IS	R	Queue 1 Interrupt status This bit indicates that there is an interrupt from Queue 1. To reset this bit, the application must read the MTL_Q1_Interrupt_Control_Status register to get the exact cause of the interrupt and clear its source. **Values:** ■ 0x0 (INACTIVE): Queue 1 Interrupt status not detected ■ 0x1 (ACTIVE): Queue 1 Interrupt status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_NUM_TXQ>1||DWC_EQOS_NUM_RXQ>1
0	Q0IS	R	Queue 0 Interrupt status This bit indicates that there is an interrupt from Queue 0. To reset this bit, the application must read Queue 0 Interrupt Control and Status register to get the exact cause of the interrupt and clear its source. **Values:** ■ 0x0 (INACTIVE): Queue 0 Interrupt status not detected ■ 0x1 (ACTIVE): Queue 0 Interrupt status detected **Value After Reset:** 0x0 **Exists:** Always

17.2.6 MTL_RxQ_DMA_Map0

- **Description:** The Receive Queue and DMA Channel Mapping 0 register is reserved in EQOS-CORE and EQOS-MTL configurations.
- **Size:** 32 bits
- **Offset:** 0xc30
- **Exists:** (DWC_EQOS_SYS>1&&DWC_EQOS_NUM_DMA_RX_CH>1)

31:29	28	27:y	n:24	23:21	20	19:y	x:16	15:13	12	11:y	x:8	7:5	4	3:y	x:0
Reserved_31_29	Q3DDMACH	Reserved_27_y	Q3MDMACH	Reserved_23_21	Q2DDMACH	Reserved_19_y	Q2MDMACH	Reserved_15_13	Q1DDMACH	Reserved_11_y	Q1MDMACH	Reserved_7_5	Q0DDMACH	Reserved_3_y	Q0MDMACH

Table 17-228 Fields for Register: MTL_RxQ_DMA_Map0

Bits	Name	Memory Access	Description
31:29	Reserved_31_29	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
28	Q3DDMACH	R/W	Queue 3 Enabled for Dynamic (per packet) DMA Channel Selection When set, this bit indicates that the packets received in Queue 3 are routed to a particular DMA channel as decided in the MAC Receiver based on the DMA channel number programmed in the L3-L4 filter registers, or the Ethernet DA address. When reset, this bit indicates that the packets received in Queue 3 are routed to the DMA Channel programmed in the Q3MDMACH field (Bits[26:24]). **Values:** ■ 0x0 (DISABLE): Queue 3 disabled for DA-based DMA Channel Selection ■ 0x1 (ENABLE): Queue 3 enabled for DA-based DMA Channel Selection **Value After Reset:** 0x0 **Exists:** DWC_EQOS_NUM_RXQ>=4
27:y	Reserved_27_y	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[y]:** DWC_EQOS_NUM_DMA_RX_CHW + 24

Bits	Name	Memory Access	Description
n:24	Q3MDMACH	R/W	Queue 3 Mapped to DMA Channel This field controls the routing of the received packet in Queue 3 to the DMA channel: ■ 000: DMA Channel 0 ■ 001: DMA Channel 1 ■ 010: DMA Channel 2 ■ 011: DMA Channel 3 ■ 100: DMA Channel 4 ■ 101: DMA Channel 5 ■ 110: DMA Channel 6 ■ 111: DMA Channel 7 This field is valid when the Q3DDMACH field is reset. **Note:** The width of this field depends on the number of RX DMA channels and not all the values may be valid in some configurations. For example, if the number of RX DMA channels selected is 2, only 000 and 001 are valid, the others are reserved **Value After Reset:** 0x0 **Exists:** DWC_EQOS_NUM_RXQ>=4
23:21	Reserved_23_21	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
20	Q2DDMACH	R/W	Queue 2 Enabled for DA-based DMA Channel Selection When set, this bit indicates that the packets received in Queue 2 are routed to a particular DMA channel as decided in the MAC Receiver based on the DMA channel number programmed in the L3-L4 filter registers, or the Ethernet DA address. When reset, this bit indicates that the packets received in Queue 2 are routed to the DMA Channel programmed in the Q2MDMACH field (Bits[18:16]). **Values:** ■ 0x0 (DISABLE): Queue 2 disabled for DA-based DMA Channel Selection ■ 0x1 (ENABLE): Queue 2 enabled for DA-based DMA Channel Selection **Value After Reset:** 0x0 **Exists:** DWC_EQOS_NUM_RXQ>=3
19:y	Reserved_19_y	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[y]:** DWC_EQOS_NUM_DMA_RX_CHW + 16

Bits	Name	Memory Access	Description
x:16	Q2MDMACH	R/W	Queue 2 Mapped to DMA Channel This field controls the routing of the received packet in Queue 2 to the DMA channel: ■ 000: DMA Channel 0 ■ 001: DMA Channel 1 ■ 010: DMA Channel 2 ■ 011: DMA Channel 3 ■ 100: DMA Channel 4 ■ 101: DMA Channel 5 ■ 110: DMA Channel 6 ■ 111: DMA Channel 7 This field is valid when the Q2DDMACH field is reset. **Value After Reset:** 0x0 **Exists:** DWC_EQOS_NUM_RXQ>=3 **Range Variable[x]:** DWC_EQOS_NUM_DMA_RX_CHW + 15
15:13	Reserved_15_13	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
12	Q1DDMACH	R/W	Queue 1 Enabled for DA-based DMA Channel Selection When set, this bit indicates that the packets received in Queue 1 are routed to a particular DMA channel as decided in the MAC Receiver based on the DMA channel number programmed in the L3-L4 filter registers, or the Ethernet DA address. When reset, this bit indicates that the packets received in Queue 1 are routed to the DMA Channel programmed in the Q1MDMACH field (Bits[10:8]). **Values:** ■ 0x0 (DISABLE): Queue 1 disabled for DA-based DMA Channel Selection ■ 0x1 (ENABLE): Queue 1 enabled for DA-based DMA Channel Selection **Value After Reset:** 0x0 **Exists:** DWC_EQOS_NUM_RXQ!=1
11:y	Reserved_11_y	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[y]:** DWC_EQOS_NUM_DMA_RX_CHW + 8

Bits	Name	Memory Access	Description
x:8	Q1MDMACH	R/W	Queue 1 Mapped to DMA Channel This field controls the routing of the received packet in Queue 1 to the DMA channel: ■ 000: DMA Channel 0 ■ 001: DMA Channel 1 ■ 010: DMA Channel 2 ■ 011: DMA Channel 3 ■ 100: DMA Channel 4 ■ 101: DMA Channel 5 ■ 110: DMA Channel 6 ■ 111: DMA Channel 7 This field is valid when the Q1DDMACH field is reset. The width of this field depends on the number of RX DMA channels and not all the values may be valid in some configurations. For example, if the number of RX DMA channels selected is 2, only 000 and 001 are valid, the other bits are reserved. **Value After Reset:** 0x0 **Exists:** DWC_EQOS_NUM_RXQ!=1 **Range Variable[x]:** DWC_EQOS_NUM_DMA_RX_CHW + 7
7:5	Reserved_7_5	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
4	Q0DDMACH	R/W	Queue 0 Enabled for DA-based DMA Channel Selection When set, this bit indicates that the packets received in Queue 0 are routed to a particular DMA channel as decided in the MAC Receiver based on the DMA channel number programmed in the L3-L4 filter registers, or the Ethernet DA address. When reset, this bit indicates that the packets received in Queue 0 are routed to the DMA Channel programmed in the Q0MDMACH field. **Values:** ■ 0x0 (DISABLE): Queue 0 disabled for DA-based DMA Channel Selection ■ 0x1 (ENABLE): Queue 0 enabled for DA-based DMA Channel Selection **Value After Reset:** 0x0 **Exists:** Always
3:y	Reserved_3_y	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[y]:** DWC_EQOS_NUM_DMA_RX_CHW

Bits	Name	Memory Access	Description
x:0	Q0MDMACH	R/W	Queue 0 Mapped to DMA Channel This field controls the routing of the packet received in Queue 0 to the DMA channel: ■ 000: DMA Channel 0 ■ 001: DMA Channel 1 ■ 010: DMA Channel 2 ■ 011: DMA Channel 3 ■ 100: DMA Channel 4 ■ 101: DMA Channel 5 ■ 110: DMA Channel 6 ■ 111: DMA Channel 7 This field is valid when the Q0DDMACH field is reset. The width of this field depends on the number of RX DMA channels and not all the values may be valid in some configurations. For example, if the number of RX DMA channels selected is 2, only 000 and 001 are valid, the other bits are reserved. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[x]:** DWC_EQOS_NUM_DMA_RX_CHW - 1

17.2.7 MTL_RxQ_DMA_Map1

- **Description:** The Receive Queue and DMA Channel Mapping 1 register is reserved in EQOS-CORE and EQOS-MTL configurations.
- **Size:** 32 bits
- **Offset:** 0xc34
- **Exists:** (DWC_EQOS_SYS>1&&DWC_EQOS_RX_Q4_EN&&DWC_EQOS_NUM_DMA_RX_CH>1)

31:29	28	27:y	x:24	23:21	20	19:y	x:16	15:13	12	11:y	x:8	7:5	4	3:y	x:0
Reserved_31_29	Q7DDMACH	Reserved_27_y	Q7MDMACH	Reserved_23_21	Q6DDMACH	Reserved_19_y	Q6MDMACH	Reserved_15_13	Q5DDMACH	Reserved_11_y	Q5MDMACH	Reserved_7_5	Q4DDMACH	Reserved_3_y	Q4MDMACH

Table 17-229 Fields for Register: MTL_RxQ_DMA_Map1

Bits	Name	Memory Access	Description
31:29	Reserved_31_29	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
28	Q7DDMACH	R/W	Queue 7 Enabled for DA-based DMA Channel Selection When set, this bit indicates that the packets received in Queue 7 are routed to a particular DMA channel as decided in the MAC Receiver based on the DMA channel number programmed in the L3-L4 filter registers, or the Ethernet DA address. When reset, this bit indicates that the packets received in Queue 7 are routed to the DMA Channel programmed in the Q7MDMACH field. **Values:** ■ 0x0 (DISABLE): Queue 5 disabled for DA-based DMA Channel Selection ■ 0x1 (ENABLE): Queue 5 enabled for DA-based DMA Channel Selection **Value After Reset:** 0x0 **Exists:** DWC_EQOS_NUM_RXQ==8
27:y	Reserved_27_y	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[y]:** DWC_EQOS_NUM_DMA_RX_CHW + 24

Bits	Name	Memory Access	Description
x:24	Q7MDMACH	R/W	Queue 7 Mapped to DMA Channel This field controls the routing of the packet received in Queue 7 to the DMA channel: ■ 000: DMA Channel 0 ■ 001: DMA Channel 1 ■ 010: DMA Channel 2 ■ 011: DMA Channel 3 ■ 100: DMA Channel 4 ■ 101: DMA Channel 5 ■ 110: DMA Channel 6 ■ 111: DMA Channel 7 This field is valid when the Q7DDMACH field is reset. The width of this field depends on the number of RX DMA channels and not all the values may be valid in some configurations. For example, if the number of RX DMA channels selected is 2, only 000 and 001 are valid, the other bits are reserved. **Value After Reset:** 0x0 **Exists:** DWC_EQOS_NUM_RXQ==8 **Range Variable[x]:** DWC_EQOS_NUM_DMA_RX_CHW + 23
23:21	Reserved_23_21	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
20	Q6DDMACH	R/W	Queue 6 Enabled for DA-based DMA Channel Selection When set, this bit indicates that the packets received in Queue 6 are routed to a particular DMA channel as decided in the MAC Receiver based on the DMA channel number programmed in the L3-L4 filter registers, or the Ethernet DA address. When reset, this bit indicates that the packets received in Queue 6 are routed to the DMA Channel programmed in the Q6MDMACH field. **Values:** ■ 0x0 (DISABLE): Queue 6 disabled for DA-based DMA Channel Selection ■ 0x1 (ENABLE): Queue 6 enabled for DA-based DMA Channel Selection **Value After Reset:** 0x0 **Exists:** DWC_EQOS_NUM_RXQ>=7
19:y	Reserved_19_y	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[y]:** DWC_EQOS_NUM_DMA_RX_CHW + 16

Bits	Name	Memory Access	Description
x:16	Q6MDMACH	R/W	Queue 6 Mapped to DMA Channel This field controls the routing of the packet received in Queue 6 to the DMA channel: ■ 000: DMA Channel 0 ■ 001: DMA Channel 1 ■ 010: DMA Channel 2 ■ 011: DMA Channel 3 ■ 100: DMA Channel 4 ■ 101: DMA Channel 5 ■ 110: DMA Channel 6 ■ 111: DMA Channel 7 This field is valid when the Q6DDMACH field is reset. The width of this field depends on the number of RX DMA channels and not all the values may be valid in some configurations. For example, if the number of RX DMA channels selected is 2, only 000 and 001 are valid, the other bits are reserved. **Value After Reset:** 0x0 **Exists:** DWC_EQOS_NUM_RXQ>=7 **Range Variable[x]:** DWC_EQOS_NUM_DMA_RX_CHW + 15
15:13	Reserved_15_13	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
12	Q5DDMACH	R/W	Queue 5 Enabled for DA-based DMA Channel Selection When set, this bit indicates that the packets received in Queue 5 are routed to a particular DMA channel as decided in the MAC Receiver based on the DMA channel number programmed in the L3-L4 filter registers, or the Ethernet DA address. When reset, this bit indicates that the packets received in Queue 5 are routed to the DMA Channel programmed in the Q5MDMACH field. **Values:** ■ 0x0 (DISABLE): Queue 5 disabled for DA-based DMA Channel Selection ■ 0x1 (ENABLE): Queue 5 enabled for DA-based DMA Channel Selection **Value After Reset:** 0x0 **Exists:** DWC_EQOS_NUM_RXQ>=6
11:y	Reserved_11_y	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[y]:** DWC_EQOS_NUM_DMA_RX_CHW + 8

Bits	Name	Memory Access	Description
x:8	Q5MDMACH	R/W	Queue 5 Mapped to DMA Channel This field controls the routing of the packets received in Queue 5 to the DMA channel: ■ 000: DMA Channel 0 ■ 001: DMA Channel 1 ■ 010: DMA Channel 2 ■ 011: DMA Channel 3 ■ 100: DMA Channel 4 ■ 101: DMA Channel 5 ■ 110: DMA Channel 6 ■ 111: DMA Channel 7 This field is valid when the Q5DDMACH field is reset. The width of this field depends on the number of RX DMA channels and not all the values may be valid in some configurations. For example, if the number of RX DMA channels selected is 2, only 000 and 001 are valid, the other bits are reserved. **Value After Reset:** 0x0 **Exists:** DWC_EQOS_NUM_RXQ>=6 **Range Variable[x]:** DWC_EQOS_NUM_DMA_RX_CHW + 7
7:5	Reserved_7_5	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
4	Q4DDMACH	R/W	Queue 4 Enabled for DA-based DMA Channel Selection When set, this bit indicates that the packets received in Queue 4 are routed to a particular DMA channel as decided in the MAC Receiver based on the DMA channel number programmed in the L3-L4 filter registers, or the Ethernet DA address. When reset, this bit indicates that the packets received in Queue 4 are routed to the DMA Channel programmed in the Q4MDMACH field. **Values:** ■ 0x0 (DISABLE): Queue 4 disabled for DA-based DMA Channel Selection ■ 0x1 (ENABLE): Queue 4 enabled for DA-based DMA Channel Selection **Value After Reset:** 0x0 **Exists:** DWC_EQOS_NUM_RXQ>=5
3:y	Reserved_3_y	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[y]:** DWC_EQOS_NUM_DMA_RX_CHW

Bits	Name	Memory Access	Description
x:0	Q4MDMACH	R/W	Queue 4 Mapped to DMA Channel This field controls the routing of the packet received in Queue 4 to the DMA channel: ■ 000: DMA Channel 0 ■ 001: DMA Channel 1 ■ 010: DMA Channel 2 ■ 011: DMA Channel 3 ■ 100: DMA Channel 4 ■ 101: DMA Channel 5 ■ 110: DMA Channel 6 ■ 111: DMA Channel 7 This field is valid when the Q4DDMACH field is reset. The width of this field depends on the number of RX DMA channels and not all the values may be valid in some configurations. For example, if the number of RX DMA channels selected is 2, only 000 and 001 are valid, the other bits are reserved. **Value After Reset:** 0x0 **Exists:** DWC_EQOS_NUM_RXQ>=5 **Range Variable[x]:** DWC_EQOS_NUM_DMA_RX_CHW - 1

17.2.8 MTL_TBS_CTRL

- **Description:** This register controls the operation of Time Based Scheduling.
- **Size:** 32 bits
- **Offset:** 0xc40
- **Exists:** (DWC_EQOS_TBS)

31:8	7	6:4	3:2	1	0
LEOS	Reserved_7	LEGOS	Reserved_3_2	LEOV	ESTM

Table 17-230 Fields for Register: MTL_TBS_CTRL

Bits	Name	Memory Access	Description
31:8	LEOS	R/W	Launch Expiry Offset The value in units of 256 nanoseconds that has to be added to the Launch time to compute the Launch Expiry time. Value valid only when LEOV is set. Max value: 999,999,999 ns, additionally should be smaller than CTR-1 value when ESTM mode is set since this value is a modulo CTR value. **Value After Reset:** 0x0 **Exists:** Always
7	Reserved_7	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
6:4	LEGOS	R/W	Launch Expiry GSN Offset The number GSN slots that has to be added to the Launch GSN to compute the Launch Expiry time. Value valid only when LEOV is set. **Value After Reset:** 0x0 **Exists:** DWC_EQOS_AV_EST
3:2	Reserved_3_2	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
1	LEOV	R/W	Launch Expiry Offset Valid When set indicates the LEOS field is valid. When not set, indicates the Launch Expiry Offset is not valid and the MTL must not check for Launch expiry time. **Values:** ■ 0x0 (INVALID): LEOS field is invalid ■ 0x1 (VALID): LEOS field is valid **Value After Reset:** 0x0 **Exists:** Always
0	ESTM	R/W	EST offset Mode When this bit is set, the Launch Time value used in Time Based Scheduling is interpreted as an EST offset value and is added to the Base Time Register (BTR) of the current list. When reset, the Launch Time value is used as an absolute value that should be compared with the System time [39:8]. **Values:** ■ 0x0 (DISABLE): EST offset Mode is disabled ■ 0x1 (ENABLE): EST offset Mode is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_AV_EST

17.2.9 MTL_EST_Control

- **Description:** This register controls the operation of Enhancements to Scheduled Transmission (IEEE802.1Qbv).
- **Size:** 32 bits
- **Offset:** 0xc50
- **Exists:** (DWC_EQOS_AV_EST)

31:24	23:12	11	10:y	x:8	7:6	5	4	3:2	1	0
PTOV	CTOV	Reserved_11	Reserved_10_y	TILS	LCSE	DFBS	DDBF	Reserved_2_3	SSWL	EEST

Table 17-231 Fields for Register: MTL_EST_Control

Bits	Name	Memory Access	Description
31:24	PTOV	R/W	PTP Time Offset Value The value of PTP Clock period multiplied by 6 in nanoseconds. This value is needed to avoid transmission overruns at the beginning of the installation of a new GCL. **Value After Reset:** 0x0 **Exists:** Always
23:12	CTOV	R/W	Current Time Offset Value Provides a 12 bit time offset value in nano second that is added to the current time to compensate for all the implementation pipeline delays such as the CDC sync delay, buffering delays, data path delays etc. This offset helps to ensure that the impact of gate controls is visible on the line exactly at the pre-determined schedule (or as close to the schedule as possible). **Value After Reset:** 0x0 **Exists:** Always
11	Reserved_11	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
10:y	Reserved_10_y	R	**Value After Reset:** 0x0 **Exists:** [<functionof> (DWC_EQOS_AV_EST ? 3 : 0)<3] **Range Variable[y]:** DWC_EQOS_EST_TISW + 8

Bits	Name	Memory Access	Description
x:8	TILS	R/W	Time Interval Left Shift Amount This field provides the left shift amount for the programmed Time Interval values used in the Gate Control Lists. ■ 000: No left shift needed (equal to x1ns) ■ 001: Left shift TI by 1 bit (equal to x2ns) ■ 010: Left shift TI by 2 bits (equal to x4ns) ■ . ■ . ■ 100: Left shift TI by 7 bits (equal to x128ns) Based on the configuration one or more bits of this field should be treated as Reserved/Read-Only. **Value After Reset:** 0x0 **Exists:** [<functionof> (DWC_EQOS_AV_EST ? 3 : 0)>0] **Range Variable[x]:** DWC_EQOS_EST_TISW + 7
7:6	LCSE	R/W	Loop Count to report Scheduling Error Programmable number of GCL list iterations before reporting an HLBS error defined in EST_Status register. **Values:** ■ 0x0 (4_ITERNS): 4 iterations ■ 0x1 (8_ITERNS): 8 iterations ■ 0x2 (16_ITERNS): 16 iterations ■ 0x3 (32_ITERNS): 32 iterations **Value After Reset:** 0x0 **Exists:** Always
5	DFBS	R/W	Drop Frames causing Scheduling Error When set frames reported to cause HOL Blocking due to not getting scheduled (HLBS field of EST_Status register) after 4,8,16,32 (based on LCSE field of this register) GCL iterations are dropped. **Values:** ■ 0x0 (DONT_DROP): Do not Drop Frames causing Scheduling Error ■ 0x1 (DROP): Drop Frames causing Scheduling Error **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
4	DDBF	R/W	Do not Drop frames during Frame Size Error When set, frames are not be dropped during Head-of-Line blocking due to Frame Size Error (HLBF field of EST_Status register). **Values:** ■ 0x0 (DROP): Drop frames during Frame Size Error ■ 0x1 (DONT_DROP): Do not Drop frames during Frame Size Error **Value After Reset:** 0x0 **Exists:** Always
3:2	Reserved_2_3	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
1	SSWL	R/W	Switch to S/W owned list When set indicates that the software has programmed that list that it currently owns (SWOL) and the hardware should switch to the new list based on the new BTR. Hardware clears this bit when the switch to the SWOL happens to indicate the completion of the switch or when an BTR error (BTRE in Status register) is set. When BTRE is set this bit is cleared but SWOL is not updated as the switch was not successful. Access restriction applies. Setting 1 sets. Self-cleared. Setting 0 has no effect. **Values:** ■ 0x0 (DISABLE): Switch to S/W owned list is disabled ■ 0x1 (ENABLE): Switch to S/W owned list is enabled **Value After Reset:** 0x0 **Exists:** Always
0	EEST	R/W	Enable EST When reset, the gate control list processing is halted and all gates are assumed to be in Open state. Should be set for the hardware to start processing the gate control lists. During the toggle from 0 to 1, the gate control list processing starts only after the SSWL bit it set. When DWC_EQOS_ASP_ECC is selected during the configuration, if any uncorrectable error is detected in the EST memory the hardware will reset this bit and disable the EST function. **Values:** ■ 0x0 (DISABLE): EST is disabled ■ 0x1 (ENABLE): EST is enabled **Value After Reset:** 0x0 **Exists:** Always

17.2.10 MTL_EST_Status

- **Description:** This register provides Status related to Enhancements to Scheduled Transmission (IEEE802.1Qbv).
- **Size:** 32 bits
- **Offset:** 0xc58
- **Exists:** (DWC_EQOS_AV_EST)

31:20	19:16	15:12	11:8	7	6:5	4	3	2	1	0
Reserved_31_20	CGSN	Reserved_15_12	BTRL	SWOL	Reserved_6_5	CGCE	HLBS	HLBF	BTRE	SWLC

Table 17-232 Fields for Register: MTL_EST_Status

Bits	Name	Memory Access	Description
31:20	Reserved_31_20	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
19:16	CGSN	R	Current GCL Slot Number Indicates the slot number of the GCL list. Slot number is a modulo 16 count of the GCL List loops executed so far. Even if a new GCL list is installed, the count is incremental. **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_TBS)
15:12	Reserved_15_12	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
11:8	BTRL	R	BTR Error Loop Count Provides the minimum count (N) for which the equation Current Time =< New BTR + (N * New Cycle Time) becomes true. N = "1111" indicates the iterations exceeded the value of 8 and the hardware was not able to update New BTR to be equal to or greater than Current Time. Software intervention is needed to update the New BTR. Value cleared when BTRE field of this register is cleared. **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
7	SWOL	R	S/W owned list When '0' indicates Gate control list number "0" is owned by software and when "1" indicates the Gate Control list "1" is owned by the software. Any reads/writes by the software (using indirect access via GCL_Control) is directed to the list indicated by this value by default. The inverse of this value is treated as HWOL. R/W operations performed by hardware are directed to the list pointed by HWOL by default. **Values:** ■ 0x0 (INACTIVE): Gate control list number "0" is owned by software ■ 0x1 (ACTIVE): Gate control list number "1" is owned by software **Value After Reset:** 0x0 **Exists:** Always
6:5	Reserved_6_5	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
4	CGCE	R/W	Constant Gate Control Error This error occurs when the list length (LLR) is 1 and the programmed Time Interval (TI) value after the optional Left Shifting is less than or equal to the Cycle Time (CTR). The above programming implies Gates are either always Closed or always Open based on the Gate Control values; the same effect can be achieved by other simpler (non TSN) programming mechanisms. Since the implementation does not support such a programming an error is reported. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. **Values:** ■ 0x0 (INACTIVE): Constant Gate Control Error not detected ■ 0x1 (ACTIVE): Constant Gate Control Error detected **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
3	HLBS	R	Head-Of-Line Blocking due to Scheduling Set when the frame is not able to win arbitration and get scheduled even after 4 iterations of the GCL. Indicates to software a potential programming error. The one hot encoded values of the Queue Numbers that are not able to make progress are indicated in the MTL_EST_Sch_Error register. Bit cleared when MTL_EST_Sch_Error register is all zeros. **Values:** ■ 0x0 (INACTIVE): Head-Of-Line Blocking due to Scheduling not detected ■ 0x1 (ACTIVE): Head-Of-Line Blocking due to Scheduling detected **Value After Reset:** 0x0 **Exists:** Always
2	HLBF	R	Head-Of-Line Blocking due to Frame Size Set when HOL Blocking is noticed on one or more Queues as a result of none of the Time Intervals of gate open in the GCL being greater than or equal to the duration needed for frame size (or frame fragment size when preemption is enabled) transmission. The one hot encoded Queue numbers that are experiencing HLBF are indicated in the MTL_EST_Frm_Size_Error register. Additionally, the first Queue number that experienced HLBF along with the frame size is captured in MTL_EST_Frm_Size_Capture register. Bit cleared when MTL_EST_Frame_Size_ Error register is all zeros. **Values:** ■ 0x0 (INACTIVE): Head-Of-Line Blocking due to Frame Size not detected ■ 0x1 (ACTIVE): Head-Of-Line Blocking due to Frame Size detected **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
1	BTRE	R/W	BTR Error When "1" indicates a programming error in the BTR of SWOL where the programmed value is less than current time. If the BTRL = "1111", SWOL is not updated and Software should reprogram the BTR to a value greater than current time and then set SSWL to reinitiate the switch to SWOL. Else if the value of BTRL < "1111", SWOL is updated and this field indicates the number of iterations (of + CycleTime) taken by hardware to update the BTR to a value greater than Current Time. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. **Values:** ■ 0x0 (INACTIVE): BTR Error not detected ■ 0x1 (ACTIVE): BTR Error detected **Value After Reset:** 0x0 **Exists:** Always
0	SWLC	R/W	Switch to S/W owned list Complete When "1" indicates the hardware has successfully switched to the SWOL, and the SWOL bit has been updated to that effect. Cleared when the SSWL of EST_Control register transitions from 0 to 1, or on a software write. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. **Values:** ■ 0x0 (INACTIVE): Switch to S/W owned list Complete not detected ■ 0x1 (ACTIVE): Switch to S/W owned list Complete detected **Value After Reset:** 0x0 **Exists:** Always

17.2.11 MTL_EST_Sch_Error

- **Description:** This register provides the One Hot encoded Queue Numbers that are having the Scheduling related error (timeout).
- **Size:** 32 bits
- **Offset:** 0xc60
- **Exists:** (DWC_EQOS_AV_EST)

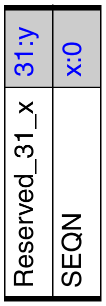


Table 17-233 Fields for Register: MTL_EST_Sch_Error

Bits	Name	Memory Access	Description
31:y	Reserved_31_x	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[y]:** DWC_EQOS_NUM_TXQ
x:0	SEQN	R/W	Schedule Error Queue Number The One Hot Encoded Queue Numbers that have experienced error/timeout described in HLBS field of status register. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[x]:** DWC_EQOS_NUM_TXQ - 1

17.2.12 MTL_EST_Frm_Size_Error

- **Description:** This register provides the One Hot encoded Queue Numbers that are having the Frame Size related error.
- **Size:** 32 bits
- **Offset:** 0xc64
- **Exists:** (DWC_EQOS_AV_EST)

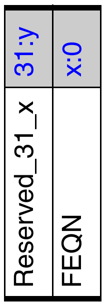


Table 17-234 Fields for Register: MTL_EST_Frm_Size_Error

Bits	Name	Memory Access	Description
31:y	Reserved_31_x	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[y]:** DWC_EQOS_NUM_TXQ
x:0	FEQN	R/W	Frame Size Error Queue Number The One Hot Encoded Queue Numbers that have experienced error described in HLBF field of status register. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[x]:** DWC_EQOS_NUM_TXQ - 1

17.2.13 MTL_EST_Frm_Size_Capture

- **Description:** This register captures the Frame Size and Queue Number of the first occurrence of the Frame Size related error. Up on clearing it captures the data of immediate next occurrence of a similar error.
- **Size:** 32 bits
- **Offset:** 0xc68
- **Exists:** (DWC_EQOS_AV_EST)

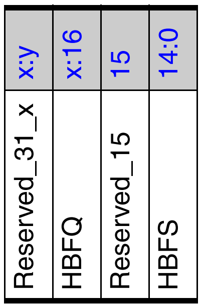


Table 17-235 Fields for Register: MTL_EST_Frm_Size_Capture

Bits	Name	Memory Access	Description
x:y	Reserved_31_x	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[x]:** DWC_EQOS_NUM_TXQ>4?13:DWC_EQOS_NUM_TXQ>2?14:15 + DWC_EQOS_NUM_TXQ>4?19:DWC_EQOS_NUM_TXQ>2?18:17 - 1 **Range Variable[y]:** DWC_EQOS_NUM_TXQ>4?19:DWC_EQOS_NUM_TXQ>2?18:17
x:16	HBFQ	R	Queue Number of HLBF Captures the binary value of the of the first Queue (number) experiencing HLBF error (see HLBF field of status register). Value once written is not altered by any subsequent queue errors of similar nature. Once cleared the queue number of the next occurring HLBF error is captured. Width is based on the number of Tx Queues configured; remaining bits are Read-Only. Cleared when MTL_EST_Frm_Size_Error register is all zeros. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[x]:** DWC_EQOS_NUM_TXQ>4?3:DWC_EQOS_NUM_TXQ>2?2:1 + 15

Bits	Name	Memory Access	Description
15	Reserved_15	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
14:0	HBFS	R	Frame Size of HLBF Captures the Frame Size of the dropped frame related to queue number indicated in HBFQ field of this register. Contents of this register should be considered invalid, if this field is zero. Cleared when MTL_EST_Frm_Size_Error register is all zeros. **Value After Reset:** 0x0 **Exists:** Always

17.2.14 MTL_EST_Intr_Enable

- **Description:** This register implements the Interrupt Enable bits for the various events that generate an interrupt. Bit positions have a 1 to 1 correlation with the status bit positions in MTL_ETS_Status register.
- **Size:** 32 bits
- **Offset:** 0xc70
- **Exists:** (DWC_EQOS_AV_EST)

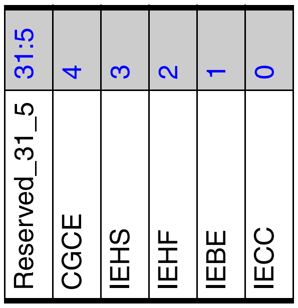


Table 17-236 Fields for Register: MTL_EST_Intr_Enable

Bits	Name	Memory Access	Description
31:5	Reserved_31_5	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
4	CGCE	R/W	Interrupt Enable for CGCE When set, generates interrupt when the Constant Gate Control Error occurs and is indicated in the status. When reset this event does not generate an interrupt **Values:** ■ 0x0 (DISABLE): Interrupt for CGCE is disabled ■ 0x1 (ENABLE): Interrupt for CGCE is enabled **Value After Reset:** 0x0 **Exists:** Always
3	IEHS	R/W	Interrupt Enable for HLBS When set, generates interrupt when the Head-of-Line Blocking due to Scheduling issue and is indicated in the status. When reset this event does not generate an interrupt. **Values:** ■ 0x0 (DISABLE): Interrupt for HLBS is disabled ■ 0x1 (ENABLE): Interrupt for HLBS is enabled **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
2	IEHF	R/W	Interrupt Enable for HLBF When set, generates interrupt when the Head-of-Line Blocking due to Frame Size error occurs and is indicated in the status. When reset this event does not generate an interrupt. **Values:** ■ 0x0 (DISABLE): Interrupt for HLBF is disabled ■ 0x1 (ENABLE): Interrupt for HLBF is enabled **Value After Reset:** 0x0 **Exists:** Always
1	IEBE	R/W	Interrupt Enable for BTR Error When set, generates interrupt when the BTR Error occurs and is indicated in the status. When reset this event does not generate an interrupt. **Values:** ■ 0x0 (DISABLE): Interrupt for BTR Error is disabled ■ 0x1 (ENABLE): Interrupt for BTR Error is enabled **Value After Reset:** 0x0 **Exists:** Always
0	IECC	R/W	Interrupt Enable for Switch List When set, generates interrupt when the configuration change is successful and the hardware has switched to the new list. When reset this event does not generate an interrupt. **Values:** ■ 0x0 (DISABLE): Interrupt for Switch List is disabled ■ 0x1 (ENABLE): Interrupt for Switch List is enabled **Value After Reset:** 0x0 **Exists:** Always

17.2.15 MTL_EST_GCL_Control

- **Description:** This register provides the control information for reading/writing to the Gate Control lists.
- **Size:** 32 bits
- **Offset:** 0xc80
- **Exists:** (DWC_EQOS_AV_EST)

31:24	23:22	21	20	19:y	x:8	7:6	5	4	3	2	1	0
Reserved_31_24	ESTEIEC	ESTEIEE	ERR0	Reserved_19_y	ADDR	Reserved_7_6	DBGB	DBGM	Reserved_3	GCRR	R1W0	SRWO

Table 17-237 Fields for Register: MTL_EST_GCL_Control

Bits	Name	Memory Access	Description
31:24	Reserved_31_24	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
23:22	ESTEIEC	* Varies	ECC Inject Error Control for EST Memory When EIEE bit of this register is set, following are the errors inserted based on the value encoded in this field. This filed will be valid only if DWC_EQOS_ASP_ECC feature is selected during the configuration, else it will be reserved. **Values:** ■ 0x0 (1BIT): Insert 1 bit error ■ 0x1 (2BIT): Insert 2 bit errors ■ 0x2 (3BIT): Insert 3 bit errors ■ 0x3 (1BIT_ADDR): Insert 1 bit error in address field **Value After Reset:** 0x0 **Exists:** Always **Memory Access:** ((DWC_EQOS_ASP_ECC)?\"read-write\":\"read-only\")

Bits	Name	Memory Access	Description
21	ESTEIEE	* Varies	EST ECC Inject Error Enable When set along with EEST bit of MTL_EST_Control register, enables the ECC error injection feature. When reset, disables the ECC error injection feature. **Values:** ■ 0x0 (DISABLE): EST ECC Inject Error is disabled ■ 0x1 (ENABLE): EST ECC Inject Error is enabled **Value After Reset:** 0x0 **Exists:** Always **Memory Access:** ((DWC_EQOS_ASP_ECC)?\"read-write\":\"read-only\")
20	ERR0	R/W	When set indicates the last write operation was aborted as software writes to GCL and GCL registers is prohibited when SSWL bit of MTL_EST_Control Register is set. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. **Values:** ■ 0x0 (DISABLE): ERR0 is disabled ■ 0x1 (ENABLE): ERR1 is enabled **Value After Reset:** 0x0 **Exists:** Always
19:y	Reserved_19_y	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[y]:** DWC_EQOS_EMAW + 8

Bits	Name	Memory Access	Description
x:8	ADDR	R/W	Gate Control List Address: (GCLA when GCRR is "0"). Provides the address (row number) of the Gate Control List at which the R/W operation has to be performed. By default the Gate Control List pointed by SWOL of MTL_EST_Status is selected for R/W, however if the DBGM bit of this register is set, a debug mode access is given to R/W from DBGB. The width of this field is dependent on DWC_EQOS_EST_DEP; unused bits should be treated as read only. Gate Control list Related Registers Address: (GCRA when GCRR is "1"). By default the GCL related register set pointed by SWOL of MTL_EST_Status is selected for R/W, however if the DBGM bit of this register is set, a debug mode access is given to R/W from DBGB. Lower 3 bits are only used in this mode, higher order bits are treated as dont cares. ■ 000: BTR Low (31:0) ■ 001: BTR High (63:31) ■ 010: CTR Low (31:0) ■ 011: CTR High (39:32) ■ 100: TER (31:0) ■ 101: LLR (n:0) (where n is (log{DWC_EQOS_EST_DEP} / log2)) ■ Others: Reserved **Value After Reset:** 0x0 **Exists:** Always **Range Variable[x]:** DWC_EQOS_EMAW + 7
7:6	Reserved_7_6	R/W	Reserved. **Value After Reset:** 0x0 **Exists:** Always
5	DBGB	R/W	Debug Mode Bank Select When set to "0" indicates R/W in debug mode should be directed to Bank 0 (GCL0 and corresponding Time related registers). When set to "1" indicates R/W in debug mode should be directed to Bank 1 (GCL1 and corresponding Time related registers). This value is used when DBGM is set and overrides by value of SWOL which is normally used. **Values:** ■ 0x0 (BANK0): R/W in debug mode should be directed to Bank 0 ■ 0x1 (BANK1): R/W in debug mode should be directed to Bank 1 **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
4	DBGM	R/W	Debug Mode When set to "1" indicates R/W in debug mode where the memory bank (for GCL and Time related registers) is explicitly provided by DBGB value, when set to "0" SWOL bit is used to determine which bank to use. **Values:** ■ 0x0 (DISABLE): Debug Mode is disabled ■ 0x1 (ENABLE): Debug Mode is enabled **Value After Reset:** 0x0 **Exists:** Always
3	Reserved_3	R/W	Reserved. **Value After Reset:** 0x0 **Exists:** Always
2	GCRR	R/W	Gate Control Related Registers When set to "1" indicates the R/W access is for the GCL related registers (BTR, CTR, TER, LLR) whose address is provided by GCRA. When "0" indicates R/W should be directed to GCL from the address provided by GCLA. **Values:** ■ 0x0 (DISABLE): Gate Control Related Registers are disabled ■ 0x1 (ENABLE): Gate Control Related Registers are enabled **Value After Reset:** 0x0 **Exists:** Always
1	R1W0	R/W	Read '1', Write '0': When set to '1': Read Operation When set to '0': Write Operation. **Values:** ■ 0x0 (WRITE): Write Operation ■ 0x1 (READ): Read Operation **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
0	SRWO	R/W	Start Read/Write Op When set indicates a Read/Write Op has started and is in progress. When reset by hardware indicates the R/W Op has completed or an error has occurred (when bit 20 is set) Reads: Data can be read from MTL_EST_GCL_Data register after this bit is reset Writes: MTL_EST_GCL_Data should be programmed with write data before setting SRWO. Access restriction applies. Setting 1 sets. Self-cleared. Setting 0 has no effect. **Values:** ■ 0x0 (DISABLE): Start Read/Write Op disabled ■ 0x1 (ENABLE): Start Read/Write Op enabled **Value After Reset:** 0x0 **Exists:** Always

17.2.16 MTL_EST_GCL_Data

- **Description:** This register holds the read data or write data in case of reads and writes respectively.
- **Size:** 32 bits
- **Offset:** 0xc84
- **Exists:** (DWC_EQOS_AV_EST)

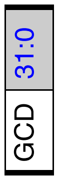


Table 17-238 Fields for Register: MTL_EST_GCL_Data

Bits	Name	Memory Access	Description
31:0	GCD	R/W	Gate Control Data The data corresponding to the address selected in the GCL_Control register. Used for both Read and Write operations. **Value After Reset:** 0x0 **Exists:** Always

17.2.17 MTL_FPE_CTRL_STS

- **Description:** This register controls the operation of, and provides status for Frame Preemption (IEEE802.1Qbu/802.3br).
- **Size:** 32 bits
- **Offset:** 0xc90
- **Exists:** (DWC_EQOS_FPE)

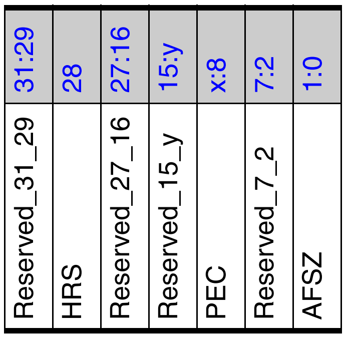


Table 17-239 Fields for Register: MTL_FPE_CTRL_STS

Bits	Name	Memory Access	Description
31:29	Reserved_31_29	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
28	HRS	R	Hold/Release Status ■ 1: Indicates a Set-and-Hold-MAC operation was last executed and the pMAC is in Hold State. ■ 0: Indicates a Set-and-Release-MAC operation was last executed and the pMAC is in Release State. **Values:** ■ 0x0 (SET_REL): Indicates a Set-and-Release-MAC operation was last executed and the pMAC is in Release State ■ 0x1 (SET_HOLD): Indicates a Set-and-Hold-MAC operation was last executed and the pMAC is in Hold State **Value After Reset:** 0x0 **Exists:** Always
27:16	Reserved_27_16	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
15:y	Reserved_15_y	R	Reserved. **Value After Reset:** 0x0 **Exists:** DWC_EQOS_NUM_TXQ!=8 **Range Variable[y]:** DWC_EQOS_NUM_TXQ + 8

Bits	Name	Memory Access	Description
x:8	PEC	R/W	Preemption Classification When set indicates the corresponding Queue must be classified as preemptable, when '0' Queue is classified as express. When both EST (Qbv) and Preemption are enabled, Queue-0 is always assumed to be preemptable. When EST (Qbv) is enabled Queues categorized as preemptable here are always assumed to be in "Open" state in the Gate Control List. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[x]:** DWC_EQOS_NUM_TXQ + 7
7:2	Reserved_7_2	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
1:0	AFSZ	R/W	Additional Fragment Size used to indicate, in units of 64 bytes, the minimum number of bytes over 64 bytes required in non-final fragments of preempted frames. The minimum non-final fragment size is (AFSZ +1) * 64 bytes **Value After Reset:** 0x0 **Exists:** Always

17.2.18 MTL_FPE_Advance

- **Description:** This register holds the Hold and Release Advance time.
- **Size:** 32 bits
- **Offset:** 0xc94
- **Exists:** (DWC_EQOS_FPE&&DWC_EQOS_AV_EST)

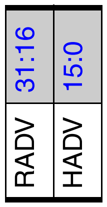


Table 17-240 Fields for Register: MTL_FPE_Advance

Bits	Name	Memory Access	Description
31:16	RADV	R/W	Release Advance The maximum time in nanoseconds that can elapse between issuing a RELEASE to the MAC and the MAC being ready to resume transmission of preemptable frames, in the absence of there being any express frames available for transmission. **Value After Reset:** 0x0 **Exists:** Always
15:0	HADV	R/W	Hold Advance The maximum time in nanoseconds that can elapse between issuing a HOLD to the MAC and the MAC ceasing to transmit any preemptable frame that is in the process of transmission or any preemptable frames that are queued for transmission. **Value After Reset:** 0x0 **Exists:** Always

17.2.19 MTL_RXP_Control_Status

- **Description:** The MTL_RXP_Control_Status register establishes the operating mode of Rx Parser and provides some status.
- **Size:** 32 bits
- **Offset:** 0xca0
- **Exists:** (DWC_EQOS_FRP_EN)

31	30:y	23:16	15:y	7:0
RXPI	Reserved_30_x	NPE	Reserved_15_x	NVE

Table 17-241 Fields for Register: MTL_RXP_Control_Status

Bits	Name	Memory Access	Description
31	RXPI	R	RX Parser in Idle state This status bit is set to 1 when the Rx parser is in Idle State and waiting for a new packet for processing. This bit is used as a handshake with software when parser gets disables. After disabling, when bit is set then software can update the Rx parser instruction table. **Values:** ■ 0x0 (INACTIVE): RX Parser not in Idle state ■ 0x1 (ACTIVE): RX Parser in Idle state **Value After Reset:** 0x1 **Exists:** Always
30:y	Reserved_30_x	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[y]:** DWC_EQOS_FRP_OK_INDEXW + 16
23:16	NPE	R/W	Number of parsable entries in the Instruction table This control indicates the number of parsable entries in the Instruction Memory. This is used in Rx parser logic to detect programming Error. In case number of parsed entries for a packet is more than this entry then NPEOVIS bit in the MTL_RXP_Interrupt_Control_Status register is set. **Value After Reset:** (DWC_EQOS_FRP_ENTRIES-1) **Exists:** Always

Bits	Name	Memory Access	Description
15:y	Reserved_15_x	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[y]:** DWC_EQOS_FRP_OK_INDEXW
7:0	NVE	R/W	Number of valid entries in the Instruction table This control indicates the number of valid entries in the Instruction Memory. This is used in Rx parser logic to detect any programming Error. In case while parsing Table address (memory address) found to be more than this entry then NVEOVIS bit in the MTL_RXP_Interrupt_Control_Status register is set. Note: The minimum value of this should be 2. **Value After Reset:** (DWC_EQOS_FRP_ENTRIES-1) **Exists:** Always

17.2.20 MTL_RXP_Interrupt_Control_Status

- **Description:** The MTL_RXP_Interrupt_Control_Status registers provides enable control for the interrupts and provides interrupt status.
- **Size:** 32 bits
- **Offset:** 0xca4
- **Exists:** (DWC_EQOS_FRP_EN)

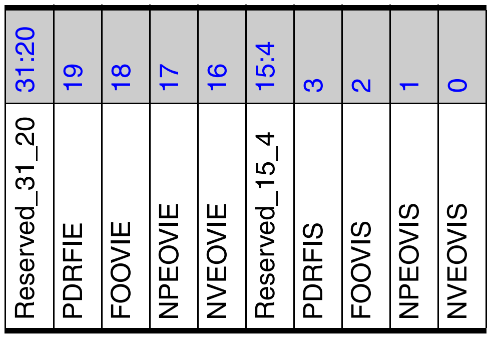


Table 17-242 Fields for Register: MTL_RXP_Interrupt_Control_Status

Bits	Name	Memory Access	Description
31:20	Reserved_31_20	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
19	PDRFIE	R/W	Packet Drop due to RF Interrupt Enable When this bit is set, the PDRFIS interrupt is enabled. When this bit is reset, the PDRFIS interrupt is disabled. **Values:** ■ 0x0 (DISABLE): Packet Drop due to RF Interrupt is disabled ■ 0x1 (ENABLE): Packet Drop due to RF Interrupt is enabled **Value After Reset:** 0x0 **Exists:** Always
18	FOOVIE	R/W	Frame Offset Overflow Interrupt Enable When this bit is set, the FOOVIS interrupt is enabled. When this bit is reset, the FOOVIS interrupt is disabled. **Values:** ■ 0x0 (DISABLE): Frame Offset Overflow Interrupt is disabled ■ 0x1 (ENABLE): Frame Offset Overflow Interrupt is enabled **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
17	NPEOVIE	R/W	Number of Parsable Entries Overflow Interrupt Enable When this bit is set, the NPEOVIS interrupt is enabled. When this bit is reset, the NPEOVIS interrupt is disabled. **Values:** ■ 0x0 (DISABLE): Number of Parsable Entries Overflow Interrupt is disabled ■ 0x1 (ENABLE): Number of Parsable Entries Overflow Interrupt is enabled **Value After Reset:** 0x0 **Exists:** Always
16	NVEOVIE	R/W	Number of Valid Entries Overflow Interrupt Enable When this bit is set, the NVEOVIS interrupt is enabled. When this bit is reset, the NVEOVIS interrupt is disabled. **Values:** ■ 0x0 (DISABLE): Number of Valid Entries Overflow Interrupt is disabled ■ 0x1 (ENABLE): Number of Valid Entries Overflow Interrupt is enabled **Value After Reset:** 0x0 **Exists:** Always
15:4	Reserved_15_4	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
3	PDRFIS	R/W	Packet Dropped due to RF Interrupt Status If the Rx Parser result says to drop the packet by setting RF=1 in the instruction memory, then this bit is set to 1. This bit is cleared when the application writes 1 to this bit. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. **Values:** ■ 0x0 (INACTIVE): Packet Dropped due to RF Interrupt Status not detected ■ 0x1 (ACTIVE): Packet Dropped due to RF Interrupt Status detected **Value After Reset:** 0x0 **Exists:** Always **Testable:** untestable

Bits	Name	Memory Access	Description
2	FOOVIS	R/W	Frame Offset Overflow Interrupt Status While parsing if the Instruction table entry's 'Frame Offset' found to be more than EOF offset, then then this bit is set. This bit is cleared when the application writes 1 to this bit. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. **Values:** ■ 0x0 (INACTIVE): Frame Offset Overflow Interrupt Status not detected ■ 0x1 (ACTIVE): Frame Offset Overflow Interrupt Status detected **Value After Reset:** 0x0 **Exists:** Always **Testable:** untestable
1	NPEOVIS	R/W	Number of Parsable Entries Overflow Interrupt Status While parsing a packet if the number of parsed entries found to be more than NPE[] (Number of Parseable Entries in MTL_RXP_Control register),then this bit is set to 1. This bit is cleared when the application writes 1 to this bit. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. **Values:** ■ 0x0 (INACTIVE): Number of Parsable Entries Overflow Interrupt Status not detected ■ 0x1 (ACTIVE): Number of Parsable Entries Overflow Interrupt Status detected **Value After Reset:** 0x0 **Exists:** Always **Testable:** untestable
0	NVEOVIS	R/W	Number of Valid Entries Overflow Interrupt Status While parsing if the Instruction address found to be more than NVE (Number of Valid Entries in MTL_RXP_Control register), then this bit is set to 1. This bit is cleared when the application writes 1 to this bit. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. **Values:** ■ 0x0 (INACTIVE): Number of Valid Entries Overflow Interrupt Status not detected ■ 0x1 (ACTIVE): Number of Valid Entries Overflow Interrupt Status detected **Value After Reset:** 0x0 **Exists:** Always **Testable:** untestable

17.2.21 MTL_RXP_Drop_Cnt

- **Description:** The MTL_RXP_Drop_Cnt register provides the drop count of Rx Parser initiated drops.
- **Size:** 32 bits
- **Offset:** 0xca8
- **Exists:** (DWC_EQOS_FRP_EN)

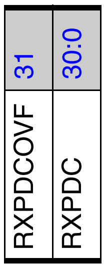


Table 17-243 Fields for Register: MTL_RXP_Drop_Cnt

Bits	Name	Memory Access	Description
31	RXPDCOVF	R	Rx Parser Drop Counter Overflow Bit When set, this bit indicates that the MTL_RXP_Drop_cnt (RXPDC) Counter field crossed the maximum limit. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): Rx Parser Drop count overflow not occurred ■ 0x1 (ACTIVE): Rx Parser Drop count overflow occurred **Value After Reset:** 0x0 **Exists:** Always
30:0	RXPDC	R	Rx Parser Drop count This 31-bit counter is implemented whenever a Rx Parser Drops a packet due to RF =1. The counter is cleared when the register is read. **Value After Reset:** 0x0 **Exists:** Always

17.2.22 MTL_RXP_Error_Cnt

- **Description:** The MTL_RXP_Error_Cnt register provides the Rx Parser related error occurrence count.
- **Size:** 32 bits
- **Offset:** 0xcac
- **Exists:** (DWC_EQOS_FRP_EN)

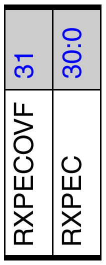


Table 17-244 Fields for Register: MTL_RXP_Error_Cnt

Bits	Name	Memory Access	Description
31	RXPECOVF	R	Rx Parser Error Counter Overflow Bit When set, this bit indicates that the MTL_RXP_Error_cnt (RXPEC) Counter field crossed the maximum limit. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): Rx Parser Error count overflow not occurred ■ 0x1 (ACTIVE): Rx Parser Error count overflow occurred **Value After Reset:** 0x0 **Exists:** Always
30:0	RXPEC	R	Rx Parser Error count This 31-bit counter is implemented whenever a Rx Parser encounters following Error scenarios ■ Entry address >= NVE[] ■ Number Parsed Entries >= NPE[] ■ Entry address > EOF data entry address The counter is cleared when the register is read. **Value After Reset:** 0x0 **Exists:** Always

17.2.23 MTL_RXP_Indirect_Acc_Control_Status

- **Description:** The MTL_RXP_Indirect_Acc_Control_Status register provides the Indirect Access control and status for Rx Parser memory.
- **Size:** 32 bits
- **Offset:** 0xcb0
- **Exists:** (DWC_EQOS_FRP_EN)

31	30:23	22:21	20	19:17	16	15:y	x:0
STARTBUSY	Reserved_30_23	RXPEIEC	RXPEIEE	Reserved_19_17	WRRDN	Reserved_15_x	ADDR

Table 17-245 Fields for Register: MTL_RXP_Indirect_Acc_Control_Status

Bits	Name	Memory Access	Description
31	STARTBUSY	R/W	FRP Instruction Table Access Busy When this bit is set to 1 by the software then it indicates to start the Read/Write operation from/to the Rx Parser Memory. Software should read this bit as 0 before issuing read or write request to access the Parser Memory Instructions. This bit when set to 1 indicates that hardware is busy until its gets cleared by hardware and software should not issue any read or write operation. **Values:** ■ 0x0 (INACTIVE): hardware not busy ■ 0x1 (ACTIVE): hardware is busy (Read/Write operation from/to the Rx Parser Memory) **Value After Reset:** 0x0 **Exists:** Always **Testable:** untestable
30:23	Reserved_30_23	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
22:21	RXPEIEC	R/W	ECC Inject Error Control for Rx Parser Memory When EIEE bit of this register is set, following are the errors inserted based on the value encoded in this field. **Values:** ■ 0x0 (1BIT): Insert 1 bit error ■ 0x1 (2BIT): Insert 2 bit errors ■ 0x2 (3BIT): Insert 3 bit errors ■ 0x3 (1BIT_ADDR): Insert 1 bit error in address field **Value After Reset:** 0x0 **Exists:** DWC_EQOS_ASP_ECC **Testable:** untestable
20	RXPEIEE	R/W	ECC Inject Error Enable for Rx Parser Memory When set, enables the ECC error injection feature. When reset, disables the ECC error injection feature. **Values:** ■ 0x0 (DISABLE): ECC Inject Error for Rx Parser Memory is disabled ■ 0x1 (ENABLE): ECC Inject Error for Rx Parser Memory is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_ASP_ECC **Testable:** untestable
19:17	Reserved_19_17	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
16	WRRDN	R/W	Read Write Control When this bit is set to 1 indicates the write operation to the Rx Parser Memory. When this bit is set to 0 indicates the read operation to the Rx Parser Memory. **Values:** ■ 0x0 (READ): Read operation to the Rx Parser Memory ■ 0x1 (WRITE): Write operation to the Rx Parser Memory **Value After Reset:** 0x0 **Exists:** Always
15:y	Reserved_15_x	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[y]:** DWC_EQOS_FRP_IND_ADDDW

Bits	Name	Memory Access	Description
x:0	ADDR	R/W	FRP Instruction Table Offset Address This field indicates the ADDR of the 32-bit entry in Rx parser instruction table. Each entry has 128-bit (4x32-bit words). The X depends on the configurable DWC_EQOS_FRP_ENTRIES If DWC_EQOS_FRP_ENTRIES == 256 , then X = 9 If DWC_EQOS_FRP_ENTRIES = 128 , then X = 8 IF DWC_EQOS_FRP_ENTRIES = 64, then X = 7 This is can be written by the software before issuing any Read/Write command. The hardware auto-increments this field after the read-write operation gets completed. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[x]:** (DWC_EQOS_FRP_IND_ADDDW) - 1

17.2.24 MTL_RXP_Indirect_Acc_Data

- **Description:** The MTL_RXP_Indirect_Acc_Data registers holds the data associated to Indirect Access to Rx Parser memory.
- **Size:** 32 bits
- **Offset:** 0xcb4
- **Exists:** (DWC_EQOS_FRP_EN)

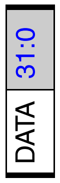


Table 17-246 Fields for Register: MTL_RXP_Indirect_Acc_Data

Bits	Name	Memory Access	Description
31:0	DATA	R	FRP Instruction Table Write/Read Data Software should write this register before issuing any write command. The hardware provides the read data from the Rx Parser Memory for read operation when STARTBUSY =0 after read command. **Value After Reset:** 0x0 **Exists:** Always

17.2.25 MTL_ECC_Control

- **Description:** The MTL_ECC_Control register establishes the operating mode of ECC related to MTL memories.
- **Size:** 32 bits
- **Offset:** 0xcc0
- **Exists:** (DWC_EQOS_ASP_ECC)

31:9	8	7:5	4	3	2	1	0
Reserved_31_9	MEEAO	Reserved_7_5	TSOEE	MRXPEE	MESTEE	MRXEE	MTXEE

Table 17-247 Fields for Register: MTL_ECC_Control

Bits	Name	Memory Access	Description
31:9	Reserved_31_9	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
8	MEEAO	R/W	MTL ECC Error Address Status Over-ride When set, the following error address fields will hold the last valid address where the error is detected. When reset, the following error address fields will hold the first address where the error is detected. EUEAS/ECEAS of MTL_ECC_Err_Addr_Status register. **Values:** ■ 0x0 (DISABLE): MTL ECC Error Address Status Over-ride is disabled ■ 0x1 (ENABLE): MTL ECC Error Address Status Over-ride is enabled **Value After Reset:** 0x0 **Exists:** Always
7:5	Reserved_7_5	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
4	TSOEE	R/W	TSO memory ECC Enable When set to 1, enables the ECC feature for TSO memory in DMA. When set to zero, disables the ECC feature for TSO memory in DMA. **Values:** ■ 0x0 (DISABLE): TSO memory ECC is disabled ■ 0x1 (ENABLE): TSO memory ECC is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_TSO_MEM_EN
3	MRXPEE	R/W	MTL Rx Parser ECC Enable When set to 1, enables the ECC feature for Rx Parser memory. When set to zero, disables the ECC feature for Rx Parser memory. **Values:** ■ 0x0 (DISABLE): MTL Rx Parser ECC is disabled ■ 0x1 (ENABLE): MTL Rx Parser ECC is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_FRP_EN
2	MESTEE	R/W	MTL EST ECC Enable When set to 1, enables the ECC feature for EST memory. When set to zero, disables the ECC feature for EST memory. **Values:** ■ 0x0 (DISABLE): MTL EST ECC is disabled ■ 0x1 (ENABLE): MTL EST ECC is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_AV_EST
1	MRXEE	R/W	MTL Rx FIFO ECC Enable When set to 1, enables the ECC feature for MTL Rx FIFO memory. When set to zero, disables the ECC feature for MTL Rx FIFO memory. **Values:** ■ 0x0 (DISABLE): MTL Rx FIFO ECC is disabled ■ 0x1 (ENABLE): MTL Rx FIFO ECC is enabled **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
0	MTXEE	R/W	MTL Tx FIFO ECC Enable When set to 1, enables the ECC feature for MTL Tx FIFO memory. When set to zero, disables the ECC feature for MTL Tx FIFO memory. **Values:** ■ 0x0 (DISABLE): MTL Tx FIFO ECC is disabled ■ 0x1 (ENABLE): MTL Tx FIFO ECC is enabled **Value After Reset:** 0x0 **Exists:** Always

17.2.26 MTL_Safety_Interrupt_Status

- **Description:** The MTL_Safety_Interrupt_Status registers provides Safety interrupt status.
- **Size:** 32 bits
- **Offset:** 0xcc4
- **Exists:** (DWC_EQOS_ASP_ECC)

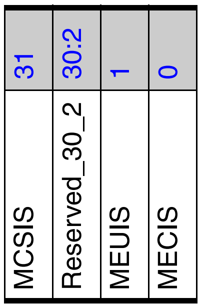


Table 17-248 Fields for Register: MTL_Safety_Interrupt_Status

Bits	Name	Memory Access	Description
31	MCSIS	R	MAC Safety Uncorrectable Interrupt Status Indicates an uncorrectable Safety-related Interrupt is set in the MAC module. MAC_DPP_FSM_Interrupt_Status register should be read when this bit is set, to get the cause of the Safety Interrupt in MAC. **Values:** ■ 0x0 (INACTIVE): MAC Safety Uncorrectable Interrupt Status not detected ■ 0x1 (ACTIVE): MAC Safety Uncorrectable Interrupt Status detected **Value After Reset:** 0x0 **Exists:** Always
30:2	Reserved_30_2	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
1	MEUIS	R	MTL ECC Uncorrectable error Interrupt Status This bit indicates that an uncorrectable error interrupt event in the MTL ECC safety feature. To get the exact cause of the interrupt the application should read the MTL_ECC_Interrupt_Status register. **Values:** ■ 0x0 (INACTIVE): MTL ECC Uncorrectable error Interrupt Status not detected ■ 0x1 (ACTIVE): MTL ECC Uncorrectable error Interrupt Status detected **Value After Reset:** 0x0 **Exists:** Always
0	MECIS	R	MTL ECC Correctable error Interrupt Status This bit indicates that a correctable error interrupt event in the MTL ECC safety feature. To get the exact cause of the interrupt the application should read the MTL_ECC_Interrupt_Status register. **Values:** ■ 0x0 (INACTIVE): MTL ECC Correctable error Interrupt Status not detected ■ 0x1 (ACTIVE): MTL ECC Correctable error Interrupt Status detected **Value After Reset:** 0x0 **Exists:** Always

17.2.27 MTL_ECC_Interrupt_Enable

- **Description:** The MTL_ECC_Interrupt_Enable register provides enable bits for the ECC interrupts.
- **Size:** 32 bits
- **Offset:** 0xcc8
- **Exists:** (DWC_EQOS_ASP_ECC)

31:13	12	11:9	8	7:5	4	3:1	0
Reserved_31_13	RPCEIE	Reserved_11_9	ECEIE	Reserved_7_5	RXCEIE	Reserved_3_1	TXCEIE

Table 17-249 Fields for Register: MTL_ECC_Interrupt_Enable

Bits	Name	Memory Access	Description
31:13	Reserved_31_13	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
12	RPCEIE	R/W	Rx Parser memory Correctable Error Interrupt Enable When set, generates an interrupt when an uncorrectable error is detected at the Rx Parser memory interface. It is indicated in RPCES status bit of MTL_ECC_Interrupt_Status register. When reset this event does not generates an interrupt. **Values:** ■ 0x0 (DISABLE): Rx Parser memory Correctable Error Interrupt is disabled ■ 0x1 (ENABLE): Rx Parser memory Correctable Error Interrupt is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_FRP_EN
11:9	Reserved_11_9	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
8	ECEIE	R/W	EST memory Correctable Error Interrupt Enable When set, generates an interrupt when a correctable error is detected at the MTL EST memory interface. It is indicated in the ECES bit of MTL_ECC_Interrupt_Status register. When reset this event does not generates an interrupt. **Values:** ■ 0x0 (DISABLE): EST memory Correctable Error Interrupt is disabled ■ 0x1 (ENABLE): EST memory Correctable Error Interrupt is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_AV_EST
7:5	Reserved_7_5	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
4	RXCEIE	R/W	Rx memory Correctable Error Interrupt Enable When set, generates an interrupt when a correctable error is detected at the MTL Rx memory interface. It is indicated in the RXCES bit of MTL_ECC_Interrupt_Status register. When reset this event does not generates an interrupt. **Values:** ■ 0x0 (DISABLE): Rx memory Correctable Error Interrupt is disabled ■ 0x1 (ENABLE): Rx memory Correctable Error Interrupt is enabled **Value After Reset:** 0x0 **Exists:** Always
3:1	Reserved_3_1	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
0	TXCEIE	R/W	Tx memory Correctable Error Interrupt Enable When set, generates an interrupt when a correctable error is detected at the MTL Tx memory interface. It is indicated in the TXCES bit of MTL_ECC_Interrupt_Status register. When reset this event does not generates an interrupt. **Values:** ■ 0x0 (DISABLE): Tx memory Correctable Error Interrupt is disabled ■ 0x1 (ENABLE): Tx memory Correctable Error Interrupt is enabled **Value After Reset:** 0x0 **Exists:** Always

17.2.28 MTL_ECC_Interrupt_Status

- **Description:** The MTL_ECC_Interrupt_Status register provides MTL ECC Interrupt Status.
- **Size:** 32 bits
- **Offset:** 0xccc
- **Exists:** (DWC_EQOS_ASP_ECC)

31:15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved_31_15	RPUES	RPAMS	RPCES	Reserved_11	EUES	EAMS	ECES	Reserved_7	RXUES	RXAMS	RXCES	Reserved_3	TXUES	TXAMS	TXCES

Table 17-250 Fields for Register: MTL_ECC_Interrupt_Status

Bits	Name	Memory Access	Description
31:15	Reserved_31_15	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
14	RPUES	R/W	Rx Parser memory Uncorrectable Error Status When set, indicates that an uncorrectable error is detected at Rx Parser memory interface. **Values:** ■ 0x0 (INACTIVE): Rx Parser memory Uncorrectable Error Status not detected ■ 0x1 (ACTIVE): Rx Parser memory Uncorrectable Error Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_FRP_EN **Testable:** untestable
13	RPAMS	R/W	MTL Rx Parser memory Address Mismatch Status This bit when set indicates that address mismatch is found for address bus of Rx Parser memory. **Values:** ■ 0x0 (INACTIVE): MTL Rx Parser memory Address Mismatch Status not detected ■ 0x1 (ACTIVE): MTL Rx Parser memory Address Mismatch Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_FRP_EN **Testable:** untestable

Bits	Name	Memory Access	Description
12	RPCES	R/W	MTL Rx Parser memory Correctable Error Status This bit when set indicates that correctable error is detected at RX Parser memory interface. **Values:** ■ 0x0 (INACTIVE): MTL Rx Parser memory Correctable Error Status not detected ■ 0x1 (ACTIVE): MTL Rx Parser memory Correctable Error Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_FRP_EN **Testable:** untestable
11	Reserved_11	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
10	EUES	R/W	MTL EST memory Uncorrectable Error Status When set, indicates that an uncorrectable error is detected at MTL EST memory interface. **Values:** ■ 0x0 (INACTIVE): MTL EST memory Uncorrectable Error Status not detected ■ 0x1 (ACTIVE): MTL EST memory Uncorrectable Error Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_AV_EST **Testable:** untestable
9	EAMS	R/W	MTL EST memory Address Mismatch Status This bit when set indicates that address mismatch is found for address bus of MTL EST memory. **Values:** ■ 0x0 (INACTIVE): MTL EST memory Address Mismatch Status not detected ■ 0x1 (ACTIVE): MTL EST memory Address Mismatch Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_AV_EST **Testable:** untestable

Bits	Name	Memory Access	Description
8	ECES	R/W	MTL EST memory Correctable Error Status This bit when set indicates that correctable error is detected at the MTL EST memory. **Values:** ■ 0x0 (INACTIVE): MTL EST memory Correctable Error Status not detected ■ 0x1 (ACTIVE): MTL EST memory Correctable Error Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_AV_EST **Testable:** untestable
7	Reserved_7	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
6	RXUES	R/W	MTL Rx memory Uncorrectable Error Status When set, indicates that an uncorrectable error is detected at the MTL Rx memory interface. **Values:** ■ 0x0 (INACTIVE): MTL Rx memory Uncorrectable Error Status not detected ■ 0x1 (ACTIVE): MTL Rx memory Uncorrectable Error Status detected **Value After Reset:** 0x0 **Exists:** Always **Testable:** untestable
5	RXAMS	R/W	MTL Rx memory Address Mismatch Status This bit when set indicates that address mismatch is found for address bus of the MTL Rx memory. **Values:** ■ 0x0 (INACTIVE): MTL Rx memory Address Mismatch Status not detected ■ 0x1 (ACTIVE): MTL Rx memory Address Mismatch Status detected **Value After Reset:** 0x0 **Exists:** Always **Testable:** untestable

Bits	Name	Memory Access	Description
4	RXCES	R/W	MTL Rx memory Correctable Error Status This bit when set indicates that correctable error is detected at the MTL Rx memory. **Values:** ■ 0x0 (INACTIVE): MTL Rx memory correctable Error Status not detected ■ 0x1 (ACTIVE): MTL Rx memory correctable Error Status detected **Value After Reset:** 0x0 **Exists:** Always **Testable:** untestable
3	Reserved_3	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
2	TXUES	R/W	MTL Tx memory Uncorrectable Error Status When set, indicates that an uncorrectable error is detected at the MTL TX memory interface. **Values:** ■ 0x0 (INACTIVE): MTL Tx memory Uncorrectable Error Status not detected ■ 0x1 (ACTIVE): MTL Tx memory Uncorrectable Error Status detected **Value After Reset:** 0x0 **Exists:** Always **Testable:** untestable
1	TXAMS	R/W	MTL Tx memory Address Mismatch Status This bit when set indicates that address mismatch is found for address bus of the MTL Tx memory. **Values:** ■ 0x0 (INACTIVE): MTL Tx memory Address Mismatch Status not detected ■ 0x1 (ACTIVE): MTL Tx memory Address Mismatch Status detected **Value After Reset:** 0x0 **Exists:** Always **Testable:** untestable

Bits	Name	Memory Access	Description
0	TXCES	R/W	MTL Tx memory Correctable Error Status This bit when set indicates that a correctable error is detected at the MTL Tx memory. **Values:** ■ 0x0 (INACTIVE): MTL Tx memory Correctable Error Status not detected ■ 0x1 (ACTIVE): MTL Tx memory Correctable Error Status detected **Value After Reset:** 0x0 **Exists:** Always **Testable:** untestable

17.2.29 MTL_ECC_Err_Sts_Rctl

- **Description:** The MTL_ECC_Err_Sts_Rctl register establishes the control for ECC Error status capture.
- **Size:** 32 bits
- **Offset:** 0xcd0
- **Exists:** (DWC_EQOS_ASP_ECC)

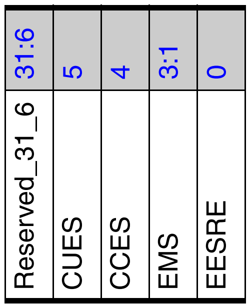


Table 17-251 Fields for Register: MTL_ECC_Err_Sts_Rctl

Bits	Name	Memory Access	Description
31:6	Reserved_31_6	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
5	CUES	R/W	Clear Uncorrectable Error Status When this bit is set along with EESRE bit of this register, based on the EMS field of this register, the respective memory's uncorrectable error address and uncorrectable error count values will be cleared upon reading. Hardware resets this bit when all the error status values are cleared. **Values:** ■ 0x0 (INACTIVE): Clear Uncorrectable Error Status not detected ■ 0x1 (ACTIVE): Clear Uncorrectable Error Status detected **Value After Reset:** 0x0 **Exists:** Always **Testable:** untestable

Bits	Name	Memory Access	Description
4	CCES	R/W	Clear Correctable Error Status When this bit is set along with EESRE bit of this register, based on the EMS field of this register, the respective memory's correctable error address and correctable error count values will be cleared upon reading. Hardware resets this bit when all the error status values are cleared. **Values:** ■ 0x0 (INACTIVE): Clear Correctable Error Status not detected ■ 0x1 (ACTIVE): Clear Correctable Error Status detected **Value After Reset:** 0x0 **Exists:** Always **Testable:** untestable
3:1	EMS	R/W	MTL ECC Memory Selection When EESRE bit of this register is set, this field indicates which memory's error status value to be read. The memory selection encoding is as described below. **Values:** ■ 0x0 (TX_MEM): MTL Tx memory ■ 0x1 (RX_MEM): MTL Rx memory ■ 0x2 (EST_MEM): MTL EST memory ■ 0x3 (RXP_MEM): MTL Rx Parser memory ■ 0x4 (TSO_MEM): DMA TSO memory **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
0	EESRE	R/W	MTL ECC Error Status Read Enable When this bit is set, based on the EMS field of this register, the respective memory's error status values will be captured as described below ■ The correctable and uncorrectable error count values will be captured into MTL_ECC_Err_Cnt_Status register ■ The address location's of correctable and uncorrectable errors will be captured into MTL_ECC_Err_Addr_Status register. Hardware resets this bit when all the status values are captured into the MTL_ECC_Err_Cnt_Status and MTL_ECC_Err_Addr_Status registers. **Values:** ■ 0x0 (DISABLE): MTL ECC Error Status Read is disabled ■ 0x1 (ENABLE): MTL ECC Error Status Read is enabled **Value After Reset:** 0x0 **Exists:** Always **Testable:** untestable

17.2.30 MTL_ECC_Err_Addr_Status

- **Description:** The MTL_ECC_Err_Addr_Status register provides the memory addresses for the correctable and uncorrectable errors.
- **Size:** 32 bits
- **Offset:** 0xcd4
- **Exists:** (DWC_EQOS_ASP_ECC)

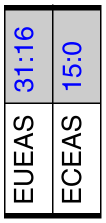


Table 17-252 Fields for Register: MTL_ECC_Err_Addr_Status

Bits	Name	Memory Access	Description
31:16	EUEAS	R	MTL ECC Uncorrectable Error Address Status Based on the EMS field of MTL_ECC_Err_Sts_Rctl register, this field holds the respective memory's address locations for which an uncorrectable error or address mismatch is detected. When MEEAO bit of MTL_ECC_Control register is set, this field holds the last valid address of memory for which either an uncorrectable error or an address mismatch is detected. When MEEAO bit of MTL_ECC_Control register is reset, this field holds the first address of the memory for which either an uncorrectable error or address mismatch is detected. **Value After Reset:** 0x0 **Exists:** Always
15:0	ECEAS	R	MTL ECC Correctable Error Address Status Based on the EMS field of MTL_ECC_Err_Sts_Rctl register, this field holds the respective memory's address locations for which a correctable error is detected. When MEEAO bit of MTL_ECC_Control register is set, this field holds the last valid address of memory for which correctable error or address mismatch is detected. When MEEAO bit of MTL_ECC_Control register is reset, this field holds the first address of the memory for which correctable error is detected. **Value After Reset:** 0x0 **Exists:** Always

17.2.31 MTL_ECC_Err_Cntr_Status

- **Description:** The MTL_ECC_Err_Cntr_Status register provides ECC Error count for Correctable and uncorrectable errors.
- **Size:** 32 bits
- **Offset:** 0xcd8
- **Exists:** (DWC_EQOS_ASP_ECC)

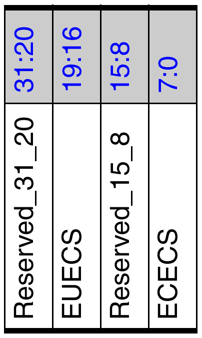


Table 17-253 Fields for Register: MTL_ECC_Err_Cntr_Status

Bits	Name	Memory Access	Description
31:20	Reserved_31_20	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
19:16	EUECS	R	MTL ECC Uncorrectable Error Counter Status Based on the EMS field of MTL_ECC_Err_Cntr_Rctl register, this field holds the respective memory's uncorrectable error count value. **Value After Reset:** 0x0 **Exists:** Always
15:8	Reserved_15_8	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
7:0	ECECS	R	MTL ECC Correctable Error Counter Status Based on the EMS field of MTL_ECC_Err_Cntr_Rctl register, this field holds the respective memory's correctable error count value. **Value After Reset:** 0x0 **Exists:** Always

17.2.32 MTL_DPP_Control

- **Description:** The MTL_DPP_Control establishes the operating mode of Data Parity protection and error injection.
- **Size:** 32 bits
- **Offset:** 0xce0
- **Exists:** ((DWC_EQOS_ASP_ALL))

31:14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved_31_14	IPECW	IPEASW	IPERD	IPETD	IPETSO	IPEDDC	IPEMRF	IPEMTS	IPEMC	IPEID	Reserved_3	EPSI	OPE	EDPP

Table 17-254 Fields for Register: MTL_DPP_Control

Bits	Name	Memory Access	Description
31:14	Reserved_31_14	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
13	IPECW	R/W	Insert Parity error in CSR Read data parity generator When set to 1, parity bit of first valid data generated by the CSR parity generator (or at PG10 as shown in Fig.AXI slave Interface Data path parity protection) is flipped. Hardware will clear this bit once respective parity bit is flipped. **Values:** ■ 0x0 (DISABLE): Insert Parity error in CSR Read data parity generator is disabled ■ 0x1 (ENABLE): Insert Parity error in CSR Read data parity generator is enabled **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_AXI_SLAVE||DWC_EQOS_ASP_PPE) **Testable:** untestable

Bits	Name	Memory Access	Description
12	IPEASW	R/W	Insert Parity error in AXI Slave Write data parity generator When set to 1, parity bit of first valid data generated by the AXI parity generator is (or at PG9 as shown in Fig.AXI slave Interface Data path parity protection) flipped. Hardware will clear this bit once respective parity bit is flipped. **Values:** ■ 0x0 (DISABLE): Insert Parity error in AXI Slave Write data parity generator is disabled ■ 0x1 (ENABLE): Insert Parity error in AXI Slave Write data parity generator is enabled **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_CSR_PORT==3||DWC_EQOS_CSR_PORT==5) **Testable:** untestable
11	IPERD	R/W	Insert Parity error in Rx write-back Descriptor parity generator When set to 1, parity bit of first valid data generated by the DMA Rx write-back descriptor parity generator(or at PG8 as shown in Fig.Receive data path parity protection) is flipped. **Values:** ■ 0x0 (DISABLE): Insert Parity error in Rx write-back Descriptor parity generator is disabled ■ 0x1 (ENABLE): Insert Parity error in Rx write-back Descriptor parity generator is enabled **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_SYS>=2) **Testable:** untestable
10	IPETD	R/W	Insert Parity error in Tx write-back Descriptor parity generator When set to 1, parity bit of first valid data generated by the DMA Tx write-back descriptor parity generator(or at PG4 as shown in Fig.Transmit data path parity protection) is flipped. Hardware will clear this bit once respective parity bit is flipped. **Values:** ■ 0x0 (DISABLE): Insert Parity error in Tx write-back Descriptor parity generator is disabled ■ 0x1 (ENABLE): Insert Parity error in Tx write-back Descriptor parity generator is enabled **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_SYS>=2) **Testable:** untestable

Bits	Name	Memory Access	Description
9	IPETSO	R/W	Insert Parity Error in DMA TSO parity generator When set to 1, parity bit of first valid data generated by the DMA TSO parity generator is (or at PG3 as shown in Fig.Transmit data path parity protection) flipped. Hardware will clear this bit once respective parity bit is flipped. **Values:** ■ 0x0 (DISABLE): Insert Parity Error in DMA TSO parity generator is disabled ■ 0x1 (ENABLE): Insert Parity Error in DMA TSO parity generator is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_TSO_EN **Testable:** untestable
8	IPEDDC	R/W	Insert Parity Error in DMA DTX Control word parity generator When set to 1, parity bit of first valid data generated by the DMA DTX Control word parity generator (or at PG2 as shown in Fig.Transmit data path parity protection) is flipped. Hardware will clear this bit once respective parity bit is flipped. **Values:** ■ 0x0 (DISABLE): Insert Parity Error in DMA DTX Control word parity generator is disabled ■ 0x1 (ENABLE): Insert Parity Error in DMA DTX Control word parity generator is enabled **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_SYS>=2) **Testable:** untestable
7	IPEMRF	R/W	Insert Parity Error in MTL Rx FIFO read control parity generator When set to 1, parity bit of first valid data generated by the MTL Rx FIFO read control parity generator (or at PG7 as shown in Fig.Receive data path parity protection) is flipped. Hardware will clear this bit once respective parity bit is flipped. **Values:** ■ 0x0 (DISABLE): Insert Parity Error in MTL Rx FIFO read control parity generator is disabled ■ 0x1 (ENABLE): Insert Parity Error in MTL Rx FIFO read control parity generator is enabled **Value After Reset:** 0x0 **Exists:** Always **Testable:** untestable

Bits	Name	Memory Access	Description
6	IPEMTS	R/W	Insert Parity Error in MTL Tx Status parity generator When set to 1, parity bit of first valid data generated by the MTL Tx Status parity generator (or at PG6 as shown in Fig.Transmit data path parity protection) is flipped. Hardware will clear this bit once respective parity bit is flipped. **Values:** ■ 0x0 (DISABLE): Insert Parity Error in MTL Tx Status parity generator is disabled ■ 0x1 (ENABLE): Insert Parity Error in MTL Tx Status parity generator is enabled **Value After Reset:** 0x0 **Exists:** !DWC_EQOS_TX_STS_DROP **Testable:** untestable
5	IPEMC	R/W	Insert Parity Error in MTL checksum parity generator When set to 1, parity bit of first valid data generated by the MTL checksum parity generator (or at PG5 as shown in Fig.Transmit data path parity protection) is flipped. Hardware will clear this bit once the respective parity bit is flipped. **Values:** ■ 0x0 (DISABLE): Insert Parity Error in MTL checksum parity generator is disabled ■ 0x1 (ENABLE): Insert Parity Error in MTL checksum parity generator is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_TX_COE **Testable:** untestable

Bits	Name	Memory Access	Description
4	IPEID	R/W	Insert Parity Error in Interface Data parity generator When set to 1, parity bit of first valid input data generated by the Interface data parity generator (or at PG1 as shown in Fig.Transmit data path parity protection) is flipped. Following are the input data bus on which parity bits are generated based on configuration selected In AHB Config, hrdata_i In AXI config, rdata_m_i In DMA Config, mdc_rdata_i In MTL Config, ati_data_i Hardware will clear this bit once the respective parity bit is flipped. **Values:** ■ 0x0 (DISABLE): Insert Parity Error in Interface Data parity generator is disabled ■ 0x1 (ENABLE): Insert Parity Error in Interface Data parity generator is enabled **Value After Reset:** 0x0 **Exists:** !DWC_EQOS_ASP_PPE **Testable:** untestable
3	Reserved_3	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
2	EPSI	R/W	Enable Parity on Slave Interface port When set to 1, enables the parity check for the slave interface ports and disables the internal generation of parity for the input slave data port. When set to 0, disables the parity check for the slave interface ports and enables the internal parity generation for the input slave data port. **Values:** ■ 0x0 (DISABLE): Parity on Slave Interface port is disabled ■ 0x1 (ENABLE): Parity on Slave Interface port is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_ASP_PPE **Testable:** untestable
1	OPE	R/W	Odd Parity Enable When set to 1, enables odd parity protection on all the external interfaces and when set to 0, enables even parity protection on all the external interfaces. **Values:** ■ 0x0 (DISABLE): Odd Parity is disabled ■ 0x1 (ENABLE): Odd Parity is enabled **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
0	EDPP	R/W	Enable Data path Parity Protection When set to 1, enables the parity protection for EQOS datapath by generating and checking the parity on EQOS datapath. When set to 0, disables the parity protection for EQOS datapath. **Values:** ■ 0x0 (DISABLE): Data path Parity Protection is disabled ■ 0x1 (ENABLE): Data path Parity Protection is enabled **Value After Reset:** 0x0 **Exists:** Always

17.3 EQOS_MTL_Q0 Registers

17.3.1 MTL_TxQ0_Operation_Mode

- **Description:** The Queue 0 Transmit Operation Mode register establishes the Transmit queue operating modes and commands.
- **Size:** 32 bits
- **Offset:** 0xd00
- **Exists:** (DWC_EQOS_NUM_TXQ>0&&!DWC_EQOS_CORE)

31:y	x:16	15:7	6:4	3:2	1	0
Reserved_31_y	TQS	Reserved_15_7	TTC	TXQEN	TSF	FTQ

Table 17-255 Fields for Register: MTL_TxQ0_Operation_Mode

Bits	Name	Memory Access	Description
31:y	Reserved_31_y	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
x:16	TQS	R/W	Transmit Queue Size This field indicates the size of the allocated Transmit queues in blocks of 256 bytes. The TQS field is read-write only if the number of Tx Queues more than one, the reset value is 0x0 and indicates size of 256 bytes. When the number of Tx Queues is one, the field is read-only and the configured TX FIFO size in blocks of 256 bytes is reflected in the reset value. The width of this field depends on the Tx memory size selected in your configuration. For example, if the memory size is 2048, the width of this field is 3 bits: LOG2(2048/256) = LOG2(8) = 3 bits **Value After Reset:** (DWC_EQOS_L256_TXMS_MAX) **Exists:** (DWC_EQOS_NUM_TXQ>1) **Range Variable[x]:** DWC_EQOS_L256_TXMS + 15
15:7	Reserved_15_7	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
6:4	TTC	R/W	Transmit Threshold Control These bits control the threshold level of the MTL Tx Queue. The transmission starts when the packet size within the MTL Tx Queue is larger than the threshold. In addition, full packets with length less than the threshold are also transmitted. These bits are used only when the TSF bit is reset. **Values:** ■ 0x0 (32BYTES): 32 ■ 0x1 (64BYTES): 64 ■ 0x2 (96BYTES): 96 ■ 0x3 (128BYTES): 128 ■ 0x4 (192BYTES): 192 ■ 0x5 (256BYTES): 256 ■ 0x6 (384BYTES): 384 ■ 0x7 (512BYTES): 512 **Value After Reset:** 0x0 **Exists:** Always
3:2	TXQEN	R/W	Transmit Queue Enable This field is used to enable/disable the transmit queue 0. ■ 2'b00: Not enabled ■ 2'b01: Reserved ■ 2'b10: Enabled ■ 2'b11: Reserved This field is Read Only in Single Queue configurations and Read Write in Multiple Queue configurations. **Note**: In multiple Tx queues configuration, all the queues are disabled by default. Enable the Tx queue by programming this field. **Values:** ■ 0x0 (DISABLE): Not enabled ■ 0x1 (EN_IF_AV): Enable in AV mode (Reserved in non-AV) ■ 0x2 (ENABLE): Enabled ■ 0x3 (RSVD2): Reserved **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_NUM_TXQ>1)

Bits	Name	Memory Access	Description
1	TSF	R/W	Transmit Store and Forward When this bit is set, the transmission starts when a full packet resides in the MTL Tx queue. When this bit is set, the TTC values specified in Bits[6:4] of this register are ignored. This bit should be changed only when the transmission is stopped. **Values:** ■ 0x0 (DISABLE): Transmit Store and Forward is disabled ■ 0x1 (ENABLE): Transmit Store and Forward is enabled **Value After Reset:** 0x0 **Exists:** Always
0	FTQ	R/W	Flush Transmit Queue When this bit is set, the Tx queue controller logic is reset to its default values. Therefore, all the data in the Tx queue is lost or flushed. This bit is internally reset when the flushing operation is complete. Until this bit is reset, you should not write to the MTL_TxQ1_Operation_Mode register. The data which is already accepted by the MAC transmitter is not flushed. It is scheduled for transmission and results in underflow and runt packet transmission. **Note**: The flush operation is complete only when the Tx queue is empty and the application has accepted the pending Tx Status of all transmitted packets. To complete this flush operation, the PHY Tx clock (clk_tx_i) should be active. Access restriction applies. Setting 1 sets. Self-cleared. Setting 0 has no effect. **Values:** ■ 0x0 (DISABLE): Flush Transmit Queue is disabled ■ 0x1 (ENABLE): Flush Transmit Queue is enabled **Value After Reset:** 0x0 **Exists:** Always **Testable:** untestable

17.3.2 MTL_TxQ0_Underflow

- **Description:** The Queue 0 Underflow Counter register contains the counter for packets aborted because of Transmit queue underflow and packets missed because of Receive queue packet flush
- **Size:** 32 bits
- **Offset:** 0xd04
- **Exists:** (DWC_EQOS_NUM_TXQ>0&&!DWC_EQOS_CORE)

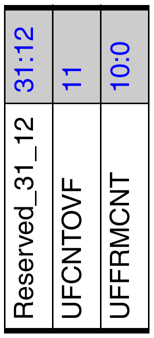


Table 17-256 Fields for Register: MTL_TxQ0_Underflow

Bits	Name	Memory Access	Description
31:12	Reserved_31_12	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
11	UFCNTOVF	R	Overflow Bit for Underflow Packet Counter This bit is set every time the Tx queue Underflow Packet Counter field overflows, that is, it has crossed the maximum count. In such a scenario, the overflow packet counter is reset to all-zeros and this bit indicates that the rollover happened. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): Overflow not detected for Underflow Packet Counter ■ 0x1 (ACTIVE): Overflow detected for Underflow Packet Counter **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
10:0	UFFRMCNT	R	Underflow Packet Counter This field indicates the number of packets aborted by the controller because of Tx Queue Underflow. This counter is incremented each time the MAC aborts outgoing packet because of underflow. The counter is cleared when this register is read with mci_be_i[0] at 1'b1. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Value After Reset:** 0x0 **Exists:** Always

17.3.3 MTL_TxQ0_Debug

- **Description:** The Queue 0 Transmit Debug register gives the debug status of various blocks related to the Transmit queue.
- **Size:** 32 bits
- **Offset:** 0xd08
- **Exists:** (DWC_EQOS_NUM_TXQ>0&&!DWC_EQOS_CORE)

31:23	22:20	19	18:16	15:6	5	4	3	2:1	0
Reserved_31_23	STXSTSF	Reserved_19	PTXQ	Reserved_15_6	TXSTSFSTS	TXQSTS	TWCSTS	TRCSTS	TXQPAUSED

Table 17-257 Fields for Register: MTL_TxQ0_Debug

Bits	Name	Memory Access	Description
31:23	Reserved_31_23	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
22:20	STXSTSF	R	Number of Status Words in Tx Status FIFO of Queue This field indicates the current number of status in the Tx Status FIFO of this queue. When the DTXSTS bit of MTL_Operation_Mode register is set to 1, this field does not reflect the number of status words in Tx Status FIFO. **Value After Reset:** 0x0 **Exists:** !DWC_EQOS_TX_STS_DROP
19	Reserved_19	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
18:16	PTXQ	R	Number of Packets in the Transmit Queue This field indicates the current number of packets in the Tx Queue. When the DTXSTS bit of MTL_Operation_Mode register is set to 1, this field does not reflect the number of packets in the Transmit queue. **Value After Reset:** 0x0 **Exists:** !DWC_EQOS_TX_STS_DROP

Bits	Name	Memory Access	Description
15:6	Reserved_15_6	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
5	TXSTSFSTS	R	MTL Tx Status FIFO Full Status When high, this bit indicates that the MTL Tx Status FIFO is full. Therefore, the MTL cannot accept any more packets for transmission. **Values:** ■ 0x0 (INACTIVE): MTL Tx Status FIFO Full status is not detected ■ 0x1 (ACTIVE): MTL Tx Status FIFO Full status is detected **Value After Reset:** 0x0 **Exists:** !DWC_EQOS_TX_STS_DROP
4	TXQSTS	R	MTL Tx Queue Not Empty Status When this bit is high, it indicates that the MTL Tx Queue is not empty and some data is left for transmission. **Values:** ■ 0x0 (INACTIVE): MTL Tx Queue Not Empty status is not detected ■ 0x1 (ACTIVE): MTL Tx Queue Not Empty status is detected **Value After Reset:** 0x0 **Exists:** Always
3	TWCSTS	R	MTL Tx Queue Write Controller Status When high, this bit indicates that the MTL Tx Queue Write Controller is active, and it is transferring the data to the Tx Queue. **Values:** ■ 0x0 (INACTIVE): MTL Tx Queue Write Controller status is not detected ■ 0x1 (ACTIVE): MTL Tx Queue Write Controller status is detected **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
2:1	TRCSTS	R	MTL Tx Queue Read Controller Status This field indicates the state of the Tx Queue Read Controller: **Values:** ■ 0x0 (IDLE): Idle state ■ 0x1 (READ): Read state (transferring data to the MAC transmitter) ■ 0x2 (WAIT): Waiting for pending Tx Status from the MAC transmitter ■ 0x3 (FLUSH): Flushing the Tx queue because of the Packet Abort request from the MAC **Value After Reset:** 0x0 **Exists:** Always
0	TXQPAUSED	R	Transmit Queue in Pause When this bit is high and the Rx flow control is enabled, it indicates that the Tx Queue is in the Pause condition (in the full-duplex only mode) because of the following: ■ Reception of the PFC packet for the priorities assigned to the Tx Queue when PFC is enabled ■ Reception of 802.3x Pause packet when PFC is disabled **Values:** ■ 0x0 (INACTIVE): Transmit Queue in Pause status is not detected ■ 0x1 (ACTIVE): Transmit Queue in Pause status is detected **Value After Reset:** 0x0 **Exists:** Always

17.3.4 MTL_TxQ0_ETS_Status

- **Description:** The Queue 0 ETS Status register provides the average traffic transmitted in Queue 0.
- **Size:** 32 bits
- **Offset:** 0xd14
- **Exists:** (DWC_EQOS_NUM_TXQ>1&&!DWC_EQOS_CORE)

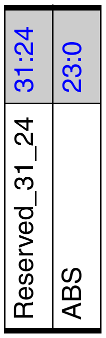


Table 17-258 Fields for Register: MTL_TxQ0_ETS_Status

Bits	Name	Memory Access	Description
31:24	Reserved_31_24	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
23:0	ABS	R	Average Bits per Slot This field contains the average transmitted bits per slot. When the DCB operation is enabled for Queue 0, this field is computed over every 10 million bit times slot (4 ms in 2500 Mbps; 10 ms in 1000 Mbps; 100 ms in 100 Mbps). The maximum value is 0x989680. **Value After Reset:** 0x0 **Exists:** Always

17.3.5 MTL_TxQ0_Quantum_Weight

- **Description:** The Queue 0 Quantum or Weights register contains the quantum value for Deficit Weighted Round Robin (DWRR), weights for the Weighted Round Robin (WRR), and Weighted Fair Queuing (WFQ) for Queue 0.
- **Size:** 32 bits
- **Offset:** 0xd18
- **Exists:** (DWC_EQOS_NUM_TXQ>1&&!DWC_EQOS_CORE)

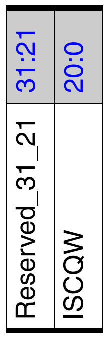


Table 17-259 Fields for Register: MTL_TxQ0_Quantum_Weight

Bits	Name	Memory Access	Description
31:21	Reserved_31_21	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
20:0	ISCQW	R/W	Quantum or Weights When the DCB operation is enabled with DWRR algorithm for Queue 0 traffic, this field contains the quantum value in bytes to be added to credit during every queue scanning cycle. The maximum value is 0x1312D0 bytes. When DCB operation is enabled with WFQ algorithm for Queue 0 traffic, this field contains the weight for this queue. The maximum value is 0x3FFF where weight of 0 indicates 100% bandwidth. Bits[20:14] must be written to zero. The higher the programmed weights lesser the bandwidth allocated for the particular Transmit Queue. This is because the weights are used to compute the packet finish time (weights*packet_size). Lesser the finish time, higher the probability of the packet getting scheduled first and using more bandwidth. When DCB operation or generic queuing operation is enabled with WRR algorithm for Queue 0 traffic, this field contains the weight for this queue. The maximum value is 0x64. Bits [20:7] must be written to zero. **Value After Reset:** 0x0 **Exists:** Always

17.3.6 MTL_Q0_Interrupt_Control_Status

- **Description:** This register contains the interrupt enable and status bits for the queue 0 interrupts.
- **Size:** 32 bits
- **Offset:** 0xd2c
- **Exists:** ((DWC_EQOS_NUM_RXQ>0||DWC_EQOS_NUM_TXQ>0)&&!DWC_EQOS_CORE)

31:25	24	23:17	16	15:10	9	8	7:2	1	0
Reserved_31_25	RXOIE	Reserved_23_17	RXOVFIS	Reserved_15_10	ABPSIE	TXUIE	Reserved_7_2	ABPSIS	TXUNFIS

Table 17-260 Fields for Register: MTL_Q0_Interrupt_Control_Status

Bits	Name	Memory Access	Description
31:25	Reserved_31_25	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
24	RXOIE	R/W	Receive Queue Overflow Interrupt Enable When this bit is set, the Receive Queue Overflow interrupt is enabled. When this bit is reset, the Receive Queue Overflow interrupt is disabled. **Values:** ■ 0x0 (DISABLE): Receive Queue Overflow Interrupt is disabled ■ 0x1 (ENABLE): Receive Queue Overflow Interrupt is enabled **Value After Reset:** 0x0 **Exists:** 0<DWC_EQOS_NUM_RXQ
23:17	Reserved_23_17	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
16	RXOVFIS	R/W	Receive Queue Overflow Interrupt Status This bit indicates that the Receive Queue had an overflow while receiving the packet. If a partial packet is transferred to the application, the overflow status is set in RDES3[21]. This bit is cleared when the application writes 1 to this bit. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. **Values:** ■ 0x0 (INACTIVE): Receive Queue Overflow Interrupt Status not detected ■ 0x1 (ACTIVE): Receive Queue Overflow Interrupt Status detected **Value After Reset:** 0x0 **Exists:** 0<DWC_EQOS_NUM_RXQ **Testable:** untestable
15:10	Reserved_15_10	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
9	ABPSIE	R/W	Average Bits Per Slot Interrupt Enable When this bit is set, the MAC asserts the sbd_intr_o or mci_intr_o interrupt when the average bits per slot status is updated. When this bit is cleared, the interrupt is not asserted for such an event. **Values:** ■ 0x0 (DISABLE): Average Bits Per Slot Interrupt is disabled ■ 0x1 (ENABLE): Average Bits Per Slot Interrupt is enabled **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_NUM_TXQ>1)&&(0<DWC_EQOS_NUM_TXQ)
8	TXUIE	R/W	Transmit Queue Underflow Interrupt Enable When this bit is set, the Transmit Queue Underflow interrupt is enabled. When this bit is reset, the Transmit Queue Underflow interrupt is disabled. **Values:** ■ 0x0 (DISABLE): Transmit Queue Underflow Interrupt Status is disabled ■ 0x1 (ENABLE): Transmit Queue Underflow Interrupt Status is enabled **Value After Reset:** 0x0 **Exists:** 0<DWC_EQOS_NUM_TXQ
7:2	Reserved_7_2	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
1	ABPSIS	R/W	Average Bits Per Slot Interrupt Status When set, this bit indicates that the MAC has updated the ABS value. This bit is cleared when the application writes 1 to this bit. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. **Values:** ■ 0x0 (INACTIVE): Average Bits Per Slot Interrupt Status not detected ■ 0x1 (ACTIVE): Average Bits Per Slot Interrupt Status detected **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_NUM_TXQ>1)&&(0<DWC_EQOS_NUM_TXQ) **Testable:** untestable
0	TXUNFIS	R/W	Transmit Queue Underflow Interrupt Status This bit indicates that the Transmit Queue had an underflow while transmitting the packet. Transmission is suspended and an Underflow Error TDES3[2] is set. This bit is cleared when the application writes 1 to this bit. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. **Values:** ■ 0x0 (INACTIVE): Transmit Queue Underflow Interrupt Status not detected ■ 0x1 (ACTIVE): Transmit Queue Underflow Interrupt Status detected **Value After Reset:** 0x0 **Exists:** 0<DWC_EQOS_NUM_TXQ **Testable:** untestable

17.3.7 MTL_RxQ0_Operation_Mode

- **Description:** The Queue 0 Receive Operation Mode register establishes the Receive queue operating modes and command.
  The RFA and RFD fields are not backward compatible with the RFA and RFD fields of 4.00a release
- **Size:** 32 bits
- **Offset:** 0xd30
- **Exists:** (DWC_EQOS_NUM_RXQ>0&&!DWC_EQOS_CORE)

31:y	x:20	19:y	x:14	13:y	x:8	7	6	5	4	3	2	1:0
Reserved_31_y	RQS	Reserved_19_y	RFD	Reserved_13_y	RFA	EHFC	DIS_TCP_EF	RSF	FEP	FUP	Reserved_2	RTC

Table 17-261 Fields for Register: MTL_RxQ0_Operation_Mode

Bits	Name	Memory Access	Description
31:y	Reserved_31_y	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
x:20	RQS	R/W	Receive Queue Size This field indicates the size of the allocated Receive queues in blocks of 256 bytes. The RQS field is read-write only if the number of Rx Queues more than one, the reset value is 0x0 and indicates size of 256 bytes. When the number of Rx Queues is one, the field is read-only and the configured RX FIFO size in blocks of 256 bytes is reflected in the reset value. The width of this field depends on the Rx memory size selected in your configuration. For example, if the memory size is 2048, the width of this field is 3 bits: LOG2(2048/256) = LOG2(8) = 3 bits **Value After Reset:** 0x0 **Exists:** DWC_EQOS_NUM_RXQ>1 **Range Variable[x]:** DWC_EQOS_L256_RXMS + 19
19:y	Reserved_19_y	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[y]:** DWC_EQOS_FLCW + 14

Bits	Name	Memory Access	Description
x:14	RFD	R/W	Threshold for Deactivating Flow Control (in half-duplex and full-duplex modes) These bits control the threshold (fill-level of Rx queue) at which the flow control is de-asserted after activation: ■ 0: Full minus 1 KB, that is, FULL 1 KB ■ 1: Full minus 1.5 KB, that is, FULL 1.5 KB ■ 2: Full minus 2 KB, that is, FULL 2 KB ■ 3: Full minus 2.5 KB, that is, FULL 2.5 KB ■ ... ■ 62: Full minus 32 KB, that is, FULL 32 KB ■ 63: Full minus 32.5 KB, that is, FULL 32.5 KB The de-assertion is effective only after flow control is asserted. Note: The value must be programmed in such a way to make sure that the threshold is a positive number. When the EHFC is set high, these values are applicable only when the Rx queue size determined by the RQS field of this register, is equal to or greater than 4 KB. For a given queue size, the values ranges between 0 and the encoding for FULL minus (QSIZE - 0.5 KB) and all other values are illegal. Here the term FULL and QSIZE refers to the queue size determined by the RQS field of this register. The width of this field depends on RX FIFO size selected during the configuration. Remaining bits are reserved and read only. **Value After Reset:** 0x0 **Exists:** DWC_EQOS_RXFIFO_SIZE>2048 **Range Variable[x]:** DWC_EQOS_FLCW + 13
13:y	Reserved_13_y	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[y]:** DWC_EQOS_FLCW + 8
x:8	RFA	R/W	Threshold for Activating Flow Control (in half-duplex and full-duplex These bits control the threshold (fill-level of Rx queue) at which the flow control is activated: For more information on encoding for this field, see RFD. **Value After Reset:** 0x0 **Exists:** DWC_EQOS_RXFIFO_SIZE>2048 **Range Variable[x]:** DWC_EQOS_FLCW + 7

Bits	Name	Memory Access	Description
7	EHFC	R/W	Enable Hardware Flow Control When this bit is set, the flow control signal operation, based on the fill-level of Rx queue, is enabled. When reset, the flow control operation is disabled. **Values:** ■ 0x0 (DISABLE): Hardware Flow Control is disabled ■ 0x1 (ENABLE): Hardware Flow Control is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_RXFIFO_SIZE>2048
6	DIS_TCP_EF	R/W	Disable Dropping of TCP/IP Checksum Error Packets When this bit is set, the MAC does not drop the packets which only have the errors detected by the Receive Checksum Offload engine. Such packets have errors only in the encapsulated payload. There are no errors (including FCS error) in the Ethernet packet received by the MAC. When this bit is reset, all error packets are dropped if the FEP bit is reset. **Values:** ■ 0x0 (ENABLE): Dropping of TCP/IP Checksum Error Packets is enabled ■ 0x1 (DISABLE): Dropping of TCP/IP Checksum Error Packets is disabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_RX_COE
5	RSF	R/W	Receive Queue Store and Forward When this bit is set, the DWC_ether_qos reads a packet from the Rx queue only after the complete packet has been written to it, ignoring the RTC field of this register. When this bit is reset, the Rx queue operates in the Threshold (cut-through) mode, subject to the threshold specified by the RTC field of this register. **Values:** ■ 0x0 (DISABLE): Receive Queue Store and Forward is disabled ■ 0x1 (ENABLE): Receive Queue Store and Forward is enabled **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
4	FEP	R/W	Forward Error Packets When this bit is reset, the Rx queue drops packets with error status (CRC error, GMII_ER, watchdog timeout, or overflow). However, if the start byte (write) pointer of a packet is already transferred to the read controller side (in Threshold mode), the packet is not dropped. When this bit is set, all packets except the runt error packets are forwarded to the application or DMA. If the RSF bit is set and the Rx queue overflows when a partial packet is written, the packet is dropped irrespective of the setting of this bit. However, if the RSF bit is reset and the Rx queue overflows when a partial packet is written, a partial packet may be forwarded to the application or DMA. **Values:** ■ 0x0 (DISABLE): Forward Error Packets is disabled ■ 0x1 (ENABLE): Forward Error Packets is enabled **Value After Reset:** 0x0 **Exists:** Always
3	FUP	R/W	Forward Undersized Good Packets When this bit is set, the Rx queue forwards the undersized good packets (packets with no error and length less than 64 bytes), including pad-bytes and CRC. When this bit is reset, the Rx queue drops all packets of less than 64 bytes, unless a packet is already transferred because of the lower value of Rx Threshold, for example, RTC = 01. **Values:** ■ 0x0 (DISABLE): Forward Undersized Good Packets is disabled ■ 0x1 (ENABLE): Forward Undersized Good Packets is enabled **Value After Reset:** 0x0 **Exists:** Always
2	Reserved_2	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
1:0	RTC	R/W	Receive Queue Threshold Control These bits control the threshold level of the MTL Rx queue (in bytes): The received packet is transferred to the application or DMA when the packet size within the MTL Rx queue is larger than the threshold. In addition, full packets with length less than the threshold are automatically transferred. This field is valid only when the RSF bit is zero. This field is ignored when the RSF bit is set to 1. **Values:** ■ 0x0 (64BYTE): 64 ■ 0x1 (32BYTE): 32 ■ 0x2 (96BYTE): 96 ■ 0x3 (128BYTE): 128 **Value After Reset:** 0x0 **Exists:** Always

17.3.8 MTL_RxQ0_Missed_Packet_Overflow_Cnt

- **Description:** The Queue 0 Missed Packet and Overflow Counter register contains the counter for packets missed because of Receive queue packet flush and packets discarded because of Receive queue overflow.
- **Size:** 32 bits
- **Offset:** 0xd34
- **Exists:** (DWC_EQOS_NUM_RXQ>0&&!DWC_EQOS_CORE)

31:28	27	26:16	15:12	11	10:0
Reserved_31_28	MISCNTOVF	MISPKTCNT	Reserved_15_12	OVFCNTOVF	OVFPKTCNT

Table 17-262 Fields for Register: MTL_RxQ0_Missed_Packet_Overflow_Cnt

Bits	Name	Memory Access	Description
31:28	Reserved_31_28	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
27	MISCNTOVF	R	Missed Packet Counter Overflow Bit When set, this bit indicates that the Rx Queue Missed Packet Counter crossed the maximum limit. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): Missed Packet Counter overflow not detected ■ 0x1 (ACTIVE): Missed Packet Counter overflow detected **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
26:16	MISPKTCNT	R	Missed Packet Counter This field indicates the number of packets missed by the DWC_ether_qos because the application asserted ari_pkt_flush_i[] for this queue. This counter is incremented each time the application issues ari_pkt_flush_i[] for this queue. This counter is reset when this register is read with mci_be_i[0] at 1b1. In EQOS-DMA, EQOS-AXI, and EQOS-AHB configurations, This counter is incremented by 1 when the DMA discards the packet because of buffer unavailability. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Value After Reset:** 0x0 **Exists:** Always
15:12	Reserved_15_12	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
11	OVFCNTOVF	R	Overflow Counter Overflow Bit When set, this bit indicates that the Rx Queue Overflow Packet Counter field crossed the maximum limit. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): Overflow Counter overflow not detected ■ 0x1 (ACTIVE): Overflow Counter overflow detected **Value After Reset:** 0x0 **Exists:** Always
10:0	OVFPKTCNT	R	Overflow Packet Counter This field indicates the number of packets discarded by the DWC_ether_qos because of Receive queue overflow. This counter is incremented each time the DWC_ether_qos discards an incoming packet because of overflow. This counter is reset when this register is read with mci_be_i[0] at 1'b1. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Value After Reset:** 0x0 **Exists:** Always

17.3.9 MTL_RxQ0_Debug

- **Description:** The Queue 0 Receive Debug register gives the debug status of various blocks related to the Receive queue.
- **Size:** 32 bits
- **Offset:** 0xd38
- **Exists:** (DWC_EQOS_NUM_RXQ>0&&!DWC_EQOS_CORE)

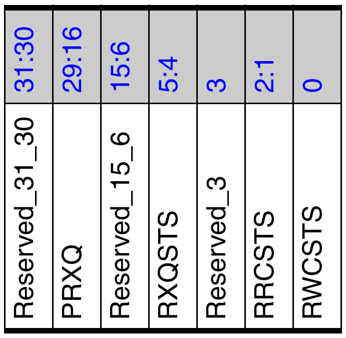


Table 17-263 Fields for Register: MTL_RxQ0_Debug

Bits	Name	Memory Access	Description
31:30	Reserved_31_30	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
29:16	PRXQ	R	Number of Packets in Receive Queue This field indicates the current number of packets in the Rx Queue. The theoretical maximum value for this field is 256KB/16B = 16K Packets, that is, Max_Queue_Size/Min_Packet_Size. **Value After Reset:** 0x0 **Exists:** Always
15:6	Reserved_15_6	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
5:4	RXQSTS	R	MTL Rx Queue Fill-Level Status This field gives the status of the fill-level of the Rx Queue: **Values:** ■ 0x0 (EMPTY): Rx Queue empty ■ 0x1 (BLW_THR): Rx Queue fill-level below flow-control deactivate threshold ■ 0x2 (ABV_THR): Rx Queue fill-level above flow-control activate threshold ■ 0x3 (FULL): Rx Queue full **Value After Reset:** 0x0 **Exists:** Always
3	Reserved_3	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
2:1	RRCSTS	R	MTL Rx Queue Read Controller State This field gives the state of the Rx queue Read controller: **Values:** ■ 0x0 (IDLE): Idle state ■ 0x1 (READ_DATA): Reading packet data ■ 0x2 (READ_STS): Reading packet status (or timestamp) ■ 0x3 (FLUSH): Flushing the packet data and status **Value After Reset:** 0x0 **Exists:** Always
0	RWCSTS	R	MTL Rx Queue Write Controller Active Status When high, this bit indicates that the MTL Rx queue Write controller is active, and it is transferring a received packet to the Rx Queue. **Values:** ■ 0x0 (INACTIVE): MTL Rx Queue Write Controller Active Status not detected ■ 0x1 (ACTIVE): MTL Rx Queue Write Controller Active Status detected **Value After Reset:** 0x0 **Exists:** Always

17.3.10 MTL_RxQ0_Control

- **Description:** The Queue Receive Control register controls the receive arbitration and passing of received packets to the application.
- **Size:** 32 bits
- **Offset:** 0xd3c
- **Exists:** (DWC_EQOS_NUM_RXQ>1&&!DWC_EQOS_CORE)

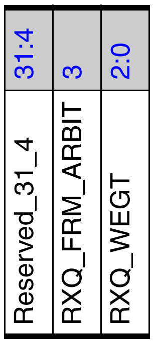


Table 17-264 Fields for Register: MTL_RxQ0_Control

Bits	Name	Memory Access	Description
31:4	Reserved_31_4	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
3	RXQ_FRM_ARBIT	R/W	Receive Queue Packet Arbitration When this bit is set, the DWC_ether_qos drives the packet data to the ARI interface such that the entire packet data of currently-selected queue is transmitted before switching to other queue. When this bit is reset, the DWC_ether_qos drives the packet data to the ARI interface such that the following amount of data of currently-selected queue is transmitted before switching to other queue: ■ PBL amount of data (indicated by ari_qN_pbl_i[]) or ■ Complete data of a packet The status and the timestamp are not a part of the PBL data. Therefore, the DWC_ether_qos drives the complete status (including timestamp status) during first PBL request for the packet (in store-and-forward mode) or the last PBL request for the packet (in Threshold mode). **Values:** ■ 0x0 (DISABLE): Receive Queue Packet Arbitration is disabled ■ 0x1 (ENABLE): Receive Queue Packet Arbitration is enabled **Value After Reset:** ((DWC_EQOS_MTL_SUBSYS&&DWC_EQOS_NUM_RXQ>1)?\"0x1\":\"0x0\") **Exists:** ((DWC_EQOS_MTL_SUBSYS&&DWC_EQOS_NUM_RXQ>1)||(!DWC_EQOS_MTL_SUBSYS&&DWC_EQOS_NUM_DMA_RX_CH>1&&DWC_EQOS_NUM_RXQ>1))
2:0	RXQ_WEGT	R/W	Receive Queue Weight This field indicates the weight assigned to the Rx Queue 0. The weight is used as the number of continuous PBL or packets requests (depending on the RXQ_FRM_ARBIT) allocated to the queue in one arbitration cycle. **Value After Reset:** 0x0 **Exists:** Always

17.4 EQOS_MTL_Q1 Registers

17.4.1 MTL_TxQ(#i)_Operation_Mode (for i = 1; i <= DWC_EQOS_NUM_TXQ-1)

- **Description:** The Queue 1 Transmit Operation Mode register establishes the Transmit queue operating modes and commands.
- **Size:** 32 bits
- **Offset:** (0x0040*i)+0x0D00
- **Exists:** (DWC_EQOS_NUM_TXQ>1&&!DWC_EQOS_CORE)

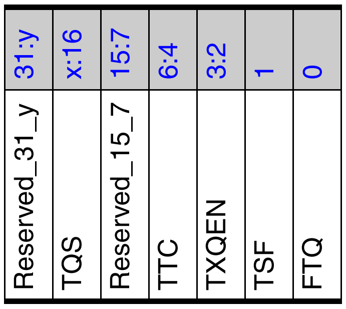


Table 17-265 Fields for Register: MTL_TxQ(#i)_Operation_Mode (for i = 1; i <= DWC_EQOS_NUM_TXQ-1)

Bits	Name	Memory Access	Description
31:y	Reserved_31_y	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
x:16	TQS	R/W	Transmit Queue Size This field indicates the size of the allocated Transmit queues in blocks of 256 bytes. The TQS field is read-write only if the number of Tx Queues more than one, the reset value is 0x0 and indicates size of 256 bytes. When the number of Tx Queues is one, the field is read-only and the configured TX FIFO size in blocks of 256 bytes is reflected in the reset value. The width of this field depends on the Tx memory size selected in your configuration. For example, if the memory size is 2048, the width of this field is 3 bits: LOG2(2048/256) = LOG2(8) = 3 bits **Value After Reset:** (DWC_EQOS_L256_TXMS_MAX) **Exists:** (DWC_EQOS_NUM_TXQ>1) **Range Variable[x]:** DWC_EQOS_L256_TXMS + 15
15:7	Reserved_15_7	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
6:4	TTC	R/W	Transmit Threshold Control These bits control the threshold level of the MTL Tx Queue. The transmission starts when the packet size within the MTL Tx Queue is larger than the threshold. In addition, full packets with length less than the threshold are also transmitted. These bits are used only when the TSF bit is reset. **Values:** ■ 0x0 (32BYTES): 32 ■ 0x1 (64BYTES): 64 ■ 0x2 (96BYTES): 96 ■ 0x3 (128BYTES): 128 ■ 0x4 (192BYTES): 192 ■ 0x5 (256BYTES): 256 ■ 0x6 (384BYTES): 384 ■ 0x7 (512BYTES): 512 **Value After Reset:** 0x0 **Exists:** Always
3:2	TXQEN	R/W	Transmit Queue Enable This field is used to enable/disable the transmit queue 0. ■ 2'b00: Not enabled ■ 2'b01: Enable in AV mode (Reserved when **Enable Audio Video Bridging** is not selected while configuring the core) ■ 2'b10: Enabled ■ 2'b11: Reserved **Note**: In multiple Tx queues configuration, all the queues are disabled by default. Enable the Tx queue by programming this field. **Values:** ■ 0x0 (DISABLE): Not enabled ■ 0x1 (EN_IF_AV): Enable in AV mode (Reserved in non-AV) ■ 0x2 (ENABLE): Enabled ■ 0x3 (RSVD2): Reserved **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_NUM_TXQ>1)

Bits	Name	Memory Access	Description
1	TSF	R/W	Transmit Store and Forward When this bit is set, the transmission starts when a full packet resides in the MTL Tx queue. When this bit is set, the TTC values specified in Bits[6:4] of this register are ignored. This bit should be changed only when the transmission is stopped. **Values:** ■ 0x0 (DISABLE): Transmit Store and Forward is disabled ■ 0x1 (ENABLE): Transmit Store and Forward is enabled **Value After Reset:** 0x0 **Exists:** Always
0	FTQ	R/W	Flush Transmit Queue When this bit is set, the Tx queue controller logic is reset to its default values. Therefore, all the data in the Tx queue is lost or flushed. This bit is internally reset when the flushing operation is complete. Until this bit is reset, you should not write to the MTL_TxQ1_Operation_Mode register. The data which is already accepted by the MAC transmitter is not flushed. It is scheduled for transmission and results in underflow and runt packet transmission. **Note**: The flush operation is complete only when the Tx queue is empty and the application has accepted the pending Tx Status of all transmitted packets. To complete this flush operation, the PHY Tx clock (clk_tx_i) should be active. Access restriction applies. Setting 1 sets. Self-cleared. Setting 0 has no effect. **Values:** ■ 0x0 (DISABLE): Flush Transmit Queue is disabled ■ 0x1 (ENABLE): Flush Transmit Queue is enabled **Value After Reset:** 0x0 **Exists:** Always **Testable:** untestable

17.4.2 MTL_TxQ(#i)_Underflow (for i = 1; i <= DWC_EQOS_NUM_TXQ-1)

- **Description:** The Queue 1 Underflow Counter register contains the counter for packets aborted because of Transmit queue underflow and packets missed because of Receive queue packet flush
- **Size:** 32 bits
- **Offset:** (0x0040*i)+0x0D04
- **Exists:** (DWC_EQOS_NUM_TXQ>1&&!DWC_EQOS_CORE)

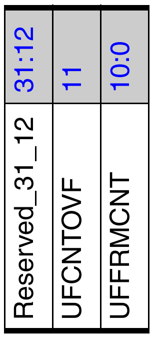


Table 17-266 Fields for Register: MTL_TxQ(#i)_Underflow (for i = 1; i <= DWC_EQOS_NUM_TXQ-1)

Bits	Name	Memory Access	Description
31:12	Reserved_31_12	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
11	UFCNTOVF	R	Overflow Bit for Underflow Packet Counter This bit is set every time the Tx queue Underflow Packet Counter field overflows, that is, it has crossed the maximum count. In such a scenario, the overflow packet counter is reset to all-zeros and this bit indicates that the rollover happened. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): Overflow not detected for Underflow Packet Counter ■ 0x1 (ACTIVE): Overflow detected for Underflow Packet Counter **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
10:0	UFFRMCNT	R	Underflow Packet Counter This field indicates the number of packets aborted by the controller because of Tx Queue Underflow. This counter is incremented each time the MAC aborts outgoing packet because of underflow. The counter is cleared when this register is read with mci_be_i[0] at 1'b1. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Value After Reset:** 0x0 **Exists:** Always

17.4.3 MTL_TxQ(#i)_Debug (for i = 1; i <= DWC_EQOS_NUM_TXQ-1)

- **Description:** The Queue 1 Transmit Debug register gives the debug status of various blocks related to the Transmit queue.
- **Size:** 32 bits
- **Offset:** (0x0040*i)+0x0D08
- **Exists:** (DWC_EQOS_NUM_TXQ>1&&!DWC_EQOS_CORE)

31:23	22:20	19	18:16	15:6	5	4	3	2:1	0
Reserved_31_23	STXSTSF	Reserved_19	PTXQ	Reserved_15_6	TXSTSFSTS	TXQSTS	TWCSTS	TRCSTS	TXQPAUSED

Table 17-267 Fields for Register: MTL_TxQ(#i)_Debug (for i = 1; i <= DWC_EQOS_NUM_TXQ-1)

Bits	Name	Memory Access	Description
31:23	Reserved_31_23	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
22:20	STXSTSF	R	Number of Status Words in Tx Status FIFO of Queue This field indicates the current number of status in the Tx Status FIFO of this queue. When the DTXSTS bit of MTL_Operation_Mode register is set to 1, this field does not reflect the number of status words in Tx Status FIFO. **Value After Reset:** 0x0 **Exists:** !DWC_EQOS_TX_STS_DROP
19	Reserved_19	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
18:16	PTXQ	R	Number of Packets in the Transmit Queue This field indicates the current number of packets in the Tx Queue. When the DTXSTS bit of MTL_Operation_Mode register is set to 1, this field does not reflect the number of packets in the Transmit queue. **Value After Reset:** 0x0 **Exists:** !DWC_EQOS_TX_STS_DROP

Bits	Name	Memory Access	Description
15:6	Reserved_15_6	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
5	TXSTSFSTS	R	MTL Tx Status FIFO Full Status When high, this bit indicates that the MTL Tx Status FIFO is full. Therefore, the MTL cannot accept any more packets for transmission. **Values:** ■ 0x0 (INACTIVE): MTL Tx Status FIFO Full status is not detected ■ 0x1 (ACTIVE): MTL Tx Status FIFO Full status is detected **Value After Reset:** 0x0 **Exists:** !DWC_EQOS_TX_STS_DROP
4	TXQSTS	R	MTL Tx Queue Not Empty Status When this bit is high, it indicates that the MTL Tx Queue is not empty and some data is left for transmission. **Values:** ■ 0x0 (INACTIVE): MTL Tx Queue Not Empty status is not detected ■ 0x1 (ACTIVE): MTL Tx Queue Not Empty status is detected **Value After Reset:** 0x0 **Exists:** Always
3	TWCSTS	R	MTL Tx Queue Write Controller Status When high, this bit indicates that the MTL Tx Queue Write Controller is active, and it is transferring the data to the Tx Queue. **Values:** ■ 0x0 (INACTIVE): MTL Tx Queue Write Controller status is not detected ■ 0x1 (ACTIVE): MTL Tx Queue Write Controller status is detected **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
2:1	TRCSTS	R	MTL Tx Queue Read Controller Status This field indicates the state of the Tx Queue Read Controller: **Values:** ■ 0x0 (IDLE): Idle state ■ 0x1 (READ): Read state (transferring data to the MAC transmitter) ■ 0x2 (WAIT): Waiting for pending Tx Status from the MAC transmitter ■ 0x3 (FLUSH): Flushing the Tx queue because of the Packet Abort request from the MAC **Value After Reset:** 0x0 **Exists:** Always
0	TXQPAUSED	R	Transmit Queue in Pause When this bit is high and the Rx flow control is enabled, it indicates that the Tx Queue is in the Pause condition (in the full-duplex only mode) because of the following: ■ Reception of the PFC packet for the priorities assigned to the Tx Queue when PFC is enabled ■ Reception of 802.3x Pause packet when PFC is disabled **Values:** ■ 0x0 (INACTIVE): Transmit Queue in Pause status is not detected ■ 0x1 (ACTIVE): Transmit Queue in Pause status is detected **Value After Reset:** 0x0 **Exists:** Always

17.4.4 MTL_TxQ(#i)_ETS_Control (for i = 1; i <= DWC_EQOS_NUM_TXQ-1)

- **Description:** The Queue ETS Control register controls the enhanced transmission selection operation.
- **Size:** 32 bits
- **Offset:** (0x0040*i)+0x0D10
- **Exists:** (DWC_EQOS_TX_AV_EN&&(DWC_EQOS_TX-_AV_STRTQ<=1)&&(DWC_EQOS_NUM_TXQ>1))

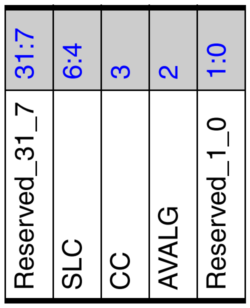


Table 17-268 Fields for Register: MTL_TxQ(#i)_ETS_Control (for i = 1; i <= DWC_EQOS_NUM_TXQ-1)

Bits	Name	Memory Access	Description
31:7	Reserved_31_7	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
6:4	SLC	R/W	Slot Count If the credit-based shaper algorithm is enabled, the software can program the number of slots (of duration programmed in DMA_CH[n]_Slot_Interval register) over which the average transmitted bits per slot, provided in the MTL_TxQ[n]_ETS_Status register, need to be computed for Queue. The encoding is as follows: **Values:** ■ 0x0 (1_SLOT): 1 slot ■ 0x1 (2_SLOT): 2 slots ■ 0x2 (4_SLOT): 4 slots ■ 0x3 (8_SLOT): 8 slots ■ 0x4 (16_SLOT): 16 slots ■ 0x5 (RSVD): Reserved **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
3	CC	R/W	Credit Control When this bit is set, the accumulated credit parameter in the credit-based shaper algorithm logic is not reset to zero when there is positive credit and no packet to transmit in Channel 1. The credit accumulates even when there is no packet waiting in Channel 1 and another channel is transmitting. When this bit is reset, the accumulated credit parameter in the credit-based shaper algorithm logic is set to zero when there is positive credit and no packet to transmit in Channel 1. When there is no packet waiting in Channel 1 and other channel is transmitting, no credit is accumulated. **Values:** ■ 0x0 (DISABLE): Credit Control is disabled ■ 0x1 (ENABLE): Credit Control is enabled **Value After Reset:** 0x0 **Exists:** Always
2	AVALG	R/W	AV Algorithm When Queue 1 is programmed for AV, this field configures the scheduling algorithm for this queue: This bit when set, indicates credit based shaper algorithm (CBS) is selected for Queue 1 traffic. When reset, strict priority is selected. **Values:** ■ 0x0 (DISABLE): CBS Algorithm is disabled ■ 0x1 (ENABLE): CBS Algorithm is enabled **Value After Reset:** 0x0 **Exists:** Always
1:0	Reserved_1_0	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always

17.4.5 MTL_TxQ(#i)_ETS_Status (for i = 1; i <= DWC_EQOS_NUM_TXQ-1)

- **Description:** The Queue 1 ETS Status register provides the average traffic transmitted in Queue 1.
- **Size:** 32 bits
- **Offset:** (0x0040*i)+0x0D14
- **Exists:** (DWC_EQOS_NUM_TXQ>1&&!DWC_EQOS_CORE)

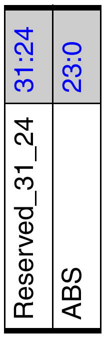


Table 17-269 Fields for Register: MTL_TxQ(#i)_ETS_Status (for i = 1; i <= DWC_EQOS_NUM_TXQ-1)

Bits	Name	Memory Access	Description
31:24	Reserved_31_24	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
23:0	ABS	R	Average Bits per Slot This field contains the average transmitted bits per slot. If AV operation is enabled, this field is computed over number of slots, programmed in the SLC field of MTL_TxQ[n]_ETS_CONTROL register. The maximum value of this field is 0x6_4000 in 100 Mbps, 0x3E_8000 in 1000 Mbps and 9C_4000 in 2500 Mbps mode respectively. When the DCB operation is enabled for Queue, this field is computed over every 10 million bit times slot (4 ms in 2500 Mbps; 10 ms in 1000 Mbps; 100 ms in 100 Mbps). The maximum value is 0x989680. **Value After Reset:** 0x0 **Exists:** Always

17.4.6 MTL_TxQ(#i)_Quantum_Weight (for i = 1; i <= DWC_EQOS_NUM_TXQ-1)

- **Description:** The Queue 1 idleSlopeCredit, Quantum or Weights register provides the average traffic transmitted in Queue 1.
- **Size:** 32 bits
- **Offset:** (0x0040*i)+0x0D18
- **Exists:** (DWC_EQOS_NUM_TXQ>1&&!DWC_EQOS_CORE)

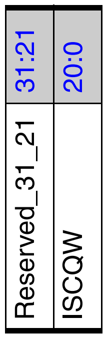


Table 17-270 Fields for Register: MTL_TxQ(#i)_Quantum_Weight (for i = 1; i <= DWC_EQOS_NUM_TXQ-1)

Bits	Name	Memory Access	Description
31:21	Reserved_31_21	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
20:0	ISCQW	R/W	idleSlopeCredit, Quantum or Weights ■ **idleSlopeCredit** When AV feature is enabled, this field contains the idleSlopeCredit value required for the credit-based shaper algorithm for Queue 1. This is the rate of change of credit in bits per cycle (40 ns for 100 Mbps; 8 ns for 1000 Mbps; 3.2 ns for 2500 Mbps) when the credit is increasing. The software should program this field with computed credit in bits per cycle scaled by 1,024. The maximum value is portTransmitRate, that is, 0x2000 in 1000/2500 Mbps mode and 0x1000 in 100 Mbps mode. Bits[20:14] must be written to zero. ■ **Quantum** When the DCB operation is enabled with DWRR algorithm for Queue 1 traffic, this field contains the quantum value in bytes to be added to credit during every queue scanning cycle. The maximum value is 0x1312D0 bytes. ■ **Weights** When DCB operation is enabled with WFQ algorithm for Queue 1 traffic, this field contains the weight for this queue. The maximum value is 0x3FFF where weight of 0 indicates 100% bandwidth. Bits[20:14] must be written to zero. When DCB operation or generic queuing operation is enabled with WRR algorithm for Queue 1 traffic, this field contains the weight for this queue. The maximum value is 0x64. Bits [20:7] must be written to zero.
20:0...(cont.)	ISCQW.	R/W	■ **Note 1:** In multiple Queue configuration this field in respective per queue register must be programmed to some non-zero value when multiple queues are enabled or single queue other than Q0 is enabled. This field need not be programmed when only Q0 is enabled. In general, when WRR algorithm is selected a non-zero value must be programmed on both Receive and Transmit. In Receive, the register is MTL_Operation_Mode register. ■ **Note 2:** For WFQ algorithm, higher the programmed weights lesser the bandwidth allocated for that Transmit Queue. The finish time is not a function of particular packet alone but it is as per the formula: (previous_finish_time of particular Transmit Queue + (weights*packet_size)) ■ **Note 3:** The weights programmed do not correspond to the number of packets but the fraction of bandwidth or time allocated for particular queue w.r.t. total BW or time. **Value After Reset:** 0x0 **Exists:** Always

17.4.7 MTL_TxQ(#i)_SendSlopeCredit (for i = 1; i <= DWC_EQOS_NUM_TXQ-1)

- **Description:** The sendSlopeCredit register contains the sendSlope credit value required for the credit-based shaper algorithm for the Queue.
- **Size:** 32 bits
- **Offset:** (0x0040*i)+0x0D1C
- **Exists:** (DWC_EQOS_TX_AV_EN&&(DWC_EQOS_TX-_AV_STRTQ<=1)&&(DWC_EQOS_NUM_TXQ>1))

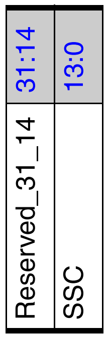


Table 17-271 Fields for Register: MTL_TxQ(#i)_SendSlopeCredit (for i = 1; i <= DWC_EQOS_NUM_TXQ-1)

Bits	Name	Memory Access	Description
31:14	Reserved_31_14	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
13:0	SSC	R/W	sendSlopeCredit Value When AV operation is enabled, this field contains the sendSlopeCredit value required for credit-based shaper algorithm for Queue 1. This is the rate of change of credit in bits per cycle (40 ns, 8 ns and 3.2 ns for 100 Mbps, 1000 Mbps and 2500 Mbps respectively) when the credit is decreasing. The software should program this field with computed credit in bits per cycle scaled by 1,024. The maximum value is portTransmitRate, that is, 0x2000 in 1000/2500 Mbps mode and 0x1000 in 100 Mbps mode. This field should be programmed with absolute sendSlopeCredit value. The credit-based shaper logic subtracts it from the accumulated credit when Channel 1 is selected for transmission. **Value After Reset:** 0x0 **Exists:** Always

17.4.8 MTL_TxQ(#i)_HiCredit (for i = 1; i <= DWC_EQOS_NUM_TXQ-1)

- **Description:** The hiCredit register contains the hiCredit value required for the credit-based shaper algorithm for the Queue.
- **Size:** 32 bits
- **Offset:** (0x0040*i)+0x0D20
- **Exists:** (DWC_EQOS_TX_AV_EN&&(DWC_EQOS_TX-_AV_STRTQ<=1)&&(DWC_EQOS_NUM_TXQ>1))

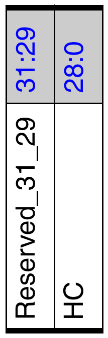


Table 17-272 Fields for Register: MTL_TxQ(#i)_HiCredit (for i = 1; i <= DWC_EQOS_NUM_TXQ-1)

Bits	Name	Memory Access	Description
31:29	Reserved_31_29	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
28:0	HC	R/W	hiCredit Value When the AV feature is enabled, this field contains the hiCredit value required for the credit-based shaper algorithm. This is the maximum value that can be accumulated in the credit parameter. This is specified in bits scaled by 1,024. The maximum value is maxInterferenceSize, that is, best-effort maximum packet size (16,384 bytes or 131,072 bits). The value to be specified is 131,072 * 1,024 = 134,217,728 or 0x0800_0000. **Value After Reset:** 0x0 **Exists:** Always

17.4.9 MTL_TxQ(#i)_LoCredit (for i = 1; i <= DWC_EQOS_NUM_TXQ-1)

- **Description:** The loCredit register contains the loCredit value required for the credit-based shaper algorithm for the Queue.
- **Size:** 32 bits
- **Offset:** (0x0040*i)+0x0D24
- **Exists:** (DWC_EQOS_TX_AV_EN&&(DWC_EQOS_TX-_AV_STRTQ<=1)&&(DWC_EQOS_NUM_TXQ>1))

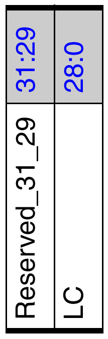


Table 17-273 Fields for Register: MTL_TxQ(#i)_LoCredit (for i = 1; i <= DWC_EQOS_NUM_TXQ-1)

Bits	Name	Memory Access	Description
31:29	Reserved_31_29	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
28:0	LC	R/W	loCredit Value When AV operation is enabled, this field contains the loCredit value required for the credit-based shaper algorithm. This is the minimum value that can be accumulated in the credit parameter. This is specified in bits scaled by 1,024. The maximum value to be programmed is corresponds to twice the maxFrameSize transmitted from this queue. If the maxFrameSize is 8192 bytes, then (8192*2) * 8 * 1024 = 134,217,728 or 0x0800_0000. Because it is a negative value, the programmed value is 2's complement of the value, that is, 0x1800_0000. **Value After Reset:** 0x0 **Exists:** Always

17.4.10 MTL_Q(#i)_Interrupt_Control_Status) (for i = 1; i <= max(DWC_EQOS_NUM_TXQ-1,DWC_EQOS_NUM_RXQ-1))

- **Description:** This register contains the interrupt enable and status bits for the queue 1 interrupts.
- **Size:** 32 bits
- **Offset:** (0x0040*i)+0x0D2C
- **Exists:** ((DWC_EQOS_NUM_RXQ>1||DWC_EQOS_NUM_TXQ>1)&&!DWC_EQOS_CORE)

31:25	24	23:17	16	15:10	9	8	7:2	1	0
Reserved_31_25	RXOIE	Reserved_23_17	RXOVFIS	Reserved_15_10	ABPSIE	TXUIE	Reserved_7_2	ABPSIS	TXUNFIS

Table 17-274 Fields for Register: MTL_Q(#i)_Interrupt_Control_Status) (for i = 1; i <= max(DWC_EQOS_NUM_TXQ-1,DWC_EQOS_NUM_RXQ-1))

Bits	Name	Memory Access	Description
31:25	Reserved_31_25	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
24	RXOIE	R/W	Receive Queue Overflow Interrupt Enable When this bit is set, the Receive Queue Overflow interrupt is enabled. When this bit is reset, the Receive Queue Overflow interrupt is disabled. **Values:** ■ 0x0 (DISABLE): Receive Queue Overflow Interrupt is disabled ■ 0x1 (ENABLE): Receive Queue Overflow Interrupt is enabled **Value After Reset:** 0x0 **Exists:** 1<DWC_EQOS_NUM_RXQ
23:17	Reserved_23_17	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
16	RXOVFIS	R/W	Receive Queue Overflow Interrupt Status This bit indicates that the Receive Queue had an overflow while receiving the packet. If a partial packet is transferred to the application, the overflow status is set in RDES3[21]. This bit is cleared when the application writes 1 to this bit. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. **Values:** ■ 0x0 (INACTIVE): Receive Queue Overflow Interrupt Status not detected ■ 0x1 (ACTIVE): Receive Queue Overflow Interrupt Status detected **Value After Reset:** 0x0 **Exists:** 1<DWC_EQOS_NUM_RXQ **Testable:** untestable
15:10	Reserved_15_10	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
9	ABPSIE	R/W	Average Bits Per Slot Interrupt Enable When this bit is set, the MAC asserts the sbd_intr_o or mci_intr_o interrupt when the average bits per slot status is updated. When this bit is cleared, the interrupt is not asserted for such an event. **Values:** ■ 0x0 (DISABLE): Average Bits Per Slot Interrupt is disabled ■ 0x1 (ENABLE): Average Bits Per Slot Interrupt is enabled **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_NUM_TXQ>1)&&(1<DWC_EQOS_NUM_TXQ)
8	TXUIE	R/W	Transmit Queue Underflow Interrupt Enable When this bit is set, the Transmit Queue Underflow interrupt is enabled. When this bit is reset, the Transmit Queue Underflow interrupt is disabled. **Values:** ■ 0x0 (DISABLE): Transmit Queue Underflow Interrupt Status is disabled ■ 0x1 (ENABLE): Transmit Queue Underflow Interrupt Status is enabled **Value After Reset:** 0x0 **Exists:** 1<DWC_EQOS_NUM_TXQ
7:2	Reserved_7_2	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
1	ABPSIS	R/W	Average Bits Per Slot Interrupt Status When set, this bit indicates that the MAC has updated the ABS value. This bit is cleared when the application writes 1 to this bit. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. **Values:** ■ 0x0 (INACTIVE): Average Bits Per Slot Interrupt Status not detected ■ 0x1 (ACTIVE): Average Bits Per Slot Interrupt Status detected **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_NUM_TXQ>1)&&(1<DWC_EQOS_NUM_TXQ) **Testable:** untestable
0	TXUNFIS	R/W	Transmit Queue Underflow Interrupt Status This bit indicates that the Transmit Queue had an underflow while transmitting the packet. Transmission is suspended and an Underflow Error TDES3[2] is set. This bit is cleared when the application writes 1 to this bit. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. **Values:** ■ 0x0 (INACTIVE): Transmit Queue Underflow Interrupt Status not detected ■ 0x1 (ACTIVE): Transmit Queue Underflow Interrupt Status detected **Value After Reset:** 0x0 **Exists:** 1<DWC_EQOS_NUM_TXQ **Testable:** untestable

17.4.11 MTL_RxQ(#i)_Operation_Mode (for i = 1; i <= DWC_EQOS_NUM_RXQ-1)

- **Description:** The Queue 1 Receive Operation Mode register establishes the Receive queue operating modes and command.
  The RFA and RFD fields are not backward compatible with the RFA and RFD fields of 4.00a release
- **Size:** 32 bits
- **Offset:** (0x0040*i)+0x0D30
- **Exists:** (DWC_EQOS_NUM_RXQ>1&&!DWC_EQOS_CORE)

31:y	x:20	19:y	x:14	13:y	x:8	7	6	5	4	3	2	1:0
Reserved_31_y	RQS	Reserved_19_y	RFD	Reserved_13_y	RFA	EHFC	DIS_TCP_EF	RSF	FEP	FUP	Reserved_2	RTC

Table 17-275 Fields for Register: MTL_RxQ(#i)_Operation_Mode (for i = 1; i <= DWC_EQOS_NUM_RXQ-1)

Bits	Name	Memory Access	Description
31:y	Reserved_31_y	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
x:20	RQS	R/W	Receive Queue Size This field indicates the size of the allocated Receive queues in blocks of 256 bytes. The RQS field is read-write only if the number of Rx Queues more than one, the reset value is 0x0 and indicates size of 256 bytes. When the number of Rx Queues is one, the field is read-only and the configured RX FIFO size in blocks of 256 bytes is reflected in the reset value. The width of this field depends on the Rx memory size selected in your configuration. For example, if the memory size is 2048, the width of this field is 3 bits: LOG2(2048/256) = LOG2(8) = 3 bits **Value After Reset:** 0x0 **Exists:** DWC_EQOS_NUM_RXQ>1 **Range Variable[x]:** DWC_EQOS_L256_RXMS + 19
19:y	Reserved_19_y	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[y]:** DWC_EQOS_FLCW + 14

Bits	Name	Memory Access	Description
x:14	RFD	R/W	Threshold for Deactivating Flow Control (in half-duplex and full-duplex modes) These bits control the threshold (fill-level of Rx queue) at which the flow control is de-asserted after activation: ■ 0: Full minus 1 KB, that is, FULL 1 KB ■ 1: Full minus 1.5 KB, that is, FULL 1.5 KB ■ 2: Full minus 2 KB, that is, FULL 2 KB ■ 3: Full minus 2.5 KB, that is, FULL 2.5 KB ■ ... ■ 62: Full minus 32 KB, that is, FULL 32 KB ■ 63: Full minus 32.5 KB, that is, FULL 32.5 KB The de-assertion is effective only after flow control is asserted. Note: The value must be programmed in such a way to make sure that the threshold is a positive number. When the EHFC is set high, these values are applicable only when the Rx queue size determined by the RQS field of this register, is equal to or greater than 4 KB. For a given queue size, the values ranges between 0 and the encoding for FULL minus (QSIZE - 0.5 KB) and all other values are illegal. Here the term FULL and QSIZE refers to the queue size determined by the RQS field of this register. The width of this field depends on RX FIFO size selected during the configuration. Remaining bits are reserved and read only. **Value After Reset:** 0x0 **Exists:** DWC_EQOS_RXFIFO_SIZE>2048 **Range Variable[x]:** DWC_EQOS_FLCW + 13
13:y	Reserved_13_y	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[y]:** DWC_EQOS_FLCW + 8
x:8	RFA	R/W	Threshold for Activating Flow Control (in half-duplex and full-duplex These bits control the threshold (fill-level of Rx queue) at which the flow control is activated: For more information on encoding for this field, see RFD. **Value After Reset:** 0x0 **Exists:** DWC_EQOS_RXFIFO_SIZE>2048 **Range Variable[x]:** DWC_EQOS_FLCW + 7

Bits	Name	Memory Access	Description
7	EHFC	R/W	Enable Hardware Flow Control When this bit is set, the flow control signal operation, based on the fill-level of Rx queue, is enabled. When reset, the flow control operation is disabled. **Values:** ■ 0x0 (DISABLE): Hardware Flow Control is disabled ■ 0x1 (ENABLE): Hardware Flow Control is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_RXFIFO_SIZE>2048
6	DIS_TCP_EF	R/W	Disable Dropping of TCP/IP Checksum Error Packets When this bit is set, the MAC does not drop the packets which only have the errors detected by the Receive Checksum Offload engine. Such packets have errors only in the encapsulated payload. There are no errors (including FCS error) in the Ethernet packet received by the MAC. When this bit is reset, all error packets are dropped if the FEP bit is reset. **Values:** ■ 0x0 (ENABLE): Dropping of TCP/IP Checksum Error Packets is enabled ■ 0x1 (DISABLE): Dropping of TCP/IP Checksum Error Packets is disabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_RX_COE
5	RSF	R/W	Receive Queue Store and Forward When this bit is set, the DWC_ether_qos reads a packet from the Rx queue only after the complete packet has been written to it, ignoring the RTC field of this register. When this bit is reset, the Rx queue operates in the Threshold (cut-through) mode, subject to the threshold specified by the RTC field of this register. **Values:** ■ 0x0 (DISABLE): Receive Queue Store and Forward is disabled ■ 0x1 (ENABLE): Receive Queue Store and Forward is enabled **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
4	FEP	R/W	Forward Error Packets When this bit is reset, the Rx queue drops packets with error status (CRC error, GMII_ER, watchdog timeout, or overflow). However, if the start byte (write) pointer of a packet is already transferred to the read controller side (in Threshold mode), the packet is not dropped. When this bit is set, all packets except the runt error packets are forwarded to the application or DMA. If the RSF bit is set and the Rx queue overflows when a partial packet is written, the packet is dropped irrespective of the setting of this bit. However, if the RSF bit is reset and the Rx queue overflows when a partial packet is written, a partial packet may be forwarded to the application or DMA. **Values:** ■ 0x0 (DISABLE): Forward Error Packets is disabled ■ 0x1 (ENABLE): Forward Error Packets is enabled **Value After Reset:** 0x0 **Exists:** Always
3	FUP	R/W	Forward Undersized Good Packets When this bit is set, the Rx queue forwards the undersized good packets (packets with no error and length less than 64 bytes), including pad-bytes and CRC. When this bit is reset, the Rx queue drops all packets of less than 64 bytes, unless a packet is already transferred because of the lower value of Rx Threshold, for example, RTC = 01. **Values:** ■ 0x0 (DISABLE): Forward Undersized Good Packets is disabled ■ 0x1 (ENABLE): Forward Undersized Good Packets is enabled **Value After Reset:** 0x0 **Exists:** Always
2	Reserved_2	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
1:0	RTC	R/W	Receive Queue Threshold Control These bits control the threshold level of the MTL Rx queue (in bytes): The received packet is transferred to the application or DMA when the packet size within the MTL Rx queue is larger than the threshold. In addition, full packets with length less than the threshold are automatically transferred. This field is valid only when the RSF bit is zero. This field is ignored when the RSF bit is set to 1. **Values:** ■ 0x0 (64BYTE): 64 ■ 0x1 (32BYTE): 32 ■ 0x2 (96BYTE): 96 ■ 0x3 (128BYTE): 128 **Value After Reset:** 0x0 **Exists:** Always

17.4.12 MTL_RxQ(#i)_Missed_Packet_Overflow_Cnt (for i = 1; i <= DWC_EQOS_NUM_RXQ-1)

- **Description:** The Queue 1 Missed Packet and Overflow Counter register contains the counter for packets missed because of Receive queue packet flush and packets discarded because of Receive queue overflow.
- **Size:** 32 bits
- **Offset:** (0x0040*i)+0x0D34
- **Exists:** (DWC_EQOS_NUM_RXQ>1&&!DWC_EQOS_CORE)

31:28	27	26:16	15:12	11	10:0
Reserved_31_28	MISCNTOVF	MISPKTCNT	Reserved_15_12	OVFCNTOVF	OVFPKTCNT

Table 17-276 Fields for Register: MTL_RxQ(#i)_Missed_Packet_Overflow_Cnt (for i = 1; i <= DWC_EQOS_NUM_RXQ-1)

Bits	Name	Memory Access	Description
31:28	Reserved_31_28	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
27	MISCNTOVF	R	Missed Packet Counter Overflow Bit When set, this bit indicates that the Rx Queue Missed Packet Counter crossed the maximum limit. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): Missed Packet Counter overflow not detected ■ 0x1 (ACTIVE): Missed Packet Counter overflow detected **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
26:16	MISPKTCNT	R	Missed Packet Counter This field indicates the number of packets missed by the DWC_ether_qos because the application asserted ari_pkt_flush_i[] for this queue. This counter is incremented each time the application issues ari_pkt_flush_i[] for this queue. This counter is reset when this register is read with mci_be_i[0] at 1b1. In EQOS-DMA, EQOS-AXI, and EQOS-AHB configurations, This counter is incremented by 1 when the DMA discards the packet because of buffer unavailability. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Value After Reset:** 0x0 **Exists:** Always
15:12	Reserved_15_12	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
11	OVFCNTOVF	R	Overflow Counter Overflow Bit When set, this bit indicates that the Rx Queue Overflow Packet Counter field crossed the maximum limit. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): Overflow Counter overflow not detected ■ 0x1 (ACTIVE): Overflow Counter overflow detected **Value After Reset:** 0x0 **Exists:** Always
10:0	OVFPKTCNT	R	Overflow Packet Counter This field indicates the number of packets discarded by the DWC_ether_qos because of Receive queue overflow. This counter is incremented each time the DWC_ether_qos discards an incoming packet because of overflow. This counter is reset when this register is read with mci_be_i[0] at 1'b1. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Value After Reset:** 0x0 **Exists:** Always

17.4.13 MTL_RxQ(#i)_Debug (for i = 1; i <= DWC_EQOS_NUM_RXQ-1)

- **Description:** The Queue 1 Receive Debug register gives the debug status of various blocks related to the Receive queue.
- **Size:** 32 bits
- **Offset:** (0x0040*i)+0x0D38
- **Exists:** (DWC_EQOS_NUM_RXQ>1&&!DWC_EQOS_CORE)

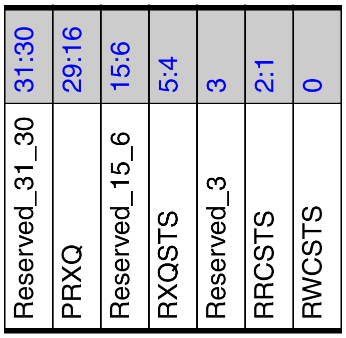


Table 17-277 Fields for Register: MTL_RxQ(#i)_Debug (for i = 1; i <= DWC_EQOS_NUM_RXQ-1)

Bits	Name	Memory Access	Description
31:30	Reserved_31_30	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
29:16	PRXQ	R	Number of Packets in Receive Queue This field indicates the current number of packets in the Rx Queue. The theoretical maximum value for this field is 256KB/16B = 16K Packets, that is, Max_Queue_Size/Min_Packet_Size. **Value After Reset:** 0x0 **Exists:** Always
15:6	Reserved_15_6	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
5:4	RXQSTS	R	MTL Rx Queue Fill-Level Status This field gives the status of the fill-level of the Rx Queue: **Values:** ■ 0x0 (EMPTY): Rx Queue empty ■ 0x1 (BLW_THR): Rx Queue fill-level below flow-control deactivate threshold ■ 0x2 (ABV_THR): Rx Queue fill-level above flow-control activate threshold ■ 0x3 (FULL): Rx Queue full **Value After Reset:** 0x0 **Exists:** Always
3	Reserved_3	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
2:1	RRCSTS	R	MTL Rx Queue Read Controller State This field gives the state of the Rx queue Read controller: **Values:** ■ 0x0 (IDLE): Idle state ■ 0x1 (READ_DATA): Reading packet data ■ 0x2 (READ_STS): Reading packet status (or timestamp) ■ 0x3 (FLUSH): Flushing the packet data and status **Value After Reset:** 0x0 **Exists:** Always
0	RWCSTS	R	MTL Rx Queue Write Controller Active Status When high, this bit indicates that the MTL Rx queue Write controller is active, and it is transferring a received packet to the Rx Queue. **Values:** ■ 0x0 (INACTIVE): MTL Rx Queue Write Controller Active Status not detected ■ 0x1 (ACTIVE): MTL Rx Queue Write Controller Active Status detected **Value After Reset:** 0x0 **Exists:** Always

17.4.14 MTL_RxQ(#i)_Control (for i = 1; i <= DWC_EQOS_NUM_RXQ-1)

- **Description:** The Queue Receive Control register controls the receive arbitration and passing of received packets to the application.
- **Size:** 32 bits
- **Offset:** (0x0040*i)+0x0D3C
- **Exists:** (DWC_EQOS_NUM_RXQ>1&&!DWC_EQOS_CORE)

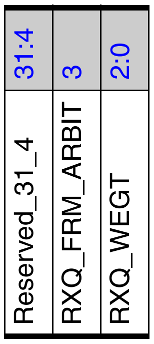


Table 17-278 Fields for Register: MTL_RxQ(#i)_Control (for i = 1; i <= DWC_EQOS_NUM_RXQ-1)

Bits	Name	Memory Access	Description
31:4	Reserved_31_4	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
3	RXQ_FRM_ARBIT	R/W	Receive Queue Packet Arbitration When this bit is set, the DWC_ether_qos drives the packet data to the ARI interface such that the entire packet data of currently-selected queue is transmitted before switching to other queue. When this bit is reset, the DWC_ether_qos drives the packet data to the ARI interface such that the following amount of data of currently-selected queue is transmitted before switching to other queue: ■ PBL amount of data (indicated by ari_qN_pbl_i[]) or ■ Complete data of a packet The status and the timestamp are not a part of the PBL data. Therefore, the DWC_ether_qos drives the complete status (including timestamp status) during first PBL request for the packet (in store-and-forward mode) or the last PBL request for the packet (in Threshold mode). **Values:** ■ 0x0 (DISABLE): Receive Queue Packet Arbitration is disabled ■ 0x1 (ENABLE): Receive Queue Packet Arbitration is enabled **Value After Reset:** ((DWC_EQOS_MTL_SUBSYS&&DWC_EQOS_NUM_RXQ>1)?\"0x1\":\"0x0\") **Exists:** ((DWC_EQOS_MTL_SUBSYS&&DWC_EQOS_NUM_RXQ>1)||(!DWC_EQOS_MTL_SUBSYS&&DWC_EQOS_NUM_DMA_RX_CH>1&&DWC_EQOS_NUM_RXQ>1))
2:0	RXQ_WEGT	R/W	Receive Queue Weight This field indicates the weight assigned to the Rx Queue 0. The weight is used as the number of continuous PBL or packets requests (depending on the RXQ_FRM_ARBIT) allocated to the queue in one arbitration cycle. **Value After Reset:** 0x0 **Exists:** Always

17.5 EQOS_DMA Registers

17.5.1 DMA_Mode

- **Description:** The Bus Mode register establishes the bus operating modes for the DMA.
- **Size:** 32 bits
- **Offset:** 0x1000
- **Exists:** (DWC_EQOS_SYS>1)

31:18	17:16	15	14:12	11	10	9	8	7:5	4:2	1	0
Reserved_31_18	INTM	Reserved_15	PR	TXPR	Reserved_10	ARBC	DSPW	Reserved_7_5	TAA	DA	SWR

Table 17-279 Fields for Register: DMA_Mode

Bits	Name	Memory Access	Description
31:18	Reserved_31_18	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
17:16	INTM	R/W	Interrupt Mode This field defines the interrupt mode of DWC_ether_qos. The behavior of the following outputs changes depending on the following settings: ■ sbd_perch_tx_intr_o[] (Transmit Per Channel Interrupt) ■ sbd_perch_rx_intr_o[] (Receive Per Channel Interrupt) ■ sbd_intr_o (Common Interrupt) It also changes the behavior of the RI/TI bits in the DMA_CH0_Status. ■ 00: sbd_perch_* are pulse signals for each TX/RX packet transfer completion events (irrespective of whether corresponding interrupts are enabled) for which IOC bits are enabled in descriptor. sbd_intr_o is also asserted when corresponding interrupts are enabled and cleared only when software clears the corresponding RI/TI status bits. ■ 01: sbd_perch_* are level signals asserted on TX/RX packet transfer completion event when corresponding interrupts are enabled and de-asserted when the software clears the corresponding RI/TI status bits. The sbd_intr_o is not asserted for these TX/RX packet transfer completion events. ■ 10: sbd_perch_* are level signals asserted on TX/RX packet transfer completion event when corresponding interrupts are enabled and de-asserted when the software clears the corresponding RI/TI status bits. However, the signal is asserted again if the same event occurred again before it was cleared. The sbd_intr_o is not asserted for these TX/RX packet transfer completion events. ■ 11: Reserved For more details please refer Table "DWC_ether_qos Transfer Complete Interrupt Behavior". **Values:** ■ 0x0 (MODE0): See above description ■ 0x1 (MODE1): See above description ■ 0x2 (MODE2): See above description ■ 0x3 (RSVD): Reserved **Value After Reset:** 0x0 **Exists:** Always
15	Reserved_15	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
14:12	PR	R/W	Priority Ratio These bits control the priority ratio in weighted round-robin arbitration between the Rx DMA and Tx DMA. These bits are valid only when the DA bit is reset. The priority ratio is Rx:Tx or Tx:Rx depending on whether the TXPR bit is reset or set. **Values:** ■ 0x0 (R_1_1): The priority ratio is 1:1 ■ 0x1 (R_2_1): The priority ratio is 2:1 ■ 0x2 (R_3_1): The priority ratio is 3:1 ■ 0x3 (R_4_1): The priority ratio is 4:1 ■ 0x4 (R_5_1): The priority ratio is 5:1 ■ 0x5 (R_6_1): The priority ratio is 6:1 ■ 0x6 (R_7_1): The priority ratio is 7:1 ■ 0x7 (R_8_1): The priority ratio is 8:1 **Value After Reset:** 0x0 **Exists:** !(DWC_EQOS_SYS==4||DWC_EQOS_SYS==5)
11	TXPR	R/W	Transmit Priority When set, this bit indicates that the Tx DMA has higher priority than the Rx DMA during arbitration for the system-side bus. **Values:** ■ 0x0 (DISABLE): Transmit Priority is disabled ■ 0x1 (ENABLE): Transmit Priority is enabled **Value After Reset:** 0x0 **Exists:** !(DWC_EQOS_SYS==4||DWC_EQOS_SYS==5)
10	Reserved_10	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
9	ARBC	R/W	Reserved. **Value After Reset:** 0x0 **Exists:** !(DWC_EQOS_SYS==4||DWC_EQOS_SYS==5)
8	DSPW	R/W	Descriptor Posted Write When this bit is set to 0, the descriptor writes are always non-posted. When this bit is set to 1, the descriptor writes are non-posted only when IOC (Interrupt on completion) is set in last descriptor, otherwise the descriptor writes are always posted. **Values:** ■ 0x0 (DISABLE): Descriptor Posted Write is disabled ■ 0x1 (ENABLE): Descriptor Posted Write is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_SYS==4||DWC_EQOS_SYS==5

Bits	Name	Memory Access	Description
7:5	Reserved_7_5	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
4:2	TAA	R/W	Transmit Arbitration Algorithm This field is used to select the arbitration algorithm for the Transmit side when multiple Tx DMAs are selected. **Values:** ■ 0x0 (FP): Fixed priority (Channel 0 has the lowest priority and the last channel has the highest priority) ■ 0x1 (WSP): Weighted Strict Priority (WSP) ■ 0x2 (WRR): Weighted Round-Robin (WRR) ■ 0x3 (RSVD): Reserved (for 3'b011 to 3'b111) **Value After Reset:** 0x0 **Exists:** !DWC_EQOS_AXI_SUBSYS&&DWC_EQOS_NUM_DMA_TX_CH>1
1	DA	R/W	DMA Tx or Rx Arbitration Scheme This bit specifies the arbitration scheme between the Transmit and Receive paths of all channels: ■ 0: Weighted Round-Robin with Rx:Tx or Tx:Rx The priority between the paths is according to the priority specified in Bits[14:12] and the priority weight is specified in the TXPR bit. ■ 1: Fixed Priority The Tx path has priority over the Rx path when the TXPR bit is set. Otherwise, the Rx path has priority over the Tx path. **Values:** ■ 0x0 (WRR): Weighted Round-Robin with Rx:Tx or Tx:Rx ■ 0x1 (FP): Fixed Priority **Value After Reset:** 0x0 **Exists:** !(DWC_EQOS_SYS==4||DWC_EQOS_SYS==5)

Bits	Name	Memory Access	Description
0	SWR	R/W	Software Reset When this bit is set, the MAC and the DMA controller reset the logic and all internal registers of the DMA, MTL, and MAC. This bit is automatically cleared after the reset operation is complete in all DWC_ether_qos clock domains. Before reprogramming any DWC_ether_qos register, a value of zero should be read in this bit. This bit must be read at least 4 CSR clock cycles after it is written to 1. **Note**: The reset operation is complete only when all resets in all active clock domains are de-asserted. Therefore, it is essential that all PHY inputs clocks (applicable for the selected PHY interface) are present for software reset completion. The time to complete the software reset operation depends on the frequency of the slowest active clock. Access restriction applies. Setting 1 sets. Self-cleared. Setting 0 has no effect. **Values:** ■ 0x0 (DISABLE): Software Reset is disabled ■ 0x1 (ENABLE): Software Reset is enabled **Value After Reset:** 0x0 **Exists:** Always **Testable:** untestable

17.5.2 DMA_SysBus_Mode

- **Description:** The System Bus mode register controls the behavior of the AHB or AXI master. It mainly controls burst splitting and number of outstanding requests.
- **Size:** 32 bits
- **Offset:** 0x1004
- **Exists:** (DWC_EQOS_SYS>1)

31	30	29:y	x:24	23:y	x:16	15	14	13	12	11	10	9:8	7	6	5	4	3	2	1	0
EN_LPI	LPI_XIT_PKT	Reserved_29_y	WR_OSR_LMT	Reserved_23_y	RD_OSR_LMT	RB	MB	ONEKBBE	AAL	EAME	AALE	Reserved_9_8	BLEN256	BLEN128	BLEN64	BLEN32	BLEN16	BLEN8	BLEN4	FB

Table 17-280 Fields for Register: DMA_SysBus_Mode

Bits	Name	Memory Access	Description
31	EN_LPI	R/W	Enable Low Power Interface (LPI) When set to 1, this bit enables the LPI mode supported by the EQOS-AXI configuration and accepts the LPI request from the AXI System Clock controller. When set to 0, this bit disables the LPI mode and always denies the LPI request from the AXI System Clock controller. **Values:** ■ 0x0 (DISABLE): Low Power Interface (LPI) is disabled ■ 0x1 (ENABLE): Low Power Interface (LPI) is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_SYS==4||DWC_EQOS_SYS==5
30	LPI_XIT_PKT	R/W	Unlock on Magic Packet or Remote Wake-Up Packet When set to 1, this bit enables the AXI master to come out of the LPI mode only when the magic packet or remote wake-up packet is received. When set to 0, this bit enables the AXI master to come out of the LPI mode when any packet is received. **Values:** ■ 0x0 (DISABLE): Unlock on Magic Packet or Remote Wake-Up Packet is disabled ■ 0x1 (ENABLE): Unlock on Magic Packet or Remote Wake-Up Packet is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_AXI_SUBSYS&&DWC_EQOS_PMT_EN

Bits	Name	Memory Access	Description
29:y	Reserved_29_y	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[y]:** DWC_EQOS_WOSRLW + 24
x:24	WR_OSR_LMT	R/W	AXI Maximum Write Outstanding Request Limit This value limits the maximum outstanding request on the AXI write interface. Maximum outstanding requests = WR_OSR_LMT + 1 **Note**: ■ Bit 26 is reserved if DWC_ETHER_QOS_AXI_MAX-_WR_REQ = 4 ■ Bit 27 is reserved if DWC_ETHER_QOS_AXI_MAX-_WR_REQ!= 16 **Value After Reset:** ((DWC_EQOS_AXI_SUBSYS)?\"0x1\":\"0x0\") **Exists:** DWC_EQOS_SYS==4||DWC_EQOS_SYS==5 **Range Variable[x]:** DWC_EQOS_WOSRLW + 23
23:y	Reserved_23_y	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[y]:** DWC_EQOS_ROSRLW + 16
x:16	RD_OSR_LMT	R/W	AXI Maximum Read Outstanding Request Limit This value limits the maximum outstanding request on the AXI read interface. Maximum outstanding requests = RD_OSR_LMT + 1 **Note**: ■ Bit 18 is reserved if parameter DWC_ETHER_QOS_AXI_-MAX_RD_REQ = 4 ■ Bit 19 is reserved if parameter DWC_ETHER_QOS_AXI_-MAX_RD_REQ!= 16 **Value After Reset:** ((DWC_EQOS_AXI_SUBSYS)?\"0x1\":\"0x0\") **Exists:** DWC_EQOS_SYS==4||DWC_EQOS_SYS==5 **Range Variable[x]:** DWC_EQOS_ROSRLW + 15

Bits	Name	Memory Access	Description
15	RB	R/W	Rebuild INCRx Burst When this bit is set high and the AHB master gets SPLIT, RETRY, or Early Burst Termination (EBT) response, the AHB master interface rebuilds the pending beats of any initiated burst transfer with INCRx and SINGLE transfers. By default, the AHB master interface rebuilds pending beats of an EBT with an unspecified (INCR) burst. **Values:** ■ 0x0 (DISABLE): Rebuild INCRx Burst is disabled ■ 0x1 (ENABLE): Rebuild INCRx Burst is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_SYS==3
14	MB	R/W	Mixed Burst When this bit is set high and the FB bit is low, the AHB master performs undefined bursts transfers (INCR) for burst length of 16 or more. For burst length of 16 or less, the AHB master performs fixed burst transfers (INCRx and SINGLE). **Values:** ■ 0x0 (DISABLE): Mixed Burst is disabled ■ 0x1 (ENABLE): Mixed Burst is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_SYS==3
13	ONEKBBE	R/W	1 KB Boundary Crossing Enable for the EQOS-AXI Master When set, the burst transfers performed by the EQOS-AXI master do not cross 1 KB boundary. When reset, the burst transfers performed by the EQOS-AXI master do not cross 4 KB boundary. **Values:** ■ 0x0 (DISABLE): 1 KB Boundary Crossing for the EQOS-AXI Master Beats is disabled ■ 0x1 (ENABLE): 1 KB Boundary Crossing for the EQOS-AXI Master Beats is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_SYS==4||DWC_EQOS_SYS==5
12	AAL	R/W	Address-Aligned Beats When this bit is set to 1, the EQOS-AXI or EQOS-AHB master performs address-aligned burst transfers on Read and Write channels. **Values:** ■ 0x0 (DISABLE): Address-Aligned Beats is disabled ■ 0x1 (ENABLE): Address-Aligned Beats is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_SYS>2

Bits	Name	Memory Access	Description
11	EAME	R/W	Enhanced Address Mode Enable. When this bit is set to 1, the DMA master enables the enhanced address mode (40-bit or 48-bit addressing mode). In this mode, the DMA engine uses either the 40- or 48-bit address, depending on the configuration. **Values:** ■ 0x0 (DISABLE): Enhanced Address Mode is disabled ■ 0x1 (ENABLE): Enhanced Address Mode is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_ADDRWIDTH != 32
10	AALE	R/W	Automatic AXI LPI enable When set to 1, enables the AXI master to enter into LPI state when there is no activity in the DWC_ether_qos for number of system clock cycles programmed in the LPIEI field of AXI_LPI_Entry_Interval register. **Values:** ■ 0x0 (DISABLE): Automatic AXI LPI is disabled ■ 0x1 (ENABLE): Automatic AXI LPI is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_SYS==4||DWC_EQOS_SYS==5
9:8	Reserved_9_8	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
7	BLEN256	R/W	AXI Burst Length 256 When this bit is set to 1, the EQOS-AXI master can select a burst length of 256 on the AXI interface. **Values:** ■ 0x0 (DISABLE): No effect ■ 0x1 (ENABLE): AXI Burst Length 256 **Value After Reset:** 0x0 **Exists:** DWC_EQOS_AXI_BL>128
6	BLEN128	R/W	AXI Burst Length 128 When this bit is set to 1, the EQOS-AXI master can select a burst length of 128 on the AXI interface. **Values:** ■ 0x0 (DISABLE): No effect ■ 0x1 (ENABLE): AXI Burst Length 128 **Value After Reset:** 0x0 **Exists:** DWC_EQOS_AXI_BL>64

Bits	Name	Memory Access	Description
5	BLEN64	R/W	AXI Burst Length 64 When this bit is set to 1, the EQOS-AXI master can select a burst length of 64 on the AXI interface. **Values:** ■ 0x0 (DISABLE): No effect ■ 0x1 (ENABLE): AXI Burst Length 64 **Value After Reset:** 0x0 **Exists:** DWC_EQOS_AXI_BL>32
4	BLEN32	R/W	AXI Burst Length 32 When this bit is set to 1, the EQOS-AXI master can select a burst length of 32 on the AXI interface. **Values:** ■ 0x0 (DISABLE): No effect ■ 0x1 (ENABLE): AXI Burst Length 32 **Value After Reset:** 0x0 **Exists:** DWC_EQOS_AXI_BL>16
3	BLEN16	R/W	AXI Burst Length 16 When this bit is set to 1 or the FB bit is set to 0, the EQOS-AXI master can select a burst length of 16 on the AXI interface. When the FB bit is set to 0, setting this bit has no effect. **Values:** ■ 0x0 (DISABLE): No effect ■ 0x1 (ENABLE): AXI Burst Length 16 **Value After Reset:** 0x0 **Exists:** DWC_EQOS_SYS==4||DWC_EQOS_SYS==5
2	BLEN8	R/W	AXI Burst Length 8 When this bit is set to 1 or the FB bit is set to 0, the EQOS-AXI master can select a burst length of 8 on the AXI interface. When the FB bit is set to 0, setting this bit has no effect. **Values:** ■ 0x0 (DISABLE): No effect ■ 0x1 (ENABLE): AXI Burst Length 8 **Value After Reset:** 0x0 **Exists:** DWC_EQOS_SYS==4||DWC_EQOS_SYS==5

Bits	Name	Memory Access	Description
1	BLEN4	R/W	AXI Burst Length 4 When this bit is set to 1 or the FB bit is set to 0, the EQOS-AXI master can select a burst length of 4 on the AXI interface. When the FB bit is set to 0, setting this bit has no effect. **Values:** ■ 0x0 (DISABLE): No effect ■ 0x1 (ENABLE): AXI Burst Length 4 **Value After Reset:** 0x0 **Exists:** DWC_EQOS_SYS==4||DWC_EQOS_SYS==5
0	FB	R/W	Fixed Burst Length **For EQOS-AXI Configurations:** When this bit is set to 1, the EQOS-AXI master initiates burst transfers of specified lengths as given below. ■ Burst transfers of fixed burst lengths as indicated by the BLEN256, BLEN128, BLEN64, BLEN32, BLEN16, BLEN8, or BLEN4 field ■ Burst transfers of length 1 When this bit is set to 0, the EQOS-AXI master initiates burst transfers that are equal to or less than the maximum allowed burst length programmed in Bits[7:1]. **For EQOS-AHB Configurations:** When this bit is set to 1, the AHB master initiates burst transfers of specified length (INCRx or SINGLE). When this bit is set to 0, the AHB master initiates transfers of unspecified length (INCR) or SINGLE transfers. **For EQOS-DMA Configurations:** The value of this bit is driven on the mdc_burst_count_o output signal. **Values:** ■ 0x0 (DISABLE): Fixed Burst Length is disabled ■ 0x1 (ENABLE): Fixed Burst Length is enabled **Value After Reset:** 0x0 **Exists:** Always

17.5.3 DMA_Interrupt_Status

- **Description:** The application reads this Interrupt Status register during interrupt service routine or polling to determine the interrupt status of DMA channels, MTL queues, and the MAC.
- **Size:** 32 bits
- **Offset:** 0x1008
- **Exists:** (DWC_EQOS_SYS>1)

31:18	17	16	15:8	7	6	5	4	3	2	1	0
Reserved_31_18	MACIS	MTLIS	Reserved_15_8	DC7IS	DC6IS	DC5IS	DC4IS	DC3IS	DC2IS	DC1IS	DC0IS

Table 17-281 Fields for Register: DMA_Interrupt_Status

Bits	Name	Memory Access	Description
31:18	Reserved_31_18	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
17	MACIS	R	MAC Interrupt Status This bit indicates an interrupt event in the MAC. To reset this bit to 1'b0, the software must read the corresponding register in the MAC to get the exact cause of the interrupt and clear its source. **Values:** ■ 0x0 (INACTIVE): MAC Interrupt Status not detected ■ 0x1 (ACTIVE): MAC Interrupt Status detected **Value After Reset:** 0x0 **Exists:** Always
16	MTLIS	R	MTL Interrupt Status This bit indicates an interrupt event in the MTL. To reset this bit to 1'b0, the software must read the corresponding register in the MTL to get the exact cause of the interrupt and clear its source. **Values:** ■ 0x0 (INACTIVE): MTL Interrupt Status not detected ■ 0x1 (ACTIVE): MTL Interrupt Status detected **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
15:8	Reserved_15_8	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
7	DC7IS	R	DMA Channel 7 Interrupt Status This bit indicates an interrupt event in DMA Channel 7. To reset this bit to 1'b0, the software must read the corresponding register in DMA Channel 7 to get the exact cause of the interrupt and clear its source. **Values:** ■ 0x0 (INACTIVE): DMA Channel 7 Interrupt Status not detected ■ 0x1 (ACTIVE): DMA Channel 7 Interrupt Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_NUM_DMA_CSR_CH==8
6	DC6IS	R	DMA Channel 6 Interrupt Status This bit indicates an interrupt event in DMA Channel 6. To reset this bit to 1'b0, the software must read the corresponding register in DMA Channel 6 to get the exact cause of the interrupt and clear its source. **Values:** ■ 0x0 (INACTIVE): DMA Channel 6 Interrupt Status not detected ■ 0x1 (ACTIVE): DMA Channel 6 Interrupt Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_NUM_DMA_CSR_CH>6
5	DC5IS	R	DMA Channel 5 Interrupt Status This bit indicates an interrupt event in DMA Channel 5. To reset this bit to 1'b0, the software must read the corresponding register in DMA Channel 5 to get the exact cause of the interrupt and clear its source. **Values:** ■ 0x0 (INACTIVE): DMA Channel 5 Interrupt Status not detected ■ 0x1 (ACTIVE): DMA Channel 5 Interrupt Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_NUM_DMA_CSR_CH>5

Bits	Name	Memory Access	Description
4	DC4IS	R	DMA Channel 4 Interrupt Status This bit indicates an interrupt event in DMA Channel 4. To reset this bit to 1'b0, the software must read the corresponding register in DMA Channel 4 to get the exact cause of the interrupt and clear its source. **Values:** ■ 0x0 (INACTIVE): DMA Channel 4 Interrupt Status not detected ■ 0x1 (ACTIVE): DMA Channel 4 Interrupt Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_NUM_DMA_CSR_CH>4
3	DC3IS	R	DMA Channel 3 Interrupt Status This bit indicates an interrupt event in DMA Channel 3. To reset this bit to 1'b0, the software must read the corresponding register in DMA Channel 3 to get the exact cause of the interrupt and clear its source. **Values:** ■ 0x0 (INACTIVE): DMA Channel 3 Interrupt Status not detected ■ 0x1 (ACTIVE): DMA Channel 3 Interrupt Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_NUM_DMA_CSR_CH>3
2	DC2IS	R	DMA Channel 2 Interrupt Status This bit indicates an interrupt event in DMA Channel 2. To reset this bit to 1'b0, the software must read the corresponding register in DMA Channel 2 to get the exact cause of the interrupt and clear its source. **Values:** ■ 0x0 (INACTIVE): DMA Channel 2 Interrupt Status not detected ■ 0x1 (ACTIVE): DMA Channel 2 Interrupt Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_NUM_DMA_CSR_CH>2
1	DC1IS	R	DMA Channel 1 Interrupt Status This bit indicates an interrupt event in DMA Channel 1. To reset this bit to 1'b0, the software must read the corresponding register in DMA Channel 1 to get the exact cause of the interrupt and clear its source. **Values:** ■ 0x0 (INACTIVE): DMA Channel 1 Interrupt Status not detected ■ 0x1 (ACTIVE): DMA Channel 1 Interrupt Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_NUM_DMA_CSR_CH!=1

Bits	Name	Memory Access	Description
0	DC0IS	R	DMA Channel 0 Interrupt Status This bit indicates an interrupt event in DMA Channel 0. To reset this bit to 1'b0, the software must read the corresponding register in DMA Channel 0 to get the exact cause of the interrupt and clear its source. **Values:** ■ 0x0 (INACTIVE): DMA Channel 0 Interrupt Status not detected ■ 0x1 (ACTIVE): DMA Channel 0 Interrupt Status detected **Value After Reset:** 0x0 **Exists:** Always

17.5.4 DMA_Debug_Status0

- **Description:** The Debug Status 0 register gives the Receive and Transmit process status for DMA Channel 0-Channel 2 for debugging purpose.
- **Size:** 32 bits
- **Offset:** 0x100c
- **Exists:** (DWC_EQOS_SYS>1)

31:28	27:24	23:20	19:16	15:12	11:8	7:2	1	0
TPS2	RPS2	TPS1	RPS1	TPS0	RPS0	Reserved_7_2	AXRHSTS	AXWHSTS

Table 17-282 Fields for Register: DMA_Debug_Status0

Bits	Name	Memory Access	Description
31:28	TPS2	R	DMA Channel 2 Transmit Process State This field indicates the Tx DMA FSM state for Channel 2. The MSB of this field always returns 0. This field does not generate an interrupt. **Values:** ■ 0x0 (STOP): Stopped (Reset or Stop Transmit Command issued) ■ 0x1 (RUN_FTTD): Running (Fetching Tx Transfer Descriptor) ■ 0x2 (RUN_WS): Running (Waiting for status) ■ 0x3 (RUN_RDS): Running (Reading Data from system memory buffer and queuing it to the Tx buffer (Tx FIFO)) ■ 0x4 (TSTMP_WS): Timestamp write state ■ 0x5 (RSVD): Reserved for future use ■ 0x6 (SUSPND): Suspended (Tx Descriptor Unavailable or Tx Buffer Underflow) ■ 0x7 (RUN_CTD): Running (Closing Tx Descriptor) **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_NUM_DMA_TX_CH>2)

Bits	Name	Memory Access	Description
27:24	RPS2	R	DMA Channel 2 Receive Process State This field indicates the Rx DMA FSM state for Channel 2. The MSB of this field always returns 0. This field does not generate an interrupt. **Values:** ■ 0x0 (STOP): Stopped (Reset or Stop Receive Command issued) ■ 0x1 (RUN_FRTD): Running (Fetching Rx Transfer Descriptor) ■ 0x2 (RSVD): Reserved for future use ■ 0x3 (RUN_WRP): Running (Waiting for Rx packet) ■ 0x4 (SUSPND): Suspended (Rx Descriptor Unavailable) ■ 0x5 (RUN_CRD): Running (Closing the Rx Descriptor) ■ 0x6 (TSTMP): Timestamp write state ■ 0x7 (RUN_TRP): Running (Transferring the received packet data from the Rx buffer to the system memory) **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_NUM_DMA_RX_CH>2)
23:20	TPS1	R	DMA Channel 1 Transmit Process State This field indicates the Tx DMA FSM state for Channel 1. The MSB of this field always returns 0. This field does not generate an interrupt. **Values:** ■ 0x0 (STOP): Stopped (Reset or Stop Transmit Command issued) ■ 0x1 (RUN_FTTD): Running (Fetching Tx Transfer Descriptor) ■ 0x2 (RUN_WS): Running (Waiting for status) ■ 0x3 (RUN_RDS): Running (Reading Data from system memory buffer and queuing it to the Tx buffer (Tx FIFO)) ■ 0x4 (TSTMP_WS): Timestamp write state ■ 0x5 (RSVD): Reserved for future use ■ 0x6 (SUSPND): Suspended (Tx Descriptor Unavailable or Tx Buffer Underflow) ■ 0x7 (RUN_CTD): Running (Closing Tx Descriptor) **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_NUM_DMA_TX_CH>1)

Bits	Name	Memory Access	Description
19:16	RPS1	R	DMA Channel 1 Receive Process State This field indicates the Rx DMA FSM state for Channel 1. The MSB of this field always returns 0. This field does not generate an interrupt. **Values:** ■ 0x0 (STOP): Stopped (Reset or Stop Receive Command issued) ■ 0x1 (RUN_FRTD): Running (Fetching Rx Transfer Descriptor) ■ 0x2 (RSVD): Reserved for future use ■ 0x3 (RUN_WRP): Running (Waiting for Rx packet) ■ 0x4 (SUSPND): Suspended (Rx Descriptor Unavailable) ■ 0x5 (RUN_CRD): Running (Closing the Rx Descriptor) ■ 0x6 (TSTMP): Timestamp write state ■ 0x7 (RUN_TRP): Running (Transferring the received packet data from the Rx buffer to the system memory) **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_NUM_DMA_RX_CH>1)
15:12	TPS0	R	DMA Channel 0 Transmit Process State This field indicates the Tx DMA FSM state for Channel 0. The MSB of this field always returns 0. This field does not generate an interrupt. **Values:** ■ 0x0 (STOP): Stopped (Reset or Stop Transmit Command issued) ■ 0x1 (RUN_FTTD): Running (Fetching Tx Transfer Descriptor) ■ 0x2 (RUN_WS): Running (Waiting for status) ■ 0x3 (RUN_RDS): Running (Reading Data from system memory buffer and queuing it to the Tx buffer (Tx FIFO)) ■ 0x4 (TSTMP_WS): Timestamp write state ■ 0x5 (RSVD): Reserved for future use ■ 0x6 (SUSPND): Suspended (Tx Descriptor Unavailable or Tx Buffer Underflow) ■ 0x7 (RUN_CTD): Running (Closing Tx Descriptor) **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
11:8	RPS0	R	DMA Channel 0 Receive Process State This field indicates the Rx DMA FSM state for Channel 0. The MSB of this field always returns 0. This field does not generate an interrupt. **Values:** ■ 0x0 (STOP): Stopped (Reset or Stop Receive Command issued) ■ 0x1 (RUN_FRTD): Running (Fetching Rx Transfer Descriptor) ■ 0x2 (RSVD): Reserved for future use ■ 0x3 (RUN_WRP): Running (Waiting for Rx packet) ■ 0x4 (SUSPND): Suspended (Rx Descriptor Unavailable) ■ 0x5 (RUN_CRD): Running (Closing the Rx Descriptor) ■ 0x6 (TSTMP): Timestamp write state ■ 0x7 (RUN_TRP): Running (Transferring the received packet data from the Rx buffer to the system memory) **Value After Reset:** 0x0 **Exists:** Always
7:2	Reserved_7_2	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
1	AXRHSTS	R	AXI Master Read Channel Status When high, this bit indicates that the read channel of the AXI master is active, and it is transferring the data. **Values:** ■ 0x0 (INACTIVE): AXI Master Read Channel Status not detected ■ 0x1 (ACTIVE): AXI Master Read Channel Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_SYS==4||DWC_EQOS_SYS==5

Bits	Name	Memory Access	Description
0	AXWHSTS	R	AXI Master Write Channel or AHB Master Status EQOS-AXI Configuration: When high, this bit indicates that the write channel of the AXI master is active, and it is transferring data. EQOS-AHB Configuration: When high, this bit indicates that the AHB master FSMs are in the non-idle state. **Values:** ■ 0x0 (INACTIVE): AXI Master Write Channel or AHB Master Status not detected ■ 0x1 (ACTIVE): AXI Master Write Channel or AHB Master Status detected **Value After Reset:** 0x0 **Exists:** DWC_EQOS_SYS>2

17.5.5 DMA_Debug_Status1

- **Description:** The Debug Status1 register gives the Receive and Transmit process status for DMA Channel 3-Channel 6.
- **Size:** 32 bits
- **Offset:** 0x1010
- **Exists:** ((DWC_EQOS_SYS>1)&&(DWC_EQOS_NUM_DMA_TX_CH>3||DWC_EQOS_NUM_D-MA_RX_CH>3))

31:28	27:24	23:20	19:16	15:12	11:8	7:4	3:0
TPS6	RPS6	TPS5	RPS5	TPS4	RPS4	TPS3	RPS3

Table 17-283 Fields for Register: DMA_Debug_Status1

Bits	Name	Memory Access	Description
31:28	TPS6	R	DMA Channel 6 Transmit Process State This field indicates the Tx DMA FSM state for Channel 6. The MSB of this field always returns 0. This field does not generate an interrupt. **Values:** ■ 0x0 (STOP): Stopped (Reset or Stop Transmit Command issued) ■ 0x1 (RUN_FTTD): Running (Fetching Tx Transfer Descriptor) ■ 0x2 (RUN_WS): Running (Waiting for status) ■ 0x3 (RUN_RDS): Running (Reading Data from system memory buffer and queuing it to the Tx buffer (Tx FIFO)) ■ 0x4 (TSTMP_WS): Timestamp write state ■ 0x5 (RSVD): Reserved for future use ■ 0x6 (SUSPND): Suspended (Tx Descriptor Unavailable or Tx Buffer Underflow) ■ 0x7 (RUN_CTD): Running (Closing Tx Descriptor) **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_NUM_DMA_TX_CH>6)

Bits	Name	Memory Access	Description
27:24	RPS6	R	DMA Channel 6 Receive Process State This field indicates the Rx DMA FSM state for Channel 6. The MSB of this field always returns 0. This field does not generate an interrupt. **Values:** ■ 0x0 (STOP): Stopped (Reset or Stop Receive Command issued) ■ 0x1 (RUN_FRTD): Running (Fetching Rx Transfer Descriptor) ■ 0x2 (RSVD): Reserved for future use ■ 0x3 (RUN_WRP): Running (Waiting for Rx packet) ■ 0x4 (SUSPND): Suspended (Rx Descriptor Unavailable) ■ 0x5 (RUN_CRD): Running (Closing the Rx Descriptor) ■ 0x6 (TSTMP): Timestamp write state ■ 0x7 (RUN_TRP): Running (Transferring the received packet data from the Rx buffer to the system memory) **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_NUM_DMA_RX_CH>6)
23:20	TPS5	R	DMA Channel 5 Transmit Process State This field indicates the Tx DMA FSM state for Channel 5. The MSB of this field always returns 0. This field does not generate an interrupt. **Values:** ■ 0x0 (STOP): Stopped (Reset or Stop Transmit Command issued) ■ 0x1 (RUN_FTTD): Running (Fetching Tx Transfer Descriptor) ■ 0x2 (RUN_WS): Running (Waiting for status) ■ 0x3 (RUN_RDS): Running (Reading Data from system memory buffer and queuing it to the Tx buffer (Tx FIFO)) ■ 0x4 (TSTMP_WS): Timestamp write state ■ 0x5 (RSVD): Reserved for future use ■ 0x6 (SUSPND): Suspended (Tx Descriptor Unavailable or Tx Buffer Underflow) ■ 0x7 (RUN_CTD): Running (Closing Tx Descriptor) **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_NUM_DMA_TX_CH>5)

Bits	Name	Memory Access	Description
19:16	RPS5	R	DMA Channel 5 Receive Process State This field indicates the Rx DMA FSM state for Channel 5. The MSB of this field always returns 0. This field does not generate an interrupt. **Values:** ■ 0x0 (STOP): Stopped (Reset or Stop Receive Command issued) ■ 0x1 (RUN_FRTD): Running (Fetching Rx Transfer Descriptor) ■ 0x2 (RSVD): Reserved for future use ■ 0x3 (RUN_WRP): Running (Waiting for Rx packet) ■ 0x4 (SUSPND): Suspended (Rx Descriptor Unavailable) ■ 0x5 (RUN_CRD): Running (Closing the Rx Descriptor) ■ 0x6 (TSTMP): Timestamp write state ■ 0x7 (RUN_TRP): Running (Transferring the received packet data from the Rx buffer to the system memory) **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_NUM_DMA_RX_CH>5)
15:12	TPS4	R	DMA Channel 4 Transmit Process State This field indicates the Tx DMA FSM state for Channel 4. The MSB of this field always returns 0. This field does not generate an interrupt. **Values:** ■ 0x0 (STOP): Stopped (Reset or Stop Transmit Command issued) ■ 0x1 (RUN_FTTD): Running (Fetching Tx Transfer Descriptor) ■ 0x2 (RUN_WS): Running (Waiting for status) ■ 0x3 (RUN_RDS): Running (Reading Data from system memory buffer and queuing it to the Tx buffer (Tx FIFO)) ■ 0x4 (TSTMP_WS): Timestamp write state ■ 0x5 (RSVD): Reserved for future use ■ 0x6 (SUSPND): Suspended (Tx Descriptor Unavailable or Tx Buffer Underflow) ■ 0x7 (RUN_CTD): Running (Closing Tx Descriptor) **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_NUM_DMA_TX_CH>4)

Bits	Name	Memory Access	Description
11:8	RPS4	R	DMA Channel 4 Receive Process State This field indicates the Rx DMA FSM state for Channel 4. The MSB of this field always returns 0. This field does not generate an interrupt. **Values:** ■ 0x0 (STOP): Stopped (Reset or Stop Receive Command issued) ■ 0x1 (RUN_FRTD): Running (Fetching Rx Transfer Descriptor) ■ 0x2 (RSVD): Reserved for future use ■ 0x3 (RUN_WRP): Running (Waiting for Rx packet) ■ 0x4 (SUSPND): Suspended (Rx Descriptor Unavailable) ■ 0x5 (RUN_CRD): Running (Closing the Rx Descriptor) ■ 0x6 (TSTMP): Timestamp write state ■ 0x7 (RUN_TRP): Running (Transferring the received packet data from the Rx buffer to the system memory) **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_NUM_DMA_RX_CH>4)
7:4	TPS3	R	DMA Channel 3 Transmit Process State This field indicates the Tx DMA FSM state for Channel 3. The MSB of this field always returns 0. This field does not generate an interrupt. **Values:** ■ 0x0 (STOP): Stopped (Reset or Stop Transmit Command issued) ■ 0x1 (RUN_FTTD): Running (Fetching Tx Transfer Descriptor) ■ 0x2 (RUN_WS): Running (Waiting for status) ■ 0x3 (RUN_RDS): Running (Reading Data from system memory buffer and queuing it to the Tx buffer (Tx FIFO)) ■ 0x4 (TSTMP_WS): Timestamp write state ■ 0x5 (RSVD): Reserved for future use ■ 0x6 (SUSPND): Suspended (Tx Descriptor Unavailable or Tx Buffer Underflow) ■ 0x7 (RUN_CTD): Running (Closing Tx Descriptor) **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_NUM_DMA_TX_CH>3)

Bits	Name	Memory Access	Description
3:0	RPS3	R	DMA Channel 3 Receive Process State This field indicates the Rx DMA FSM state for Channel 3. The MSB of this field always returns 0. This field does not generate an interrupt. **Values:** ■ 0x0 (STOP): Stopped (Reset or Stop Receive Command issued) ■ 0x1 (RUN_FRTD): Running (Fetching Rx Transfer Descriptor) ■ 0x2 (RSVD): Reserved for future use ■ 0x3 (RUN_WRP): Running (Waiting for Rx packet) ■ 0x4 (SUSPND): Suspended (Rx Descriptor Unavailable) ■ 0x5 (RUN_CRD): Running (Closing the Rx Descriptor) ■ 0x6 (TSTMP): Timestamp write state ■ 0x7 (RUN_TRP): Running (Transferring the received packet data from the Rx buffer to the system memory) **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_NUM_DMA_RX_CH>3)

17.5.6 DMA_Debug_Status2

- **Description:** The Debug Status Register 2 gives the Receive and Transmit process status for DMA Channel 7.
- **Size:** 32 bits
- **Offset:** 0x1014
- **Exists:** ((DWC_EQOS_SYS>1)&&(DWC_EQOS_NUM_DMA_TX_CH>7||DWC_EQOS_NUM_DMA_RX_CH>7))

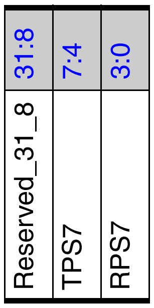


Table 17-284 Fields for Register: DMA_Debug_Status2

Bits	Name	Memory Access	Description
31:8	Reserved_31_8	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
7:4	TPS7	R	DMA Channel 7 Transmit Process State This field indicates the Tx DMA FSM state for Channel 7. The MSB of this field always returns 0. This field does not generate an interrupt. **Values:** ■ 0x0 (STOP): Stopped (Reset or Stop Transmit Command issued) ■ 0x1 (RUN_FTTD): Running (Fetching Tx Transfer Descriptor) ■ 0x2 (RUN_WS): Running (Waiting for status) ■ 0x3 (RUN_RDS): Running (Reading Data from system memory buffer and queuing it to the Tx buffer (Tx FIFO)) ■ 0x4 (TSTMP_WS): Timestamp write state ■ 0x5 (RSVD): Reserved for future use ■ 0x6 (SUSPND): Suspended (Tx Descriptor Unavailable or Tx Buffer Underflow) ■ 0x7 (RUN_CTD): Running (Closing Tx Descriptor) **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_NUM_DMA_TX_CH>7)

Bits	Name	Memory Access	Description
3:0	RPS7	R	DMA Channel 7 Receive Process State This field indicates the Rx DMA FSM state for Channel 7. The MSB of this field always returns 0. This field does not generate an interrupt. **Values:** ■ 0x0 (STOP): Stopped (Reset or Stop Receive Command issued) ■ 0x1 (RUN_FRTD): Running (Fetching Rx Transfer Descriptor) ■ 0x2 (RSVD): Reserved for future use ■ 0x3 (RUN_WRP): Running (Waiting for Rx packet) ■ 0x4 (SUSPND): Suspended (Rx Descriptor Unavailable) ■ 0x5 (RUN_CRD): Running (Closing the Rx Descriptor) ■ 0x6 (TSTMP): Timestamp write state ■ 0x7 (RUN_TRP): Running (Transferring the received packet data from the Rx buffer to the system memory) **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_NUM_DMA_RX_CH>7)

17.5.7 AXI4_Tx_AR_ACE_Control

- **Description:** This register is used to control the AXI4 Cache Coherency Signals for read transactions by all the Transmit DMA channels. The following signals of the AXI4 interface are driven with different values as programmed for corresponding type (descriptor, buffer1, buffer2) of access.
  - arcache_m_o[3:0]
  - ardomain_m_o[1:0]
- **Size:** 32 bits
- **Offset:** 0x1020
- **Exists:** (DWC_EQOS_SYS>4)

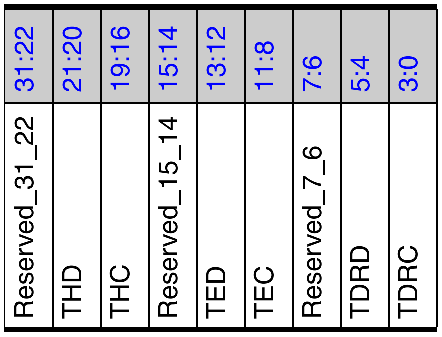


Table 17-285 Fields for Register: AXI4_Tx_AR_ACE_Control

Bits	Name	Memory Access	Description
31:22	Reserved_31_22	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
21:20	THD	R/W	Transmit DMA First Packet Buffer or TSO Header Domain Control When TSO is NOT enabled, This field is used to drive ardomain_o[1:0] signal when Transmit DMA is accessing First Buffer of the Packet (First valid buffer with FD being set in the TDES3 of the Descriptor). When TSO is enabled, This field is used to drive ardomain_o[1:0] signal when the Transmit DMA is accessing the TSO Header data. **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
19:16	THC	R/W	Transmit DMA First Packet Buffer or TSO Header Cache Control When TSO is NOT enabled, This field is used to drive arcache_o[3:0] signal when Transmit DMA is accessing First Buffer of the Packet (First valid buffer with FD being set in the TDES3 of the Descriptor).. When TSO is enabled, This field is used to drive arcache_o[3:0] signal when the Transmit DMA is accessing the TSO Header data. **Value After Reset:** 0x0 **Exists:** Always
15:14	Reserved_15_14	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
13:12	TED	R/W	Transmit DMA Extended Packet Buffer or TSO Payload Domain Control When TSO is NOT enabled, This field is used to drive ardomain_o[1:0] signal when Transmit DMA is accessing the extended buffers (when packet is distributed across multiple buffers). When TSO is enabled, This field is used to drive ardomain_o[1:0] signal when the Transmit DMA is accessing the TSO payload data. **Value After Reset:** 0x0 **Exists:** Always
11:8	TEC	R/W	Transmit DMA Extended Packet Buffer or TSO Payload Cache Control When TSO is NOT enabled, This field is used to drive arcache_o[3:0] signal when Transmit DMA is accessing the extended buffers (when packet is distributed across multiple buffers). When TSO is enabled, This field is used to drive arcache_o[3:0] signal when the Transmit DMA is accessing the TSO payload data. **Value After Reset:** 0x0 **Exists:** Always
7:6	Reserved_7_6	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
5:4	TDRD	R/W	Transmit DMA Read Descriptor Domain Control This field is used to drive ardomain_o[1:0] signal when Transmit DMA engines access the Descriptor. **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
3:0	TDRC	R/W	Transmit DMA Read Descriptor Cache Control This field is used to drive arcache_o[3:0] signal when Transmit DMA engines access the Descriptor. **Value After Reset:** 0x0 **Exists:** Always

17.5.8 AXI4_Rx_AW_ACE_Control

- **Description:** This register is used to control the AXI4 Cache Coherency Signals for write transactions by all the Receive DMA channels. The following signals of the AXI4 interface are driven with different values as programmed for corresponding type (descriptor, buffer1, buffer2) of access.
  - awcache_m_o[3:0]
  - awdomain_m_o[1:0]
- **Size:** 32 bits
- **Offset:** 0x1024
- **Exists:** (DWC_EQOS_SYS>4)

31:30	29:28	27:24	23:22	21:20	19:16	15:14	13:12	11:8	7:6	5:4	3:0
Reserved_31_30	RDD	RDC	Reserved_23_22	RHD	RHC	Reserved_15_14	RPD	RPC	Reserved_7_6	RDWD	RDWC

Table 17-286 Fields for Register: AXI4_Rx_AW_ACE_Control

Bits	Name	Memory Access	Description
31:30	Reserved_31_30	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
29:28	RDD	R/W	Receive DMA Buffer Domain Control This field is used to drive the awdomain_o[1:0] signal when Receive DMA is accessing the Buffer when Header and payload are NOT separated. **Value After Reset:** 0x0 **Exists:** Always
27:24	RDC	R/W	Receive DMA Buffer Cache Control This field is used to drive awcache_o[3:0] signal when Receive DMA is accessing the Buffer when Header and payload are NOT separated. **Value After Reset:** 0x0 **Exists:** Always
23:22	Reserved_23_22	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
21:20	RHD	R/W	Receive DMA Header Domain Control This field is used to drive awdomain_o[1:0] and signal when Receive DMA is accessing the header Buffer when Header and payload are separated. **Value After Reset:** 0x0 **Exists:** Always
19:16	RHC	R/W	Receive DMA Header Cache Control This field is used to drive awcache_o[3:0] and signal when Receive DMA is accessing the header Buffer when Header and payload are separated. **Value After Reset:** 0x0 **Exists:** Always
15:14	Reserved_15_14	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
13:12	RPD	R/W	Receive DMA Payload Domain Control This field is used to drive awdomain_o[1:0] signal when Receive DMA is accessing the Payload Buffer when Header and payload are separated. **Value After Reset:** 0x0 **Exists:** Always
11:8	RPC	R/W	Receive DMA Payload Cache Control This field is used to drive awcache_o[3:0] signal when Receive DMA is accessing the Payload Buffer when Header and payload are separated. **Value After Reset:** 0x0 **Exists:** Always
7:6	Reserved_7_6	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
5:4	RDWD	R/W	Receive DMA Write Descriptor Domain Control This field is used to drive awdomain_o[1:0] signal when Receive DMA accesses the Descriptor. **Value After Reset:** 0x0 **Exists:** Always
3:0	RDWC	R/W	Receive DMA Write Descriptor Cache Control This field is used to drive awcache_o[3:0] signal when Receive DMA accesses the Descriptor. **Value After Reset:** 0x0 **Exists:** Always

17.5.9 AXI4_TxRx_AWAR_ACE_Control

- **Description:** This register is used to control the AXI4 Cache Coherency Signals for Descriptor write transactions by all the TxDMA channels and Descriptor read transactions by all the RxDMA channels. It also controls the values to be driven on awprot_m_o and arprot_m_o.
- **Size:** 32 bits
- **Offset:** 0x1028
- **Exists:** (DWC_EQOS_SYS>4)

31:23	22:20	19	18:16	15:14	13:12	11:8	7:6	5:4	3:0
Reserved_31_23	WRP	Reserved_19	RDP	Reserved_15_14	RDRD	RDRC	Reserved_7_6	TDWD	TDWC

Table 17-287 Fields for Register: AXI4_TxRx_AWAR_ACE_Control

Bits	Name	Memory Access	Description
31:23	Reserved_31_23	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
22:20	WRP	R/W	DMA Write Protection control This field is used to drive awprot_m_o[2:0] signal on the AXI Write Channel. **Value After Reset:** 0x0 **Exists:** Always
19	Reserved_19	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
18:16	RDP	R/W	DMA Read Protection control This field is used to drive arprot_m_o[2:0] signal during all read requests. **Value After Reset:** 0x0 **Exists:** Always
15:14	Reserved_15_14	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
13:12	RDRD	R/W	Receive DMA Read Descriptor Domain control This field is used to drive ardomain_o[1:0] signal when Receive DMA engines read the Descriptor. **Value After Reset:** 0x0 **Exists:** Always
11:8	RDRC	R/W	Receive DMA Read Descriptor Cache control This field is used to drive arcache_o[3:0] signal when Receive DMA engines read the Descriptor. **Value After Reset:** 0x0 **Exists:** Always
7:6	Reserved_7_6	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
5:4	TDWD	R/W	Transmit DMA Write Descriptor Domain control This field is used to drive awdomain_o[1:0] signal when Transmit DMA write to the Descriptor. **Value After Reset:** 0x0 **Exists:** Always
3:0	TDWC	R/W	Transmit DMA Write Descriptor Cache control This field is used to drive awcache_o[3:0] signal when Transmit DMA writes to the Descriptor. **Value After Reset:** 0x0 **Exists:** Always

17.5.10 AXI_LPI_Entry_Interval

- **Description:** This register is used to control the AXI LPI entry interval.
- **Size:** 32 bits
- **Offset:** 0x1040
- **Exists:** (DWC_EQOS_SYS>3)

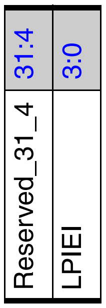


Table 17-288 Fields for Register: AXI_LPI_Entry_Interval

Bits	Name	Memory Access	Description
31:4	Reserved_31_4	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
3:0	LPIEI	R/W	LPI Entry Interval Contains the number of system clock cycles, multiplied by 64, to wait for an activity in the DWC_ether_qos to enter into the AXI low power state 0 indicates 64 clock cycles **Value After Reset:** 0x0 **Exists:** Always

17.5.11 DMA_TBS_CTRL

- **Description:** This register is used to control the TBS attributes.
- **Size:** 32 bits
- **Offset:** 0x1050
- **Exists:** (DWC_EQOS_SYS>1)&&DWC_EQOS_TBS

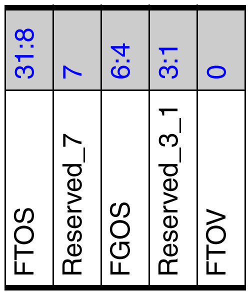


Table 17-289 Fields for Register: DMA_TBS_CTRL

Bits	Name	Memory Access	Description
31:8	FTOS	R/W	Fetch Time Offset The value in units of 256 nanoseconds, that has to be deducted from the Launch time to compute the Fetch Time. Max value: 999,999,999 ns, additionally should be smaller than CTR-1 value when ESTM mode is set since this value is a modulo CTR value. **Value After Reset:** 0x0 **Exists:** Always
7	Reserved_7	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
6:4	FGOS	R/W	Fetch GSN Offset The number GSN slots that must be deducted from the Launch GSN to compute the Fetch GSN. Value valid only when FTOV is set. **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_AV_EST)
3:1	Reserved_3_1	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
0	FTOV	R/W	Fetch Time Offset Valid When set indicates the FTOS field is valid. When not set, indicates the Fetch Offset is not valid and the DMA engine can fetch the frames from host memory without any time restrictions. **Values:** ■ 0x0 (INVALID): Fetch Time Offset is invalid ■ 0x1 (VALID): Fetch Time Offset is valid **Value After Reset:** 0x0 **Exists:** Always

17.5.12 DMA_Safety_Interrupt_Status

- **Description:** This register indicates summary (whether error occured in DMA/MTL/MAC and correctable/uncorrectable) of the Automotive Safety related error interrupts.
- **Size:** 32 bits
- **Offset:** 0x1080
- **Exists:** (DWC_EQOS_ASP_ECC)

31	30	29	28	27:2	1	0
MCSIS	Reserved_30	MSUIS	MSCIS	Reserved_27_2	DEUIS	DECIS

Table 17-290 Fields for Register: DMA_Safety_Interrupt_Status

Bits	Name	Memory Access	Description
31	MCSIS	R	MAC Safety Uncorrectable Interrupt Status Indicates a uncorrectable Safety related Interrupt is set in the MAC module. MAC_DPP_FSM_Interrupt_Status register should be read when this bit is set, to get the cause of the Safety Interrupt in MAC. **Values:** ■ 0x0 (INACTIVE): MAC Safety Uncorrectable Interrupt Status not detected ■ 0x1 (ACTIVE): MAC Safety Uncorrectable Interrupt Status detected **Value After Reset:** 0x0 **Exists:** Always
30	Reserved_30	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
29	MSUIS	R	MTL Safety Uncorrectable error Interrupt Status This bit indicates an uncorrectable error interrupt event in MTL. To get exact cause of the interrupt the software should read the MTL_Safety_Interrupt_Status register. **Values:** ■ 0x0 (INACTIVE): MTL Safety Uncorrectable error Interrupt Status not detected ■ 0x1 (ACTIVE): MTL Safety Uncorrectable error Interrupt Status detected **Value After Reset:** 0x0 **Exists:** Always
28	MSCIS	R	MTL Safety Correctable error Interrupt Status This bit indicates a correctable error interrupt event in MTL. To get exact cause of the interrupt the software should read the MTL_Safety_Interrupt_Status register. **Values:** ■ 0x0 (INACTIVE): MTL Safety Correctable error Interrupt Status not detected ■ 0x1 (ACTIVE): MTL Safety Correctable error Interrupt Status detected **Value After Reset:** 0x0 **Exists:** Always
27:2	Reserved_27_2	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
1	DEUIS	R	DMA ECC Uncorrectable error Interrupt Status This bit indicates an interrupt event in the DMA ECC safety feature. To get the exact cause of the interrupt the application should read the DMA_ECC_Interrupt_Status register. **Values:** ■ 0x0 (INACTIVE): DMA ECC Uncorrectable error Interrupt Status not detected ■ 0x1 (ACTIVE): DMA ECC Uncorrectable error Interrupt Status detected **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
0	DECIS	R	DMA ECC Correctable error Interrupt Status This bit indicates an interrupt event in the DMA ECC safety feature. To get the exact cause of the interrupt the application should read the DMA_ECC_Interrupt_Status register. **Values:** ■ 0x0 (INACTIVE): DMA ECC Correctable error Interrupt Status not detected ■ 0x1 (ACTIVE): DMA ECC Correctable error Interrupt Status detected **Value After Reset:** 0x0 **Exists:** Always

17.5.13 DMA_ECC_Interrupt_enable

- **Description:** This register is used to enable the Automotive Safety related TSO memory ECC error interrupt.
- **Size:** 32 bits
- **Offset:** 0x1084
- **Exists:** (DWC_EQOS_ASP_ECC&&DWC_EQOS_TSO_MEM_EN)

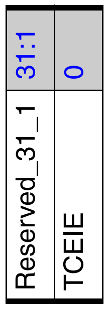


Table 17-291 Fields for Register: DMA_ECC_Interrupt_enable

Bits	Name	Memory Access	Description
31:1	Reserved_31_1	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
0	TCEIE	R/W	TSO memory Correctable Error Interrupt Enable When set, generates an interrupt when a correctable error is detected at the DMA TSO memory interface. It is indicated in the TCES bit of DMA_ECC_Interrupt_Status register. When reset this event does not generates an interrupt. **Values:** ■ 0x0 (DISABLE): TSO memory Correctable Error Interrupt is disabled ■ 0x1 (ENABLE): TSO memory Correctable Error Interrupt is enabled **Value After Reset:** 0x0 **Exists:** Always

17.5.14 DMA_ECC_Interrupt_Status

- **Description:** This register indicates the Automotive Safety related TSO memory ECC error interrupt status.
- **Size:** 32 bits
- **Offset:** 0x1088
- **Exists:** (DWC_EQOS_ASP_ECC&&DWC_EQOS_TSO_MEM_EN)

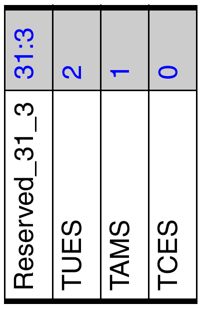


Table 17-292 Fields for Register: DMA_ECC_Interrupt_Status

Bits	Name	Memory Access	Description
31:3	Reserved_31_3	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
2	TUES	R/W	DMA TSO memory Uncorrectable Error status When set, indicates that an uncorrectable error is detected at DMA TSO memory interface. **Values:** ■ 0x0 (INACTIVE): DMA TSO memory Uncorrectable Error status not detected ■ 0x1 (ACTIVE): DMA TSO memory Uncorrectable Error status detected **Value After Reset:** 0x0 **Exists:** Always **Testable:** untestable
1	TAMS	R/W	DMA TSO memory Address Mismatch status This bit when set indicates that address mismatch is found for address bus of DMA TSO memory. **Values:** ■ 0x0 (INACTIVE): DMA TSO memory Address Mismatch status not detected ■ 0x1 (ACTIVE): DMA TSO memory Address Mismatch status detected **Value After Reset:** 0x0 **Exists:** Always **Testable:** untestable

Bits	Name	Memory Access	Description
0	TCES	R/W	DMA TSO memory Correctable Error status This bit when set indicates that correctable error is detected at DMA TSO memory interface. **Values:** ■ 0x0 (INACTIVE): DMA TSO memory Correctable Error status not detected ■ 0x1 (ACTIVE): DMA TSO memory Correctable Error status detected **Value After Reset:** 0x0 **Exists:** Always **Testable:** untestable

17.6 EQOS_DMA_CH0 Registers

17.6.1 DMA_CH(#i)_Control (for i = 0; i <= DWC_EQOS_NUM_DMA_CSR_CH-1)

- **Description:** The DMA Channel*i* Control register specifies the MSS value for segmentation, length to skip between two descriptors, and also the features such as header splitting and 8xPBL mode.
- **Size:** 32 bits
- **Offset:** (0x0080*i)+0x1100
- **Exists:** DWC_EQOS_SYS>1

31:25	24	23:21	20:18	17	16	15:14	13:0
Reserved_31_25	SPH	Reserved_23_21	DSL	Reserved_17	PBLx8	Reserved_15_14	MSS

Table 17-293 Fields for Register: DMA_CH(#i)_Control (for i = 0; i <= DWC_EQOS_NUM_DMA_CSR_CH-1)

Bits	Name	Memory Access	Description
31:25	Reserved_31_25	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
24	SPH	R/W	Split Headers When this bit is set, the DMA splits the header and payload in the Receive path. The DMA writes the header to the Buffer Address1 of RDES0. The DMA writes the payload to the buffer to which the Buffer Address2 is pointing. The software must ensure that the header fits into the Receive buffers. If the header length exceeds the receive buffer size, the DMA does not split the header and payload. This bit is available only if Enable Split Header Structure option is selected. **Values:** ■ 0x0 (DISABLE): Split Headers feature is disabled ■ 0x1 (ENABLE): Split Headers feature is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_SPLIT_HDR_EN
23:21	Reserved_23_21	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
20:18	DSL	R/W	Descriptor Skip Length This bit specifies the Word, Dword, or Lword number (depending on the 32-bit, 64-bit, or 128-bit bus) to skip between two unchained descriptors. The address skipping starts from the end of the current descriptor to the start of the next descriptor. When the DSL value is equal to zero, the DMA takes the descriptor table as contiguous. **Value After Reset:** 0x0 **Exists:** Always
17	Reserved_17	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
16	PBLx8	R/W	8xPBL mode When this bit is set, the PBL value programmed in Bits[21:16] in DMA_CH0_Tx_Control and Bits[21:16] in DMA_CH0_Rx_Control is multiplied by eight times. Therefore, the DMA transfers the data in 8, 16, 32, 64, 128, and 256 beats depending on the PBL value. **Values:** ■ 0x0 (DISABLE): 8xPBL mode is disabled ■ 0x1 (ENABLE): 8xPBL mode is enabled **Value After Reset:** 0x0 **Exists:** Always
15:14	Reserved_15_14	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
13:0	MSS	R/W	Maximum Segment Size This field specifies the maximum segment size that should be used while segmenting the packet. This field is valid only if the TSE bit of DMA_CH0_Tx_Control register is set. The value programmed in this field must be more than the configured Datawidth in bytes. It is recommended to use a MSS value of 64 bytes or more. **Value After Reset:** 0x0 **Exists:** DWC_EQOS_TSO_EN&&(DWC_EQOS_NUM_DMA_TSO_CH>0)

17.6.2 DMA_CH(#i)_Tx_Control (for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1)

- **Description:** The DMA Channel*i* Transmit Control register controls the Tx features such as PBL, TCP segmentation, and Tx Channel weights.
- **Size:** 32 bits
- **Offset:** (0x0080*i)+0x1104
- **Exists:** DWC_EQOS_SYS>1

31:29	28	27:24	23	22	21:16	15	14:13	12	11:5	4	3:1	0
Reserved_31_29	EDSE	TQOS	Reserved_23	ETIC	TxPBL	IPBL	TSE_MODE	TSE	Reserved_11_5	OSF	TCW	ST

Table 17-294 Fields for Register: DMA_CH(#i)_Tx_Control (for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1)

Bits	Name	Memory Access	Description
31:29	Reserved_31_29	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
28	EDSE	R/W	Enhanced Descriptor Enable When this bit is set, the corresponding channel uses Enhanced Descriptors that are 32 Bytes for both Normal and Context Descriptors. When reset, the corresponding channel uses the descriptors that are 16 Bytes. **Values:** ■ 0x0 (DISABLE): Enhanced Descriptor is disabled ■ 0x1 (ENABLE): Enhanced Descriptor is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_TBS
27:24	TQOS	R/W	Transmit QOS. This field is used to drive arqos_m_o[3:0] or awqos_m_o[3:0] output signals for all transactions of DMA Tx Channel0. **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_SYS>4)
23	Reserved_23	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
22	ETIC	R/W	Early Transmit Interrupt Control When this bit is set, Early Transmit Interrupt (ETI) status is set after completion of transfer of data from buffers of a transmit descriptor in which IOC bit (TDES2[31]) is set. When this bit is reset, ETI is set only after a complete packet is transferred to the MTL TX FIFO memory. **Values:** ■ 0x0 (DISABLE): Early Transmit Interrupt is disabled ■ 0x1 (ENABLE): Early Transmit Interrupt is enabled **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_AHB_SUBSYS&&!DWC_EQOS_CSR_SLV_CLK)
21:16	TxPBL	R/W	Transmit Programmable Burst Length These bits indicate the maximum number of beats to be transferred in one DMA block data transfer. The DMA always attempts max burst as specified in PBL each time it starts a burst transfer on the application bus. You can program PBL with any of the following values: 1, 2, 4, 8, 16, or 32. Any other value results in undefined behavior. To transfer more than 32 beats, perform the following steps: 1. Set the 8xPBL mode in DMA_CH0_Control register. 2. Set the TxPBL. Note: The maximum value of TxPBL must be less than or equal to half the Tx Queue size (TQS field of MTL_TxQ[n]_Operation_Mode register) in terms of beats. This is required so that the Tx Queue has space to store at least another Tx PBL worth of data while the MTL Tx Queue Controller is transferring data to MAC. For example, in 64-bit data width configurations the total locations in Tx Queue of size 512 bytes is 64, TxPBL and 8xPBL needs to be programmed to less than or equal to 32. **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
15	IPBL	R/W	Ignore PBL Requirement When this bit is set, the DMA does not check for PBL number of locations in the MTL before initiating a transfer. If space is not available, the MTL may use handshaking to slow the DMA. Note: This bit/mode must not be used when multiple Transmit DMA Channels are enabled as it may block other Transmit and Receive DMA Channels from accessing the Read Data Channel of AXI bus until space is available in Transmit Queue for current transfer. **Values:** ■ 0x0 (DISABLE): Ignore PBL Requirement is disabled ■ 0x1 (ENABLE): Ignore PBL Requirement is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_SYS==4||DWC_EQOS_SYS==5
14:13	TSE_MODE	R/W	TSE Mode ■ 00: TSO/USO (segmentation functionality is enabled). In this mode, the setting of TSE bit enables the TSO/USO segmentation. ■ 01: UFO with Checksum (UDP Fragmentation over IPv4 with Checksum). In this mode, the setting of TSE bit enables the UDP fragmentation functionality with Checksum for all the UDP packets. ■ 10: UFO without Checksum (UDP Fragmentation over IPv4 with Checksum). In this mode, the setting of TSE bit enables the UDP fragmentation functionality without Checksum for all the UDP packets. ■ 11: Reserved **Values:** ■ 0x0 (TSO_USO): TSO/USO ■ 0x1 (UFOWC): UFO with Checksum ■ 0x2 (UFOWOC): UFO without Checksum ■ 0x3 (RSVD): Reserved **Value After Reset:** 0x0 **Exists:** DWC_EQOS_UFO_EN

Bits	Name	Memory Access	Description
12	TSE	R/W	TCP Segmentation Enabled When this bit is set, the DMA performs the TCP segmentation or UDP Segmentation/Fragmentation for packets in this channel. The TCP segmentation or UDP packet's segmentation/Fragmentation is done only for those packets for which the TSE bit (TDES0[19]) is set in the Tx Normal descriptor.When this bit is set, the TxPBL value must be greater than 4. **Values:** ■ 0x0 (DISABLE): TCP Segmentation is disabled ■ 0x1 (ENABLE): TCP Segmentation is enabled **Value After Reset:** 0x0 **Exists:** DWC_EQOS_TSO_EN
11:5	Reserved_11_5	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
4	OSF	R/W	Operate on Second Packet When this bit is set, it instructs the DMA to process the second packet of the Transmit data even before the status for the first packet is obtained. **Values:** ■ 0x0 (DISABLE): Operate on Second Packet disabled ■ 0x1 (ENABLE): Operate on Second Packet enabled **Value After Reset:** 0x0 **Exists:** Always
3:1	TCW	R/W	Transmit Channel Weight This field indicates the weight assigned to the corresponding Transmit channel. When reset is complete, this field is set to 0 for all channels by default, resulting in equal weights to all channels. **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_SYS>1&&DWC_EQOS_SYS<4)&&(DWC_EQOS_NUM_DMA_TX_CH>1)

Bits	Name	Memory Access	Description
0	ST	R/W	Start or Stop Transmission Command When this bit is set, transmission is placed in the Running state. The DMA checks the Transmit list at the current position for a packet to be transmitted. The DMA tries to acquire descriptor from either of the following positions: ■ The current position in the list This is the base address of the Transmit list set by the DMA_CH0_TxDesc_List_Address register. ■ The position at which the transmission was previously stopped If the DMA does not own the current descriptor, the transmission enters the Suspended state and the TBU bit of the DMA_CH0_Status register is set. The Start Transmission command is effective only when the transmission is stopped. If the command is issued before setting the DMA_CH0_TxDesc_List_Address register, the DMA behavior is unpredictable. When this bit is reset, the transmission process is placed in the Stopped state after completing the transmission of the current packet. The Next Descriptor position in the Transmit list is saved, and it becomes the current position when the transmission is restarted. To change the list address, you need to program DMA_CH0_TxDesc_List_Address register with a new value when this bit is reset. The new value is considered when this bit is set again. The stop transmission command is effective only when the transmission of the current packet is complete or the transmission is in the Suspended state. **Values:** ■ 0x0 (STOP): Stop Transmission Command ■ 0x1 (START): Start Transmission Command **Value After Reset:** 0x0 **Exists:** Always **Testable:** untestable

17.6.3 DMA_CH(#i)_Rx_Control (for i = 0; i <= DWC_EQOS_NUM_DMA_RX_CH-1)

- **Description:** The DMA Channel*i* Receive Control register controls the Rx features such as PBL, buffer size, and extended status.
- **Size:** 32 bits
- **Offset:** (0x0080*i)+0x1108
- **Exists:** DWC_EQOS_SYS>1

31	30:28	27:24	23	22	21:16	15	14:y	x:1	0
RPF	Reserved_30_28	RQOS	Reserved_23	ERIC	RxPBL	Reserved_15	RBSZ_13_y	RBSZ_x_0	SR

Table 17-295 Fields for Register: DMA_CH(#i)_Rx_Control (for i = 0; i <= DWC_EQOS_NUM_DMA_RX_CH-1)

Bits	Name	Memory Access	Description
31	RPF	R/W	Rx Packet Flush. When this bit is set to 1, then DWC_ether_qos automatically flushes the packet from the Rx Queues destined to this DMA Rx Channel, when it is stopped. When this bit remains set and the DMA is re-started by the software driver, the packets residing in the Rx Queues that were received when this RxDMA was stopped, get flushed out. The packets that are received by the MAC after the RxDMA is re-started are routed to the RxDMA. The flushing happens on the Read side of the Rx Queue. When this bit is set to 0, the DWC_ether_qos not flush the packet in the Rx Queue destined to this RxDMA Channel when it is STOP state. This may in turn cause head-of-line blocking in the corresponding RxQueue. **Values:** ■ 0x0 (DISABLE): Rx Packet Flush is disabled ■ 0x1 (ENABLE): Rx Packet Flush is enabled **Value After Reset:** 0x0 **Exists:** Always
30:28	Reserved_30_28	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
27:24	RQOS	R/W	Rx AXI4 QOS. This field is used to drive arqos_m_o[3:0] or awqos_m_o[3:0] output signals for all transactions of DMA Rx Channel0. **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_SYS>4)
23	Reserved_23	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
22	ERIC	R/W	Early Receive Interrupt Control When this bit is set, Early Receive Interrupt (ERI) status is set after completion of every burst transfer of data from the Rx DMA to the buffer. When this bit is reset, ERI is set only after a complete buffer is filled up by the RxDMA. **Values:** ■ 0x0 (DISABLE): Early Receive Interrupt is disabled ■ 0x1 (ENABLE): Early Receive Interrupt is enabled **Value After Reset:** 0x0 **Exists:** (DWC_EQOS_AHB_SUBSYS&&!DWC_EQOS_CSR_SLV_CLK)
21:16	RxPBL	R/W	Receive Programmable Burst Length These bits indicate the maximum number of beats to be transferred in one DMA block data transfer. The DMA always attempts max burst as specified in PBL each time it starts a burst transfer on the application bus. You can program PBL with any of the following values: 1, 2, 4, 8, 16, or 32. Any other value results in undefined behavior. To transfer more than 32 beats, perform the following steps: 1. Set the 8xPBL mode in the DMA_CH0_Control register. 2. Set the RxPBL. Note: The maximum value of RxPBL must be less than or equal to half the Rx Queue size (RQS field of MTL_RxQ[n]_Operation_Mode register) in terms of beats. This is required so that the Rx Queue has space to store at least another Rx PBL worth of data while the Rx DMA is transferring a block of data. For example, in 64-bit data width configurations the total locations in Rx Queue of size 512 bytes is 64, so RxPBL and 8xPBL needs to be programmed to less than or equal to 32. **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
15	Reserved_15	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
14:y	RBSZ_13_y	R/W	Receive Buffer size High RBSZ[13:0] is split into two fields higher RBSZ_13_y and lower RBSZ_x_0. The RBSZ[13:0] field indicates the size of the Rx buffers specified in bytes. The maximum buffer size is limited to 16K bytes. The buffer size is applicable to payload buffers when split headers are enabled. **Note**: The buffer size must be a multiple of 4, 8, or 16 depending on the data bus widths (32-bit, 64-bit, or 128-bit respectively). This is required even if the value of buffer address pointer is not aligned to data bus width. Hence the lower RBSZ_x_0 bits are read-only and the value is considered as all-zero. Thus the RBSZ_13_y indicates the buffer size in terms of locations (with the width same as bus-width). **Value After Reset:** 0x0 **Exists:** Always **Range Variable[y]:** DWC_EQOS_BEW + 1
x:1	RBSZ_x_0	R	Receive Buffer size Low RBSZ[13:0] is split into two fields RBSZ_13_y and RBSZ_x_0. The RBSZ_x_0 is the lower field whose width is based on data bus width of the configuration. This field is of width 2, 3, or 4 bits for 32-bit, 64-bit, or 128-bit data bus width respectively. This field is read-only (RO). **Value After Reset:** 0x0 **Exists:** Always **Range Variable[x]:** DWC_EQOS_BEW

Bits	Name	Memory Access	Description
0	SR	R/W	Start or Stop Receive When this bit is set, the DMA tries to acquire the descriptor from the Receive list and processes the incoming packets. The DMA tries to acquire descriptor from either of the following positions: ■ The current position in the list This is the address set by the DMA_CH0_RxDesc_List_Address register. ■ The position at which the Rx process was previously stopped If the DMA does not own the current descriptor, the reception is suspended and the RBU bit of the DMA_CH0_Status register is set. The Start Receive command is effective only when the reception is stopped. If the command is issued before setting the DMA_CH0_RxDesc_List_Address register, the DMA behavior is unpredictable. When this bit is reset, the Rx DMA operation is stopped after the transfer of the current packet. The next descriptor position in the Receive list is saved, and it becomes the current position after the Rx process is restarted. The Stop Receive command is effective only when the Rx process is in the Running (waiting for Rx packet) or Suspended state. **Values:** ■ 0x0 (STOP): Stop Receive ■ 0x1 (START): Start Receive **Value After Reset:** 0x0 **Exists:** Always **Testable:** untestable

17.6.4 DMA_CH(#i)_TxDesc_List_HAddress (for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1)

- **Description:** The Channel*i* Tx Descriptor List HAddress register has the higher 8 or 16 bits of the start address of the Transmit descriptor list.
  You can write to this register only when the Tx DMA has stopped, that is, the ST bit is set to zero in DMA_CH*i*_Tx_Control register. When stopped, this register can be written with a new descriptor list address. When you set the ST bit to 1, the DMA takes the newly-programmed descriptor base address. If this register is not changed when the ST bit is set to 0, the DMA takes the descriptor address where it was stopped earlier. This register must be programmed with desired the higher address before programming the DMA_CH*i*_TxDesc_List_Address register.
- **Size:** 32 bits
- **Offset:** (0x0080*i)+0x1110
- **Exists:** DWC_EQOS_SYS>1&&DWC_EQOS_ADDRWIDTH_64

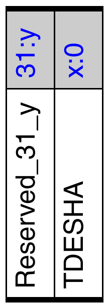


Table 17-296 Fields for Register: DMA_CH(#i)_TxDesc_List_HAddress (for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1)

Bits	Name	Memory Access	Description
31:y	Reserved_31_y	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
x:0	TDESHA	R/W	Start of Transmit List This field contains the most-significant 8 or 16 bits of the 40- or 48-bit base address of the first descriptor in the Transmit descriptor list. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[x]:** DWC_EQOS_ADDRWIDTH_MNS32 - 1

17.6.5 DMA_CH(#i)_TxDesc_List_Address (for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1)

- **Description:** The Channel*i* Tx Descriptor List Address register points the DMA to the start of Transmit descriptor list. The descriptor lists reside in the physical memory space of the application and must be Word, Dword, or Lword-aligned (for 32-bit, 64-bit, or 128-bit data bus). The DMA internally converts it to bus width aligned address by making the corresponding LSB to low.
  You can write to this register only when the Tx DMA has stopped, that is, the ST bit is set to zero in DMA_CH0_Tx_Control register. When stopped, this register can be written with a new descriptor list address. When you set the ST bit to 1, the DMA takes the newly-programmed descriptor base address. If this register is not changed when the ST bit is set to 0, the DMA takes the descriptor address where it was stopped earlier.
- **Size:** 32 bits
- **Offset:** (0x0080*i)+0x1114
- **Exists:** DWC_EQOS_SYS>1

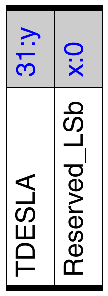


Table 17-297 Fields for Register: DMA_CH(#i)_TxDesc_List_Address (for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1)

Bits	Name	Memory Access	Description
31:y	TDESLA	R/W	Start of Transmit List This field contains the base address of the first descriptor in the Transmit descriptor list. The DMA ignores the LSB bits (1:0, 2:0, or 3:0) for 32-bit, 64-bit, or 128-bit bus width and internally takes these bits as all-zero. Therefore, these LSB bits are read-only (RO). The width of this field depends on the configuration: ■ 31:2 for 32-bit configuration ■ 31:3 for 64-bit configuration ■ 31:4 for 128-bit configuration **Value After Reset:** 0x0 **Exists:** Always **Testable:** untestable **Range Variable[y]:** DWC_EQOS_BEW
x:0	Reserved_LSb	R	**Value After Reset:** 0x0 **Exists:** Always **Range Variable[x]:** DWC_EQOS_BEW - 1

17.6.6 DMA_CH(#i)_RxDesc_List_HAddress (for i = 0; i <= DWC_EQOS_NUM_DMA_RX_CH-1)

- **Description:** The Channel*i* Rx Descriptor List HAddress register has the higher 8 or 16 bits of the start address of the Receive descriptor list.
  Writing to this register is permitted only when reception is stopped. When stopped, this register must be written to before the receive Start command is given. You can write to this register only when Rx DMA has stopped, that is, SR bit is set to zero in the DMA_CH*i*_RxDesc_List_Address register When you set the SR bit to 1, the DMA takes the newly programmed descriptor base address.
- **Size:** 32 bits
- **Offset:** (0x0080*i)+0x1118
- **Exists:** DWC_EQOS_SYS>1&&DWC_EQOS_ADDRWIDTH_64

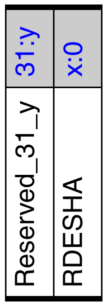


Table 17-298 Fields for Register: DMA_CH(#i)_RxDesc_List_HAddress (for i = 0; i <= DWC_EQOS_NUM_DMA_RX_CH-1)

Bits	Name	Memory Access	Description
31:y	Reserved_31_y	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
x:0	RDESHA	R/W	Start of Receive List This field contains the most-significant 8 or 16 bits of the 40-bit or 48-bit base address of the first descriptor in the Rx Descriptor list. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[x]:** DWC_EQOS_ADDRWIDTH_MNS32 - 1

17.6.7 DMA_CH(#i)_RxDesc_List_Address (for i = 0; i <= DWC_EQOS_NUM_DMA_RX_CH-1)

- **Description:** The Channel*i* Rx Descriptor List Address register points the DMA to the start of Receive descriptor list.
  This register points to the start of the Receive Descriptor List. The descriptor lists reside in the physical memory space of the application and must be Word, Dword, or Lword-aligned (for 32-bit, 64-bit, or 128-bit data bus). The DMA internally converts it to bus width aligned address by making the corresponding LS bits low. Writing to this register is permitted only when reception is stopped. When stopped, this register must be written to before the receive Start command is given. You can write to this register only when Rx DMA has stopped, that is, SR bit is set to zero in DMA_CH0_Rx_Control register. When stopped, this register can be written with a new descriptor list address.
  When you set the SR bit to 1, the DMA takes the newly programmed descriptor base address.
- **Size:** 32 bits
- **Offset:** (0x0080*i)+0x111C
- **Exists:** DWC_EQOS_SYS>1

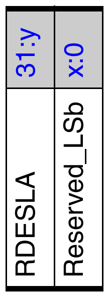


Table 17-299 Fields for Register: DMA_CH(#i)_RxDesc_List_Address (for i = 0; i <= DWC_EQOS_NUM_DMA_RX_CH-1)

Bits	Name	Memory Access	Description
31:y	RDESLA	R/W	Start of Receive List This field contains the base address of the first descriptor in the Rx Descriptor list. The DMA ignores the LSB bits (1:0, 2:0, or 3:0) for 32-bit, 64-bit, or 128-bit bus width and internally takes these bits as all-zero. Therefore, these LSB bits are read-only (RO). The width of this field depends on the configuration: ■ 31:2 for 32-bit configuration ■ 31:3 for 64-bit configuration ■ 31:4 for 128-bit configuration **Value After Reset:** 0x0 **Exists:** Always **Testable:** untestable **Range Variable[y]:** DWC_EQOS_BEW
x:0	Reserved_LSb	R	**Value After Reset:** 0x0 **Exists:** Always **Range Variable[x]:** DWC_EQOS_BEW - 1

17.6.8 DMA_CH(#i)_TxDesc_Tail_Pointer (for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1)

- **Description:** The Channel*i* Tx Descriptor Tail Pointer register points to an offset from the base and indicates the location of the last valid descriptor.
- **Size:** 32 bits
- **Offset:** (0x0080*i)+0x1120
- **Exists:** DWC_EQOS_SYS>1

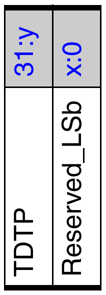


Table 17-300 **Fields for Register: DMA_CH(#i)_TxDesc_Tail_Pointer (for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1)**

Bits	Name	Memory Access	Description
31:y	TDTP	R/W	Transmit Descriptor Tail Pointer This field contains the tail pointer for the Tx descriptor ring. The software writes the tail pointer to add more descriptors to the Tx channel. The hardware tries to transmit all packets referenced by the descriptors between the head and the tail pointer registers. The width of this field depends on the configuration: ■ 31:2 for 32-bit configuration ■ 31:3 for 64-bit configuration ■ 31:4 for 128-bit configuration **Value After Reset:** 0x0 **Exists:** Always **Range Variable[y]:** DWC_EQOS_BEW
x:0	Reserved_LSb	R	**Value After Reset:** 0x0 **Exists:** Always **Range Variable[x]:** DWC_EQOS_BEW - 1

17.6.9 DMA_CH(#i)_RxDesc_Tail_Pointer (for i = 0; i <= DWC_EQOS_NUM_DMA_RX_CH-1)

- **Description:** The Channel*i* Rx Descriptor Tail Pointer Points to an offset from the base and indicates the location of the last valid descriptor.
- **Size:** 32 bits
- **Offset:** (0x0080*i)+0x1128
- **Exists:** DWC_EQOS_SYS>1

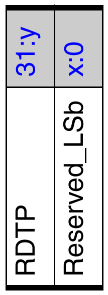


Table 17-301 **Fields for Register: DMA_CH(#i)_RxDesc_Tail_Pointer (for i = 0; i <= DWC_EQOS_NUM_DMA_RX_CH-1)**

Bits	Name	Memory Access	Description
31:y	RDTP	R/W	Receive Descriptor Tail Pointer This field contains the tail pointer for the Rx descriptor ring. The software writes the tail pointer to add more descriptors to the Rx channel. The hardware tries to write all received packets to the descriptors referenced between the head and the tail pointer registers. The width of this field depends on the configuration: ■ 31:2 for 32-bit configuration ■ 31:3 for 64-bit configuration ■ 31:4 for 128-bit configuration **Value After Reset:** 0x0 **Exists:** Always **Range Variable[y]:** DWC_EQOS_BEW
x:0	Reserved_LSb	R	**Value After Reset:** 0x0 **Exists:** Always **Range Variable[x]:** DWC_EQOS_BEW - 1

17.6.10 DMA_CH(#i)_TxDesc_Ring_Length (for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1)

- **Description:** The Tx Descriptor Ring Length register contains the length of the Transmit descriptor ring.
- **Size:** 32 bits
- **Offset:** (0x0080*i)+0x112C
- **Exists:** DWC_EQOS_SYS>1

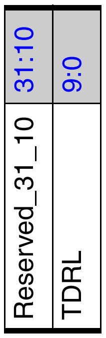


Table 17-302 Fields for Register: DMA_CH(#i)_TxDesc_Ring_Length (for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1)

Bits	Name	Memory Access	Description
31:10	Reserved_31_10	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
9:0	TDRL	R/W	Transmit Descriptor Ring Length This field sets the maximum number of Tx descriptors in the circular descriptor ring. The maximum number of descriptors is limited to 1K descriptors. Synopsys recommends a minimum ring descriptor length of 4. For example, You can program any value up to 0x3FF in this field. This field is 10 bits wide, if you program 0x3FF, you can have 1024 descriptors. If you want to have 10 descriptors, program it to a value of 0x9. **Value After Reset:** 0x0 **Exists:** Always

17.6.11 DMA_CH(#i)_RxDesc_Ring_Length (for i = 0; i <= DWC_EQOS_NUM_DMA_RX_CH-1)

- **Description:** The Channel*i* Rx Descriptor Ring Length register contains the length of the Receive descriptor circular ring.
- **Size:** 32 bits
- **Offset:** (0x0080*i)+0x1130
- **Exists:** DWC_EQOS_SYS>1

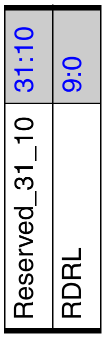


Table 17-303 Fields for Register: DMA_CH(#i)_RxDesc_Ring_Length (for i = 0; i <= DWC_EQOS_NUM_DMA_RX_CH-1)

Bits	Name	Memory Access	Description
31:10	Reserved_31_10	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
9:0	RDRL	R/W	Receive Descriptor Ring Length This register sets the maximum number of Rx descriptors in the circular descriptor ring. The maximum number of descriptors is limited to 1K descriptors. For example, You can program any value up to 0x3FF in this field. This field is 10 bits wide, if you program 0x3FF, you can have 1024 descriptors. If you want to have 10 descriptors, program it to a value of 0x9. **Value After Reset:** 0x0 **Exists:** Always

17.6.12 DMA_CH(#i)_Interrupt_Enable (for i = 0; i <= DWC_EQOS_NUM_DMA_CSR_CH-1)

- **Description:** The Channel*i* Interrupt Enable register enables the interrupts reported by the Status register.
- **Size:** 32 bits
- **Offset:** (0x0080*i)+0x1134
- **Exists:** DWC_EQOS_SYS>1

31:16	15	14	13	12	11	10	9	8	7	6	5:3	2	1	0
Reserved_31_16	NIE	AIE	CDEE	FBEE	ERIE	ETIE	RWTE	RSE	RBUE	RIE	Reserved_5_3	TBUE	TXSE	TIE

Table 17-304 Fields for Register: DMA_CH(#i)_Interrupt_Enable (for i = 0; i <= DWC_EQOS_NUM_DMA_CSR_CH-1)

Bits	Name	Memory Access	Description
31:16	Reserved_31_16	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
15	NIE	R/W	Normal Interrupt Summary Enable When this bit is set, the normal interrupt summary is enabled. This bit enables the following interrupts in the DMA_CH0_Status register: ■ Bit 0: Transmit Interrupt ■ Bit 2: Transmit Buffer Unavailable ■ Bit 6: Receive Interrupt ■ Bit 11: Early Receive Interrupt When this bit is reset, the normal interrupt summary is disabled. **Values:** ■ 0x0 (DISABLE): Normal Interrupt Summary is disabled ■ 0x1 (ENABLE): Normal Interrupt Summary is enabled **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
14	AIE	R/W	Abnormal Interrupt Summary Enable When this bit is set, the abnormal interrupt summary is enabled. This bit enables the following interrupts in the DMA_CH0_Status register: ■ Bit 1: Transmit Process Stopped ■ Bit 7: Rx Buffer Unavailable ■ Bit 8: Receive Process Stopped ■ Bit 9: Receive Watchdog Timeout ■ Bit 10: Early Transmit Interrupt ■ Bit 12: Fatal Bus Error ■ Bit 13: Context Descriptor Error When this bit is reset, the abnormal interrupt summary is disabled. **Values:** ■ 0x0 (DISABLE): Abnormal Interrupt Summary is disabled ■ 0x1 (ENABLE): Abnormal Interrupt Summary is enabled **Value After Reset:** 0x0 **Exists:** Always
13	CDEE	R/W	Context Descriptor Error Enable When this bit is set along with the AIE bit, the Descriptor error interrupt is enabled. When this bit is reset, the Descriptor error interrupt is disabled. **Values:** ■ 0x0 (DISABLE): Context Descriptor Error is disabled ■ 0x1 (ENABLE): Context Descriptor Error is enabled **Value After Reset:** 0x0 **Exists:** Always
12	FBEE	R/W	Fatal Bus Error Enable When this bit is set along with the AIE bit, the Fatal Bus error interrupt is enabled. When this bit is reset, the Fatal Bus Error error interrupt is disabled. **Values:** ■ 0x0 (DISABLE): Fatal Bus Error is disabled ■ 0x1 (ENABLE): Fatal Bus Error is enabled **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
11	ERIE	R/W	Early Receive Interrupt Enable When this bit is set along with the NIE bit, the Early Receive interrupt is enabled. When this bit is reset, the Early Receive interrupt is disabled. **Values:** ■ 0x0 (DISABLE): Early Receive Interrupt is disabled ■ 0x1 (ENABLE): Early Receive Interrupt is enabled **Value After Reset:** 0x0 **Exists:** ((DWC_EQOS_SYS>1)&&(DWC_EQOS_NUM_DMA_RX_CH>0))
10	ETIE	R/W	Early Transmit Interrupt Enable When this bit is set along with the AIE bit, the Early Transmit interrupt is enabled. When this bit is reset, the Early Transmit interrupt is disabled. **Values:** ■ 0x0 (DISABLE): Early Transmit Interrupt is disabled ■ 0x1 (ENABLE): Early Transmit Interrupt is enabled **Value After Reset:** 0x0 **Exists:** ((DWC_EQOS_SYS>1)&&(DWC_EQOS_NUM_DMA_TX_CH>0))
9	RWTE	R/W	Receive Watchdog Timeout Enable When this bit is set along with the AIE bit, the Receive Watchdog Timeout interrupt is enabled. When this bit is reset, the Receive Watchdog Timeout interrupt is disabled. **Values:** ■ 0x0 (DISABLE): Receive Watchdog Timeout is disabled ■ 0x1 (ENABLE): Receive Watchdog Timeout is enabled **Value After Reset:** 0x0 **Exists:** ((DWC_EQOS_SYS>1)&&(DWC_EQOS_NUM_DMA_RX_CH>0))
8	RSE	R/W	Receive Stopped Enable When this bit is set along with the AIE bit, the Receive Stopped Interrupt is enabled. When this bit is reset, the Receive Stopped interrupt is disabled. **Values:** ■ 0x0 (DISABLE): Receive Stopped is disabled ■ 0x1 (ENABLE): Receive Stopped is enabled **Value After Reset:** 0x0 **Exists:** ((DWC_EQOS_SYS>1)&&(DWC_EQOS_NUM_DMA_RX_CH>0))

Bits	Name	Memory Access	Description
7	RBUE	R/W	Receive Buffer Unavailable Enable When this bit is set along with the AIE bit, the Receive Buffer Unavailable interrupt is enabled. When this bit is reset, the Receive Buffer Unavailable interrupt is disabled. **Values:** ■ 0x0 (DISABLE): Receive Buffer Unavailable is disabled ■ 0x1 (ENABLE): Receive Buffer Unavailable is enabled **Value After Reset:** 0x0 **Exists:** ((DWC_EQOS_SYS>1)&&(DWC_EQOS_NUM_DMA_RX_CH>0))
6	RIE	R/W	Receive Interrupt Enable When this bit is set along with the NIE bit, the Receive Interrupt is enabled. When this bit is reset, the Receive Interrupt is disabled. **Values:** ■ 0x0 (DISABLE): Receive Interrupt is disabled ■ 0x1 (ENABLE): Receive Interrupt is enabled **Value After Reset:** 0x0 **Exists:** ((DWC_EQOS_SYS>1)&&(DWC_EQOS_NUM_DMA_RX_CH>0))
5:3	Reserved_5_3	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
2	TBUE	R/W	Transmit Buffer Unavailable Enable When this bit is set along with the NIE bit, the Transmit Buffer Unavailable interrupt is enabled. When this bit is reset, the Transmit Buffer Unavailable interrupt is disabled. **Values:** ■ 0x0 (DISABLE): Transmit Buffer Unavailable is disabled ■ 0x1 (ENABLE): Transmit Buffer Unavailable is enabled **Value After Reset:** 0x0 **Exists:** ((DWC_EQOS_SYS>1)&&(DWC_EQOS_NUM_DMA_TX_CH>0))

Bits	Name	Memory Access	Description
1	TXSE	R/W	Transmit Stopped Enable When this bit is set along with the AIE bit, the Transmission Stopped interrupt is enabled. When this bit is reset, the Transmission Stopped interrupt is disabled. **Values:** ■ 0x0 (DISABLE): Transmit Stopped is disabled ■ 0x1 (ENABLE): Transmit Stopped is enabled **Value After Reset:** 0x0 **Exists:** ((DWC_EQOS_SYS>1)&&(DWC_EQOS_NUM_DMA_TX_CH >0))
0	TIE	R/W	Transmit Interrupt Enable When this bit is set along with the NIE bit, the Transmit Interrupt is enabled. When this bit is reset, the Transmit Interrupt is disabled. **Values:** ■ 0x0 (DISABLE): Transmit Interrupt is disabled ■ 0x1 (ENABLE): Transmit Interrupt is enabled **Value After Reset:** 0x0 **Exists:** ((DWC_EQOS_SYS>1)&&(DWC_EQOS_NUM_DMA_TX_CH >0))

17.6.13 DMA_CH(#i)_Rx_Interrupt_Watchdog_Timer (for i = 0; i <= DWC_EQOS_NUM_DMA_RX_CH-1)

- **Description:** The Receive Interrupt Watchdog Timer register indicates the watchdog timeout for Receive Interrupt (RI) from the DMA. When this register is written with a non-zero value, it enables the watchdog timer for the RI bit of the DMA_CH*i*_Status register.
- **Size:** 32 bits
- **Offset:** (0x0080*i)+0x1138
- **Exists:** DWC_EQOS_SYS>1

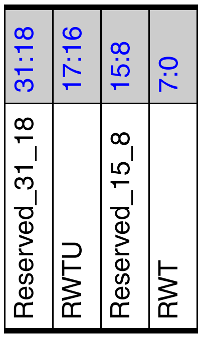


Table 17-305 Fields for Register: DMA_CH(#i)_Rx_Interrupt_Watchdog_Timer (for i = 0; i <= DWC_EQOS_NUM_DMA_RX_CH-1)

Bits	Name	Memory Access	Description
31:18	Reserved_31_18	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
17:16	RWTU	R/W	Receive Interrupt Watchdog Timer Count Units This fields indicates the number of system clock cycles corresponding to one unit in RWT field. ■ 2'b00: 256 ■ 2'b01: 512 ■ 2'b10: 1024 ■ 2'b11: 2048 For example, when RWT=2 and RWTU=1, the watchdog timer is set for 2*512=1024 system clock cycles. **Value After Reset:** 0x0 **Exists:** Always **Testable:** untestable
15:8	Reserved_15_8	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
7:0	RWT	R/W	Receive Interrupt Watchdog Timer Count This field indicates the number of system clock cycles, multiplied by factor indicated in RWTU field, for which the watchdog timer is set. The watchdog timer is triggered with the programmed value after the Rx DMA completes the transfer of a packet for which the RI bit is not set in the DMA_CH(#i)_Status register, because of the setting of Interrupt Enable bit in the corresponding descriptor RDES3[30]. When the watchdog timer runs out, the RI bit is set and the timer is stopped. The watchdog timer is reset when the RI bit is set high because of automatic setting of RI as per the Interrupt Enable bit RDES3[30] of any received packet. **Value After Reset:** 0x0 **Exists:** Always **Testable:** untestable

17.6.14 DMA_CH(#i)_Slot_Function_Control_Status (for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1)

- **Description:** The Slot Function Control and Status register contains the control bits for slot function and the status for Transmit path.
- **Size:** 32 bits
- **Offset:** (0x0080*i)+0x113C
- **Exists:** DWC_EQOS_SYS>1&&DWC_EQOS_AV_SLOT_EN

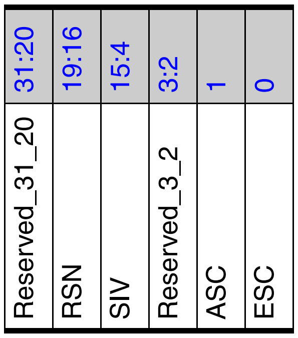


Table 17-306 Fields for Register: DMA_CH(#i)_Slot_Function_Control_Status (for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1)

Bits	Name	Memory Access	Description
31:20	Reserved_31_20	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
19:16	RSN	R	Reference Slot Number This field gives the current value of the reference slot number in the DMA. It is used for slot comparison. **Value After Reset:** 0x0 **Exists:** Always **Testable:** untestable
15:4	SIV	R/W	Slot Interval Value This field controls the period of the slot interval in which the TxDMA fetches the scheduled packets. A value of 0 specifies the slot interval of 1 us while the maximum value 4095 specifies the slot interval of 4096us. The default/reset value is 0x07C which corresponds to slot interval of 125us **Value After Reset:** 0x7c **Exists:** Always
3:2	Reserved_3_2	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
1	ASC	R/W	Advance Slot Check When set, this bit enables the DMA to fetch the data from the buffer when the slot number (SLOTNUM) programmed in the Tx descriptor is ■ equal to the reference slot number given in the RSN field or ■ ahead of the reference slot number by up to two slots This bit is applicable only when the ESC bit is set. **Values:** ■ 0x0 (DISABLE): Advance Slot Check is disabled ■ 0x1 (ENABLE): Advance Slot Check is enabled **Value After Reset:** 0x0 **Exists:** Always
0	ESC	R/W	Enable Slot Comparison When set, this bit enables the checking of the slot numbers programmed in the Tx descriptor with the current reference given in the RSN field. The DMA fetches the data from the corresponding buffer only when the slot number is ■ equal to the reference slot number or ■ ahead of the reference slot number by one slot When reset, this bit disables the checking of the slot numbers. The DMA fetches the data immediately after the descriptor is processed. **Note:** The UFO (UDP Fragmentation over IPv4)/TSO/USO should not be enabled along with TBS/AVB Slot number check. The UFO/TSO/USO involves multiple packets/segments/fragments transmission for single packet received from application and the slot number check are applicable for fetching of only first segment/fragment. As a result it might be difficult for software to specify slot number for subsequent packets. **Values:** ■ 0x0 (DISABLE): Slot Comparison is disabled ■ 0x1 (ENABLE): Slot Comparison is enabled **Value After Reset:** 0x0 **Exists:** Always

17.6.15 DMA_CH(#i)_Current_App_TxDesc (for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1)

- **Description:** The Channel*i* Current Application Transmit Descriptor register points to the current Transmit descriptor read by the DMA.
- **Size:** 32 bits
- **Offset:** (0x0080*i)+0x1144
- **Exists:** DWC_EQOS_SYS>1

Table 17-307 Fields for Register: DMA_CH(#i)_Current_App_TxDesc (for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1)

Bits	Name	Memory Access	Description
31:0	CURTDESAPTR	R	Application Transmit Descriptor Address Pointer The DMA updates this pointer during Tx operation. This pointer is cleared on reset. **Value After Reset:** 0x0 **Exists:** Always

17.6.16 DMA_CH(#i)_Current_App_RxDesc (for i = 0; i <= DWC_EQOS_NUM_DMA_RX_CH-1)

- **Description:** The Channel*i* Current Application Receive Descriptor register points to the current Receive descriptor read by the DMA.
- **Size:** 32 bits
- **Offset:** (0x0080*i)+0x114C
- **Exists:** DWC_EQOS_SYS>1

31:0
CURRDESAPTR

Table 17-308 Fields for Register: DMA_CH(#i)_Current_App_RxDesc (for i = 0; i <= DWC_EQOS_NUM_DMA_RX_CH-1)

Bits	Name	Memory Access	Description
31:0	CURRDESAPTR	R	Application Receive Descriptor Address Pointer The DMA updates this pointer during Rx operation. This pointer is cleared on reset. **Value After Reset:** 0x0 **Exists:** Always

17.6.17 DMA_CH(#i)_Current_App_TxBuffer_H (for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1)

- **Description:** The Channel*i* Current Application Transmit Buffer Address High register has the higher 8 or 16 bits of the current address of the Transmit buffer address read by the DMA.
- **Size:** 32 bits
- **Offset:** (0x0080*i)+0x1150
- **Exists:** DWC_EQOS_SYS>1&&DWC_EQOS_ADDRWIDTH_64&&(DWC_EQOS_NUM_DMA_TX_CH>0)

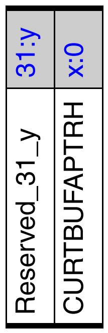


Table 17-309 Fields for Register: DMA_CH(#i)_Current_App_TxBuffer_H (for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1)

Bits	Name	Memory Access	Description
31:y	Reserved_31_y	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
x:0	CURTBUFAPTRH	R	Application Transmit Buffer Address Pointer The DMA updates this pointer during Tx operation. This pointer is cleared on reset. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[x]:** DWC_EQOS_ADDRWIDTH_MNS32 - 1

17.6.18 DMA_CH(#i)_Current_App_TxBuffer (for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1)

- **Description:** The Channel*i* Current Application Transmit Buffer Address register points to the current Tx buffer address read by the DMA.
- **Size:** 32 bits
- **Offset:** (0x0080*i)+0x1154
- **Exists:** DWC_EQOS_SYS>1

Table 17-310 Fields for Register: DMA_CH(#i)_Current_App_TxBuffer (for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1)

Bits	Name	Memory Access	Description
31:0	CURTBUFAPTR	R	Application Transmit Buffer Address Pointer The DMA updates this pointer during Tx operation. This pointer is cleared on reset. **Value After Reset:** 0x0 **Exists:** Always

17.6.19 DMA_CH(#i)_Current_App_RxBuffer_H (for i = 0; i <= DWC_EQOS_NUM_DMA_RX_CH-1)

- **Description:** The Channel*i* Current Application Receive Buffer Address High register has the higher 8 or 16 bits of the current address of the Receive buffer address read by the DMA.
- **Size:** 32 bits
- **Offset:** (0x0080*i)+0x1158
- **Exists:** DWC_EQOS_SYS>1&&DWC_EQOS_ADDRWIDTH_64&&(DWC_EQOS_NUM_DMA_RX_CH>0)

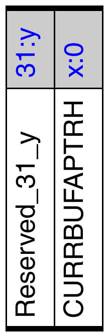


Table 17-311 Fields for Register: DMA_CH(#i)_Current_App_RxBuffer_H (for i = 0; i <= DWC_EQOS_NUM_DMA_RX_CH-1)

Bits	Name	Memory Access	Description
31:y	Reserved_31_y	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
x:0	CURRBUFAPTRH	R	Application Receive Buffer Address Pointer The DMA updates this pointer during Rx operation. This pointer is cleared on reset. **Value After Reset:** 0x0 **Exists:** Always **Range Variable[x]:** DWC_EQOS_ADDRWIDTH_MNS32 - 1

17.6.20 DMA_CH(#i)_Current_App_RxBuffer (for i = 0; i <= DWC_EQOS_NUM_DMA_RX_CH-1)

- **Description:** The Channel 0 Current Application Receive Buffer Address register points to the current Rx buffer address read by the DMA.
- **Size:** 32 bits
- **Offset:** (0x0080*i)+0x115C
- **Exists:** DWC_EQOS_SYS>1

31:0
CURRBUFAPTR

Table 17-312 Fields for Register: DMA_CH(#i)_Current_App_RxBuffer (for i = 0; i <= DWC_EQOS_NUM_DMA_RX_CH-1)

Bits	Name	Memory Access	Description
31:0	CURRBUFAPTR	R	Application Receive Buffer Address Pointer The DMA updates this pointer during Rx operation. This pointer is cleared on reset. **Value After Reset:** 0x0 **Exists:** Always

17.6.21 DMA_CH(#i)_Status (for i = 0; i <= DWC_EQOS_NUM_DMA_CSR_CH-1)

- **Description:** The software driver (application) reads the Status register during interrupt service routine or polling to determine the status of the DMA.
  Note: The number of DMA_CH[n]_Status register in the configuration is the higher of number of Rx DMA Channels and Tx DMA Channels.
- **Size:** 32 bits
- **Offset:** (0x0080*i)+0x1160
- **Exists:** DWC_EQOS_SYS>1

31:22	21:19	18:16	15	14	13	12	11	10	9	8	7	6	5:3	2	1	0
Reserved_31_22	REB	TEB	NIS	AIS	CDE	FBE	ERI	ETI	RWT	RPS	RBU	RI	Reserved_5_3	TBU	TPS	TI

Table 17-313 Fields for Register: DMA_CH(#i)_Status (for i = 0; i <= DWC_EQOS_NUM_DMA_CSR_CH-1)

Bits	Name	Memory Access	Description
31:22	Reserved_31_22	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
21:19	REB	R	Rx DMA Error Bits This field indicates the type of error that caused a Bus Error. For example, error response on the AHB or AXI interface. ■ Bit 21 ❑ 1'b1: Error during data transfer by Rx DMA ❑ 1'b0: No Error during data transfer by Rx DMA ■ Bit 20 ❑ 1'b1: Error during descriptor access ❑ 1'b0: Error during data buffer access ■ Bit 19 ❑ 1'b1: Error during read transfer ❑ 1'b0: Error during write transfer This field is valid only when the FBE bit is set. This field does not generate an interrupt. **Value After Reset:** 0x0 **Exists:** Always **Testable:** untestable

Bits	Name	Memory Access	Description
18:16	TEB	R	Tx DMA Error Bits This field indicates the type of error that caused a Bus Error. For example, error response on the AHB or AXI interface. ■ Bit 18 ❑ 1'b1: Error during data transfer by Tx DMA ❑ 1'b0: No Error during data transfer by Tx DMA ■ Bit 17 ❑ 1'b1: Error during descriptor access ❑ 1'b0: Error during data buffer access ■ Bit 16 ❑ 1'b1: Error during read transfer ❑ 1'b0: Error during write transfer This field is valid only when the FBE bit is set. This field does not generate an interrupt. **Value After Reset:** 0x0 **Exists:** Always **Testable:** untestable
15	NIS	R/W	Normal Interrupt Summary Normal Interrupt Summary bit value is the logical OR of the following bits when the corresponding interrupt bits are enabled in the DMA_CH0_Interrupt_Enable register: ■ Bit 0: Transmit Interrupt ■ Bit 2: Transmit Buffer Unavailable ■ Bit 6: Receive Interrupt ■ Bit 11: Early Receive Interrupt Only unmasked bits (interrupts for which interrupt enable is set in DMA_CH0_Interrupt_Enable register) affect the Normal Interrupt Summary bit. This is a sticky bit. You must clear this bit (by writing 1 to this bit) each time a corresponding bit which causes NIS to be set is cleared. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. **Values:** ■ 0x0 (INACTIVE): Normal Interrupt Summary status not detected ■ 0x1 (ACTIVE): Normal Interrupt Summary status detected **Value After Reset:** 0x0 **Exists:** Always **Testable:** untestable

Bits	Name	Memory Access	Description
14	AIS	R/W	Abnormal Interrupt Summary Abnormal Interrupt Summary bit value is the logical OR of the following when the corresponding interrupt bits are enabled in the DMA_CH0_Interrupt_Enable register: ■ Bit 1: Transmit Process Stopped ■ Bit 7: Receive Buffer Unavailable ■ Bit 8: Receive Process Stopped ■ Bit 10: Early Transmit Interrupt ■ Bit 12: Fatal Bus Error ■ Bit 13: Context Descriptor Error Only unmasked bits affect the Abnormal Interrupt Summary bit. This is a sticky bit. You must clear this bit (by writing 1 to this bit) each time a corresponding bit, which causes AIS to be set, is cleared. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. **Values:** ■ 0x0 (INACTIVE): Abnormal Interrupt Summary status not detected ■ 0x1 (ACTIVE): Abnormal Interrupt Summary status detected **Value After Reset:** 0x0 **Exists:** Always **Testable:** untestable
13	CDE	R/W	Context Descriptor Error This bit indicates that the DMA Tx/Rx engine received a descriptor error, which indicates invalid context in the middle of packet flow (intermediate descriptor) or all one's descriptor in Tx case and on Rx side it indicates DMA has read a descriptor with either of the buffer address as ones which is considered to be invalid. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. **Values:** ■ 0x0 (INACTIVE): Context Descriptor Error status not detected ■ 0x1 (ACTIVE): Context Descriptor Error status detected **Value After Reset:** 0x0 **Exists:** Always **Testable:** untestable

Bits	Name	Memory Access	Description
12	FBE	R/W	Fatal Bus Error This bit indicates that a bus error occurred (as described in the EB field). When this bit is set, the corresponding DMA channel engine disables all bus accesses. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. **Values:** ■ 0x0 (INACTIVE): Fatal Bus Error status not detected ■ 0x1 (ACTIVE): Fatal Bus Error status detected **Value After Reset:** 0x0 **Exists:** Always **Testable:** untestable
11	ERI	R/W	Early Receive Interrupt This bit when set indicates that the RxDMA has completed the transfer of packet data to the memory. In configs supporting ERIC, When ERIC=0, this bit is set only after the Rx DMA has filled up a complete receive buffer with packet data. When ERIC=1, this bit is set after every burst transfer of data from the Rx DMA to the buffer. The setting of RI bit automatically clears this bit. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. **Values:** ■ 0x0 (INACTIVE): Early Receive Interrupt status not detected ■ 0x1 (ACTIVE): Early Receive Interrupt status detected **Value After Reset:** 0x0 **Exists:** Always **Testable:** untestable

Bits	Name	Memory Access	Description
10	ETI	R/W	Early Transmit Interrupt This bit when set indicates that the TxDMA has completed the transfer of packet data to the MTL TXFIFO memory. In configs supporting ERIC: When ETIC=0, this bit is set only after the Tx DMA has transferred a complete packet to MTL. When ETIC=1, this bit is set after completion of (partial) packet data transfer from buffers in the Transmit descriptor in which IOC=1. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. **Values:** ■ 0x0 (INACTIVE): Early Transmit Interrupt status not detected ■ 0x1 (ACTIVE): Early Transmit Interrupt status detected **Value After Reset:** 0x0 **Exists:** Always **Testable:** untestable
9	RWT	R/W	Receive Watchdog Timeout This bit is asserted when a packet with length greater than 2,048 bytes (10,240 bytes when Jumbo Packet mode is enabled) is received. **Values:** ■ 0x0 (INACTIVE): Receive Watchdog Timeout status not detected ■ 0x1 (ACTIVE): Receive Watchdog Timeout status detected **Value After Reset:** 0x0 **Exists:** Always **Testable:** untestable
8	RPS	R/W	Receive Process Stopped This bit is asserted when the Rx process enters the Stopped state. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. **Values:** ■ 0x0 (INACTIVE): Receive Process Stopped status not detected ■ 0x1 (ACTIVE): Receive Process Stopped status detected **Value After Reset:** 0x0 **Exists:** Always **Testable:** untestable

Bits	Name	Memory Access	Description
7	RBU	R/W	Receive Buffer Unavailable This bit indicates that the application owns the next descriptor in the Receive list, and the DMA cannot acquire it. The Rx process is suspended. To resume processing Rx descriptors, the application should change the ownership of the descriptor and issue a Receive Poll Demand command. If this command is not issued, the Rx process resumes when the next recognized incoming packet is received. In ring mode, the application should advance the Receive Descriptor Tail Pointer register of a channel. This bit is set only when the DMA owns the previous Rx descriptor. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. **Values:** ■ 0x0 (INACTIVE): Receive Buffer Unavailable status not detected ■ 0x1 (ACTIVE): Receive Buffer Unavailable status detected **Value After Reset:** 0x0 **Exists:** Always **Testable:** untestable
6	RI	R/W	Receive Interrupt This bit indicates that the packet reception is complete. When packet reception is complete, Bit 31 of RDES3 is reset in the last descriptor, and the specific packet status information is updated in the descriptor. The reception remains in the Running state. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. **Values:** ■ 0x0 (INACTIVE): Receive Interrupt status not detected ■ 0x1 (ACTIVE): Receive Interrupt status detected **Value After Reset:** 0x0 **Exists:** Always **Testable:** untestable
5:3	Reserved_5_3	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
2	TBU	R/W	Transmit Buffer Unavailable This bit indicates that the application owns the next descriptor in the Transmit list, and the DMA cannot acquire it. Transmission is suspended. The TPS0 field of the DMA_Debug_Status0 register explains the Transmit Process state transitions. To resume processing the Transmit descriptors, the application should do the following: 1. Change the ownership of the descriptor by setting Bit 31 of TDES3. 2. Issue a Transmit Poll Demand command. For ring mode, the application should advance the Transmit Descriptor Tail Pointer register of a channel. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. **Values:** ■ 0x0 (INACTIVE): Transmit Buffer Unavailable status not detected ■ 0x1 (ACTIVE): Transmit Buffer Unavailable status detected **Value After Reset:** 0x0 **Exists:** Always **Testable:** untestable
1	TPS	R/W	Transmit Process Stopped This bit is set when the transmission is stopped. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. **Values:** ■ 0x0 (INACTIVE): Transmit Process Stopped status not detected ■ 0x1 (ACTIVE): Transmit Process Stopped status detected **Value After Reset:** 0x0 **Exists:** Always **Testable:** untestable

Bits	Name	Memory Access	Description
0	TI	R/W	Transmit Interrupt This bit indicates that the packet transmission is complete. When transmission is complete, Bit 31 of TDES3 is reset in the last descriptor, and the specific packet status information is updated in the descriptor. Access restriction applies. Self-set to 1 on internal event. Setting 1 clears. Setting 0 has no effect. **Values:** ■ 0x0 (INACTIVE): Transmit Interrupt status not detected ■ 0x1 (ACTIVE): Transmit Interrupt status detected **Value After Reset:** 0x0 **Exists:** Always **Testable:** untestable

17.6.22 DMA_CH(#i)_Miss_Frame_Cnt (for i = 0; i <= DWC_EQOS_NUM_DMA_RX_CH-1)

- **Description:** This register has the number of packet counter that got dropped by the DMA either due to Bus Error or due to programming RPF field in DMA_CH${i}_Rx_Control register.
- **Size:** 32 bits
- **Offset:** (0x0080*i)+0x1164
- **Exists:** DWC_EQOS_SYS>1

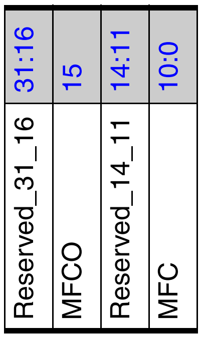


Table 17-314 Fields for Register: DMA_CH(#i)_Miss_Frame_Cnt (for i = 0; i <= DWC_EQOS_NUM_DMA_RX_CH-1)

Bits	Name	Memory Access	Description
31:16	Reserved_31_16	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always
15	MFCO	R	Overflow status of the MFC Counter When this bit is set then the MFC counter does not get incremented further. The bit gets cleared when this register is read. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): Miss Frame Counter overflow not occurred ■ 0x1 (ACTIVE): Miss Frame Counter overflow occurred **Value After Reset:** 0x0 **Exists:** Always
14:11	Reserved_14_11	R	Reserved. **Value After Reset:** 0x0 **Exists:** Always

Bits	Name	Memory Access	Description
10:0	MFC	R	Dropped Packet Counters This counter indicates the number of packet counters that are dropped by the DMA either because of bus error or because of programing RPF field in DMA_CH${i}_Rx_Control register. The counter gets cleared when this register is read. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Value After Reset:** 0x0 **Exists:** Always

17.6.23 DMA_CH(#i)_RXP_Accept_Cnt (for i = 0; i <= DWC_EQOS_NUM_DMA_RX_CH-1)

- **Description:** The DMA_CH(#i)_RXP_Accept_Cnt registers provides the count of the number of frames accepted by Rx Parser.
- **Size:** 32 bits
- **Offset:** (0x0080*i)+0x1168
- **Exists:** DWC_EQOS_SYS>1&&DWC_EQOS_FRP_EN

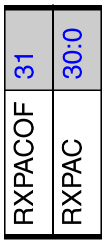


Table 17-315 Fields for Register: DMA_CH(#i)_RXP_Accept_Cnt (for i = 0; i <= DWC_EQOS_NUM_DMA_RX_CH-1)

Bits	Name	Memory Access	Description
31	RXPACOF	R	Rx Parser Accept Counter Overflow Bit When set, this bit indicates that the RXPAC Counter field crossed the maximum limit. Access restriction applies. Clears on read. Self-set to 1 on internal event. **Values:** ■ 0x0 (INACTIVE): Rx Parser Accept Counter overflow not occurred ■ 0x1 (ACTIVE): Rx Parser Accept Counter overflow occurred **Value After Reset:** 0x0 **Exists:** Always
30:0	RXPAC	R	Rx Parser Accept Counter This 31-bit counter is implemented whenever a Rx Parser Accept a packet due to AF =1. The counter is cleared when the register is read. **Value After Reset:** 0x0 **Exists:** Always

17.6.24 DMA_CH(#i)_RX_ERI_Cnt (for i = 0; i <= DWC_EQOS_NUM_DMA_RX_CH-1)

- **Description:** The DMA_CH(#i)_RX_ERI_Cnt registers provides the count of the number of times ERI was asserted.
- **Size:** 32 bits
- **Offset:** (0x0080*i)+0x1168
- **Exists:** DWC_EQOS_SYS>1

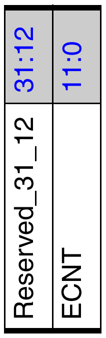


Table 17-316 Fields for Register: DMA_CH(#i)_RX_ERI_Cnt (for i = 0; i <= DWC_EQOS_NUM_DMA_RX_CH-1)

Bits	Name	Memory Access	Description
31:12	Reserved_31_12	R	**Value After Reset:** 0x0 **Exists:** Always
11:0	ECNT	R	ERI Counter When ERIC bit of DMA_CH(#i)_RX_Control register is set, this counter increments for burst transfer completed by the Rx DMA from the start of packet transfer. This counter will get reset at the start of new packet. **Value After Reset:** 0x0 **Exists:** Always

18

Internal Parameter Descriptions

Provides a description of the internal parameters that might be indirectly referenced in expressions in the Signals, Parameters, or Registers chapters. These parameters are not visible in the coreConsultant GUI and most of them are derived automatically from visible parameters. **You must not set any of these parameters directly.**

Some expressions might refer to TCL functions or procedures (sometimes identified as **function_of**) that coreConsultant uses to make calculations. The exact formula used by these TCL functions is not provided in this chapter. However, when you configure the core in coreConsultant, all TCL functions and parameters are evaluated completely; and the resulting values are displayed where appropriate in the coreConsultant GUI reports.

Table 18-1 Internal Parameters

Parameter Name	Equals To
7	
DWC_EQOS_ADDRWIDTH_64	{DWC_EQOS_ADDRWIDTH ! = 32}
DWC_EQOS_ADDRWIDTH_MNS32	{DWC_EQOS_ADDRWIDTH ==48 ? 16 : 8}
DWC_EQOS_AHB_SLAVE	{(DWC_EQOS_CSR_PORT ==2)}
DWC_EQOS_AHB_SUBSYS	{DWC_EQOS_SYS ==3}
DWC_EQOS_APB3_SLAVE	{(DWC_EQOS_CSR_PORT ==4 || DWC_EQOS_CSR_PORT ==6)}
DWC_EQOS_APB_SLAVE	{(DWC_EQOS_CSR_PORT ==1)}
DWC_EQOS_ARI_PBL_WIDTH	{ (DWC_EQOS_SYS ==1) ? (DWC_EQOS_RX_FIFO_PTR_WIDTH-1) : 8}
DWC_EQOS_ASLV_CLK	0
DWC_EQOS_ASP	{DWC_EQOS_ASP_ALL || DWC_EQOS_ASP_ECC}
DWC_EQOS_ASP_ACT	{DWC_EQOS_ASP_ALL}
DWC_EQOS_ASP_DPP	{DWC_EQOS_ASP_ALL}

Parameter Name	Equals To
DWC_EQOS_ASP_FSM	{DWC_EQOS_ASP_ALL}
DWC_EQOS_ATI_CTL_WIDTH	{(DWC_EQOS_DATAWIDTH ==32 && DWC_EQOS_SA_VLAN_INS_CTRL_EN && (DWC_EQOS_OST_EN || DWC_EQOS_TBS)) ? 2 : (DWC_EQOS_DATAWIDTH ==32 && (DWC_EQOS_OST_EN || DWC_EQOS_TBS)) ? 2 : (DWC_EQOS_DATAWIDTH ==32 && DWC_EQOS_SA_VLAN_INS_CTRL_EN) ? 1 : (DWC_EQOS_DATAWIDTH ==64 && (DWC_EQOS_OST_EN || DWC_EQOS_TBS)) ? 1 : 1}
DWC_EQOS_ATI_PBL_WIDTH	{(DWC_EQOS_SYS ==1) ? (DWC_EQOS_TX_FIFO_PTR_WIDTH-1) : 8 }
DWC_EQOS_AV_SLOT_EN	{DWC_EQOS_TX_AV_EN && DWC_EQOS_SYS>1}
DWC_EQOS_AXI4_LITE_SLAVE	{(DWC_EQOS_CSR_PORT ==5)}
DWC_EQOS_AXI_GM_ADDR_WIDTH	DWC_EQOS_ADDRWIDTH
DWC_EQOS_AXI_GM_DATA_WIDTH	DWC_EQOS_DATAWIDTH
DWC_EQOS_AXI_SLAVE	{(DWC_EQOS_CSR_PORT ==3 || DWC_EQOS_CSR_PORT ==5)}
DWC_EQOS_AXI_SUBSYS	{DWC_EQOS_SYS ==4 || DWC_EQOS_SYS ==5}
DWC_EQOS_BEW	{(DWC_EQOS_DATAWIDTH ==32) ? 2 : (DWC_EQOS_DATAWIDTH ==64) ? 3 : (DWC_EQOS_DATAWIDTH ==128) ? 4 : 5}
DWC_EQOS_BLW	{[(log(DWC_EQOS_AXI_BL)/log(2))]}
DWC_EQOS_BSA	{[(log(DWC_EQOS_DATAWIDTH/8)/log(2))]}
DWC_EQOS_COARSE_FINE_CORRECTION	2
DWC_EQOS_CORE	{DWC_EQOS_SYS == 0}
DWC_EQOS_DBG_BUS	0
DWC_EQOS_DMA_SUBSYS	{DWC_EQOS_SYS == 2}
DWC_EQOS_EMAW	{[(log(DWC_EQOS_EST_DEP)/log(2))]}
DWC_EQOS_EST_TISW	{DWC_EQOS_AV_EST ? 3 : 0}
DWC_EQOS_FHE	0
DWC_EQOS_FLCW	{(DWC_EQOS_RXFIFO_SIZE<=4096) ? 3 : (DWC_EQOS_RXFIFO_SIZE<=8192) ? 4 : (DWC_EQOS_RXFIFO_SIZE<=16384) ? 5 : 6}
DWC_EQOS_FRP_IND_ADDDW	{[(log(DWC_EQOS_FRP_ENTRIES*4)/log(2))]}

Parameter Name	Equals To
DWC_EQOS_FRP_OK_INDEXW	{[(log(DWC_EQOS_FRP_ENTRIES)/log(2))]}
DWC_EQOS_GMII_INTF_ONLY	(! DWC_EQOS_RTBI_EN && ! DWC_EQOS_RMII_EN && ! DWC_EQOS_RGMII_EN && ! DWC_EQOS_TBI_EN && ! DWC_EQOS_SGMII_EN && ! DWC_EQOS_SMII_EN && ! DWC_EQOS_REVMII_EN) && DWC_EQOS_SINGLE_PHY_INTF
DWC_EQOS_L256_RXMS	{DWC_EQOS_RXFIFO_SIZE <= 512 ? 1 : DWC_EQOS_RXFIFO_SIZE <= 1024 ? 2 : DWC_EQOS_RXFIFO_SIZE <= 2048 ? 3 : DWC_EQOS_RXFIFO_SIZE <= 4096 ? 4 : DWC_EQOS_RXFIFO_SIZE <= 8192 ? 5 : DWC_EQOS_RXFIFO_SIZE <= 16384 ? 6 : DWC_EQOS_RXFIFO_SIZE <= 32768 ? 7 : DWC_EQOS_RXFIFO_SIZE <= 65536 ? 8 : DWC_EQOS_RXFIFO_SIZE <= 131072 ? 9 : 10}
DWC_EQOS_L256_TXMS	{DWC_EQOS_TXFIFO_SIZE <= 512 ? 1 : DWC_EQOS_TXFIFO_SIZE <= 1024 ? 2 : DWC_EQOS_TXFIFO_SIZE <= 2048 ? 3 : DWC_EQOS_TXFIFO_SIZE <= 4096 ? 4 : DWC_EQOS_TXFIFO_SIZE <= 8192 ? 5 : DWC_EQOS_TXFIFO_SIZE <= 16384 ? 6 : DWC_EQOS_TXFIFO_SIZE <= 32768 ? 7 : DWC_EQOS_TXFIFO_SIZE <= 65536 ? 8 : DWC_EQOS_TXFIFO_SIZE <= 131072 ? 9 : 10}
DWC_EQOS_L256_TXMS_MAX	{DWC_EQOS_NUM_TXQ>1 ? 0 : (DWC_EQOS_TXFIFO_SIZE == 256 ? 0 : DWC_EQOS_TXFIFO_SIZE <= 512 ? 1 : DWC_EQOS_TXFIFO_SIZE <= 1024 ? 3 : DWC_EQOS_TXFIFO_SIZE <= 2048 ? 7 : DWC_EQOS_TXFIFO_SIZE <= 4096 ? 15 : DWC_EQOS_TXFIFO_SIZE <= 8192 ? 31 : DWC_EQOS_TXFIFO_SIZE <= 16384 ? 63 : DWC_EQOS_TXFIFO_SIZE <= 32768 ? 127 : DWC_EQOS_TXFIFO_SIZE <= 65536 ? 255 : DWC_EQOS_TXFIFO_SIZE <= 131072 ? 511 : 1023)}
DWC_EQOS_LP_MODE_EN	DWC_EQOS_PMT_EN
DWC_EQOS_M1000_ONLY	{DWC_EQOS_OP_MODE == 2}
DWC_EQOS_M110_ONLY	{DWC_EQOS_OP_MODE == 1}
DWC_EQOS_MEMW	{(DWC_EQOS_DATAWIDTH ==32) ? 35 : (DWC_EQOS_DATAWIDTH ==64) ? 68 : (DWC_EQOS_DATAWIDTH ==128) ? 133 : 262}
DWC_EQOS_MMC_FPE_EN	DWC_EQOS_MMC_EN && DWC_EQOS_FPE

Parameter Name	Equals To
DWC_EQOS_MMC_IPC_RX_DGRAM_BCNT_ATLEAST_ONE	{DWC_EQOS_MMC_RX_IPV4_GD_OCTET_CNT_EN || DWC_EQOS_MMC_RX_IPV4_HDRERR_OCTET_CNT_EN || DWC_EQOS_MMC_RX_IPV4_NOPAY_OCTET_CNT_EN || DWC_EQOS_MMC_RX_IPV4_FRAG_OCTET_CNT_EN || DWC_EQOS_MMC_RX_IPV6_GD_OCTET_CNT_EN || DWC_EQOS_MMC_RX_IPV6_HDRERR_OCTET_CNT_EN || DWC_EQOS_MMC_RX_IPV6_NOPAY_OCTET_CNT_EN}
DWC_EQOS_MMC_IPC_RX_EN_ATLEAST_ONE	{DWC_EQOS_MMC_IPC_RX_PKTS_ATLEAST_ONE || DWC_EQOS_MMC_IPC_RX_DGRAM_BCNT_ATLEAST_ONE || DWC_EQOS_MMC_IPC_RX_PAYLD_BCNT_ATLEAST_ONE}
DWC_EQOS_MMC_IPC_RX_PAYLD_BCNT_ATLEAST_ONE	{DWC_EQOS_MMC_RX_UDP_GD_OCTET_CNT_EN || DWC_EQOS_MMC_RX_IPV4_UDSBL_OCTET_CNT_EN || DWC_EQOS_MMC_RX_UDP_ERR_OCTET_CNT_EN || DWC_EQOS_MMC_RX_TCP_GD_OCTET_CNT_EN || DWC_EQOS_MMC_RX_TCP_ERR_OCTET_CNT_EN || DWC_EQOS_MMC_RX_ICMP_GD_OCTET_CNT_EN || DWC_EQOS_MMC_RX_ICMP_ERR_OCTET_CNT_EN}
DWC_EQOS_MMC_IPC_RX_PKTS_ATLEAST_ONE	{DWC_EQOS_MMC_RX_IPV4_GD_PKTS_CNT_EN || DWC_EQOS_MMC_RX_IPV4_HDRERR_PKTS_CNT_EN || DWC_EQOS_MMC_RX_IPV4_NOPAY_PKTS_CNT_EN || DWC_EQOS_MMC_RX_IPV4_FRAG_PKTS_CNT_EN || DWC_EQOS_MMC_RX_IPV4_UDSBL_PKTS_CNT_EN || DWC_EQOS_MMC_RX_IPV6_GD_PKTS_CNT_EN || DWC_EQOS_MMC_RX_IPV6_HDRERR_PKTS_CNT_EN || DWC_EQOS_MMC_RX_IPV6_NOPAY_PKTS_CNT_EN || DWC_EQOS_MMC_RX_UDP_GD_PKTS_CNT_EN || DWC_EQOS_MMC_RX_UDP_ERR_PKTS_CNT_EN || DWC_EQOS_MMC_RX_TCP_GD_PKTS_CNT_EN || DWC_EQOS_MMC_RX_TCP_ERR_PKTS_CNT_EN || DWC_EQOS_MMC_RX_ICMP_GD_PKTS_CNT_EN || DWC_EQOS_MMC_RX_ICMP_ERR_PKTS_CNT_EN}
DWC_EQOS_MMC_RX_BCNT_ATLEAST_ONE	{DWC_EQOS_MMC_RXOCTG_CNT_EN || DWC_EQOS_MMC_RXOCTGB_CNT_EN}
DWC_EQOS_MMC_RX_EN_ATLEAST_ONE	{DWC_EQOS_MMC_RX_PKTS_ATLEAST_ONE || DWC_EQOS_MMC_RX_BCNT_ATLEAST_ONE}
DWC_EQOS_MMC_RX_LPI_ATLEAST_ONE	{DWC_EQOS_MMC_RX_LPI_TRANSITION_CNT_EN || DWC_EQOS_MMC_RX_LPI_MICROSEC_TIMER_EN}

Parameter Name	Equals To
DWC_EQOS_MMC_RX_PKTS_ATLEAST_ONE	{DWC_EQOS_MMC_RXPKTGB_CNT_EN || DWC_EQOS_MMC_RXBCASTG_CNT_EN || DWC_EQOS_MMC_RXMCASTG_CNT_EN || DWC_EQOS_MMC_RXCRCERR_CNT_EN || DWC_EQOS_MMC_RXALGNERR_CNT_EN || DWC_EQOS_MMC_RXRUNTERR_CNT_EN || DWC_EQOS_MMC_RXJABERR_CNT_EN || DWC_EQOS_MMC_RXUNDERSZG_CNT_EN || DWC_EQOS_MMC_RXOVERSZG_CNT_EN || DWC_EQOS_MMC_RXADDR_RNG_CNT_EN || DWC_EQOS_MMC_RXUCASTG_CNT_EN || DWC_EQOS_MMC_RXLENERR_CNT_EN || DWC_EQOS_MMC_RXOUTOFRNG_TYP_CNT_EN || DWC_EQOS_MMC_RXPAUSEPKT_CNT_EN || DWC_EQOS_MMC_RXFIFOOVFL_CNT_EN || DWC_EQOS_MMC_RXVLANPKTGB_CNT_EN || DWC_EQOS_MMC_RXWDGERR_CNT_EN || DWC_EQOS_MMC_RXRCVERR_CNT_EN || DWC_EQOS_MMC_RXCTRLPKT_CNT_EN}
DWC_EQOS_MMC_TX_BCNT_ATLEAST_ONE	{DWC_EQOS_MMC_TXOCTGB_CNT_EN || DWC_EQOS_MMC_TXOCTG_CNT_EN}
DWC_EQOS_MMC_TX_EN_ATLEAST_ONE	{DWC_EQOS_MMC_TX_PKTS_ATLEAST_ONE || DWC_EQOS_MMC_TX_BCNT_ATLEAST_ONE}
DWC_EQOS_MMC_TX_LPI_ATLEAST_ONE	{DWC_EQOS_MMC_TX_LPI_TRANSITION_CNT_EN || DWC_EQOS_MMC_TX_LPI_MICROSEC_TIMER_EN}
DWC_EQOS_MMC_TX_PKTS_ATLEAST_ONE	{DWC_EQOS_MMC_TXADDR_RNG_CNT_EN || DWC_EQOS_MMC_TXPKTGB_CNT_EN || DWC_EQOS_MMC_TXBCASTG_CNT_EN || DWC_EQOS_MMC_TXMCASTG_CNT_EN || DWC_EQOS_MMC_TXUCASTGB_CNT_EN || DWC_EQOS_MMC_TXMCASTGB_CNT_EN || DWC_EQOS_MMC_TXBCASTGB_CNT_EN || DWC_EQOS_MMC_TXUNDRFLW_CNT_EN || DWC_EQOS_MMC_TXSNGLCOLG_CNT_EN || DWC_EQOS_MMC_TXMULTCOLG_CNT_EN || DWC_EQOS_MMC_TXDEFRD_CNT_EN || DWC_EQOS_MMC_TXLATECOL_CNT_EN || DWC_EQOS_MMC_TXEXSCOL_CNT_EN || DWC_EQOS_MMC_TXCARR_CNT_EN || DWC_EQOS_MMC_TXPKTG_CNT_EN || DWC_EQOS_MMC_TXEXSDEF_CNT_EN || DWC_EQOS_MMC_TXPAUSE_CNT_EN || DWC_EQOS_MMC_TXVLAN_CNT_EN || DWC_EQOS_MMC_TXOVERSZG_CNT_EN}
DWC_EQOS_MTL_SUBSYS	{DWC_EQOS_SYS == 1}

Parameter Name	Equals To
DWC_EQOS_NUM_DMA_CSR_CH	{[(DWC_EQOS_NUM_DMA_TX_CH > DWC_EQOS_NUM_DMA_RX_CH ? DWC_EQOS_NUM_DMA_TX_CH : DWC_EQOS_NUM_DMA_RX_CH)]}
DWC_EQOS_NUM_DMA_RX_CHW	{DWC_EQOS_NUM_DMA_RX_CH<=2 ? 1 : DWC_EQOS_NUM_DMA_RX_CH <=4 ? 2 : 3}
DWC_EQOS_NUM_DMA_TSO_CH_POW2	{[function_of: (DWC_EQOS_NUM_DMA_TSO_CH > 8 ? 16 : DWC_EQOS_NUM_DMA_TSO_CH > 4 ? 8 : DWC_EQOS_NUM_DMA_TSO_CH > 2 ? 4 : DWC_EQOS_NUM_DMA_TSO_CH > 1 ? 2 : 1)] }
DWC_EQOS_PCS_EN	DWC_EQOS_TBI_EN || DWC_EQOS_SGMII_EN || DWC_EQOS_RTBI_EN
DWC_EQOS_PPS_OUT_1_EN	{DWC_EQOS_PPS_OUT_NUM>=2}
DWC_EQOS_PPS_OUT_2_EN	{DWC_EQOS_PPS_OUT_NUM>=3}
DWC_EQOS_PPS_OUT_3_EN	{DWC_EQOS_PPS_OUT_NUM ==4}
DWC_EQOS_REVMII_INTF_ONLY	(DWC_EQOS_REVMII_EN && DWC_EQOS_SINGLE_PHY_INTF)
DWC_EQOS_RGMII_INTF_ONLY	(DWC_EQOS_RGMII_EN && DWC_EQOS_SINGLE_PHY_INTF)
DWC_EQOS_RGMII_OR_RTBI	DWC_EQOS_RGMII_EN || DWC_EQOS_RTBI_EN
DWC_EQOS_RMII_INTF_ONLY	(DWC_EQOS_RMII_EN && DWC_EQOS_SINGLE_PHY_INTF)
DWC_EQOS_ROSRLW	{[(log(DWC_EQOS_AXI_GM_MAX_RD_REQUESTS)/log(2))]}
DWC_EQOS_RTBI_INTF_ONLY	(DWC_EQOS_RTBI_EN && DWC_EQOS_SINGLE_PHY_INTF)
DWC_EQOS_RXD_IOW	{[function]}
DWC_EQOS_RXFIFOSZ	{DWC_EQOS_SYS ! =0 ? [(log(DWC_EQOS_RXFIFO_SIZE)/log(2)) - 7] : 0}
DWC_EQOS_RX_Q4_EN	{DWC_EQOS_NUM_RXQ>=5}
DWC_EQOS_RXQW	{(DWC_EQOS_NUM_RXQ<=2) ? 1 : (DWC_EQOS_NUM_RXQ<=4) ? 2 : 3}
DWC_EQOS_SGMII_INTF_ONLY	(DWC_EQOS_SGMII_EN && DWC_EQOS_SINGLE_PHY_INTF)
DWC_EQOS_SLVERR	{(DWC_EQOS_CSR_PORT>1)}

Parameter Name	Equals To
DWC_EQOS_SMII_INTF_ONLY	(DWC_EQOS_SMII_EN && DWC_EQOS_SINGLE_PHY_INTF)
DWC_EQOS_STRW	{[(pow(2,DWC_EQOS_BSA))]}
DWC_EQOS_TBI_INTF_ONLY	(DWC_EQOS_TBI_EN && DWC_EQOS_SINGLE_PHY_INTF)
DWC_EQOS_TSO_MEM_ADDR_WIDTH	{[((log(DWC_EQOS_TSO_MEM_SIZE*DWC_EQOS_NUM_DMA_TSO_CH_POW2)/log(2)) - (DWC_EQOS_BEW))]}
DWC_EQOS_TX_AV_EN	DWC_EQOS_AV_ENABLE && DWC_EQOS_NUM_TXQ>1 && (DWC_EQOS_TX_Q1_AV_EN || DWC_EQOS_TX_Q2_AV_EN || DWC_EQOS_TX_Q3_AV_EN || DWC_EQOS_TX_Q4_AV_EN || DWC_EQOS_TX_Q5_AV_EN || DWC_EQOS_TX_Q6_AV_EN || DWC_EQOS_TX_Q7_AV_EN)
DWC_EQOS_TX_AV_STRTQ	{DWC_EQOS_TX_Q1_AV_EN ==1 ? 1 : DWC_EQOS_TX_Q2_AV_EN ==1 ? 2 : DWC_EQOS_TX_Q3_AV_EN ==1 ? 3 : DWC_EQOS_TX_Q4_AV_EN ==1 ? 4 : DWC_EQOS_TX_Q5_AV_EN ==1 ? 5 : DWC_EQOS_TX_Q6_AV_EN ==1 ? 6 : DWC_EQOS_TX_Q7_AV_EN ==1 ? 7 : 0}
DWC_EQOS_TXD_IOW	{[function]}
DWC_EQOS_TXFCBW	{[function]}
DWC_EQOS_TXFIFOSZ	{DWC_EQOS_SYS ! =0 ? [(log(DWC_EQOS_TXFIFO_SIZE)/log(2)) - 7] : 0}
DWC_EQOS_TXQW	{(DWC_EQOS_NUM_TXQ<=2) ? 1 : (DWC_EQOS_NUM_TXQ<=4) ? 2 : 3}
DWC_EQOS_UFO_EN	{DWC_EQOS_TSO_EN && DWC_EQOS_TSO_MEM_EN && DWC_EQOS_TSO_MEM_SIZE>=128 && SNPS_RSVDPARAM_0}
DWC_EQOS_WOSRLW	{[(log(DWC_EQOS_AXI_GM_MAX_WR_REQUESTS)/log(2))]}
H	1'b1
RXFIFOSIZE	DWC_EQOS_RX_FIFO_PTR_WIDTH + ((DWC_EQOS_DATAWIDTH/64) + 2)
SNPS_RSVDPARAM_0	{[function_of: -project - -version [function_of] DWC-ETHERNET-QOS-V5_00]}

19 Descriptors

This chapter describes the DWC_ether_qos descriptors.

19.1 Overview

The DMA in the Ethernet subsystem transfers data based on a linked list of descriptors. The application creates the descriptors in the system memory. The DWC_ether_qos supports the following two types of descriptors:

- **Normal Descriptor**: Normal descriptors are used for packet data and to provide control information applicable to the packets to be transmitted.
- **Context Descriptor**: Context descriptors are used to provide control information applicable to the packet to be transmitted.

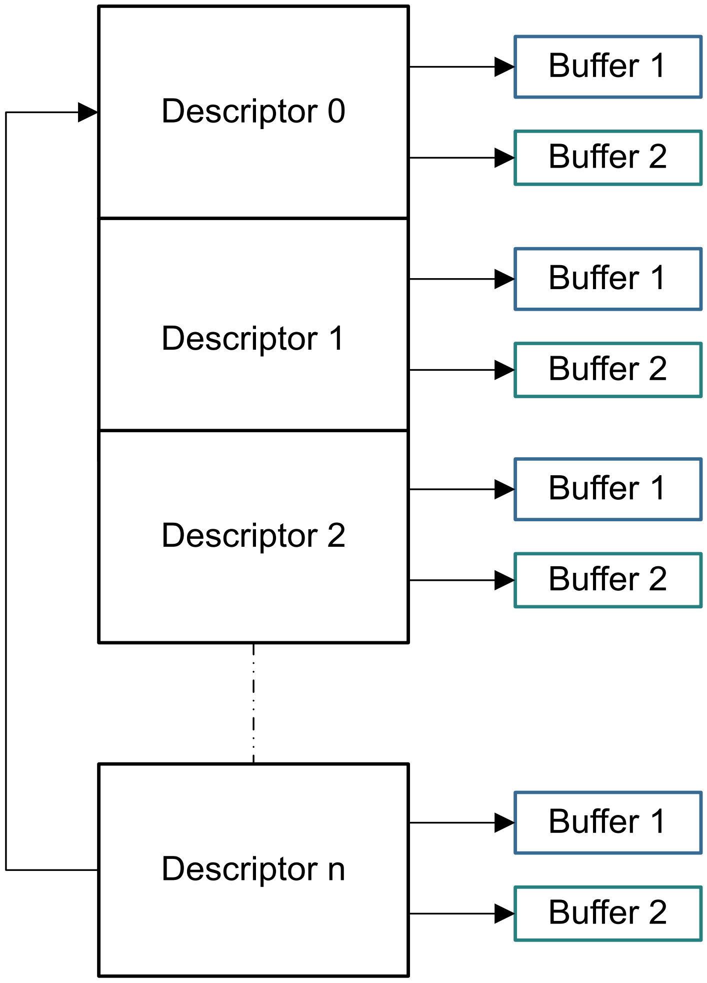
Each normal descriptor contains two buffers and two address pointers. These buffers enable the adapter port to be compatible with various types of memory management schemes.

Note There is no limit for the number of descriptors that can be used for a single packet.

19.2 Descriptor Structure

The DWC_ether_qos supports the ring structure for DMA descriptor as shown in Figure 19-1.

Figure 19-1 Descriptor Ring Structure



In Ring structure, descriptors are separated by the Word, DWord, or LWord number programmed in the DSL field of the DMA_CH#_Control register. The application needs to program the total ring length, that is, the total number of descriptors in ring span in the following registers of a DMA channel:

- Transmit Descriptor Ring Length Register (DMA_CH#_TxDesc_Ring_Length)
- Receive Descriptor Ring Length Register (DMA_CH#_RxDesc_Ring_Length)

The Descriptor Tail Pointer Register contains the pointer to the descriptor address (N). The base address and the current descriptor pointer decide the address of the current descriptor that the DMA can process. The descriptors up to one location less than the one indicated by the descriptor tail pointer (N – 1) are owned by the DMA. The DMA continues to process the descriptors until the following condition occurs:

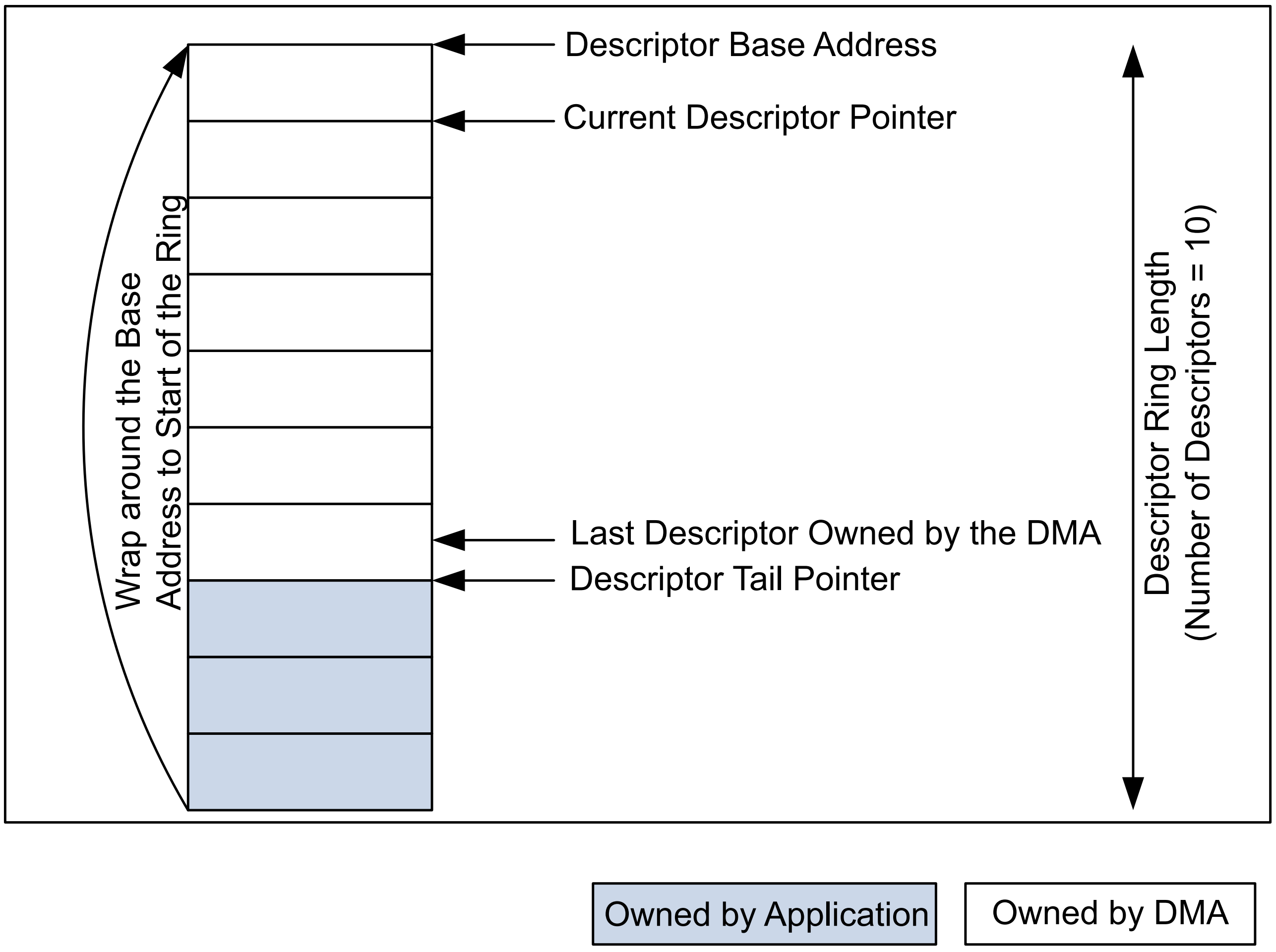
```
Current Descriptor Pointer == Descriptor Tail Pointer;
```

The DMA goes into the Suspend mode when this condition occurs. The application must perform a write to the Descriptor Tail pointer register and update the tail pointer so that the following condition is true:

```
Current Descriptor Pointer < Descriptor Tail Pointer;
```

The DMA automatically wraps around the base address when the end of ring is reached, as shown in Figure 19-2.

Figure 19-2 DMA Descriptor Ring


Descriptor Base Address
Current Descriptor Pointer
Wrap around the Base Address to Start of the Ring
Last Descriptor Owned by the DMA
Descriptor Tail Pointer
Descriptor Ring Length (Number of Descriptors = 10)
Owned by Application
Owned by DMA

For descriptors owned by the application, the OWN bit of DES3 is reset to 0. For descriptors owned by the DMA, the OWN bit is set to 1. If the application has only one descriptor in the beginning, the application sets the last descriptor address (tail pointer) to Descriptor Base Address + 1. The DMA processes the first descriptor and then waits for the application to advance the tail pointer.

19.3 Descriptor Structure for Split Header Support

DWC_ether_qos supports splitting the incoming receive packet header such that the header and the payload are stored in different buffers in the system memory. For more details about splitting header on receive packets, see "Header-Payload Split" on page 347.

Figure 19-3 shows the descriptor structure without the Split Header feature.

Figure 19-3 Descriptors without Split Header Feature

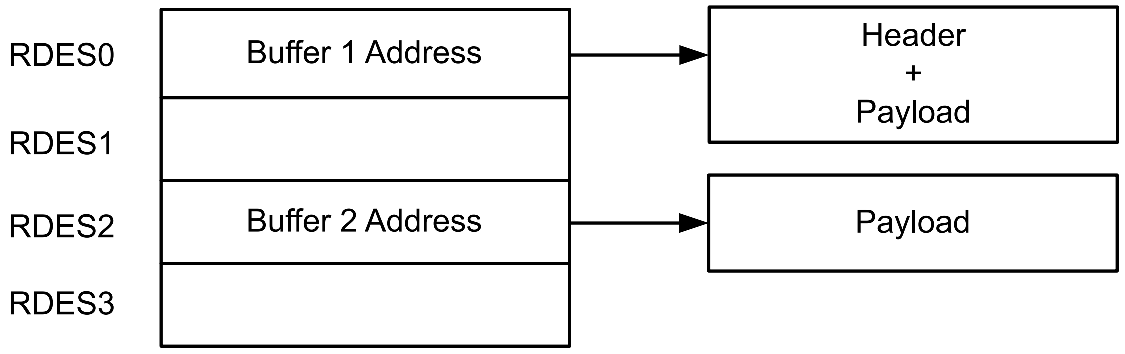
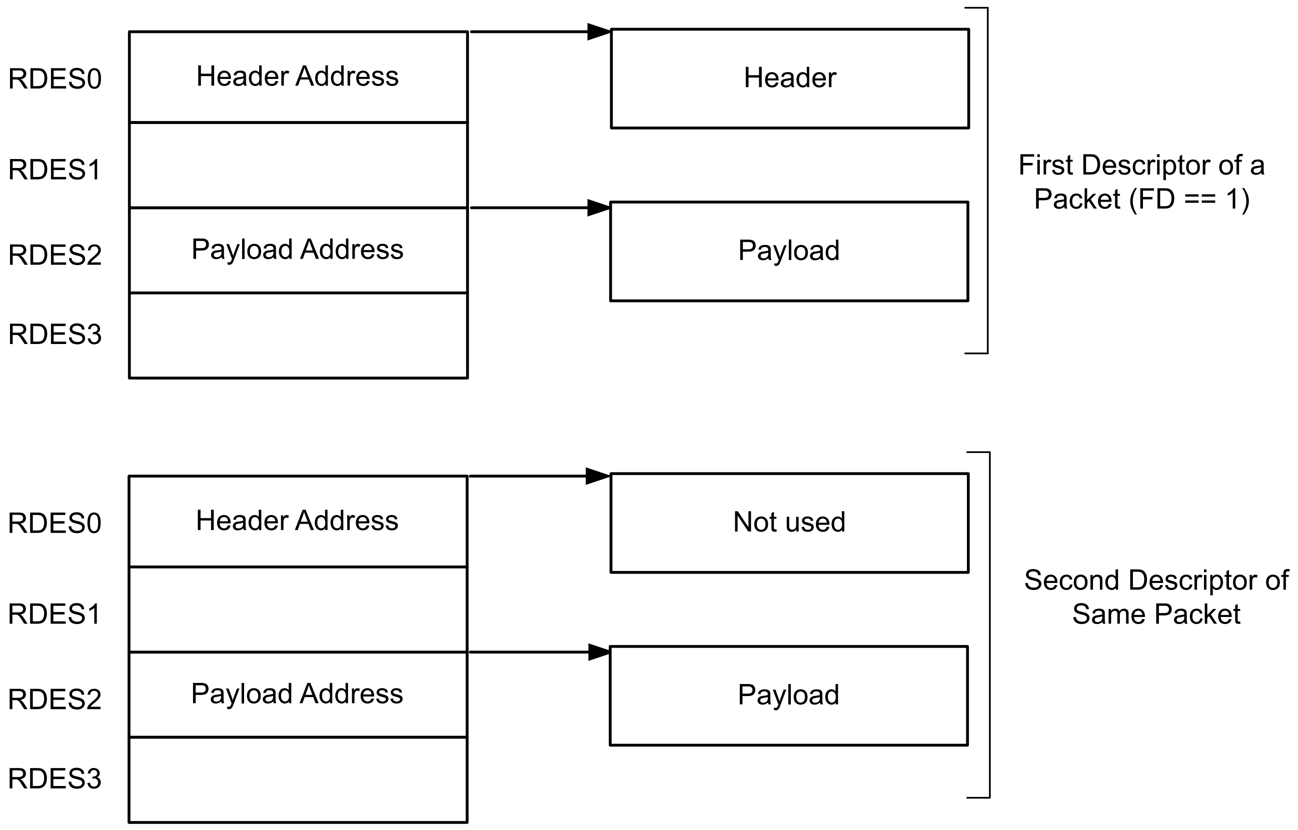


Figure 19-4 shows the descriptor structure with the Split Header feature.

Figure 19-4 Descriptors with Split Header Feature



The DMA writes the header of the received packet by using the header address to which the RDES0 in the first descriptor is pointing (FD bit of RDES3 is set). The DMA writes the payload of the received packet into the buffer address to which the RDES2 is pointing. For subsequent descriptors (FD is set to 0), the address to which RDES0 (Header address) is pointing is not used. The payload is written only to buffers to which the RDES2 (payload address) is pointing.

The DMA writes the header length in RDES2 of the first receive descriptor (RDES3[29] (FD Bit) is set) for the packet. The packet length is written in RDES3 of the last receive descriptor (RDES3[28] (LD Bit) set). The buffer length for the payload is set by the driver through the RBSZ field in the corresponding DMA Channel Register 2 (Receive Control Register). The DMA fills receive buffers fully in all except the last descriptor. The header length is taken to be the value based on the bits programmed in the MAC Extended Configuration Register HDSMS field (Bits [22:20]).

19.4 Descriptor Endianness

The descriptor addresses must be aligned to configured bus width (Word, DWord, or LWord for 32-bit, 64-bit, or 128-bit bus respectively). The data bus can be configured for either little-endian or big-endian format. Table 19-1 provides information about Figures 19-5 through Figure 19-9 that show the structure of a normal descriptor with respect to the data bus endianness.

Table 19-1 Descriptor Endianness

Data Bus Endianness	Descriptor Endianness	Figure Number
Little-endian	Same-endian	■ Figure 19-5 (for 32-bit data bus) ■ Figure 19-6 (for 64-bit data bus) ■ Figure 19-8 (for 128-bit data bus)
Big-endian	Reverse-endian	■ Figure 19-5 (for 32-bit data bus) ■ Figure 19-7 (for 64-bit data bus) ■ Figure 19-9 (for 128-bit data bus)

Figure 19-5 shows the normal Receive and Transmit descriptors for 32-bit data bus when the endian mode of data bus and descriptors is little-endian.

Figure 19-5 Rx/Tx Descriptors in Same-Endian Mode for 32-Bit Little-Endian or Reverse Endian Mode for 32-Bit Big-Endian Data Bus

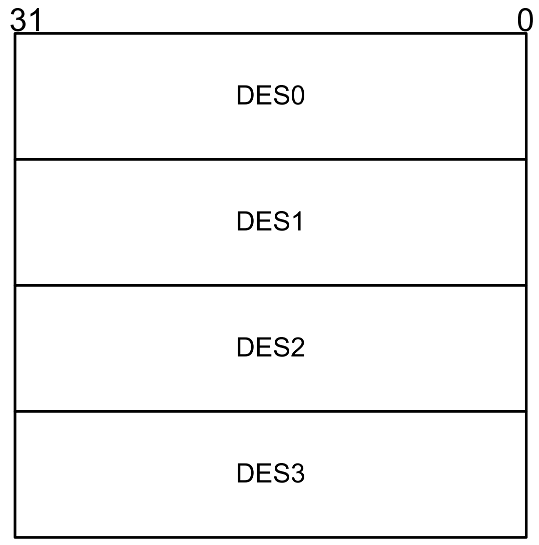


Figure 19-6 shows the normal Receive and Transmit descriptors for 64-bit data bus when the endian mode of data bus is little-endian or big-endian and the endian mode of descriptors is reverse-endian.

Figure 19-6 Rx/Tx Descriptors in Same-Endian Mode for 64-Bit, Little-Endian Data Bus

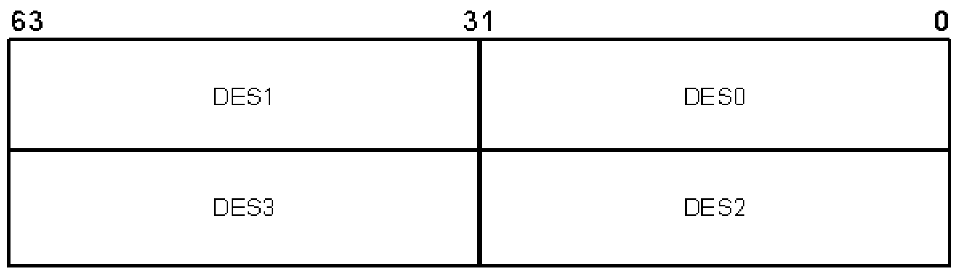


Figure 19-6 and Figure 19-7 show the normal Receive and Transmit descriptors for 64-bit data bus when the endian mode of data bus is big-endian and the endian mode of descriptors is reverse-endian. In 64-bit big-endian data bus systems, when the descriptors are configured for reverse endianness, the order of the descriptor subset is reversed from the default (Figure 19-6). For example, Bits[63:32] are DES0 while Bits[31:0] are DES1.

Figure 19-7 Rx/Tx Descriptors in Reverse-Endian Mode for 64-Bit, Big-Endian Data Bus

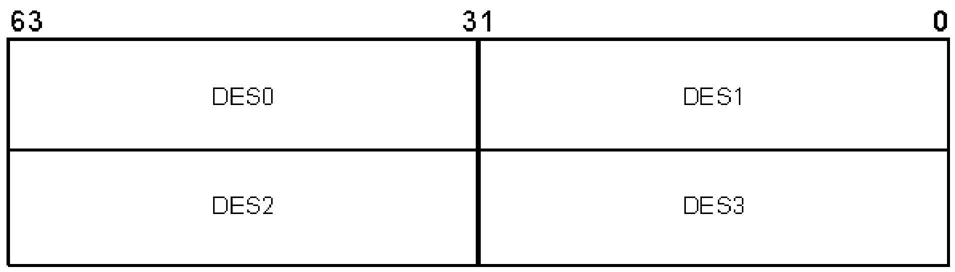


Figure 19-8 shows the normal Receive and Transmit descriptors for 128-bit data bus when the endian mode of data bus and descriptors is little-endian.

Figure 19-8 Rx/Tx Descriptors in Same-Endian Mode for 128-Bit, Little-Endian Format Data Bus

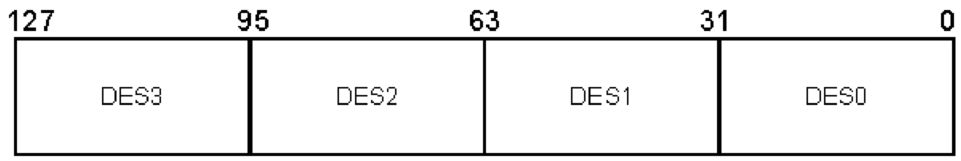
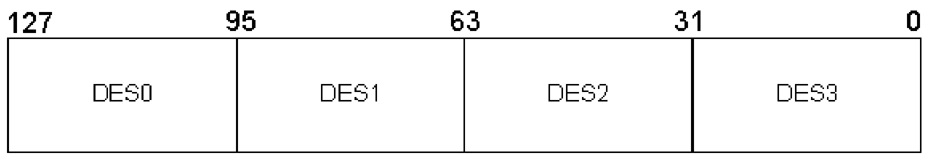


Figure 19-9 shows the normal Receive and Transmit descriptors for 128-bit data bus when the endian mode of data bus is big-endian and the endian mode of descriptors is reverse-endian. In 128-bit big-endian data bus systems, when the descriptors are configured for reverse-endian mode, the order of the descriptor subset is reversed from the default (Figure 19-6). For example, Bits[127:64] are DES0 while Bits[63:32] are DES1.

Figure 19-9 Rx/Tx Descriptors in Reverse-Endian Mode for 128-Bit, Big-Endian Format Data Bus



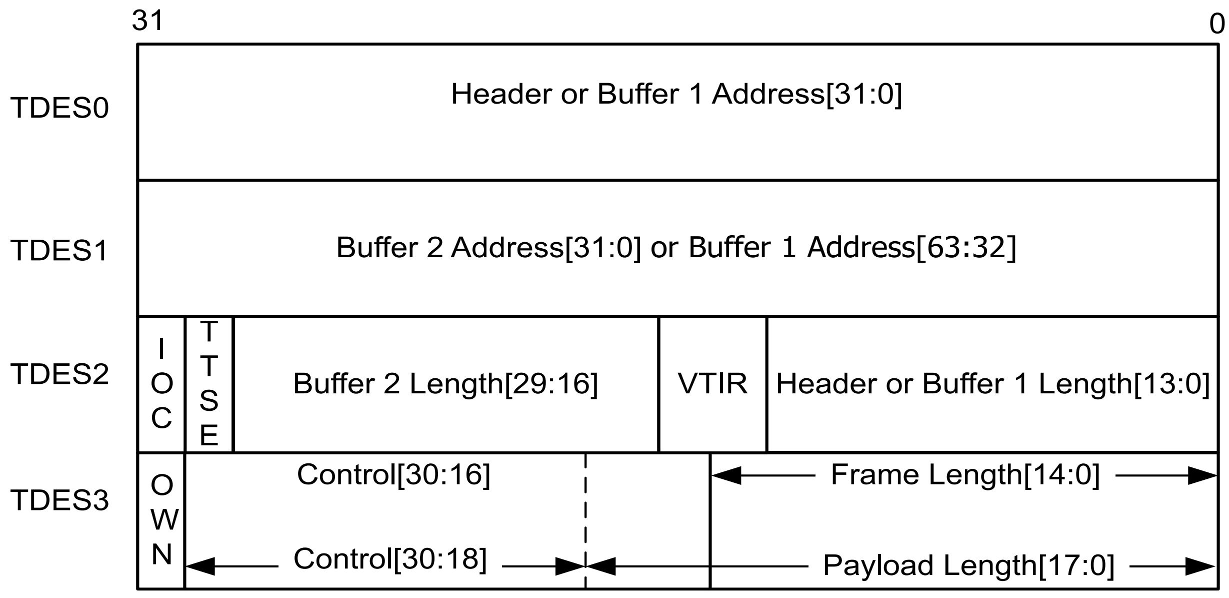
19.5 Transmit Descriptor

The DMA in DWC_ether_qos requires at least one descriptor for a transmit packet. In addition to two buffers, two byte-count buffers, and two address pointers, the transmit descriptor has control fields which can be used to control the MAC operation on per-transmit packet basis. The Transmit Normal descriptor has two formats: Read format and Write-Back format

19.5.1 Transmit Normal Descriptor (Read Format)

Figure 19-10 shows the Read Format for a Transmit normal descriptor. Table 19-2 through Table 19-7 describe the read format for the Transmit Normal Descriptors: TDES0, TDES1, TDES2, and TDES3.

Figure 19-10 Transmit Descriptor Read Format



19.5.1.1 TDES0 Normal Descriptor (Read Format)

Table 19-2 TDES0 Normal Descriptor (Read Format)

Bit	Name	Description
31:0	BUF1AP	Buffer 1 Address Pointer or TSO Header Address Pointer These bits indicate the physical address of Buffer 1. These bits indicate the TSO Header Address pointer when the following bits are set: ■ TSE bit of TDES3 ■ FD bit of TDES3

19.5.1.2 TDES1 Normal Descriptor (Read Format)

Table 19-3 TDES1 Normal Descriptor (Read Format)

Bit	Name	Description
31:0	BUF2AP	Buffer 2 or Buffer 1 Address Pointer This bit indicates the physical address of Buffer 2 when a descriptor ring structure is used. There is no limitation for the buffer address alignment. In 40- or 48-bit addressing mode, these bits indicate the most-significant 8- or 16- bits of the Buffer 1 Address Pointer.

19.5.1.3 TDES2 Normal Descriptor (Read Format)

Table 19-4 TDES2 Normal Descriptor (Read Format)

31	30	29-16	15:14	13:0
IOC	TTSE/ TMWD	B2L	VTIR	HL or B1L

Table 19-5 TDES2 Normal Descriptor (Read Format)

Bits	Name	Description
31	IOC	Interrupt on Completion This bit controls the setting of TI and ETI status bits in the DMA_CH#_Status register. When ETIC = 1 and TDES2[LD] = 0, this bit sets the ETI bit. When TDES3[LD] = 1, this bit sets the TI status bit.
30	TTSE/TMWD	Transmit Timestamp Enable or External TSO Memory Write Enable This bit enables the IEEE1588 time stamping for Transmit packet referenced by the descriptor, if TSE bit is not set. If TSE bit is set and external TSO memory is enabled, setting this bit disables external TSO memory writing for this packet.
29:16	B2L	Buffer 2 Length The driver sets this field. When set, this field indicates Buffer 2 length.
15:14	VTIR	VLAN Tag Insertion or Replacement These bits request the MAC to perform VLAN tagging or untagging before transmitting the packets. The application must set the CRC Pad Control bits appropriately when VLAN Tag Insertion, Replacement, or Deletion is enabled for the packet. The following list describes the values of these bits: ■ 2'b00: Do not add a VLAN tag. ■ 2'b01: Remove the VLAN tag from the packets before transmission. This option should be used only with the VLAN packets. ■ 2'b10: Insert a VLAN tag with the tag value programmed in the MAC_VLAN_Incl register or context descriptor. ■ 2'b11: Replace the VLAN tag in packets with the tag value programmed in the MAC_VLAN_Incl register or context descriptor. This option should be used only with the VLAN packets. These bits are valid when the Enable SA and VLAN Insertion on Tx option is selected while configuring the core.

Bits	Name	Description
13:0	HL or B1L	Header Length or Buffer 1 Length For Header length only bits [9:0] are taken. The size 13:0 is applicable only when interpreting buffer 1 length. If the TCP Segmentation Offload feature is enabled through the TSE bit of TDES3, this field is equal to the header length. When the TSE bit is set in TDES3, the header length includes the length in bytes from Ethernet Source address till the end of the TCP header. The maximum header length supported for TSO feature is 1023 bytes. The maximum header length supported for TSO feature is 1023 bytes. If the TCP Segmentation Offload feature is not enabled, this field is equal to Buffer 1 length.

19.5.1.4 TDES3 Normal Descriptor (Read Format)

Table 19-6 TDES2 Normal Descriptor (Read Format)

31	30	29	28	27:26	25:23	22:19	18	17:16	15	14:0
OWN	CTXT	FD	LD	CPC	SAIC	SLOTNUM or THL	TSE	CIC/TPL	TPL	FL/TPL

Table 19-7 TDES3 Normal Descriptor (Read Format)

Bits	Name	Description
31	OWN	Own Bit When this bit is set, it indicates that the DMA owns the descriptor. When this bit is reset, it indicates that the application owns the descriptor. The DMA clears this bit after it completes the transfer of data given in the associated buffer(s).
30	CTXT	Context Type This bit should be set to 1'b0 for normal descriptor.
29	FD	First Descriptor When this bit is set, it indicates that the buffer contains the first segment of a packet.
28	LD	Last Descriptor When this bit is set, it indicates that the buffer contains the last segment of the packet. When this bit is set, the B1L or B2L field should have a non-zero value.

Bits	Name	Description
27:26	CPC	CRC Pad Control This field controls the CRC and Pad Insertion for Tx packet. This field is valid only when the first descriptor bit (TDES3[29]) is set. The following list describes the values of Bits[27:26]: ■ 2'b00: CRC and Pad Insertion The MAC appends the cyclic redundancy check (CRC) at the end of the transmitted packet of length greater than or equal to 60 bytes. The MAC automatically appends padding and CRC to a packet with length less than 60 bytes. ■ 2'b01: CRC Insertion (Disable Pad Insertion) The MAC appends the CRC at the end of the transmitted packet but it does not append padding. The application should ensure that the padding bytes are present in the packet being transferred from the Transmit Buffer, that is, the packet being transferred from the Transmit Buffer is of length greater than or equal to 60 bytes. ■ 2'b10: Disable CRC Insertion The MAC does not append the CRC at the end of the transmitted packet. The application should ensure that the padding and CRC bytes are present in the packet being transferred from the Transmit Buffer. ■ 2'b11: CRC Replacement The MAC replaces the last four bytes of the transmitted packet with recalculated CRC bytes. The application should ensure that the padding and CRC bytes are present in the packet being transferred from the Transmit Buffer. This field is valid only for the first descriptor. Note: When the TSE bit is set, the MAC ignores this field because the CRC and pad insertion is always done for segmentation.
25:23	SAIC	SA Insertion Control These bits request the MAC to add or replace the Source Address field in the Ethernet packet with the value given in the MAC Address 0 register. The application must set the CRC Pad Control bits appropriately when SA Insertion Control is enabled for the packet. Bit 25 specifies the MAC Address Register (1 or 0) value that is used for Source Address insertion or replacement. The following list describes the values of Bits[24:23]: ■ 2'b00: Do not include the source address ■ 2'b01: Include or insert the source address. For reliable transmission, the application must provide frames without source addresses. ■ 2'b10: Replace the source address. For reliable transmission, the application must provide frames with source addresses. ■ 2'b11: Reserved These bits are valid in the EQOS-DMA, EQOS-AXI, and EQOS-AHB configurations when the Enable SA and VLAN Insertion on Tx option is selected while configuring the core and when the First Segment control bit (TDES3 [29]) is set. This field is valid only for the first descriptor.

Bits	Name	Description
22:19	SLOTNUM or THL	SLOTNUM: Slot Number Control Bits in AV Mode These bits indicate the slot interval in which the data should be fetched from the corresponding buffers addressed by TDES0 or TDES1. When the Transmit descriptor is fetched, the DMA compares the slot number value in this field with the slot interval maintained in the RSN field DMA_CH#_Slot_Function_Control_Status. It fetches the data from the buffers only if a value matches. These bits are valid only for the AV channels. THL: TCP/UDP Header Length If the TSE bit is set, this field contains the length of the TCP/UDP header. The minimum value of this field must be 5 for TCP header. The value must be equal to 2 for UDP header. This field is valid only for the first descriptor.
18	TSE	TCP Segmentation Enable When this bit is set, the DMA performs the TCP/UDP segmentation or UDP fragmentation for a packet depending on the TSE_MODE[1:0] bit of the DMA_CH(#i)_Tx_Control Register. This bit is valid only if the FD bit is set.
17:16	CIC/TPL	Checksum Insertion Control or TCP Payload Length These bits control the checksum calculation and insertion. The following list describes the bit encoding: ■ 2'b00: Checksum Insertion Disabled. ■ 2'b01: Only IP header checksum calculation and insertion are enabled. ■ 2'b10: IP header checksum and payload checksum calculation and insertion are enabled, but pseudo-header checksum is not calculated in hardware. ■ 2'b11: IP Header checksum and payload checksum calculation and insertion are enabled, and pseudo-header checksum is calculated in hardware. This field is valid when the Enable Transmit TCP/IP Checksum Offload option is selected and the TSE bit is reset. When the TSE bit is set, this field contains the upper bits [17:16] of the TCP Payload (or IP Payload for UDP fragmentation). This allows the TCP/UDP packet length field to be spanned across TDES3[17:0] to provide 256 KB packet length support. This field is valid only for the first descriptor.
15	TPL	Reserved or TCP Payload Length When the TSE bit is reset, this bit is reserved. When the TSE bit is set, this is Bit 15 of the TCP payload length [17:0]. This field is valid only when the Enable TCP Segmentation Offloading for TCP/IP Packets option is selected while configuring the core.

Bits	Name	Description
14:0	FL/TPL	Frame Length or TCP Payload Length This field is equal to the length of the packet to be transmitted in bytes. When the TSE bit is not set, this field is equal to the total length of the packet to be transmitted: `Ethernet Header Length + TCP /IP Header Length - Preamble Length - SFD Length + Ethernet Payload Length` When the TSE bit is set, this field is equal to the lower 15 bits of the TCP payload length in case of segmentation and IP payload in case of UDP fragmentation. In case of segmentation, this length does not include Ethernet header or TCP/UDP/IP header length. In case of fragmentation, this length does not include Ethernet header and IP header. When DWRR/WFQ algorithm is NOT enabled, value written into this field is not used when TSE = 0.

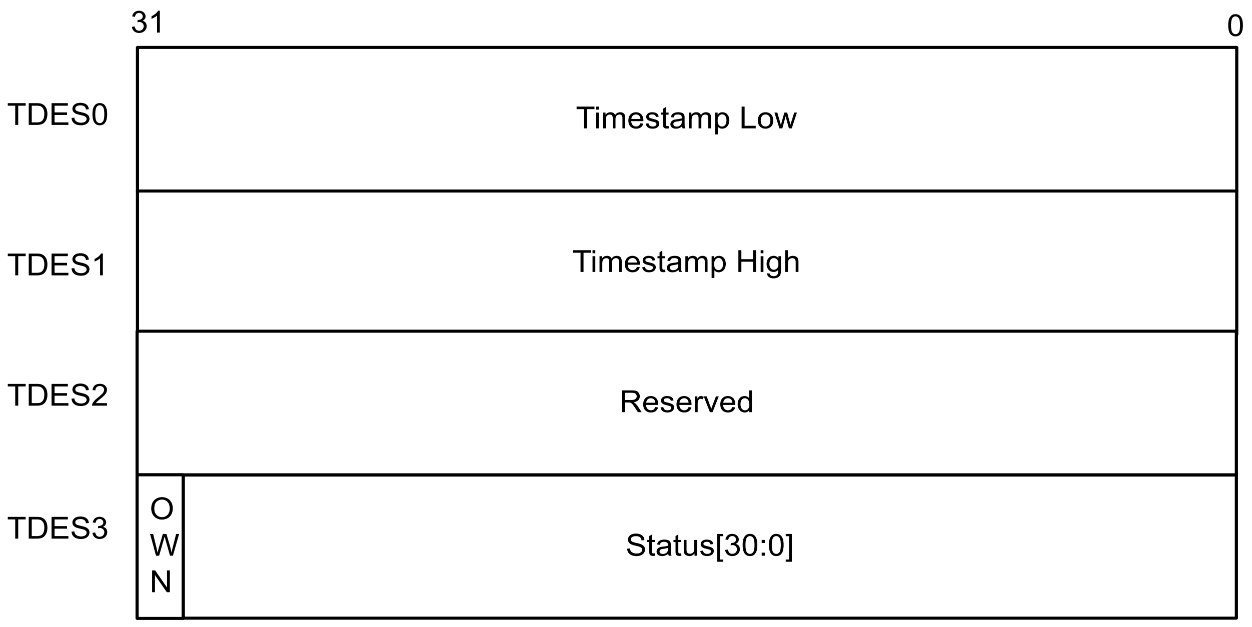
19.5.1.5 Transmit Normal Descriptor (Write-Back Format)

The write-back format of the Transmit Descriptor includes timestamp low, timestamp high, OWN, and Status bits.

The write-back format is applicable only for the last descriptor of the corresponding packet. The LD bit (TDES3[28]) is set in the descriptor where the DMA writes back the status and timestamp information for the corresponding Transmit packet.

Figure 19-11 illustrates the write-back format of the Transmit Descriptor.

Figure 19-11 Transmit Descriptor Write-Back Format



19.5.1.6 TDES0 Normal Descriptor (Write-Back Format)

As described in Table 19-8, this format is only applicable to the last descriptor of a packet.

Table 19-8 TDES0 Normal Descriptor (Write-Back Format)TDES0 Normal Descriptor (Write-Back Format)

Bit	Name	Description
31:0	TTSL	Transmit Packet Timestamp Low The DMA updates this field with least significant 32 bits of the timestamp captured for the corresponding Transmit packet. The DMA writes the timestamp only if TTSE bit of TDES2 is set in the first descriptor of the packet. This field has the timestamp only if the Last Segment bit (LS) in the descriptor is set and the Timestamp status (TTSS) bit is set.

19.5.1.7 TDES1 Normal Descriptor (Write-Back Format)

As mentioned in Table 19-9, this format is only applicable to the last descriptor of a packet.

Table 19-9 TDES1 Normal Descriptor (Write-Back Format)

Bit	Name	Description
31:0	TTSH	Transmit Packet Timestamp High The DMA updates this field with the most significant 32 bits of the timestamp captured for corresponding transmit packet. The DMA writes the timestamp only if the TTSE bit of TDES2 is set in the first descriptor of the packet. This field has the timestamp only if the Last Segment bit (LS) in the descriptor is set and Timestamp status (TTSS) bit is set.

19.5.1.8 TDES2 Normal Descriptor (Write-Back Format)

This format is applicable only to the last descriptor of a packet.

Table 19-10 TDES2 Normal Descriptor (Write-Back Format)

Bit	Description
31:0	Reserved

19.5.1.9 TDES3 Normal Descriptor (Write-Back Format)

This format is applicable only to the last descriptor of a packet.

Table 19-11 TDES3 Normal Descriptor (Write-Back Format)

31	30	29	28	27:18	17	16	15	14	13	12	11	10	9	8	7:4	3	2	1	0
OWN	CTXT	FD	LD	Rsvd	TTSS	EUE	ES	JT	FF	PCE	LoC	NC	LC	EC	CC	ED	UF	DB	IHE

Table 19-12 TDES3 Normal Descriptor (Write-Back Format)

Bit	Name	Description
31	OWN	Own Bit When this bit is set, it indicates that the DWC_ether_qos DMA owns the descriptor. The DMA clears this bit when it completes the packet transmission. After the write-back is complete, this bit is set to 'b0.
30	CTXT	Context Type This bit should be set to 'b0 for Normal descriptor.
29	FD	First Descriptor This bit indicates that the buffer contains the first segment of a packet.
28	LD	Last Descriptor This bit is set 'b1 for last descriptor of a packet. The DMA writes the status fields only in the last descriptor of the packet.
27:24	Rsvd	Reserved
23	DE	Descriptor Error When this bit is set, it indicates that the descriptor content is incorrect. The DMA sets this bit during write-back while closing the descriptor. Descriptor Errors can be: ■ Incorrect sequence from the context descriptor. For example, a location after the first descriptor for a packet. ■ All 1s ■ CTXT, LD, and FD bits set to 1. **Note 1:** When Descriptor Error occurs due to All 1s or CTXT, LD, and FD bits set to 1, the Transmit DMA closes the transmit descriptor with DE and LD bits set to 1. When IOC bit in TDES2 of corresponding first descriptor is set to 1, Transmit DMA will set the TI bit in the DMA_CH#_Status register **Note 2:** Based on CTXT, LD, and FD bits of the transmit descriptor, the subsequent descriptor might be considered as the First Descriptor (even if FD bit is not set) and partial packet is sent.
22:18	Rsvd	Reserved
17	TTSS	Tx Timestamp Status This status bit indicates that a timestamp has been captured for the corresponding transmit packet. When this bit is set, TDES0 and TDES1 have timestamp values that were captured for the Transmit packet. This field is valid only when the Last Segment control bit (TDES3 [28]) in a descriptor is set. This bit is valid only when IEEE1588 timestamping feature is enabled; otherwise, it is reserved.
16	EUE	ECC Uncorrectable Error Status Indicates the ECC uncorrectable error in the TSO memory. **Note:** Uncorrectable error in Transmit FIFO memory is reported with (Bit 13) FF = 1. This is because, all such packets are flushed by DWC_ether_qos.

Bit	Name	Description
15	ES	Error Summary This bit indicates the logical OR of the following bits: ■ TDES3[0]: IP Header Error ■ TDES3[14]: Jabber Timeout ■ TDES3[13]: Packet Flush ■ TDES3[12]: Payload Checksum Error ■ TDES3[11]: Loss of Carrier ■ TDES3[10]: No Carrier ■ TDES3[9]: Late Collision ■ TDES3[8]: Excessive Collision ■ TDES3[3]: Excessive Deferral ■ TDES3[2]: Underflow Error This bit is also set when EUE (bit 16) is set.
14	JT	Jabber Timeout This bit indicates that the MAC transmitter has experienced a jabber time-out. This bit is set only when the JD bit of the MAC_Configuration register is not set.
13	FF	Packet Flushed This bit indicates that the DMA or MTL flushed the packet because of a software flush command given by the CPU.
12	PCE	Payload Checksum Error This bit indicates that the Checksum Offload engine had a failure and did not insert any checksum into the encapsulated TCP, UDP, or ICMP payload. This failure can be either because of insufficient bytes, as indicated by the Payload Length field of the IP Header or the MTL starting to forward the packet to the MAC transmitter in Store-and-Forward mode without the checksum having been calculated yet. This second error condition only occurs when the Transmit FIFO depth is less than the length of the Ethernet packet being transmitted to avoid deadlock, the MTL starts forwarding the packet when the FIFO is full, even in the store-and-forward mode. This error can also occur when Bus Error is detected during packet transfer. When the Full Checksum Offload engine is not enabled, this bit is reserved.
11	LoC	Loss of Carrier This bit indicates that Loss of Carrier occurred during packet transmission (that is, the gmii_crs_i signal was inactive for one or more transmit clock periods during packet transmission). This is valid only for the packets transmitted without collision and when the MAC operates in the half-duplex mode.
10	NC	No Carrier This bit indicates that the carrier sense signal form the PHY was not asserted during transmission.
9	LC	Late Collision This bit indicates that packet transmission was aborted because a collision occurred after the collision window (64 byte times including Preamble in MII mode and 512 byte times including Preamble and Carrier Extension in GMII mode). This bit is not valid if Underflow Error is set.

Bit	Name	Description
8	EC	Excessive Collision This bit indicates that the transmission was aborted after 16 successive collisions while attempting to transmit the current packet. If the DR bit is set in the MAC_Configuration register, this bit is set after first collision and the transmission of the packet is aborted.
7:4	CC	Collision Count This 4-bit counter value indicates the number of collisions occurred before the packet was transmitted. The count is not valid when the EC bit is set.
3	ED	Excessive Deferral This bit indicates that the transmission ended because of excessive deferral of over 24,288 bit times (155,680 bits times in 1000 Mbps mode or Jumbo Packet enabled mode) if DC bit is set in the MAC_Configuration register. When TBS is enabled in full duplex mode and this bit is set, it indicates that the frame has been dropped after the expiry time has reached.
2	UF	Underflow Error This bit indicates that the MAC aborted the packet because the data arrived late from the system memory. The underflow error can occur because of either of the following conditions: ■ The DMA encountered an empty Transmit Buffer while transmitting the packet ■ The application filled the MTL Tx FIFO slower than the MAC transmit rate The transmission process enters the suspended state and sets the underflow bit corresponding to a queue in the MTL_Interrupt_Status register.
1	DB	Deferred Bit This bit indicates that the MAC deferred before transmitting because of presence of carrier. This bit is valid only in the half-duplex mode.
0	IHE	IP Header Error When IP Header Error is set, this bit indicates that the Checksum Offload engine detected an IP header error. This bit is valid only when Tx Checksum Offload is enabled. Otherwise, it is reserved. If COE detects an IP header error, it still inserts an IPv4 header checksum if the Ethernet Type field indicates an IPv4 payload. In full duplex mode, when EST/Qbv is enabled and this bit is set, it indicates the frame drop status due to Frame Size error or Schedule Error.

19.5.2 Transmit Context Descriptor

The context descriptor is used to provide the timestamps for one-step timestamp correction, VLAN Tag ID for VLAN insertion feature.

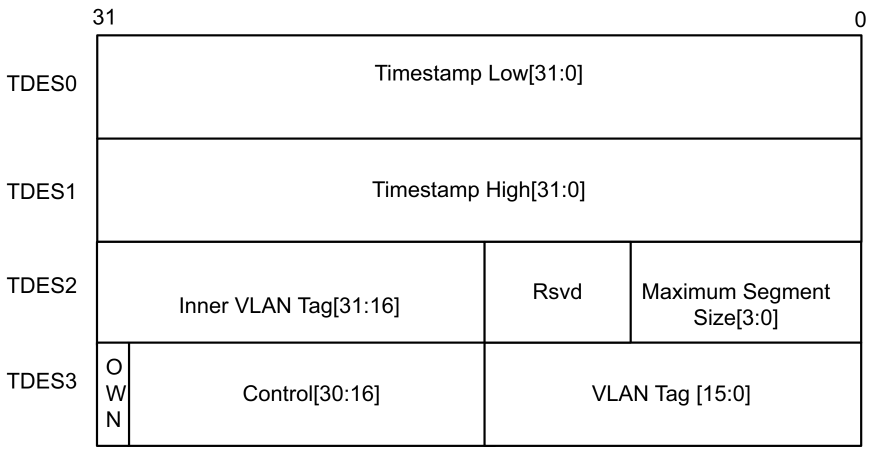
The Transmit Context descriptor can be provided any time before a packet descriptor. The context is valid for the current packet and subsequent packets. The context descriptor is used to provide the timestamps for one-step timestamp correction and VLAN Tag ID for VLAN insertion feature. Write back is done on a context descriptor only to reset the OWN bit.

Note The VLAN Tag IDs and MSS values, provided by the application in a context descriptor with their corresponding Valid bits set, are stored internally by the DMA. When the outer or inner VLAN tag is provided with the Valid bit set, the DMA always passes the last valid VLAN tag to the MTL. The application cannot invalidate the valid VLAN tag stored by the DMA. The VLAN tag is inserted or replaced based on the control inputs provided for the packet.

The Inner VLAN Tag Control input is used only for the next packet that immediately follows the context descriptor. The application must provide a context descriptor before the normal descriptor of each packet for which the DMA should use the inner VLAN Tag control input.

Figure 19-12 shows the format of the Transmit Context descriptor.

Figure 19-12 Transmit Context Descriptor Format



19.5.2.1 TDES0 Context Descriptor

Table 19-13 describes the format of the TDES0 Context descriptor.

Table 19-13 TDES0 Context Descriptor

Bit	Name	Description
31:0	TTSL	Transmit Packet Timestamp Low For one-step correction, the driver can provide the lower 32 bits of timestamp in this descriptor word. The DMA uses this value as the low word for doing one-step timestamp correction. This field is valid only if the OSTC and TCMSSV bits of TDES3 context descriptor are set.

19.5.2.2 TDES1 Context Descriptor

Table 19-14 describes the format of the TDES1 Context descriptor.

Table 19-14 TDES1 Context Descriptor

Bit	Name	Description
31:0	TTSH	Transmit Packet Timestamp High For one-step correction, the driver can provide the upper 32 bits of timestamp in this descriptor. The DMA uses this value as the high word for doing one-step timestamp correction. This field is valid only if the OSTC and TCMSSV bits of TDES3 context descriptor are set.

19.5.2.3 TDES2 Context Descriptor

Table 19-15 explains the format of the TDES2 Context descriptor.

Table 19-15 TDES2 Context Descriptor

Bit	Name	Description
31:16	IVT	Inner VLAN Tag When the IVLTV bit of TDES3 context descriptor is set and the TCMSSV and OSTC bits of TDES3 context descriptor are reset, TDES2[31:16] contains the inner VLAN Tag to be inserted in the subsequent Transmit packets.
15:14	Rsvd	Reserved
13:0	MSS	Maximum Segment Size When the Enable TCP Segmentation Offloading for TCP/IP Packets option is selected, the driver can provide maximum segment size in this field. This segment size is used while segmenting the TCP/IP payload. This field is valid only if the TCMSSV bit of TDES3 context descriptor is set and the OSTC bit of the TDES3 context descriptor is reset.

19.5.2.4 TDES3 Context Descriptor

Table 19-16 TDES3 Context Descriptor

31	30	29:28	27	26	25:24	23	22:18	17	16	15:0
OWN	CTXT	Rsvd	OSTC	TCMSSV	Rsvd	CDE	Rsvd	IVLTV	VLTV	VT

Table 19-17 details the format of the TDES3 Context descriptor.

Table 19-17 TDES3 Context Descriptor

Bit	Name	Description
31	OWN	Own Bit When this bit is set, it indicates that the DWC_ether_qos DMA owns the descriptor. When this bit is reset, it indicates that the application owns the descriptor. The DMA clears this bit immediately after the read.

Bit	Name	Description
30	CTXT	Context Type This bit should be set to 1'b1 for Context descriptor.
29:28	Rsvd	Reserved
27	OSTC	One-Step Timestamp Correction Enable When this bit is set, the DMA performs a one-step timestamp correction with reference to the timestamp values provided in TDES0 and TDES1.
26	TCMSSV	One-Step Timestamp Correction Input or MSS Valid When this bit and the OSTC bit are set, it indicates that the Timestamp Correction input provided in TDES0 and TDES1 is valid. When the OSTC bit is reset and this bit and the TSE bit of TDES3 are set in subsequent normal descriptor, it indicates that the MSS input in TDES2 is valid.
25:24	Rsvd	Reserved
23	DE	Descriptor Error When this bit is set, it indicates that the descriptor content is incorrect.The DMA sets this bit during write-back while closing the context descriptor. Descriptor Errors can be: ■ Incorrect sequence from the context descriptor. For example, a location before the first descriptor for a packet. ■ All 1s. ■ CD, LD, and FD bits set to 1. **Note 1:** When Descriptor Error occurs due to All 1s or CTXT, LD, and FD bits set to 1, the Transmit DMA closes the transmit descriptor with DE and LD bits set to 1. When IOC bit in TDES2 of corresponding first descriptor is set to 1, Transmit DMA will set the TI bit in the DMA_CH#_Status register **Note 2:** Based on CTXT, LD, and FD bits of the transmit descriptor, the subsequent descriptor might be considered as the First Descriptor (even if FD bit is not set) and partial packet is sent.
22:20	Rsvd	Reserved
19:18	IVTIR	Inner VLAN Tag Insert or Replace When this bit is set, these bits request the MAC to perform Inner VLAN tagging or un-tagging before transmitting the packets. If the packet is modified for VLAN tags, the MAC automatically recalculates and replaces the CRC bytes. The following list describes the values of these bits: ■ 2'b00: Do not add the inner VLAN tag. ■ 2'b01: Remove the inner VLAN tag from the packets before transmission. This option should be used only with the VLAN frames. ■ 2'b10: Insert an inner VLAN tag with the tag value programmed in the MAC_Inner_VLAN_Incl register or context descriptor. ■ 2'b11: Replace the inner VLAN tag in packets with the tag value programmed in the MAC_Inner_VLAN_Incl register or context descriptor. This option should be used only with the VLAN frames. These bits are valid when the **Enable SA and VLAN Insertion on Tx** and **Enable Double VLAN Processing** options are selected.

Bit	Name	Description
17	IVLTV	Inner VLAN Tag Valid When this bit is set, it indicates that the IVT field of TDES2 is valid.
16	VLTV	VLAN Tag Valid When this bit is set, it indicates that the VT field of TDES3 is valid.
15:0	VT	VLAN Tag This field contains the VLAN Tag to be inserted or replaced in the packet. This field is used as VLAN Tag only when the VLTI bit of the MAC_VLAN_Incl register is reset.

19.6 Receive Descriptor

The DMA in DWC_ether_qos attempts to read a descriptor only if the Tail Pointer is different from the Base Pointer or current pointer. It is recommended to have a descriptor ring with a length that can accommodate at least two complete packets received by the MAC. Otherwise, the performance of the DMA is impacted greatly because of the unavailability of the descriptors. In such situations, the RxFIFO in MTL becomes full and starts dropping packets.

The following Receive Descriptors are present:

- Normal descriptors
- Context descriptors

All RX descriptors are prepared by the software and given to the DMA as “Normal” Descriptors with the content as shown in Receive Normal Descriptor (Read Format). The DMA reads this descriptor and after transferring a received packet (or part of) to the buffers indicated by the descriptor, the Rx DMA will close the descriptor with the corresponding packet status. The format of this status is given in the “Receive Normal Descriptor (Write-Back Format)”.

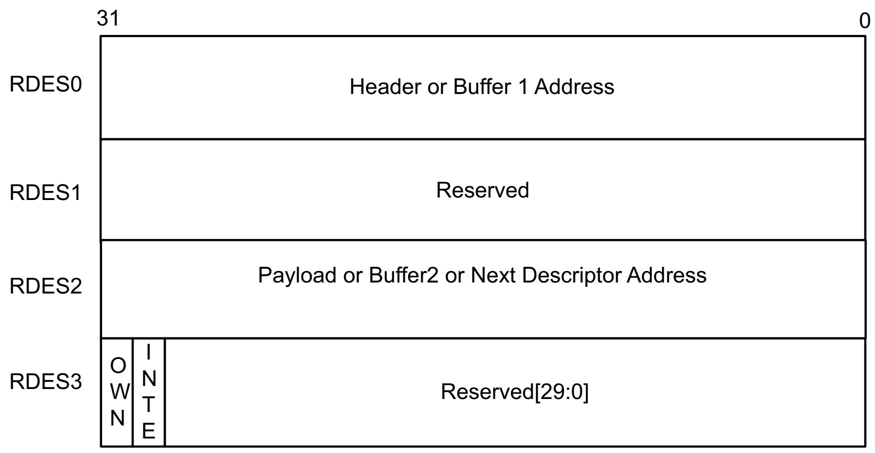
For some packets, the normal descriptor bits are not enough to write the complete status. For such packets, the RX DMA writes the extended status to the next descriptor (without processing or using the Buffers/ Pointers embedded in that descriptor). The format and content of the descriptor write back is described in “Receive Context Descriptor”.

19.6.1 Receive Normal Descriptor (Read Format)

The read format for a Receive Normal descriptor is made up of a header or Buffer 1 address, reserved field, payload or Buffer 2 or Next Descriptor address, a 30-bit reserved filed, OWN bit, and an interrupt bit.

Figure 19-13 shows the Read Format for a Receive Normal Descriptor.

Figure 19-13 Receive Normal Descriptor Read Format



31
0
RDES0
Header or Buffer 1 Address
RDES1
Reserved
RDES2
Payload or Buffer2 or Next Descriptor Address
RDES3
OWN
INTE
Reserved[29:0]

Note In the Receive Descriptor (Read Format), if the Buffer Address field is all 0s, DWC_ether_qos does not transfer data to that buffer and skips to the next buffer or next descriptor.

19.6.1.1 RDES0 Normal Descriptor (Read Format)

Table 19-18 explains the read format of the RDES0 Normal Descriptor.

Table 19-18 RDES0 Normal Descriptor (Read Format)

Bit	Name	Description
31:0	BUF1AP	Header or Buffer 1 Address Pointer When the SPH bit of Control register of a channel is reset, these bits indicate the physical address of Buffer 1. When the SPH bit is set, these bits indicate the physical address of Header Buffer where the Rx DMA writes the L2/L3/L4 header bytes of the received packet. The application can program a byte-aligned address for this buffer which means that the LS bits of this field can be non-zero. However, while transferring the start of packet, the DMA performs a Write operation with RDES0[1:0] (or RDES0[2:0]/[3:0] in case of 64-/128-bit configuration) as zero. However, the packet data is shifted as per actual offset as given by buffer address pointer. If the address pointer points to a buffer where the middle or last part of the packet is stored, the DMA ignores the offset address and writes to the full location as indicated by the data-width.

19.6.1.2 RDES1 Normal Descriptor (Read Format)

Table 19-19 describes the read format of the RDES1 Normal Descriptor.

Table 19-19 RDES1 Normal Descriptor (Read Format)

Bit	Name	Description
31:0	Reserved or BUF1AP	In 64-bit addressing mode, this field contains the most-significant 32 bits of the Buffer 1 Address Pointer. Otherwise, this field is reserved.

19.6.1.3 RDES2 Normal Descriptor (Read Format)

Table 19-20 provides the read format of the RDES2 Normal Descriptor.

Table 19-20 RDES2 Normal Descriptor (Read Format)

Bit	Name	Description
31:0	BUF2AP	Buffer 2 Address Pointer These bits indicate the physical address of Buffer 2. When the SPH bit of the DMA_CH#_Control register is set, the buffer address pointer must be bus width-aligned, that is, RDES2[3:0, 2:0, or 1:0] = 0 corresponding to 128, 64, or 32 bus width. LSBs are ignored internally. When the SPH bit of the DMA_CH#_Control register is reset, there is no limitations on the RDES2 value. However, the RxDMA uses the LS Bits of the pointer address only while transferring the start bytes of a packet. If the BUF2AP is giving the address of a buffer in which the middle or last part of a packet is stored, the DMA ignores BUF2AP[3:0 or 2:0 or 1:0] (corresponding to 128- or 64- or 32-bit data-bus) and writes to the complete location.

19.6.1.4 RDES3 Normal Descriptor (Read Format)

Table 19-21 describes the read format of the RDES3 Normal Descriptor.

31	30	19-26	25	24	23:0
OWN	IOC	Rsvd	BUF2V	BUF1V	Rsvd

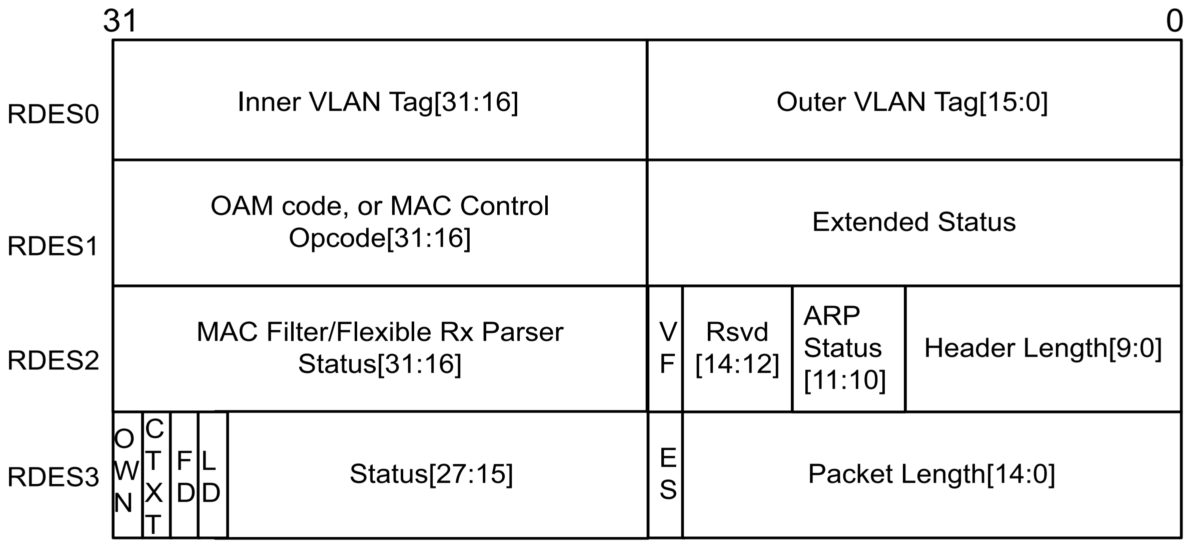
Table 19-21 RDES3 Normal Descriptor (Read Format)

Bit	Name	Description
31	OWN	Own Bit When this bit is set, it indicates that the DWC_ether_qos DMA owns the descriptor. When this bit is reset, it indicates that the application owns the descriptor. The DMA clears this bit when either of the following conditions is true: ■ The DMA completes the packet reception ■ The buffers associated with the descriptor are full
30	IOC	Interrupt Enabled on Completion When this bit is set, an interrupt is issued to the application when the DMA closes this descriptor.
29:26	Rsvd	Reserved
25	BUF2V	Buffer 2 Address Valid When this bit is set, it indicates to the DMA that the buffer 2 address specified in RDES2 is valid. The application must set this bit so that the DMA can use the address, to which the Buffer 2 address in RDES2 is pointing, to write received packet data.
24	BUF1V	Buffer 1 Address Valid When set, this indicates to the DMA that the buffer 1 address specified in RDES1 is valid. The application must set this value if the address pointed to by Buffer 1 address in RDES1 can be used by the DMA to write received packet data.
23:0	Rsvd	Reserved

19.6.2 Receive Normal Descriptor (Write-Back Format)

Figure 19-14 illustrates the write-back format for a Receive Normal descriptor.

Figure 19-14 Receive Normal Descriptor (Write-Back Format)



Note When Flexible RX Parser is enabled, RDES2[31:16] indicates Parser status, and not the MAC filter status. MAC filter status is not available when Flexible RX parser is enabled.

19.6.2.1 RDES0 Normal Descriptor (Write-Back Format)

Table 19-22 describes the write-back format for the RDES0 Normal Descriptor.

Table 19-22 RDES0 Normal Descriptor (Write-Back Format)

Bit	Name	Description
31:16	IVT	Inner VLAN Tag This field contains the Inner VLAN tag of the received packet if the RS0V bit of RDES3 is set. This is valid only when Double VLAN tag processing and VLAN tag stripping are enabled.
15:0	OVT	Outer VLAN Tag This field contains the Outer VLAN tag of the received packet if the RS0V bit of RDES3 is set.

19.6.2.2 RDES1 Normal Descriptor (Write-Back Format)

The Status fields in write-back format are valid only for the last descriptor (RDES3[28] is set). Table 19-23 provides the details of the write-back format for RDES1 Normal Descriptor.

31:16	15	14	13	12	11:8	7	6	5	4	3	2:0
OPC	TD	TSA	PV	PFT	PMT	IPCE	IPCB	IPV6	IPV4	IPHE	PT

Table 19-23 RDES1 Normal Descriptor (Write-Back Format)

Bit	Name	Description
31:16	OPC	OAM Sub-Type Code, or MAC Control Packet opcode OAM Sub-Type Code If Bits[18:16] of RDES3 are set to 3'b111, this field contains the OAM sub-type and code fields. MAC Control Packet opcode If Bits[18:16] of RDES3 are set to 3'b110, this field contains the MAC Control packet opcode field.
15	TD	Timestamp Dropped This bit indicates that the timestamp was captured for this packet but it got dropped in the MTL Rx FIFO because of overflow. This bit is available only when you select the Timestamp feature. Otherwise, this bit is reserved.
14	TSA	Timestamp Available When Timestamp is present, this bit indicates that the timestamp value is available in a context descriptor word 2 (RDES2) and word 1(RDES1). This is valid only when the Last Descriptor bit (RDES3 [28]) is set. The context descriptor is written in the next descriptor just after the last normal descriptor for a packet.
13	PV	PTP Version This bit indicates that the received PTP message has the IEEE 1588 version 2 format. When this bit is reset, it indicates the IEEE 1588 version 1 format. This bit is available only when you select the Timestamp feature. Otherwise, this bit is reserved.
12	PFT	PTP Packet Type This bit indicates that the PTP message is sent directly over Ethernet. This bit is available only when you select the Timestamp feature. Otherwise, this bit is reserved.
11:8	PMT	PTP Message Type These bits are encoded to give the type of the message received: ■ 0000: No PTP message received ■ 0001: SYNC (all clock types) ■ 0010: Follow_Up (all clock types) ■ 0011: Delay_Req (all clock types) ■ 0100: Delay_Resp (all clock types) ■ 0101: Pdelay_Req (in peer-to-peer transparent clock) ■ 0110: Pdelay_Resp (in peer-to-peer transparent clock) ■ 0111: Pdelay_Resp_Follow_Up (in peer-to-peer transparent clock) ■ 1000: Announce ■ 1001: Management ■ 1010: Signaling ■ 1011–1110: Reserved ■ 1111: PTP packet with Reserved message type These bits are available only when you select the Timestamp feature.

Bit	Name	Description
7	IPCE	IP Payload Error When this bit is set, it indicates either of the following: ■ The 16-bit IP payload checksum (that is, the TCP, UDP, or ICMP checksum) calculated by the MAC does not match the corresponding checksum field in the received segment. ■ The TCP, UDP, or ICMP segment length does not match the payload length value in the IP Header field. ■ The TCP, UDP, or ICMP segment length is less than minimum allowed segment length for TCP, UDP, or ICMP. Bit 15 (ES) of RDES3 is not set when this bit is set.
6	IPCB	IP Checksum Bypassed This bit indicates that the checksum offload engine is bypassed. This bit is available when you select the Enable Receive TCP/IP Checksum Check feature.
5	IPV6	IPv6 header Present This bit indicates that an IPV6 header is detected. When the Enable Split Header Feature option is selected and the SPH bit of Control Register of a channel is set, the IPV6 header is available in the header buffer area to which RDES0 is pointing.
4	IPV4	IPV4 Header Present This bit indicates that an IPV4 header is detected. When the SPH bit of RDES3 is set, the IPV4 header is available in the header buffer area to which RDES0 is pointing.
3	IPHE	IP Header Error When this bit is set, it indicates either of the following: ■ The 16-bit IPv4 header checksum calculated by the MAC does not match the received checksum bytes. ■ The IP datagram version is not consistent with the Ethernet Type value. ■ Ethernet packet does not have the expected number of IP header bytes. This bit is valid when either Bit 5 or Bit 4 is set. This bit is available when you select the Enable Receive TCP/IP Checksum Check feature.
2:0	PT	Payload Type These bits indicate the type of payload encapsulated in the IP datagram processed by the Receive Checksum Offload Engine (COE): ■ 3'b000: Unknown type or IP/AV payload not processed ■ 3'b001: UDP ■ 3'b010: TCP ■ 3'b011: ICMP ■ 3'b110: AV Tagged Data Packet ■ 3'b111: AV Tagged Control Packet ■ 3'b101: AV Untagged Control Packet ■ 3'b100: IGMP if IPV4 Header Present bit is set else DCB (LLDP) Control Packet If the COE does not process the payload of an IP datagram because there is an IP header error or fragmented IP, it sets these bits to 3'b000.

19.6.2.3 RDES2 Normal Descriptor (Write-Back Format)

31:29	28	27	26:19	18	17	16	15	14	13:11	10	9:0
L3L4FM	L4FM	L3FM	MADRM	HF	DAF	SAF	OTS	ITS	Rsvd	ARPNR	HL

Table 19-24 RDES2 Normal Descriptor (Write-Back Format)

Bit	Name	Description
31:29	L3L4FM	Layer 3 and Layer 4 Filter Number Matched These bits indicate the number of the Layer 3 and Layer 4 Filter that matched the received packet: ■ 000: Filter 0 ■ 001: Filter 1 ■ 010: Filter 2 ■ 011: Filter 3 ■ 100: Filter 4 ■ 101: Filter 5 ■ 110: Filter 6 ■ 111: Filter 7 This field is valid only when Bit 28 or Bit 27 is set high. When more than one filter matches, these bits give the number of lowest filter. **Note:** This status is not available when Flexible RX Parser is enabled.
28	L4FM	Layer 4 Filter Match When this bit is set, it indicates that the received packet matches one of the enabled Layer 4 Port Number fields. This status is given only when one of the following conditions is true: ■ Layer 3 fields are not enabled and all enabled Layer 4 fields match ■ All enabled Layer 3 and Layer 4 filter fields match When more than one filter matches, this bit gives the layer 4 filter status of filter indicated by Bits[31:29]. **Note:** This status is not available when Flexible RX Parser is enabled.
27	L3FM	Layer 3 Filter Match When this bit is set, it indicates that the received packet matches one of the enabled Layer 3 IP Address fields. This status is given only when one of the following conditions is true: ■ All enabled Layer 3 fields match and all enabled Layer 4 fields are bypassed ■ All enabled filter fields match When more than one filter matches, this bit gives the layer 3 filter status of filter indicated by Bits[31:29]. **Note:** This status is not available when Flexible RX Parser is enabled.

Bit	Name	Description
26:19	MADRM	MAC Address Match or Hash Value When the HF bit is reset, this field contains the MAC address register number that matched the Destination address of the received packet. This field is valid only if the DAF bit is reset. When the HF bit is set, this field contains the hash value computed by the MAC. A packet passes the hash filter when the bit corresponding to the hash value is set in the hash filter register. **Note:** This status is not available when Flexible RX Parser is enabled.
18	HF	Hash Filter Status When this bit is set, it indicates that the packet passed the MAC address hash filter. Bits[26:19] indicate the hash value. **Note:** This status is not available when Flexible RX Parser is enabled.
17	DAF/RXPI	Destination Address Filter Fail When Flexible RX Parser is disabled, and this bit is set, it indicates that the packet failed the DA Filter in the MAC. When Flexible RX Parser is enabled, this bit is set to indicate that the packet parsing is incomplete (RXPI) due to ECC error. **Note:** When this bit is set, ES bit of RDES3 is also set.
16	SAF/RXPD	SA Address Filter Fail When Flexible RX Parser is disabled, and this bit is set, it indicates that the packet failed the SA Filter in the MAC. When Flexible RX Parser is enabled, this bit is set to indicate that the packet is dropped (RXPD) by the parser. **Note:** When this bit is set, ES bit of RDES3 is also set.
15	OTS	VLAN Filter Status When set, this bit indicates that the VLAN Tag of the received packed passed the VLAN filter. This bit is valid only when DWC_EQOS_ERVFE is not enabled. If DWC_EQOS_ERVFE is enabled, the bit is redefined as Outer VLAN Tag Filter Status (OTS). This bit is valid for both Single and Double VLAN Tagged frames
14	ITS	Inner VLAN Tag Filter Status (ITS) This bit is valid only when DWC_EQOS_ERVFE is enabled. This bit is valid only for Double VLAN Tagged frames, when Double VLAN Processing is enabled. For more information, see the Filter Status topic.
13:11	Rsvd	Reserved
10	ARPNR	ARP Reply Not Generated When this bit is set, it indicates that the MAC did not generate the ARP Reply for received ARP Request packet. This bit is set when the MAC is busy transmitting ARP reply to earlier ARP request (only one ARP request is processed at a time). This bit is reserved when the Enable IPv4 ARP Offload option is not selected.

Bit	Name	Description
9:0	HL	L3/L4 Header Length This field contains the length of the header of the packet split by the MAC at L3 or L4 header boundary as identified by the MAC receiver. This field is valid only when the first descriptor bit is set (FD = 1). The header data is written to the Buffer 1 address of corresponding descriptor. If header length is zero, this field is not valid. It implies that the MAC did not identify and split the header. This field is valid when the Enable Split Header Feature option is selected.

19.6.2.4 RDES3 Normal Descriptor (Write-Back Format)

Table 19-25 describes the write-back format for the RDES3 Normal Descriptor.

31	30	29	28	27	26	25	24	23	22	21	20	19	18:16	15	14:0
OWN	CTXT	FD	LD	RS2V	RS1V	RS0V	CE	GP	RWT	OE	RE	DE	LT	ES	PL

Table 19-25 RDES3 Normal Descriptor (Write-Back Format)

Bit	Name	Description
31	OWN	Own Bit When this bit is set, it indicates that the DWC_ether_qos DMA owns the descriptor. When this bit is reset, it indicates that the application owns the descriptor. The DMA clears this bit when either of the following conditions is true: ■ The DMA completes the packet reception ■ The buffers associated with the descriptor are full
30	CTXT	Receive Context Descriptor When this bit is set, it indicates that the current descriptor is a context type descriptor. The DMA writes 1'b0 to this bit for normal receive descriptor. When CTXT and FD bits are used together, {CTXT, FD} ■ 2'b00: Intermediate Descriptor ■ 2'b01: First Descriptor ■ 2'b10: Reserved ■ 2'b11: Descriptor Error (due to all 1s) **Note:** When Descriptor Error occurs, the Receive DMA closes the receive descriptor indicating Descriptor Error. This receive descriptor is skipped and the buffer addresses are not used to write the packet data Receive DMA will set the CDE bit in DMA_CH#_Status register but not the RI bit even when IOC is set, as this is not marked as last receive descriptor for the packet. The subsequent valid receive descriptor is used to write the packet data.
29	FD	First Descriptor When this bit is set, it indicates that this descriptor contains the first buffer of the packet. If the size of the first buffer is 0, the second buffer contains the beginning of the packet. If the size of the second buffer is also 0, the next descriptor contains the beginning of the packet. See the CTXT bit description for details of using the CTXT bit and FD bit together.

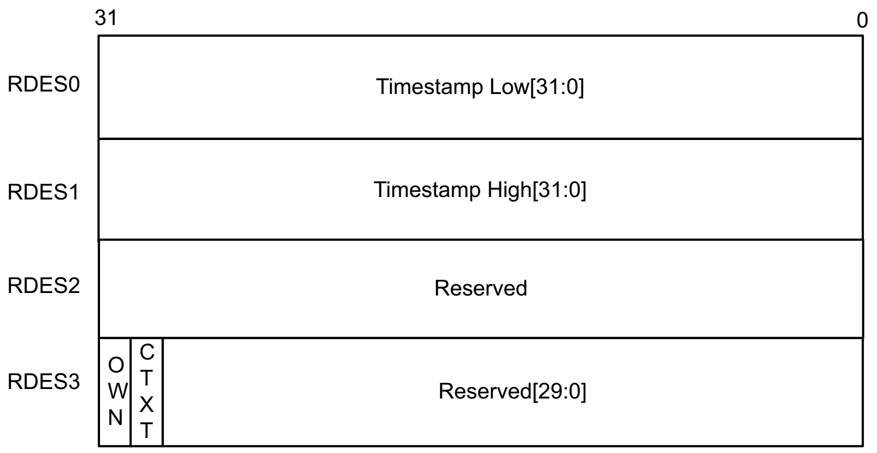
Bit	Name	Description
28	LD	Last Descriptor When this bit is set, it indicates that the buffers to which this descriptor is pointing are the last buffers of the packet.
27	RS2V	Receive Status RDES2 Valid When this bit is set, it indicates that the status in RDES2 is valid and it is written by the DMA. This bit is valid only when the LD bit of RDES3 is set.
26	RS1V	Receive Status RDES1 Valid When this bit is set, it indicates that the status in RDES1 is valid and it is written by the DMA. This bit is valid only when the LD bit of RDES3 is set.
25	RS0V	Receive Status RDES0 Valid When this bit is set, it indicates that the status in RDES0 is valid and it is written by the DMA. This bit is valid only when the LD bit of RDES3 is set.
24	CE	CRC Error When this bit is set, it indicates that a Cyclic Redundancy Check (CRC) Error occurred on the received packet. This field is valid only when the LD bit of RDES3 is set.
23	GP	Giant Packet When this bit is set, it indicates that the packet length exceeds the specified maximum Ethernet size of 1518, 1522, or 2000 bytes (9018 or 9022 bytes if jumbo packet enable is set). Note: Giant packet indicates only the packet length. It does not cause any packet truncation.
22	RWT	Receive Watchdog Timeout When this bit is set, it indicates that the Receive Watchdog Timer has expired while receiving the current packet. The current packet is truncated after watchdog timeout.
21	OE	Overflow Error When this bit is set, it indicates that the received packet is damaged because of buffer overflow in Rx FIFO. Note: This bit is set only when the DMA transfers a partial packet to the application. This happens only when the Rx FIFO is operating in the threshold mode. In the store-and-forward mode, all partial packets are dropped completely in Rx FIFO.
20	RE	Receive Error When this bit is set, it indicates that the gmii_rxer_i signal is asserted while the gmii_rxdv_i signal is asserted during packet reception. This error also includes carrier extension error in the GMII and half-duplex mode. Error can be of less or no extension, or error (rxd!= 0f) during extension.
19	DE	Dribble Bit Error When this bit is set, it indicates that the received packet has a non-integer multiple of bytes (odd nibbles). This bit is valid only in the MII Mode.

Bit	Name	Description
18:16	LT	Length/Type Field This field indicates if the packet received is a length packet or a type packet. The encoding of the 3 bits is as follows: ■ 3'b000: The packet is a length packet ■ 3'b001: The packet is a type packet. ■ 3'b011: The packet is a ARP Request packet type ■ 3'b100: The packet is a type packet with VLAN Tag ■ 3'b101: The packet is a type packet with Double VLAN Tag ■ 3'b110: The packet is a MAC Control packet type ■ 3'b111: The packet is a OAM packet type ■ 3'b010: Reserved
15	ES	Error Summary When this bit is set, it indicates the logical OR of the following bits: ■ RDES3[24]: CRC Error ■ RDES3[19]: Dribble Error ■ RDES3[20]: Receive Error ■ RDES3[22]: Watchdog Timeout ■ RDES3[21]: Overflow Error ■ RDES3[23]: Giant Packet ■ RDES2[17]: Destination Address Filter Fail, when Flexible RX Parser is enabled ■ RDES2[16]: SA Address Filter Fail, when Flexible RX Parser is enabled This field is valid only when the LD bit of RDES3 is set.
14:0	PL	Packet Length These bits indicate the byte length of the received packet that was transferred to system memory (including CRC). This field is valid when the LD bit of RDES3 is set and Overflow Error bits are reset. The packet length also includes the two bytes appended to the Ethernet packet when IP checksum calculation is enabled and the received packet is not a MAC control packet. This field is valid when the LD bit of RDES3 is set. When the Last Descriptor and Error Summary bits are not set, this field indicates the accumulated number of bytes that have been transferred for the current packet.

19.6.3 Receive Context Descriptor

This descriptor is read-only for the application. Only the DMA can write to this descriptor. The context descriptor provides information about the extended status related to the last received packet. The Bit 30 of RDES3 indicates the context type descriptor.

Figure 19-15 Receive Context Descriptor Format



19.6.3.1 RDES0 Context Descriptor

Table 19-26 RDES0 Context Descriptor

Bit	Name	Description
31:0	RTSL	Receive Packet Timestamp Low The DMA updates this field with least significant 32 bits of the timestamp captured for corresponding Receive packet. When this field and the RTSH field of RDES1 show all-ones value, the timestamp must be considered as corrupt.

19.6.3.2 RDES1 Context Descriptor

Table 19-27 RDES1 Context Descriptor

Bit	Field	Description
31:0	RTSH	Receive Packet Timestamp High The DMA updates this field with most significant 32 bits of the timestamp captured for corresponding receive packet. When this field and the RTSL field of RDES0 show all-ones value, the timestamp must be considered as corrupt.

19.6.3.3 RDES2 Context Descriptor

Table 19-28 RDES2 Context Descriptor

Bit	Description
31:0	Reserved

19.6.3.4 RDES3 Context Descriptor

31	30	29	28:0
OWN	CTXT	DE	Rsvd

Table 19-29 RDES3 Context Descriptor

Bit	Name	Description
31	OWN	Own Bit When this bit is set, it indicates that the DMA owns the descriptor. When this bit is reset, it indicates that the application owns the descriptor. The DMA clears this bit when either of the following conditions is true: ■ The DMA completes the packet reception ■ The buffers associated with the descriptor are full
30	CTXT	Receive Context Descriptor When this bit is set, it indicates that the current descriptor is a context descriptor. The DMA writes 1'b1 to this bit for context descriptor. DMA writes 2'b11 to indicate a descriptor error due to all 1s. When CTXT and DE bits are used together, {CTXT, DE} ■ 2'b00: Reserved ■ 2'b01: Reserved ■ 2'b10: Context Descriptor ■ 2'b11: Descriptor Error **Note:** When Descriptor Error occurs, the Receive DMA closes the receive descriptor indicating Descriptor Error. This receive descriptor is skipped and the buffer addresses are not used to write the packet data Receive DMA will set the CDE bit in DMA_CH#_Status register but not the RI bit even when IOC is set, as this is not marked as last receive descriptor for the packet. The subsequent valid receive descriptor is used to write the packet data.
29	DE	Descriptor Error See the CTXT bit description for details of using theDE bit along with CTXT bit.
28:0	Rsvd	Reserved

19.7 Enhanced Descriptor for Time-Based Scheduling

The time-based scheduling feature needs Enhanced Descriptors (that are 32 Bytes) to be enabled on all the DMA channels that intend to use the feature (by setting the EDSE bit of DMA_CH(#i)_TX_Control register).

The structure of 32-byte Descriptor for the Context and the Normal Descriptor in Read and Write formats are described in the following sections:

19.7.1 Enhanced Normal Descriptor - Read (32-Bit Mode)

The fields present in the first 16 bytes of the Enhanced Descriptor format of normal descriptor are as follows:

- **LTV**

  Launch Time Valid indicates the Launch Time (LT) and GSN fields present in the descriptor are valid. The LTV must be set only if the FD bit of the descriptor is not.

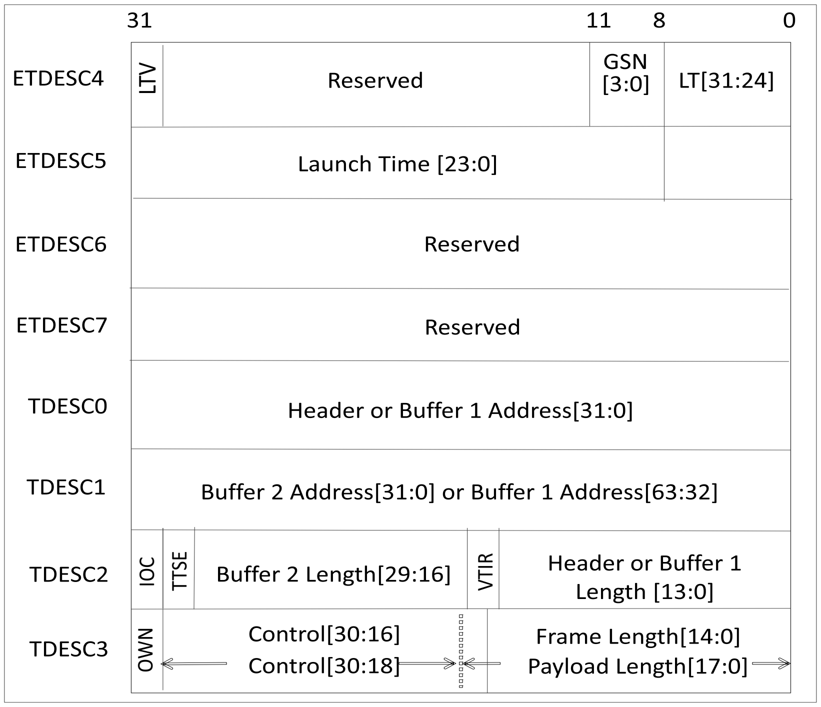
- **GSN**

  This field indicates the GCL slot number associated with the packet.

- **Launch Time (LT)**

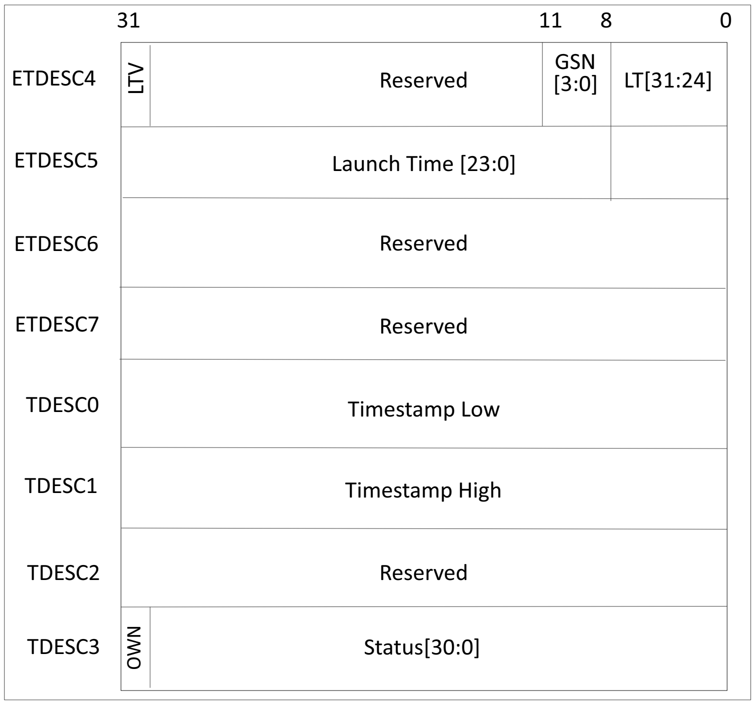
  This field indicates the Launch Time associated with the packet.

Figure 19-16 Enhanced Normal Descriptor - Read (32-Bit Mode)



19.7.2 Enhanced Normal Descriptor (Write, 32-Bit Mode)

Figure 19-17 Enhanced Normal Descriptor Write (32-Bit Mode)



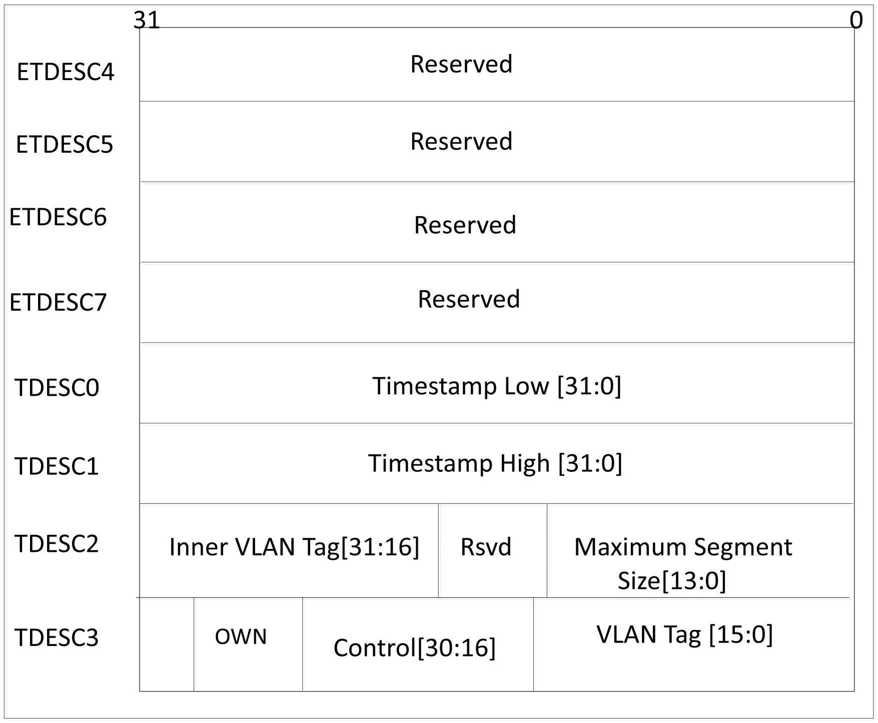
Note

- All the enhanced descriptor examples, assumes a 32-bit data width configurations. However, the same definitions are valid in 64 and 128 bit configurations.
- In both the write back formats, no modifications can be done to the extended 16 bytes. The rest of the 16 bytes (TDESC0 to TDESC3) are written back as per the previous 16 bytes Descriptor format.
- Fetch time when enabled overrides the AV slot function
- When an unaligned new GCL list (with a different CTR) is installed it is recommended not to have traffic during the installation of the new list. Any traffic during the switching of the lists might have unpredictable behavior regarding Fetch, Launch, and Launch expiry as the CTR and BTR values get updated while the frame is being processed

19.7.3 Enhanced Context Descriptor (Read, 32-bit Mode)

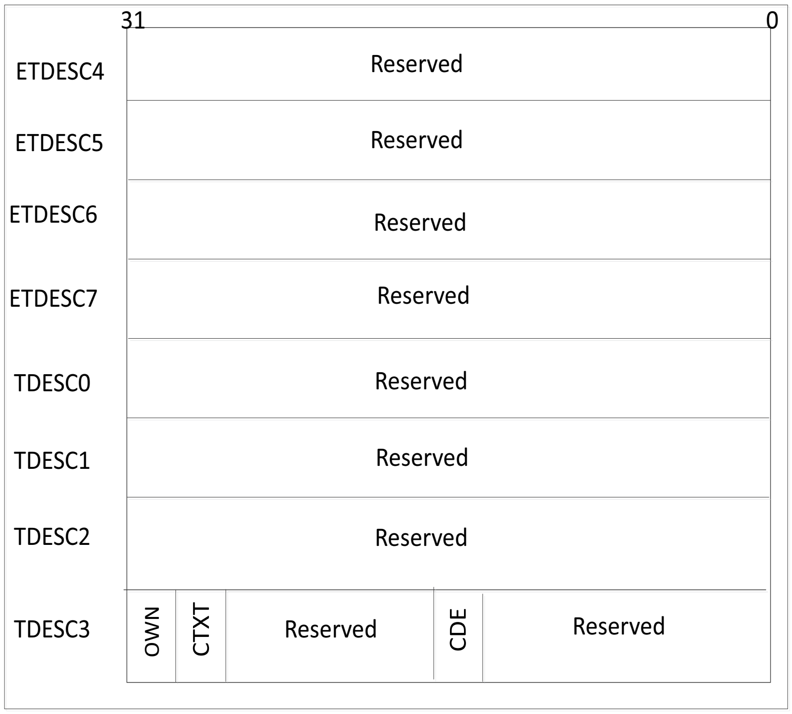
The first 16 bytes of the Enhanced context descriptor (ETDESC4 to ETDESC7) are reserved and must be zeros. The fields present in last bytes of the Descriptor (TDESC0 to TDESC3) are same as the context descriptor (TDESC0 to TDESC3) in 16 byte format.

Figure 19-18 Enhanced Context Descriptor - Read



19.7.4 Enhanced Context Descriptor (Write, 32 Bit Mode)

Figure 19-19 Enhanced Context Descriptor - Write



20 Programming

This chapter provides the instructions for initializing the DMA or MAC registers in the proper sequence. It contains the following sections:

Note

When any Register content is being transferred to a different clock domain after a write operation, there should not be any further writes to the same location until the first write is updated. Otherwise, the second write operation does not get updated to the destination clock domain. Therefore, the delay between two writes to the same register location should be at least four cycles of the destination clock (PHY Receive clock, PHY Transmit clock, or PTP clock). This limitation is removed when you select **Enable support for back-to-back register writes** during RTL configuration.

20.1 Initializing DMA

Complete the following steps to initialize the DMA:

1. Provide a software reset. This resets all of the MAC internal registers and logic (bit-0 of DMA_Mode).
2. Wait for the completion of the reset process (poll bit 0 of the DMA_Mode, which is only cleared after the reset operation is completed).
3. Program the following fields to initialize the DMA_SysBus_Mode register:
   a. AAL
   b. Fixed burst or undefined burst
   c. Burst mode values in case of AHB bus interface, OSR_LMT in case of AXI bus interface.
   d. If fixed length value is enabled, select the maximum burst length possible on the AXI Bus (bits [7:1])
4. Create a descriptor list for transmit and receive. In addition, ensure that the descriptors are owned by DMA (set bit 31 of descriptor TDES3/RDES3).

   For more information about descriptors, see “Descriptors” on page 1315.
5. Program the Transmit and Receive Ring length registers (DMA_CH(#i)_TxDesc_Ring_Length (for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1) and DMA_CH(#i)_RxDesc_Ring_Length (for i = 0; i <= DWC_EQOS_NUM_DMA_RX_CH-1)). The ring length programmed must be at least 4.

Note The descriptor address from the start to the end of the ring must not cross the 4GB boundary.

6. Initialize receive and transmit descriptor list address with the base address of the transmit and receive descriptor (DMA_CH(#i)_TxDesc_List_Address (for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1), DMA_CH(#i)_RxDesc_List_Address (for i = 0; i <= DWC_EQOS_NUM_DMA_RX_CH-1)). Also, program transmit and receive tail pointer registers indicating to the DMA about the available descriptors (DMA_CH(#i)_TxDesc_Tail_Pointer (for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1) and DMA_CH(#i)_RxDesc_Tail_Pointer (for i = 0; i <= DWC_EQOS_NUM_DMA_RX_CH-1))..

Note For 40-bit or 48-bit addressing mode, program the higher address List registers (DMA_CH[n]_TxDesc_List_HAddress, DMA_CH[n]_RxDesc_List_HAddress).
The tailpointer registers must be advanced to the location immediately after the descriptors that are set, for the DMA to know that additional descriptors are available.

7. Program the settings of the following registers for the parameters like maximum burst-length (PBL) initiated by DMA, descriptor skip lengths, OSP in case of TxDMA, RBSZ in case of RxDMA, and so on:
   - DMA_CH(#i)_Control (for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1)
   - DMA_CH(#i)_TX_Control (for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1)
   - DMA_CH(#i)_RX_Control (for i = 0; i <= DWC_EQOS_NUM_DMA_RX_CH-1)
8. Enable the interrupts by programming the DMA_CH(#i)_Interrupt_Enable (for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1) register.

9. Start the Receive and Transmit DMAs by setting SR (bit 0) of the DMA_CH(#i)_RX_Control (for i = 0; i <= DWC_EQOS_NUM_DMA_RX_CH-1) and ST (bit 0) of the DMA_CH(#i)_TX_Control (for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1) register.
10. Repeat steps 4 to 9 for all the Tx DMA and Rx DMA channels selected in the hardware.

20.2 Initializing MTL Registers

The Transaction Layer (MTL) registers must be initialized to establish the transmit and receive operating modes and commands.

Complete the following steps to initialize the MTL registers:

1. Program the Tx Scheduling (SCHALG) and Receive Arbitration Algorithm (RAA) fields in MTL_Operation_Mode to initialize the MTL operation in case of multiple Tx and Rx queues.
2. Program the Receive Queue to DMA mapping in MTL_RxQ_DMA_Map0 and MTL_RxQ_DMA_Map1 registers.
3. Program the following fields to initialize the mode of operation in the MTL_TxQ0_Operation_Mode register.
   a. Transmit Store And Forward (TSF) or Transmit Threshold Control (TTC) in case of threshold mode
   b. Transmit Queue Enable (TXQEN) to value 2‘b10 to enable Transmit Queue0
   c. Transmit Queue Size (TQS)
4. Program the following fields to initialize the mode of operation in the MTL_RxQ0_Operation_Mode register:
   a. Receive Store and Forward (RSF) or RTC in case of Threshold mode
   b. Flow Control Activation and De-activation thresholds for MTL Receive FIFO (RFA and RFD)
   c. Error Packet and undersized good Packet forwarding enable (FEP and FUP)
   d. Receive Queue Size (RQS)
5. Repeat previous two steps for all MTL Tx and Rx queues selected in the configuration.

20.3 Initializing MAC

The MAC configuration registers establish the operating mode of the MAC. These registers must be initialized before initializing the DMA.

The following MAC Initialization operations can be performed after DMA initialization. If the MAC initialization is completed before the DMA is configured, enable the MAC receiver (last step in the following sequence) only after the DMA is active. Otherwise, received frames fill the Rx FIFO and overflow.

1. Provide the MAC address registers: MAC_Address0_High and MAC_Address0_Low. If more than one MAC address is enabled in your configuration (during configuration in the coreConsultant), program the MAC addresses appropriately.
2. Program the following fields to set the appropriate filters for the incoming frames in the MAC_Packet_Filter register:
   a. Receive All
   b. Promiscuous mode
   c. Hash or Perfect Filter
   d. Unicast, multicast, broadcast, and control frames filter settings
3. Program the following fields for proper flow control in the MAC_Q0_Tx_Flow_Ctrl register:
   a. Pause time and other Pause frame control bits
   b. Transmit Flow control bits
   c. Flow Control Busy
4. Program the MAC_Interrupt_Enable register, as required, and if applicable, for your configuration.
5. Program the appropriate fields in the MAC_Configuration register. For ex: Inter-packet gap while transmission and jabber disable.
6. Set bit 0 and 1 in MAC_Configuration registers to start the MAC transmitter and receiver.

20.4 Performing Normal Receive and Transmit Operation

During normal operation of the DWC_ether_qos, normal and transmit interrupts are read, descriptors polled, the DMA is suspended (if it does not own descriptors), and values of current host transmitter or receiver descriptor pointers are read for debugging.

For normal operation, complete the following steps:

1. For normal transmit and receive interrupts, read the interrupt status. Then, poll the descriptors, reading the status of the descriptor owned by the Host (either transmit or receive).
2. Set appropriate values for the descriptors, ensuring that transmit and receive descriptors are owned by the DMA to resume the transmission and reception of data.
3. If the descriptors are not owned by the DMA (or no descriptor is available), the DMA goes into SUSPEND state. The transmission or reception can be resumed by freeing the descriptors and writing the descriptor tail pointer to Tx/Rx tail pointer register (DMA_CH[n]_TxDesc_Tail_Pointer and DMA_CH[n]_RxDesc_Tail_Pointer).
4. The values of the current host transmitter or receiver descriptor address pointer can be read for the debug process (DMA_CH[n]_Current_App_TxDesc and DMA_CH[n]_Current_App_RxDesc).
5. The values of the current host transmit buffer address pointer and receive buffer address pointer can be read for the debug process (Register DMA_CH[n]_Current_App_TxBuffer and DMA_CH[n]_Current_App_RxBuffer).

20.5 Stopping and Starting Transmission

Complete the following steps to pause the transmission for some time. The steps are provided for Channel 0.

Note The steps are provided for Channel 0.

1. Disable the Transmit DMA (if applicable) by clearing Bit 0 (ST) of DMA_CH(#i)_TX_Control (for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1) Register.
2. Wait for any previous frame transmissions to complete. You can check this by reading the appropriate bits of MTL_TxQ0_Debug Register (TRCSTS is not 01 and TXQSTS=0).
3. Disable the MAC transmitter and MAC receiver by clearing Bit (RE) and Bit 1(TE) of the MAC_Configuration Register.
4. Disable the Receive DMA (if applicable), after making sure that the data in the Rx FIFO is transferred to the system memory (by reading the appropriate bits of MTL_TxQ0_Debug Register, PRXQ=0 and RXQSTS=00).
5. Make sure that both Tx Queue and Rx Queue are empty (TXQSTS is 0 in MTL_TxQ0_Debug Register and RXQSTS is 0 in MTL_RxQ0_Debug Register).
6. To restart the operation, first start the DMAs, and then enable the MAC Transmitter and Receiver.

Note
- Do not change the configuration (such as duplex mode, speed, port, or loop back) when the MAC is actively transmitting or receiving. These parameters are changed by software only when the MAC transmitter and receiver are not active.
- Similarly, do not change the DMA-related configuration when Transmit and Receive DMA are active.

20.6 Programming Guidelines for Switching to New Descriptor List in RxDMA

Switching to a new descriptor list is different in the Rx DMA compared to the Tx DMA. Switching to a new descriptor list is permitted when the RxDMA is in SUSPEND state, as clarified by the following points:

- Generally, RxDMA prepares the descriptors in advance.
- If the RxDMA goes to SUSPEND due to descriptors not being available, a major failure occurs (software is not able to free the filled-up descriptors/buffers). If this issue is not rectified immediately, frames will be lost because of an RxFIFO Overflow. Therefore, the software is allowed to create a new descriptor list and program the RxDMA to start using it immediately, without going into STOP state.

20.7 Programming Guidelines for Switching the AHB or AXI Clock Frequency

To dynamically change the AHB or AXI clock frequency (without applying soft reset or hard reset) in EQOS-AHB or EQOS-AXI configuration, follow these steps:

1. Disable the Transmit DMA (if applicable) and wait for any previous frame transmissions to complete. When the frame transmissions is complete, the Tx FIFO becomes empty and the Tx DMA enters the STOP state. The Tx FIFO status is given in the MTL_TxQ[n]_Debug register and the status of DMA is given in the DMA_Debug_Status0 register.
2. Disable the MAC transmitter and the MAC receiver by clearing the appropriate bits in MAC Configuration Register.
3. Disable the Receive DMA (if applicable) after making sure that the data in the Rx FIFO is transferred to the system memory. The Rx FIFO empty status is given in MTL_RxQ([n]_Debug Register.
4. Ensure that the application does not perform any register read or write operation.
5. Change the frequency of the AHB or AXI clock.
6. Enable the MAC Transmitter or the MAC Receiver and the Transmit or Receive DMA.

These steps ensure that no valid data is present in the Tx FIFO or Rx FIFO at the time of clock frequency switching and prevent any data corruption.

Note To avoid glitches or short pulses, use glitch-free MUXs or other mechanisms for clock switching.

20.8 Programming Guidelines for Multi-Channel Multi-Queuing

20.8.1 Transmit

1. Program the Transmit queue size in the TQS field of MTL_TxQ[n]_Operation_Mode register. Based on the value programmed in the TQS field, the size of the queue is determined.

   In the Transmit operation, the number of channels is equal to the number of the queues. Due to this reason, the Channel-to-Queue mapping is fixed.

2. For a queue to be used, the queue needs to be enabled in TXQEN in the corresponding MTL_TxQ[n]_Operation_Mode Register. In DMA configurations, the ST bit of DMA_CH[n]_Tx_Control Register and corresponding TXQEN in MTL_TxQ[n]_Operation Mode Register needs to be enabled.

3. The scheduling method needs to be programmed in SCHALG of MTL_Operation_Mode register.

4. Program the MTL_TxQ[n]_Quantum_Weight for DCB queue as per the selected algorithm. In case of CBS algorithm in AVB queues, the MTL_TxQ[n]_ETS_Control, MTL_TxQ[n]_SendSlopeCredit, MTL_TxQ[n]_HiCredit and MTL_TxQ[n]_LoCredit registers also need to be programmed as required.

5. If DCB is enabled and PFC function is required, program MAC_TxQ_Prty_Map0 Register to assign a fixed priority to the queue. This priority assigned is used for determining if the corresponding queue should stop transmitting packet based on the received PFC packet.

20.8.2 Receive

1. Program the Receive queue size in the RQS field of MTL_RxQ[n]_Operation_Mode Register. Based on the value programmed in RQS field, the size of the queue is determined.

2. Enable the Receive Queues 0 to 7 in the fields RXQ0EN to RXQ7EN in MAC_RxQ_Ctrl0 Register for AV or DCB. In DMA configurations, SR bit of statically or dynamically mapped DMA_CH[n]_Rx_Control Register and corresponding RXQ[n]_EN in MAC_RxQ_Ctrl0 Register needs to be enabled.

3. The MAC routes the Rx packets to the Rx Queues based on following packet types:

   a. AV PTP Packets: Based on the programming of AVPTPQ in MAC_RxQ_Ctrl1 Register.

   b. AV Untagged Control packets: Based on the programming of AVCPQ in MAC_RxQ_Ctrl1 Register.

   c. Data Center Bridging (DCB) related Link Layer Discovery Protocol (LLDP) packets. Program DCBCPQ in MAC_RxQ_Ctrl1 Register to indicate to MAC which queue should get the DCB packets.

   d. VLAN Tag Priority field in VLAN Tagged packets: Program PSRQ7-0 of the MAC_RxQ_Ctrl2 and MAC_RxQ_Ctrl3 Register for the routing of tagged packets based on the USP (user Priority) field of the received packets to the Rx Queues 0 to 7.

   e. The AV tagged control and data packets are also routed based on PSRQ field of the MAC_RxQ_Ctrl2 and MAC_RxQ_Ctrl3 registers.

Note The priorities set in PSRQ7-0 should be unique.

4. If multiple RX DMA channels are enabled, the following programming should be done for proper arbitration and mapping:
   a. Program the RAA field of MTL_Operation_Mode register to select the arbitration algorithm to decide which RxQ is read out from the RxFIFO memory.
   b. Program the MTL_RxQ[n]_Control to decide the weights and the packet arbitration for each RxQ.
   c. If static mapping is programmed in MTL_RxQ_DMA_Map[n] register (RXQ[n]DADMACH is reset to 0), bits RXQx2DMA and others need to be programmed to select the channel for which each queue is mapped.
   d. Set RXQ[n]DADMACH bit in MTL_RxQ_DMA_Map0 Register to select dynamic mapping of packets in each RxQueue.
   e. In dynamic channel mapping, the routing of a packet to a specific RxDMA channel is decided by the value of DCS field in the lowest MAC Address Register.

20.8.3 Programming Guidelines for Recovering from DMA Channel Failure

When the DMA channel issues a bus error, follow these steps to recover from the failure.

20.8.3.1 Recovering from the Receive DMA Channel Failure

Follow these steps if you get bus error in the Receive DMA channel:

1. Set the RPF bit to 1. This flushes all the packets one after the other.

   This step is optional. However, setting this bit prevents HOL (head-of-line) blocking in the Rx queues when packets sent to the RXDMA are stopped due to bus error.
2. Re-program the specific Registers of the DMA channel.
3. Start the DMA channel.

20.8.3.2 Recovering from the Transmit DMA Channel Failure

1. Stop the specific DMA channel, even if it is in active state.
2. Flush the corresponding MTL queue.
3. Re-program the specific Registers of the DMA channel.
4. Start the DMA channel.

Note Due to the known limitations in the design, reprogramming the DMA channel registers might not always be successful in recovering from a Bus Error. If DWC_ether_qos is not fully functional after reprogramming the DMA, as a workaround, issue a Soft Reset to recover from the Bus Error.

20.9 Programming Guidelines for GMII Link State Transitions

20.9.1 Transmit and Receive Clocks Running when Link Down

Complete the following steps when the link is down but the Transmit and Receive clocks are running:

Note The steps are provided for Channel 0.

1. Disable the Transmit DMA (if applicable) by clearing Bit 0 (ST) of DMA_CH(#i)_TX_Control (for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1) Register.
2. Disable the MAC receiver by clearing Bit 2 (RE) of MAC_Configuration Register.
3. Wait for any previous frame transmissions to complete. You can check this by reading the appropriate bits of MTL_TxQ0_Debug Register (TRCSTS is not 01).Or, Flush the Tx FIFO for faster empty operation.
4. Disable the MAC transmitter by clearing Bit 1(TE) of the MAC_Configuration Register.
5. Make sure that both Tx Queue and Rx Queue are empty (TXQSTS is 0 in MTL_TxQ0_Debug Register and RXQSTS is 0 in MTL_RxQ0_Debug Register).
6. After the link is up, read the PHY registers to know the latest configuration and accordingly program the MAC registers.
7. Restart the operation by starting the Tx DMA, and then enabling the MAC Transmitter and Receiver.

   You do not need to disable the Rx DMA. As the Receiver is disabled, the FIFO does not get any data in the Rx FIFO.

20.9.2 Transmit and Receive Clocks Stopped when Link Down

Complete the following steps when the link is down and the Transmit and Receive clocks are stopped:

Note The steps are provided for Channel 0.

1. Disable the MAC Transmitter and Receiver by clearing bits RE and TE of MAC_Configuration Register. This does not take effect immediately as the clocks are absent.
2. Wait until the link is up and the clocks are restored.
3. Wait for the completion of the transfer of any partial frame, if any, at time of stopping of Transmit/Receive clock. This can be checked by reading the MAC_Debug Register (should be all-zero). Some old packets may still remain in the TXFIFO as the MAC Transmitter is stopped.
4. Read the PHY registers to know the latest operating mode and accordingly program the MAC registers.
5. Restart the MAC Transmitter and Receiver by setting RE and TE bits.

20.10 Programming Guidelines for IEEE 1588 Timestamping

20.10.1 Initialization Guidelines for System Time Generation

You can enable the timestamp feature by setting Bit 0 of the MAC_Timestamp_Control Register. However, it is essential that the timestamp counter be initialized after this bit is set. Complete the following steps during DWC_ether_qos initialization:

1. Mask the Timestamp Trigger interrupt by clearing the bit 16 of MAC_Interrupt_Enable Register.
2. Set Bit 0 of MAC_Timestamp_Control Register to enable timestamping.
3. Program MAC_Sub_Second_Increment Register based on the PTP clock frequency.
4. If you are using the Fine Correction approach, program MAC_Timestamp_Addend and set Bit 5 of MAC_Timestamp_Control Register.
5. Poll the MAC_Timestamp_Control Register until Bit 5 is cleared.
6. Program Bit 1 of MAC_Timestamp_Control Register to select the Fine Update method (if required).
7. Program MAC_System_Time_Seconds_Update Register and MAC_System_Time_Nanoseconds_Update Register with the appropriate time value.
8. Set Bit 2 in MAC_Timestamp_Control Register.

   The timestamp counter starts operation as soon as it is initialized with the value written in the Timestamp Update registers.

   If one-step timestamping is enabled
   a. To enable one-step timestamping, program Bit 27 of the TDES3 Context Descriptor.
   b. Program registers MAC_Timestamp_Ingress_Asym_Corr and MAC_Timestamp_Egress_Asym_Corr to update the correction field in PDelay_Req PTP messages.
9. Enable the MAC receiver and transmitter for proper timestamping.

Note If timestamp operation is disabled by clearing Bit 0 of MAC_Timestamp_Control Register, repeat all these steps to restart the timestamp operation.

20.10.2 System Time Correction

20.10.2.1 Coarse Correction Method

To synchronize or update the system time in one process (coarse correction method), complete the following steps:

1. Set the offset (positive or negative) in the Timestamp Update registers(MAC_System_Time_Seconds_Update and MAC_System_Time_Nanoseconds_Update).
2. Set Bit 3 (TSUPDT) of MAC_Timestamp_Control Register.

   The value in the Timestamp Update registers is added to or subtracted from the system time when the TSUPDT bit is cleared.

20.10.2.2 Fine Correction Method

To synchronize or update the system time to reduce system-time jitter (fine correction method), complete the following steps:

1. With the help of the algorithm explained in"System Time Register Module" on page 258 calculate the rate by which you want to make the system time increments slower or faster.
2. Update the MAC_Timestamp_Addend with the new value and set Bit 5 of the MAC_Timestamp_Control Register.
3. Wait for the time for which you want the new value of the Addend register to be active. You can do this by enabling the Timestamp Trigger interrupt after the system time reaches the target value.
4. Program the required target time in MAC_PPS[n]_Target_Time_Seconds Register and MAC_PPS[n]_Target_Time_Nanoseconds Register.
5. Enable the Timestamp interrupt in bit 12 of MAC_Interrupt_Enable register.
6. Set bit 4 in Register MAC_Timestamp_Control.
7. When this trigger causes an interrupt, read MAC_Interrupt_Status Register.
8. Reprogram MAC_Timestamp_Addend Register with the old value and set bit 5 again

20.11 Programming Guidelines for AV Feature

After you enable the AV feature in the DWC_ether_qos controller, there are several programming tasks to be followed:

- Initializing the DMA for QOS-AHB configurations only
- Enabling slot number checking
- Enabling average bits per slot reporting
- Disabling flow control for AV-enabled queues (transmit and receive flow control)

20.11.1 Initializing the DMA in Audio Video Feature

The first step in programming the AV feature in a QOS-AHB configuration is to initialize the DMA.

Use this initialization sequence for QOS-AHB/QOS-AXI/QOS-AXI4 configurations with AV feature.

1. Provide a software reset to reset all QOS internal registers and logic (bit 0 in DMA_Mode register).
2. Wait for the completion of the reset process. Poll bit 0 of the DMA_Mode register), which is cleared only after the reset operation is completed.
3. Program the fields to initialize the DMA register by setting the values in DMA_Mode register.
4. Create a proper descriptor list for transmit and receive. In addition, ensure that the DMA owns the Transmit and Receive descriptors. When OSF mode is used, at least two TX descriptors are required.

   For more information about descriptors, see "Descriptors" on page 1315.
5. Make sure that your software creates three or more different transmit or receive descriptors in the list before reusing any of the descriptors.
6. Program the Transmit and Receive Ring length registers (DMA_CH(#i)_TxDesc_Ring_Length (for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1) and DMA_CH(#i)_RxDesc_Ring_Length (for i = 0; i <= DWC_EQOS_NUM_DMA_RX_CH-1)). The ring length programmed must be at least 4.
7. Initialize receive and transmit descriptor list address with the base address of the transmit and receive descriptor (DMA_CH(#i)_TxDesc_List_Address (for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1), DMA_CH(#i)_TxDesc_List_Address (for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1)). In addition, you must program the Transmit and Receive tail pointer registers indicating to the DMA about the available descriptors (DMA_CH(#i)_TxDesc_Tail_Pointer (for i = 0; i <= DWC_EQOS_NUM_D-MA_TX_CH-1) and DMA_CH(#i)_RxDesc_Tail_Pointer (for i = 0; i <= DWC_EQOS_NUM_D-MA_RX_CH-1)).
8. Program the following fields to initialize the mode of operation in the MTL_TxQ0_Operation_Mode register:
   a. Transmit Store And Forward (TSF)
   b. Transmit Threshold Control (TTC)
   c. Transmit Queue Enable (TXQEN) to value 2‘b10 to enable Transmit Queue0
   d. Transmit Queue Size (TQS)
9. Enable the interrupts by programming the DMA_CH(#i)_Interrupt_Enable (for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1) register.
10. Repeat steps 4 through 9 for all additional channels of AV feature.

11. Program the CBS control register, idleSlope, sendSlope, hiCredit, and loCredit registers of the AV Queues.
12. Start the Receive and Transmit DMA by setting bit 0 of the DMA_CH(#i)_TX_Control (for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1) register and bit 0 of the DMA_CH(#i)_RX_Control (for i = 0; i <= DWC_EQOS_NUM_DMA_RX_CH-1) register.

20.11.2 Enabling Slot Number Checking

You can enable slot number checking feature if you want to specify the intervals at which the DMA channels mapped to AV Queues fetch the frames from the AHB/AXI system bus.

These steps should be completed after step 11 and before step 12 of "Initializing DMA".

You can use the slot number check feature to specify the intervals at which the DMA Channels mapped to AV Queues fetch the frames from the AHB/AXI system bus. This feature is useful for a uniform and periodic transfer of the AV traffic from the host memory. The feature is available only when you enable timestamping and program the MAC_Sub_Second_Increment register. Complete the following steps to enable the slot number checking:

1. Enable timestamping by following the steps described in "Initialization Guideline for System Time Generation".
2. Make sure that the SLOTNUM field (bits 22:19) of TDES3 Normal Descriptor (Read Format) contains a valid slot number. You can read the current reference slot number from the DMA_CH(#i)_Slot_Function_Control_Status (for i = 0; i <= DWC_EQOS_NUM_DMA_TX_CH-1).
3. Set Bit 0 (ESC) of the Slot Function Control and Status register of a channel to enable the slot number checking.

20.11.3 Enabling Average Bits Per Slot Reporting

You can enable the ability to report the average bits that are transmitted in a slot.

The CBS Status register of the additional AV channels provides information about the average bits that are transmitted in a slot. The software can asynchronously read this register to retrieve information about the average bits transmitted per slot. Complete the following steps to enable average bits per slot reporting:

1. Enable timestamping by following the steps described in the "Initialization Guidelines for System Time Generation" on page 1367.
2. Program Bits [6:4], SLC, of the MTL_TxQ[n]_ETS_Control register of a channel with the number of slots over which the average transmitted bits per slot need to be computed.
3. Enable Bit 9(ABPSSIE) of the MTL_Q[n]_Interrupt_Control_Status register of a channel to generate the average bits per slot interrupt.

Note

- The frequency of this interrupt depends on the value programmed in step 2. For example, when you program value 0 in the SLC field, the interrupt is generated at every 125 microsecond.
- When not required, you can disable this interrupt to stop the interrupt flooding.

4. Read Bits [16:0], ABS, from the MTL_TxQ[n]_ETS_Status register of a channel on each interrupt.

Note The software can read the ABS bits in polling mode even if the ABPSIE bit is not enabled. When high, bit 1 (ABPSIS) of the MTL_TxQ[n]_ETS_Status register indicates that a new value is updated in the ABS field.

20.11.4 Disabling Flow Control for AV Enabled Queues

20.11.4.1 Transmit Flow

- Program the EHFC (Enable Hardware Flow Control) bit of corresponding Rx Queue's MTL_RxQ[n]_Operation_Mode register to 0.

20.11.4.2 Receive Flow Control

- Program the PSTQ[n] field corresponding to AV enabled Tx Queue in MAC_TxQ_Prty_Map0/1 register to 0.

20.12 Programming Guidelines for Energy Efficient Ethernet

After you enable the Energy Efficient Ethernet (EEE) in the DWC_ether_qos controller, you can program the following:

- Entering and Exiting Tx LPI mode during the QoS initialization
- Gating off the CSR clock in the Rx LPI mode
- Gating off the CSR clock in the Tx LPI mode

20.12.1 Entering and Exiting the Tx LPI Mode

Energy Efficient Ethernet (EEE) enables the IEEE 802.3 Media Access Control (MAC) sub layer along with a family of Physical layers to operate in the Low-Power Idle (LPI) mode. In the Transmit path, the software must set the LPIEN bit of the MAC_LPI_Control_Status register to indicate to the MAC to stop transmission and initiate the LPI protocol. You can configure the Energy Efficient Ethernet (EEE) feature in coreConsultant while initializing the core.

Complete the following steps during QoS core initialization:

1. Read the PHY register through the MDIO interface, check if the remote end has the EEE capability, and then negotiate the timer values.
2. Program the PHY registers through the MDIO interface (including the RX_CLK_stoppable bit that indicates to the PHY whether to stop Rx clock in LPI mode.)
3. Program Bits[25:16] and Bits[15:0] in MAC_LPI_Timers_Control Register.
4. Read the link status of the PHY chip by using the MDIO interface and update Bit 17 of MAC_LPI_Control_Status accordingly. This update should be done whenever the link status in the PHY chip changes.
5. Program the MAC_1US_Tic_Counter as per the frequency of the clock used for accessing the CSR slave port.
6. Program the MAC_LPI_Entry_Timer register (LPIET) with the IDLE time for which the MAC should wait before entering the LPI state on its own.
7. Set LPITE and LPITXA (bit[20:19]) of MAC_LPI_Control_Status register to enable the auto-entry into LPI and auto-exit of MAC from LPI state.
8. Program the MAC_1US_Tic_Counter as per the frequency of the clock used for accessing the CSR slave port.
9. Program the MAC_LPI_Entry_Timer register (LPIET) with the IDLE time for which the MAC should wait before entering the LPI state on its own.
10. Set LPITE and LPITXA (bit[20:19]) of MAC_LPI_Control_Status register to enable the auto-entry into LPI and auto-exit of MAC from LPI state.
11. Set Bit 16 of MAC_LPI_Control_Status Register to make the MAC Transmitter enter the LPI state.

    The MAC enters the LPI mode after completing all scheduled packets and remains IDLE for the time indicated by LPIET. It sets the TLPIEN (bit[0]) after entry to LPI state.
12. When a packet is scheduled for transmission (when the TxDMA comes out of IDLE state or when a packet is presented at ATI or MTI interface), the MAC Transmitter exits LPI state automatically. It waits for TWT time before setting the TLPIEX interrupt status bit and then resume the packet transmission.

13. MAC Transmitter re-enters LPI state if it remains IDLE for LPIET time and sets the TLPIEN bit and the entry-exit cycle continues.
14. Reset LPITXEN in case the application wants to over-ride the auto-entry/exit modes and make the MAC Transmitter exit the LPI state directly.

Note

- To make the MAC enter the LPI state only after it completes the transmission of all queued frames in the Tx FIFO, you should set Bit 19 in MAC_LPI_Control_Status Register.
- To switch off the GMII Transmit Clock during the LPI state, use the sbd_tx_clk_gating_ctrl_o signal for gating the clock input.
- To switch off the CSR clock or power to the rest of the system during the LPI state, you should wait for the TLPIEN interrupt of MAC_LPI_Control_Status Register to be generated. Restore the clocks before performing step 6 when you want to come out of the LPI state.

20.12.2 Gating Off the CSR Clock in the LPI Mode

You can gate off the CSR clock to save the power when the MAC is in Low-Power Idle (LPI) mode.

20.12.2.1 Gating Off the CSR Clock in the Rx LPI Mode

The following operations are performed when the MAC receives the LPI pattern from the PHY:

1. The MAC RX enters the LPI mode and the Rx LPI entry interrupt status [RLPIEN interrupt of MAC_LPI_Control_Status Register] is set.
2. The interrupt pin (sbd_intr_o) is asserted. The sbd_intr_o interrupt is cleared when the host reads the MAC_LPI_Control_Status Register.

After the sbd_intr_o interrupt is asserted and the MAC Tx is also in the LPI mode, you can gate-off the CSR clock. If the MAC TX is not in the LPI mode when you gate off the CSR clock, the events on the MAC transmitter do not get reported or updated in the CSR.

For restoring the CSR clock, wait for the LPI exit indication from the PHY after which the MAC asserts the LPI exit interrupt on lpi_intr_o (synchronous to clk_rx_i). The lpi_intr_o interrupt is cleared when MAC_LPI_Control_Status Register is read.

20.12.2.2 Gating Off the CSR Clock in the Tx LPI Mode

The following operations are performed when Bit 16 (LPIEN) of MAC_LPI_Control_Status is set:

1. The Transmit LPI Entry interrupt (TLPIEN bit of MAC_LPI_Control_Status Register) is set.
2. The interrupt pin (sbd_intr_o) is asserted. The sbd_intr_o interrupt is cleared when the host reads the MAC_LPI_Control_Status Register.

After the sbd_intr_o interrupt is asserted and the MAC RX is also in the LPI mode, you can gate off the CSR clock. If the MAC RX is not in the LPI mode when you gate off the CSR clock, the events on the MAC receiver do not get reported or updated in the CSR.

For restoring the CSR clock, switch on the CSR clock when the MAC has to come out of the TX LPI mode. After the CSR clock is resumed, reset Bit 16 (LPIEN) of MAC_LPI_Control_Status Register to bring the MAC out of the LPI mode.

20.13 Programming Guidelines for Flexible Pulse-Per-Second Output

After you enable the Flexible Pulse-Per-Second Output feature in the DWC_ether_qos controller, you can perform the following tasks:

- "Generating Single Pulse on PPS" on page 1374
- "Generating Next Pulse on PPS" on page 1374
- "Generating a Pulse Train on PPS" on page 1374
- "Generating an Interrupt without Affecting the PPS" on page 1375

20.13.1 Generating Single Pulse on PPS

To generate single pulse on PPS:

1. Program 11 or 10 (for interrupt) in Bits [6:5], TRGTMODSEL, of MAC_PPS_Control Register. This instructs the MAC to use the Target Time registers (MAC_PPS0_Target_Time_Seconds and MAC_PPS0_Target_Time_Nanoseconds) for start time of PPS signal output.
2. Program the start time value in the Target Time registers (MAC_PPS0_Target_Time_Seconds and MAC_PPS0_Target_Time_Nanoseconds).
3. Program the width of the PPS signal output in MAC_PPS(#i)_Width (for i = 0; i <= DWC_EQOS_PPS_OUT_NUM-1) Register.
4. Program Bits [3:0], PPSCMD, of MAC_PPS_Control to 0001. This instructs the MAC to generate single pulse on the PPS signal output at the time programmed in the Target Time registers.

20.13.2 Generating Next Pulse on PPS

When the PPSCMD is executed (PPSCMD bits = 0), you can cancel the pulse generation by giving the Cancel Start Command (PPSCMD=0011) before the programmed start time elapses. You can also program the behavior of the next pulse in advance.

To program the next pulse, follow this procedure:

1. Program the start time for the next pulse in the Target Time registers. This time should be more than the time at which the falling edge occurs for the previous pulse.
2. Program the width of the next PPS signal output in MAC_PPS(#i)_Width (for i = 0; i <= DWC_EQOS_PPS_OUT_NUM-1).
3. Program Bits [3:0], PPSCMD, of MAC_PPS_Control to generate a single pulse after the time at which the previous pulse is de-asserted. This instructs the MAC to generate single pulse on the PPS signal output, at the time programmed in Target Time registers.

   If you give this command before the previous pulse becomes low, then the new command overwrites the previous command and the QOS may generate only 1 extended pulse.

20.13.3 Generating a Pulse Train on PPS

Complete the following steps to generate a pulse train on PPS:

1. Program 11 or 10 (for interrupt) in Bits [6:5], TRGTMODSEL, of MAC_PPS_Control Register. This instructs the MAC to use the Target Time registers for start time of the PPS signal output.
2. Program the start time value in the Target Time registers.

3. Program the interval value between the train of pulses on the PPS signal output inMAC_PPS(#i)_Interval (for i = 0; i <= DWC_EQOS_PPS_OUT_NUM-1) Register.
4. Program the width of the PPS signal output in MAC_PPS(#i)_Width (for i = 0; i <= DWC_EQOS_PPS_OUT_NUM-1) Register.
5. Program Bits[3:0], PPSCMD, of MAC_PPS_Control Register to 0010. This instructs the MAC to generate train of pulses on the PPS signal output with start time programmed in Target Time registers.

   By default, the PPS pulse train is free-running unless stopped by 'STOP Pulse train at time' or 'STOP Pulse Train immediately' commands.
6. Program the stop value in the Target Time registers. Ensure that Bit 31 (TSTRBUSY) of MAC_PPS(#i)_Target_Time_Nanoseconds (for i = 0; i <= DWC_EQOS_PPS_OUT_NUM-1) Register is reset before programming the Target Time registers again.
7. Program the PPSCMD field (bit 3:0) of MAC_PPS_Control to 0100. This stops the train of pulses on PPS signal output after the programmed stop time specified in Step 6 elapses.

You can stop the pulse train at any time by programming 0101 in the PPSCMD field. Similarly, you can cancel the Stop Pulse train command (given in Step 7) by programming 0110 in the PPSCMD field before the time (programmed in Step 6) elapses. You can cancel the pulse train generation by programming 0011 in the PPSCMD field before the programmed start time (in Step 2) elapses.

20.13.4 Generating an Interrupt without Affecting the PPS

The Bits [6:5], TRGTMODSEL, of the MAC_PPS_Control Register enable you to program the Target Time registers to do any one of the following:

- Generate only interrupts.
- Generate interrupts and the PPS start and stop time.
- Generate only PPS start and stop time.

Complete the following steps to program the Target Time registers to generate only interrupt event:

1. Program 00 (for interrupt) in Bits [6:5], TRGTMODSEL, of MAC_PPS_Control Register. This instructs the MAC to use the Target Time registers for target time interrupt.
2. Program a target time value in the Target Time registers. This instructs the MAC to generate an interrupt when the target time elapses.

   If Bits [6:5], TRGTMODSEL, are changed (for example, to control the PPS), then the interrupt generation is over-written with the new mode and new programmed Target Time register value.

20.14 Programming Guidelines for TSO

The TCP Segmentation Offload (TSO) engine is used to offload the TCP segmentation functions to the hardware. To program the TSO, set the TSE bit to enable TCP packet segmentation, and program descriptor fields to enable TSO for the current packet.

Complete the following steps to program TSO:

1. Program the TSE bit of corresponding DMA_CH[n]_Tx_Control register to enable TCP packet segmentation in that DMA.
2. In addition to the normal transfer descriptor setting, the following descriptor fields must be programmed to enable TSO for the current packet:
   a. Enable TSE in Bit 18 of TDES3
   b. Program the length of the un-segmented TCP/IP packet payload in bits [17:0] of TDES3 and the TCP header in bits [22:19] of TDES3.
   c. Program the maximum size of the segment in MSS of DMA_CH[n]_Control register or MSS in the context descriptor.

      If MSS field is programmed in both DMA_CH[n]_Control register and in the context descriptor, the latest software programmed sequence is considered.
3. The header of the unsegmented TCP/IP packet should be in Buffer 1 of the first descriptor and this buffer must not hold any payload bytes. The payload is allocated to Buffer 2 and the buffers of the subsequent descriptors.

Note If TSE is enabled in TDES3 for a non-TCP-IP packet, the result is unpredictable.

20.15 Programming Guidelines for UFO

DWC_ether_qos supports fragmentation of UDP packets into smaller IPv4 fragments.

Enable the IP fragmentation by programming TSE_MODE ([14:13]) bit of DMA_CH0_TX_CONTROL register as follows:

- Set to 2’b01 to enable fragmentation of UDP over IPv4 with checksum
- Set to 2’b10 to enable fragmentation of UDP over IPv4 without checksum

Note In both the cases, if the TSE ([12]) bit is set to 1 for a TCP packet, DMA enables TCP segmentation; fragmentation is not enabled, because fragmentation is supported only for UDP packets

For more details, see the description of DMA_CH0_TX_CONTROL Register in the “Register Descriptions” on page 605

20.15.1 Software Guidelines

When the software creates Transmit Descriptor of the UDP/IPv4 packet that is to be fragmented, the software must:

1. Set the desired Fragment Size (MFS) in the MSS field of TDES2 of the Context Descriptor. The MFS must be set only once, unless the MFS needs to be changed (for TCP packet targeted for segmentation). The MFS must be in multiple of 8-bytes for IP fragmentation.
2. Set TSE bit of TDES3 of normal descriptor.
3. Program THL = 2 in TDES3 of normal descriptor.
4. Program the correct value in the TPL field of TDES3 of Normal Descriptor. This value should be equal to Length of UDP Header + UDP payload.

Note

- Enable fragmentation only for IPv4; not for IPv6.
- Set the DF Flag (Do Not Fragment) in the IPv4 Header packet to 0.
- Set the Fragmentation Offset in the IPv4 Header packet to 0.

When the input packet is UDP over IPv6, do not enable IP fragmentation. Set the TSE bit to 0 in the TDES3 of normal descriptor and let the software perform the fragmentation.

20.16 Programming Guidelines for Header Payload Split Receive

The Header-Payload Split support is an optional feature that can be enabled by

1. Setting the SPH field in the DMA_CH#_Control register.
2. Selecting the Header-Payload split option by setting the SPLM field of MAC_Ext_Cfg1 register.

The DMA can process the header and payload of the received packets separately.

Note In case of L3/L4 header-payload split, if the IPv4 header length or IPv6 payload length fields are corrupt, the split might not always happen at the correct boundary

20.17 Programming Guidelines for VLAN Filtering on Receive

Complete the following steps to program VLAN filtering on receive:

1. Program MAC_VLAN_Tag register for the following bit to select the filtering method:
   a. ETV: Enable 12-Bit VLAN Tag Comparison or 16-bit VLAN Tag comparison.
   b. VTHM: VLAN Tag Hash Table Match Enable.
   c. ERIVLT: Enable inner VLAN Tag or outer VLAN Tag (to enable the inner or outer VLAN Tag filtering, Double VLAN Processing should enabled by setting EDVLP)
   d. ERSVLM: Enable Receive S-VLAN Match or C-VLAN match (for S-VLAN processing to be enabled, set ESVL)
   e. DOVLTC: Ignores VLAN Type for Tag Match
   f. VTIM: to enable VLAN Tag Inverse Match instead of the normal VLAN Tag matching
2. Program VL of MAC_VLAN_Tag Register for the 12-bit or 16-bit VLAN tag.
3. If Hash filtering of VLAN tag is enabled, program MAC_VLAN_Hash_Table Register. When ETV bit is reset, upper 4 bits of the calculated CRC-32 of VLAN Tag are inverted and used to index the content of MAC_VLAN_Hash_Table register. When ETV bit is set, upper 4 bits of calculated CRC-32 of VLAN Tag are used to index the content of MAC_VLAN_Hash_Table register. For example, when ETV bit is set, a hash value of 4b'1000 selects Bit 8 of the VLAN Hash table. When ETV bit is reset a hash value of 4'b1000 selects Bit 7 of the VLAN Hash table.

20.18 Programming Guidelines for Extended VLAN Filtering and Routing on Receive

For the indirect access of the per VLAN Tag registers, follow these steps:

- Write
  - Write the required data into the MAC VLAN Tag Data register.
  - Program the VLAN Tag Control Register's OFS field with the required Filter Register's offset and command type to the CT field. For a write command, set this bit to 0.
  - Write 1 to the OB field and wait till the OB bit is reset to do the next write. This will guarantee that the appropriate VLAN Tag Filter Register has been programmed.
- Read
  - Program the VLAN Tag Control Register's OFS Field with the required register's offset and command type to the CT field. For a read command, set this bit to 1.
  - Write 1 to the OB field and wait till the OB bit is reset. The appropriate VLAN Tag Filter register's value is available in the MAC VLAN Tag Data Register.

20.19 Programming Sequence for Queue/Channel Based VLAN Inclusion Register

To program the queue/channel-based VLAN inclusion register, complete the following steps:

Note Indirect VLAN include registers are not accessed while setting the CBTI bit.

1. Set the CBTI bit of MAC_VLAN_Incl register to 1, to enable queue/channel based VLAN Tag insertion on all transmitted packets. This bit must be set before any indirect access to the queue/channel specific MAC_VLAN_Incl(#i) register.
2. Program the VLAN Tag and VLAN Type to be inserted in packets from a particular queue/channel in VLT and CSVL fields of MAC_VLAN_Incl register; corresponding offset address in ADDR field (0 for queue/channel 0, 1 for queue/channel 1, and so on) must be set. Set the RDWR bit to 0 to indicate write access. The write to byte 0 (byte 3 in Big Endian mode) of MAC_VLAN_Incl register initiates access to indirect access MAC_VLAN_Incl(#i) register.
3. The BUSY bit of MAC_VLAN_Incl register is set by DWC_ether_qos to indicate the progress of access to indirect access MAC_VLAN_Incl(#i) register. On completion of the access, the BUSY bit is cleared. The application must not attempt subsequent access to MAC_VLAN_Incl(#i) register when the BUSY bit is 1.
4. Repeat step 2 and step 3 to program VLAN Tag and VLAN Type to be inserted in packets from the remaining queues/channels. The application must ensure that the required VLAN Tag and VLAN Type for all the queues/channels are programmed; otherwise unintended VLAN Tag and VLAN Type might be inserted.

20.20 Programming Guidelines for EST

Program the gate control values and time intervals in the Software Owned Gate Control List (SWOL) along with the other EST related registers that are described in "Register Descriptions" on page 605to appropriate values. The following sub-sections provide step by step details for programming the GCL and the other EST related registers.

20.20.1 Programming the GCL and GCL Linked Registers

Follow these steps to program the Gate Control List (GCL) and the four other registers implemented per GCL.

1. The GCL and the four other GCL-linked registers should be accessed via indirect addressing using the MTL_EST_GCL_Control and MTL_EST_GCL_Data registers. The SWOL bit of the MTL_EST_Status register indicates if GCL0 or GCL1 is owned by software.
2. To program the GCL, write the 32-bit write data to MTL_EST_GCL_Data register. Then program TL_EST_GCL_Control register to write the write address and other control information.
3. In the MTL_EST_GCL_Data register, Write Data consists of up to 8 bits (configurable) of Gate Controls and up to 24 bits (configurable) of Time Interval. Gate Close is indicated by programming a '0' and Gate Open is indicated by programming a '1'. For a 4-TC and 20-bit Time Interval Configuration, the data width is 24-bits and the remaining 8-bits are reserved/read-only. The data should be written in the following format.
   ```
   {8'h0, TC3, TC2, TC1, TC0, 20-bit Time Interval} where TCx = 0 or 1.
   ```
4. MTL_EST_GCL_Control register: Program the SRWO bit to 1 (to start a Write Op) and program the address and R/W fields appropriately.
5. Poll and check for the SRWO bit to be cleared by hardware to indicate the completion of the previous operation before initiating a new R/W operation via the same indirect addressing mode.
6. Repeat steps 3, 4, 5 until the programming of the GCL is completed.
7. Using the same indirect addressing method described above, program the BTR, CTR, TER and LLR registers. Set the GCRR bit in the MTL_EST_GCL_Control register appropriately. The GCRR bit interprets the address field as belonging to these registers (instead of the GCL).

After programming of the GCL and the related registers, program the MTL_EST_Control register to allow hardware to own and process the GCL.

When the List Length (as indicated in LLR) is 1, the associated Time Interval should be smaller than the value of the Cycle Time Register. Otherwise, an error is reported (as detailed in the Error handling section) as a single set of gate controls add no value in the TSN context.

Note The time unit in all the GCL related registers is Seconds and Nano-seconds. In cases where internally generated PTP System Time is used, the nano-seconds field must be programmed to use the Digital Rollover mode. (TSCTRLSSR field of MAC_Timestamp_Control must be set to value '1')

20.20.2 Programming the EST Registers

After completing the steps mentioned in "Programming the GCL and GCL Linked Registers" , program the MTL_EST_Control register.

1. Set the Current Time Offset Value and Time Internal Left Shift fields of the MTL_EST_Control register appropriately. Also, set the Enable EST (EEST) and Switch to SWOL (SWOL) bits.

Note

The CTOV recommendations for GMII for SPRAM configurations are:

- 96 * Tx clock period, for 32 and 64 bit data width configurations.
- 128 * Tx clock period for 128 bit data width configurations.

The CTOV recommendations for GMII for non-SPRAM configurations are:

- 30 * Tx clock period, when SA/VLAN insertion is enabled.
- 22 * Tx clock period, when SA/VLAN insertion is not enabled.

2. This will enable the hardware to own and process the new GCL and make a switch to the new GCL at the BTR value. The hardware provides an interrupt (if enabled) when the switch to the new list happens.
3. Appropriate action has to be taken by software to address any other interrupts (explained in the Interrupts section) received during the hardware execution of the GCL.

Set the SSWL (Switch to Software Owned List) bit in the MTL_EST_Control register to handoff to hardware. Software is not allowed to write to the GCL and GCL linked registers when the SSWL bit is set, because the new GCL might be in use by the hardware.

The SSWL is reset/cleared by the hardware when it successfully switches to the new list. The hardware also flips the SWOL bit to indicate the new GCL that the software owns.

To install a new GCL, program the GCL it owns (indicated by SWOL bit) as described in "Programming the GCL and GCL Linked Registers" and then program the MTL_EST_Control register as described above. Ensure that the new BTR is set to an appropriate value to avoid BTRError that might need software intervention in some cases.

To avoid transmission overruns, the packet length (frame size) information should be available at all times. Therefore, in the DMA configurations, program the packet length in the first descriptor of every Tx frame. Similarly, in the MTL configuration, provide the packet length in the control word.

Note

To avoid transmission overruns, the Packet Length provided in the Transmit Descriptor should account for the SA and VLAN insertion, if applicable, that is, Packet Length should represent the actual packet length on the Ethernet line.

20.21 Programming the Launch Time in Time-Based Scheduling

The Launch Time is programmed in the Enhanced Normal Transmit Descriptors in DMA configurations and is driven as a control word in MTL configurations as follows:

The OSTC and Launch Time features are mutually exclusive and should not be used together; in case of a conflict (if OSTC = LTV = 1 in MTL configuration), priority is given to OSTC and the Launch Time is ignored.

In DMA configuration, if a context descriptor is received with a valid OSTC values immediately before receiving a first normal descriptor with LTV bit set, then the LTV is ignored.

20.22 Programming Guidelines for Media Clock Generation and Recovery

20.22.1 Programming Guidelines for Media Clock Generation

1. To set the PPS instance in Media Clock generation mode, program the appropriate Presentation Time Control (supported generation modes "1001-1011") to PPSCTRL_PPSCMD (for 0th instance)/PPSCMD#i (for 1,2,3 instances) of MAC_PPS_Control register.
2. Based on the selected PPS instance, application must drive the appropriate trigger signal to the corresponding mcg_pst_trig_i[#i].
3. Based on the programmed mode, DUT captures the timestamp and programs it in the MAC_PPS(#i)_Target_Time_Seconds register. Then, mcgr_dma_req_o[#i] is set.
4. For every request generated by the DUT, DWC_ether_qos reads the corresponding MAC_PPS(#i)_Target_Time_Seconds register and acknowledge the read request by asserting the corresponding mcgr_dma_ack_i[#i].

20.22.2 Programming Guidelines for Media Clock Recovery

- Enable the Current Presentation Time (CPT) Counter by setting the PTGE field of MAC_Timestamp_Control register. In addition to programming the initialization values for the System Time, update the MAC_Presn_Time_Updt register with the equivalent Presentation Time init value. Only then the TSINIT field of MAC_Timestamp_Control register is set.
- The Increment values used for the System time is also used for the Current Presentation time. This is because, the increment value is in Sub-seconds and Sub-nanoseconds.
- When an update is needed, along with programming the update values for the System Time, the MAC_Presn_Time_Updt register should be updated with the equivalent Presentation Time update value (32-bit ns). Finally, TSUPDT field of MAC_Timestamp_Control register must be set for the update to happen.
- Program the MCGREN#i of MAC_PPS_Control register to enable the PPS instance to operate in MCGR mode.
- To set the PPS instance in Media Clock recovery mode, program the appropriate Presentation Time Control (supported recovery modes "0001 - 0011") to PPSCTRL_PPSCMD (for 0th instance)/PPSCMD#i (for 1,2,3 instances) of MAC_PPS_Control register.
- After PPS instance is set in recovery mode, corresponding mcgr_dma_req_o[#i] is set by the DUT. For every request generated by the DUT, program the target presentation time into the MAC_PPS(#i)_Target_Time_Seconds register and acknowledge the request by asserting the corresponding mcgr_dma_ack_i[#i]. Acknowledgment can be given before/after programming the Target presentation time
- Observe the recovered media clock on the corresponding ptp_pps_o[#i].

20.23 Programming Guidelines for ECC Protection for Memories

Programming guidelines for the ECC protection for memories:

- Enable the ECC feature for the respective memory by setting the appropriate bit in MTL_ECC_Control register.
- A correctable interrupt (sbd_sfty_ce_intr_o) or an uncorrectable interrupt (sbd_sfty_ue_intr_o) are generated to indicate to the application if any correctable/uncorrectable/address errors are detected. Appropriate status is indicated in the DMA/MTL_ECC_Interrupt_Status register.
- Debug mode is provided for each memory to specify errors

Note Enable the ECC feature before the traffic is online (or after the reset). Otherwise false interrupts might be triggered.

20.23.1 Programming Guidelines for ECC Hardware Error Injection (Debug Mode)

Follow these steps for ECC Hardware Error Injection:

- Enable the ECC error injection feature for MTL TX/RX and DMA TSO, by setting the appropriate bit in the MTL_DBG_CTL register. For more details about accessing memory in debug mode, see "Accessing Memory In Slave/Debug Mode" on page 161.
- Enable the ECC error injection feature for the EST memory by setting the appropriate bit in the EST_G-CL_Control register.
- Enable the ECC error injection for RX Parser memory by setting the appropriate bit in the MTL_RX-P_Indirect_Acc_Control_Status register. To access Rx Parser memory in debug mode, disable the Rx Parser feature by setting the FRPE bit of MTL_Opeartion_Mode register to 0.
- Where multiple CSR writes are required, for writing single data word into the memory, the application should ensure that all the CSR writes corresponding to one memory write should have the same value for the error injection control word.

20.24 Programming Guidelines for On-Chip Datapath Parity Protection

Follow these steps for on-chip datapath parity protection:

- Set the EDPP bit of MTL_DPP_Control register to enable the generation of parity and detection of parity Mismatch on datapath. An odd parity generation/detection can be enabled by setting the OPE bit of MTL_DPP_control register else by default even parity generation/detection will be active.
- An uncorrectable interrupt (sbd_sfty_ue_intr_o) is generated ti indicate to the application if any parity Mismatches are detected and the appropriate sattaus will be indicated in DMA/MTL_Safety_Interrupt_status and MAC_DPP_FSM_Interrupt_Status register.
- Debug mode is supported for each parity generator, to insert error. For details of how to insert errors, see, "Parity Error Injection" on page 410.

Note To avoid false safety interrupts, enable the datapath parity protection before the start of Rx or Tx traffic.

20.25 Programming Guidelines for FSM Parity and Timeout

Follow these steps to program the FSM parity and timeout

- Set the PRTYEN and TMOUTEN fields of MAC_FSM_Control register to enable the FSM parity and timeout feature respectively
- For parity error detection, force the FSMs to enter into a state with even number of 1s and wait for the safety uncorrectable interrupt to be set with FSMPES of MAC_DPP_FSM_Interrupt_Status.
- For Error Injection mode in FSM parity, select the appropriate clock domain for which parity error injection should be set using the [23:16] bits of MAC_FSM_Control register. Wait for the safety uncorrectable interrupt to be set with FSMPES of MAC_DPP_FSM_Interrupt_Status.
- For FSM Timeouts, program the large and normal mode values indicated by LTMRMD and NTMRMD of MAC_FSM_ACT_Timer register.
- Select large or normal mode tic generation per clock domain by using the [31:24] bits of MAC_FSM_-Control Register.
- Set the TMR field of the MAC_ACT_Timer field with the appropriate number of CSR clock cycles used to generate 1us tic.
- Force the FSM in a particular clock domain to be in an active state for Timeout duration (for example, T-2T). Check for the safety interrupt and the status bit for the corresponding clock domain getting set in [15:8] bits of MAC_DPP_FSM_Interrupt_Status register.
- For Error Injection mode in FSM timeout, select the appropriate clock domain for which timeout error injection must be set using the [15:8] bits of MAC_FSM_Control register. Wait for the safety uncorrectable interrupt along with the corresponding error status set in MAC_DPP_FSM_Interrupt_Status.
- For application / CSR interface timeout, only normal mode tic generation is used. Based on the timeouts of configured interface, appropriate fields (MSTTES, SLVTES) of MAC_DPP_FSM_interrupt_Status is set along with the safety interrupt.

A

Area and Power

This appendix provides area in terms of gate count, area differences because of scan ready and clock gating, and power dissipation estimates with and without clock gating. It contains the following sections:

A.1 Area

A.1.1 Gate Count Summary

This section summarizes the area in terms of gate count for the various modules/configurations. Gate count for the DWC_ether_qos is calculated with the following parameters:

Design Compiler Version:	L-2016.03-SP4
Technology synthesized for:	28nm
Application clock:	400 MHz
PTP Reference Clock:	50 MHz
CSR Slave Clock:	200 MHz
MII/GMII clock:	67.5/312.5 MHz for 2.5Gbps
Gate size:	NAND2x1
Synthesis Strategy:	Synopsys Design Compiler Reference Methodology (DCRM)

Note The area numbers provided are without clock-gating and scan.

Table A-1 gives the gate count numbers in 32-bit data bus configuration.

Table A-1 Area for 32-Bit Data Bus Configurations

Features	Area in Kgates (NAND2)					
	EQOS-CORE	**EQOS-MTL**	**EQOS-DMA**	**EQOS-AHB**	**EQOS-AXI**	**EQOS-AXI4**
Minimum Configuration (FD, 10/100M or 1G only, APB, Drop TxStatus, DPRAM, No optional features)	15.3	23.8	39.0	42.5	48.6	50.6
Application Master Interface						
Both Endianess	0.2					
Address-width (40-, 48-bit)	NA				2.1, 4.3	
AXI Outstanding Read Req (8, 16, 32)					0.5, 1.2, 2.8	
AXI Outstanding Write Req (8, 16, 32)					1.0, 3.0, 6.5	
CSR Port interface						
AHB (32-, 64-, 128-bit)	NA			0.5, 0.8, 1.4		
AXI4-LITE	NA				1.7	

Features	Area in Kgates (NAND2)					
	EQOS-CORE	EQOS-MTL	EQOS-DMA	EQOS-AHB	EQOS-AXI	EQOS-AXI4
AXI (32-, 64, 128-bit)	NA				2.1, 3.2, 5.1	
Different Clock for CSR (Per TX+RX)	NA	**0.4**	**5.0**	**5.6**	**7.4**	**8.7**
Enable Back2Back CSR writes	3.0	7.0				
General Features						
Add 10/100/1G support	0.50	1.00				
Half-Duplex Support	2.5					
Double VLAN	0.7	0.7 or 2.2 with SPRAM				
SA-VLAN Insrt	1.3	1.3 or 2.8 with SPRAM				
Queue/Channel Based VLAN Tag Insertion (Per TxQ)	NA	**0.2**				
Buffer Management						
Max RxFIFO Size (256KB)	NA	2.0				
Max TxFIFO Size (128KB)	NA	1.0				
Single Port RAM (SPRAM)	NA	8.5				
Debug Memory Access	NA	1.7				
No Drop TxStatus - 2 packets (per TxQ with 1588)	NA	**2.1**				
No Drop TxStatus - 8 packets (per TxQ with 1588)	NA	7.5	NA			
PHY Interfaces (All together)	**12.0**					
RGMII	0.5					
RMII	0.5					
SMII	1.0					
RevMII	1.8					
SGMII or RTBI	4.0					
TBI	5.5					
SMA (MDIO)	1.0					
Filtering Features						
Each Additional MAC Address Register (1 - 31)	**0.8**					
CDC Sync each additional MAC Address Register (1 - 31)	**0.5**					

Features	**Area in Kgates (NAND2)**					
	EQOS-CORE	**EQOS-MTL**	**EQOS-DMA**	**EQOS-AHB**	**EQOS-AXI**	**EQOS-AXI4**
Additional 32 MAC Address Registers	17.5					
CDC Sync Additional 32 MAC Address Registers	13.5					
Additional 64 MAC Address Registers	35.0					
CDC Sync Additional 64 MAC Address Registers	27.0					
DA Hash Filter (64, 128, 256)	1, 1.7, 3.0					
CDC Sync Hash Filter (64, 128, 256)	0.6, 1.1, 2.3					
VLAN Hash Filter	0.3					
Extended VLAN Filter (per set of 4 filters)	**1.2**					
Each L3-L4 Filter	**3.0**					
CDC Sync each L3-L4 filter	**1.4**					
Flexible Receiver Parser	**55**					
Flexible Receiver Parser with 64 Bytes header parsing	NA	NA	30K			
Each additional 64 byte parsing	10k					
IEEE 1588 Timestamping						
External System Time Source	7	7 (13.5 with SPRAM)				
Internal/both System Time Source	11.5	11.5 (17 with SPRAM)				
IEEE 1588 Timestamping - Higher Word	0.3					
IEEE 1588 Timestamping - Sub-Nanosecond	1.0					
Aux Snapshot (FIFO = 4, 8, 16)	3.3, 5.5, 11.5					
Each Flexible PPS	**3.9**					
PTP Offload	14.5					
One-Step Timestamping for PTP over Ethernet	6.5	6.5 (7.5 with SPRAM)				
One-Step Timestamping PTP over UDP/IP	3.5	8.0	8.5			
For SPRAM, Add to above row	NA	1.0				
Time Sensitive Networking Features						
MAC Frame Preemption	NA	25				
802.1Qbv Time Aware Shaper (EST)	NA	20.0				

Features	Area in Kgates (NAND2)					
	EQOS-CORE	EQOS -MTL	EQOS -DMA	EQOS -AHB	EQOS -AXI	EQOS -AXI4
Time Based Scheduling (TBS) without EST - per TxQ	NA	**1.5**	**2.5**			
Time Based Scheduling (TBS) with EST - 2 TxQ	NA	17.0	29.0			
Time Based Scheduling (TBS) with EST - per additional TxQ	**NA**	**5.0**	**8.0**			
TCP/IP Offloading Features						
Receive COE	2.0					
Transmit COE (Per TxQ)	NA	**7.6**				
TSO (Per TxDMA)	NA		**10.0**			
TSO - Dedicated Memory (with Debug Access)	NA		1.5 (4.0)			
Split Header in RX path	NA		4 (5 with SPRAM)			
IPv4 ARP Offload	2.5					
Other Optional Features						
GPIO	0.5					
Energy Efficient Ethernet (EEE)	2.0					
PMT-Magic packet	2.5					
PMT-Remote Wakeup (4-, 8-, 16-filters)	7.5, 13, 24					
ARP Offload	2.5					
RMON/MMC Counters - All 32-bit Tx Counters (28 nos)	14.0					
RMON/MMC Counters - All 32-bit Rx Counters (28 nos)	14.0					
RMON/MMC Counters - All 32-bit Rx IPC Counter (28 nos)	15.0					
Area Reduction per 16-bit counter	**-0.2**					
Multi-Queue, Multi-Channel Features						
Packet Duplication (32, 64, 128 MAC Addresses)	NA		2.5, 5.0, 10.0			
DCB (1TX + 2RX Q)	2.5	5.8				
DCB (2TX + 2RX Q)	3.3	15.9	23.3 (Incld additional TxDMA)			
DCB - Every Additional RxQ	**1.0**	**3.0**				
DCB - Every Additional TxQ	**0.2**	**5.0**	**12.5 (Incld additional TxDMA)**			
AVB (2TX + 1RX Q)	NA	16.0	24.0 (Incld additional TxDMA)			

Features	**Area in Kgates (NAND2)**					
	EQOS-CORE	**EQOS -MTL**	**EQOS -DMA**	**EQOS -AHB**	**EQOS -AXI**	**EQOS -AXI4**
AVB - every additional TxQ	NA	**6.8**	**15.0 (Incld additional TxDMA)**			
AVB - every additional RxQ	**1.7**	**5.0**				
For SPRAM option, Reduce per TxQ	NA	**-0.5**				
For SPRAM option, Reduce per RxQ	NA	**-2.0**				
DCB + AVB - 2TX + 2RX Q	6.0	29.0	38.0		40.0	
Each additional TxQ with AVB and DCB	**0.2**	**9.1**	**17.0 (Incld additional TxDMA)**			
Each additional TxQ w/o AVB or DCB	NA	**8.5**	**16.0 (Incld additional TxDMA)**			
Each additional RxDMA Channel	NA		**7.2**			
Automotive Safety Features						
ECC Support for all external memories	NA	8.5	11.5			
Other Safety features (DPP, FSM, Timeouts)	NA	9.7	17.2			
Maximum Configuration (1 TX + 1 RX) with SPRAM	**345.0**	**394.6**	**426.4**	**429.0**	**457.5**	**459.6**
Maximum Configuration (8 TX + 8 RX) with SPRAM	**358.0**	**759.0**	**1037**	**1040**	**1078**	**1082**

To estimate the area for your target configuration,

1. Take the area of the minimum target configuration (EQOS-CORE or EQOS-MTL, etc) with 32-bit in the first row.
2. Add the area of the optional features selected in your configuration.
3. For features such as multiple DMA, Queues, address filters and so on, multiply the per-item area (bold) with the selected number.
4. If your target configuration is 64-bit or 128-bit, add the corresponding area increase factor for relevant features as given in Table A-2 and Table A-3 respectively.

Note The calculated estimated area might have about +/- 5% variation with respect to the actual area based on the target technology library/operating conditions.

Table A-2 gives the increase in approximate area because of a 64-bit data bus configuration (in affected blocks only, with respect to the numbers given in Table A-1).

Table A-2 Area for 64-Bit Data Bus Configurations

Feature	Area in Kgates (NAND2)				
	EQOS-CORE	**EQOS -MTL**	**EQOS -DMA**	**EQOS -AHB**	**EQOS -AXI/4**
Minimum Configuration	0.7	3.8	6.8	7.5	8.0
Both Endianess	NA		0.4		
Debug Memory Access	NA	0.4			
SPRAM		3.0			
IEEE 1588 Timestamping with SPRAM	NA	1.0			
Split Header in Receive Path	NA		0.6 with SPRAM		
Each Transmit COE	NA	**3.2**			
Each TSO	NA		**3**		
TSO - Dedicated Memory (Debug Access)	NA		2.5		
First additional TxQ (AVB or DCB)	NA	1.8	3.5		
Rest additional TxQ (AVB or DCB)	NA	**0.5**	**2.2**		
Each Additional RxQ	NA	**0.3**			
For SPRAM option, Reduce per TxQ	NA	**-0.5**			
For SPRAM option, Reduce per RxQ	NA	**-0.5**			
Each additional RxDMA Channel	NA		**1.4**		
AXI Master Interface (for 8TX + 8RX)	NA				4.0
Automotive Safety Features					
ECC Support for all external memories	NA	10.1	14.0		
Other safety features (DPP, FSM, timeouts)	NA	12.3	24.5		
Total Maximum Configuration (1 TX + 1 RX)	**346.5**	**404.2**	**442**	**445**	**478.3**
Total Maximum Configuration (8 TX + 8 RX)	**359**	**795**	**1105**	**1110**	**1155**

Table A-3 gives the increase in approximate area because of a 128-bit data bus configuration (in affected blocks only, with respect to the numbers given in Table A-1).

Table A-3 Area for 128 Bit Data Bus Configurations

Feature	Area in Kgates (NAND2)				
	EQOS-CORE	**EQOS -MTL**	**EQOS -DMA**	**EQOS -AHB**	**EQOS -AXI/4**
Minimum Configuration	1.7	12.3	21.5	23.0	26.0

Feature	**Area in Kgates (NAND2)**				
	EQOS-CORE	**EQOS-MTL**	**EQOS-DMA**	**EQOS-AHB**	**EQOS-AXI/4**
Both Endianess	NA		0.8		
Debug Memory Access	NA	1.2			
SPRAM		8.0			
IEEE 1588 Timestamping with SPRAM	NA	2.0			
Split Header in Receive Path	NA		2.5 with SPRAM		
Each Transmit COE	NA	**9.7**			
Each TSO	NA		**9.0**		
TSO - Dedicated Memory (Debug Access)	NA		6.9		
First additional TxQ (AVB or DCB)	NA	**5.2**	**9.4**		
Rest Additional TxQ (AVB or DCB)	NA	**1.7**	**6**		
Each Additional RxQ	NA	**0.6**			
For SPRAM option, Reduce per TxQ	NA	**-1.5**			
For SPRAM option, Reduce per RxQ	NA	**-2.8**			
Each additional RxDMA Channel	NA		**5.7**		
AXI Master Interface (for 8TX + 8RX)	NA				10.5
Automotive Safety Features					
ECC Support for all external memories	NA	11.6	21.5		
Other safety features (DPP, FSM, timeouts)	NA	16.4	39.8		
Total Maximum Configuration (1 TX + 1 RX)	**348.1**	**427.8**	**480**	**483.5**	**522.6**
Total Maximum Configuration (8 TX + 8 RX)	**360**	**885**	**1271**	**1280**	**1328**

A.2 Power and Area Differences Due to Scan Ready, Clock Gating, and Low Power Mode

Table A-4 shows the power and area differences because of Scan Ready, Clock Gating and Low-Power Modes.

Table A-4 Power and Area Differences Because of Scan Ready, Clock Gating and Low-Power Modes

Sample Configurations for Estimating Area and Power								
Main Parameter	Configuration 1		Configuration 2			Configuration 3		
Features	■ 10/100/1G speeds		■ 10/100/1G speeds			■ 10/100/1G speeds		
	■ Async Reset		■ Async Reset			■ Async Reset		
	■ System interface = AHB		■ System interface = AHB			■ System interface = AHB		
	■ 32-bit Little Endian		■ 32-bit Little Endian			■ 32-bit Little Endian		
	■ MTL Rx/Tx FIFO size = 2048 bytes		■ MTL Rx/Tx FIFO size = 2048 bytes			■ MTL Rx/Tx FIFO size = 2048 bytes		
	■ PHY Interface = MII/GMII		■ PHY Interface = MII/GMII			■ PHY Interface = MII/GMII		
	■ 1 MAC Address		■ 1 MAC Address			■ 1 MAC Address		
	■ Optional modules = SMA		■ Optional modules = SMA, PMT (with Magic packet)			■ Optional modules = SMA, PMT (with 4 PMT RWK filters), ARP, EEE		
Operation	Normal Operation		Normal Operation		Low Power Mode	Normal Operation		Low Power Mode
Technology	Industry Standard 40LP	Industry Standard 28	Industry Standard 40LP	Industry Standard 28	Industry Standard 40LP	Industry Standard 40LP	Industry Standard 28	Industry Standard 40LP
Gate Size	NAND2X1 cells (area 0.7056)	NAND2X1 cells (area 0.50625)	NAND2X1 cells (area 0.7056)	NAND2X1 cells (area 0.50625)	NAND2X1 cells (area 0.7056)	NAND2X1 cells (area 0.7056)	NAND2X1 cells (area 0.50625)	NAND2X1 cells (area 0.7056)
Application Clock Frequency	400 MHz							
PHY Clock Frequency	125 MHz							

Sample Configurations for Estimating Area and Power									
No Clock Gating and No Scan	Gate Count (Kgates)	47.4	43.6	50	46.2	3.9	59.4	54.7	17.5
	Comb/ Seq Counts	15016/ 3866	14288/ 3868	15772/ 4106	15137/ 4108	NA	18857/ 4882	17947/ 4884	NA
	Dynamic Power (mW)	2.16	1.67	2.3	1.77	0.062	2.75	2.12	0.21
	Static Power (uW)	20	160	20	170	1.23	20	197.8	5.72
Clock Gating & Scan Ready	Percentage Gating	81.10%	81.11%	80.36%	80.31%	NA	81.67%	79.72%	NA
	Test Coverage	99.30%	99.27%	99.32%	99.27%	NA	99.37%	99.29%	NA
	Gate Count (Kgates)	50.6	47.9	54	50.8	4.3	64	61.1	18.6
	Comb/ Seq Counts	13417/ 4060	12178/ 4062	14532/ 4313	12896/ 4314	NA	17136/ 5197	15613/ 5198	NA
	Dynamic Power (mW)	0.98	0.49	1.07	0.82	0.04	1.32	0.65	0.08
	Static Power (uW)	20	160	20	160	1.22	20	200	5.42

Note

- Tetramax and PrimeTime PX, version J-2014.12-SP1 is used for test coverage estimation and power computation.
- The Area and Power numbers provided for Low Power Mode are for the Always ON domain logic.
- The Normal Operation power estimates are for simulations with continuous transmission and reception of 1500 frames each, at the Application clock frequency.
- The Low Power Mode power estimates are for UPF mode simulations with 5 frames received, the last one being valid remote wakeup frame that wakes up the MAC. In ARP-enabled configuration the MAC responds to 1 ARP request in Low Power Mode.

B Endian Support

DWC_ether_qos supports endian modes for 32-bit, 64-bit, or 128-bit data bus.

This appendix provides information about the endian modes that the DWC_ether_qos supports for 32-bit, 64-bit, or 128-bit data bus. It contains the following sections:

B.1 Introduction

The DWC_ether_qos supports the following endian modes:

- "Little-Endian Mode" on page 1398
- "Big-Endian Mode" on page 1398
- "Byte-Invariant Big-Endian Mode" on page 1398

B.1.1 Little-Endian Mode

In the little-endian (LE) mode, the lowest storage address accesses the least significant byte (LSB) lane of the data bus and the highest address accesses the most significant byte (MSB) lane. This mode is supported in all DWC_ether_qos configurations.

B.1.2 Big-Endian Mode

In the conventional big-endian (CBE) mode, the lowest storage address accesses the most significant byte lane of the data bus and the highest address accesses the least significant byte. This mode is supported in all DWC_ether_qos configurations except EQOS-AXI which supports byte-invariant big-endian mode. However, when you select the AHB slave interface in the EQOS-AXI configuration, the CBE mode is applicable to the AHB slave interface.

B.1.3 Byte-Invariant Big-Endian Mode

In the byte-invariant big-endian (BIBE) or address invariant mode, the address of each byte of memory remains unchanged when switching between the little-endian and big-endian modes. When a data item larger than a byte is loaded from or stored to the memory, the bytes making up that data item are arranged in the correct order depending on the endianness of the memory access. This mode is supported only in the EQOS-AXI configuration on the AXI bus interface.

Note Although the EQOS-AXI master port supports the byte-invariant big-endian mode, it works in the similar way as the conventional big-endian mode. This is because the EQOS-AXI master always uses transfers size equal to the AXI bus data width. Such transfers are identical in both conventional and byte-invariant big-endian modes.

B.2 32-bit Data Bus Transfers

This section provides information about the little-endian, conventional big-endian, and byte-invariant big-endian mode accesses on 32-bit data bus for Byte, HalfWord, and Word transfers. It contains the following sections:

- "Master Interface" on page 1399
  - Little-Endian Data Transfer
  - Conventional Big-Endian/ Byte-Invariant Big-Endian Data Transfer
- "Slave Interface" on page 1399
  - "Little-Endian Data Transfer" on page 1400
  - "Conventional Big-Endian Data Transfer" on page 1400
  - "Byte-Invariant Big-Endian Data Transfer" on page 1401

B.2.1 Master Interface

B.2.1.1 Little-Endian Data Transfer

Table B-1provides information about the data transfers in the little-endian mode on the DWC_ether_qos master interface.

Table B-1 32-Bit Data Bus: Little-Endian Support for Master Interface

Transfer Size	Address OffsetA[1:0]	Data[31:24]	Data[23:16]	Data[15:8]	Data[7:0]
Always Word Transfer	0	MSB	MSB-1	LSB + 1	LSB

B.2.1.2 Conventional Big-Endian/ Byte-Invariant Big-Endian Data Transfer

Table B-2 provides information about the data transfers in the conventional big-endian and byte-invariant big-endian mode on the DWC_ether_qos master interface.

Table B-2 32-Bit Data Bus: Big-Endian/ Byte-Invariant Support for Master Interface

Transfer Size	Address OffsetA[1:0]	Data[31:24]	Data[23:16]	Data[15:8]	Data[7:0]
Always Word Transfer	0	LSB	LSB + 1	MSB-1	MSB

B.2.2 Slave Interface

The DWC_ether_qos Control and Status Registers are defined as 32-bit data structure. However, this data structure can break because of the byte-lane swapping during endian conversion. To retain the 32-bit data structure in all modes, the register byte-address is defined as shown Table B-3.

For example, in little-endian mode, Bits[7:0] correspond to address A0 and in CBE or BIBE mode, Bits[31:24] correspond to address A0.

Table B-3 32-Bit Data Bus: Reference CSR Register for Slave Interface

CSR Bits	31:24	23:16	15:18	7:0
Data Notation of CSR Register	D3	D2	D 1	D0
Byte Address in LE	A3	A2	A 1	A0
Byte Address in CBE/BIBE	A0	A1	A 2	A3

B.2.2.1 Little-Endian Data Transfer

Table B-4 provides information about the data transfers in little-endian mode on the DWC_ether_qos slave interface. This is the default mode in which the lane data corresponds to the same bits in the Control and Status Registers.

Table B-4 32-Bit Data Bus: Little-Endian Support for Slave Interface

Transfer Size	Address OffsetA[1:0]	Data[31:24]	Data[23:16]	Data[15:8]	Data[7:0]
Word	0	D3	D2	D1	D0
HalfWord	0			D1	D0
HalfWord	2	D3	D2		
Byte	0				D0
Byte	1			D1	
Byte	2		D2		
Byte	3	D3			

B.2.2.2 Conventional Big-Endian Data Transfer

Table B-5 provides information about the data transfers in conventional big-endian mode on the DWC_ether_qos slave interface. In this mode, the lane data corresponds to the same bits of the Control and Status Registers (CSR) and therefore, there is no lane swapping.

Table B-5 32-Bit Data Bus: Conventional Big-Endian Support for Slave Interface

Transfer Size	Address OffsetA[1:0]	Data[31:24]	Data[23:16]	Data[15:8]	Data[7:0]
Word	0	D3	D2	D1	D0

Transfer Size	Address OffsetA[1:0]	Data[31:24]	Data[23:16]	Data[15:8]	Data[7:0]
HalfWord	0	D3	D2		
HalfWord	2			D1	D0
Byte	0	D3			
Byte	1		D2		
Byte	2			D1	
Byte	3				D0

B.2.2.3 Byte-Invariant Big-Endian Data Transfer

Table B-6 provides information about the data transfers in byte-invariant big-endian mode on the DWC_ether_qos slave interface. In this mode, a Word access (32-bit) is identical to the little-endian and conventional big-endian mode but transfers of smaller size (16-bit and 8-bit) are different. For example, a byte-access to A0 results in bits[7:0] of the AHB/AXI bus written to bits[31:24] (denoted as D3 in Table B-6) of the corresponding Control and Status register.

Table B-6 32-Bit Data Bus: Byte-Invariant Big-Endian Support for Slave Interface

Transfer Size	Address OffsetA[1:0]	Data[31:24]	Data[23:16]	Data[15:8]	Data[7:0]
Word	0	D3	D2	D1	D0
HalfWord	0			D3	D2
HalfWord	2	D1	D0		
Byte	0				D3
Byte	1			D2	
Byte	2		D1		
Byte	3	D0			

B.3 64-bit Data Bus Transfers

This section provides information about the little-endian, conventional big-endian, and byte-invariant bigendian mode accesses on 64-bit data bus for Byte, HalfWord, Word, and DWord transfers. It contains the following sections:

B.3.1 Master Interface

The AXI, AHB, or native DMA master interface initiates only DWord transfers. Therefore, this section does not describe the transfer types such as HalfWord, Byte, and Word at address offset 4.

B.3.1.1 Little-Endian Data Transfer

Table B-7 provides information about the data transfers in the little-endian mode on the DWC_ether_qos master interface.

Table B-7 64-Bit Data Bus: Little-Endian Support for Master Interface

Transfer Size	Address Offset A[2:0]	Data [63:56]	Data [55:48]	Data [47:40]	Data [39:32]	Data [31:24]	Data [23:16]	Data [15:8]	Data [7:0]
DWord	0	MSB	MSB - 1	MSB - 2	MSB - 3	LSB + 3	LSB + 2	LSB + 1	LSB

B.3.1.2 Conventional Big-Endian/ Byte-Invariant Big-Endian Data Transfer

Table B-8 provides information about the data transfers in the conventional big-endian and byte-invariant big-endian mode on the DWC_ether_qos master interface.

Table B-8 64-Bit Data Bus: Big-Endian/ Byte-Invariant Support for Master Interface

Transfer Size	Address Offset A[2:0]	Data [63:56]	Data [55:48]	Data [47:40]	Data [39:32]	Data [31:24]	Data [23:16]	Data [15:8]	Data [7:0]
DWord	0	LSB	LSB - 1	LSB + 2	LSB + 3	MSB - 3	MSB - 2	MSB - 1	MSB

B.3.2 Slave Interface

The DWC_ether_qos registers and internal bus are 32-bit wide. Therefore, the DWord (64-bit) CSR access internally get converted into two 32-bit CSR accesses as shown in Table B-9. The DWord access through slave interface is supported only with the AXI slave port. The DWord accesses are not supported on the AHB Slave port.

Table B-9 64-Bit Data Bus: Reference CSR Register for Slave Interface

CSR Bits	31:24	23:16	15:18	7:0
First CSR Register	D3	D2	D 1	D0
Second CSR Register	D7	D6	D5	D4
Address in LE	A0	A1	A 2	A3
	A7	A6	A5	A4
Address in CBE or BIBE	A0	A1	A2	A3
	A4	A5	A6	A7

B.3.2.1 Little-Endian Data Transfer

Table B-10 provides information about the data transfers in little-endian mode on the DWC_ether_qos slave interface. In Table B-10, the DWord access is applicable only for the AXI slave port.

Table B-10 64-Bit Data Bus: Little-Endian Support for Slave Interface

Transfer Size	Address Offset A[2:0]	Data [63:56]	Data [55:48]	Data [47:40]	Data [39:32]	Data [31:24]	Data [23:16]	Data [15:8]	Data [7:0]
DWord	0	D7	D6	D5	D4	D3	D2	D1	D0
Word	0					D3	D2	D1	D0
Word	4	D7	D6	D5	D4				
HalfWord	0							D1	D0
HalfWord	2					D3	D2		
HalfWord	4			D5	D4				
HalfWord	6	D7	D6						
Byte	0								D0
Byte	1							D1	
Byte	2						D2		
Byte	3					D3			
Byte	4				D4				
Byte	5			D5					
Byte	6		D6						
Byte	7	D7							

B.3.2.2 Conventional Big-Endian Data Transfer

Table B-11 provides information about the data transfers in conventional big-endian mode on the DWC_ether_qos slave interface.The DWord access is not supported on the AHB slave port and therefore, it is not described in Table B-11.

Table B-11 64-Bit Data Bus: Conventional Big-Endian Support for Slave Interface

Transfer Size	Address Offset A[2:0]	Data [63:56]	Data [55:48]	Data [47:40]	Data [39:32]	Data [31:24]	Data [23:16]	Data [15:8]	Data [7:0]
Word	0	D3	D2	D1	D0				
Word	4					D7	D6	D5	D4
HalfWord	0	D3	D2						
HalfWord	2			D1	D0				
HalfWord	4					D7	D6		
HalfWord	6							D5	D4
Byte	0	D3							
Byte	1		D2						
Byte	2			D1					
Byte	3				D0				
Byte	4					D7			
Byte	5						D6		
Byte	6							D5	
Byte	7								D4

B.3.2.3 Byte-Invariant Big-Endian Data Transfer

Table B-12 provides information about the data transfers in byte-invariant big-endian mode on the AXI slave interface. For DWord access at offset 0, Bits[63:32] of the AXI bus get transferred to bits[31:0] of the Register 0 and bits[31:0] get transferred to bits[31:0] of Register 1. For Word access at offset 0, bits[31:0] of the AXI bus get transferred to Register 0.

For HalfWord access to Address A0, bits[15:0] of the AXI bus get transferred to bits[31:16] of the Control and Status Register.

Table B-12 64-Bit Data Bus: Byte-Invariant Big-Endian Support for Slave Interface

Transfer Size	Address Offset A[2:0]	Data [63:56]	Data [55:48]	Data [47:40]	Data [39:32]	Data [31:24]	Data [23:16]	Data [15:8]	Data [7:0]
DWord	0	D3	D2	D1	D0	D7	D6	D5	D4

Transfer Size	Address Offset A[2:0]	Data [63:56]	Data [55:48]	Data [47:40]	Data [39:32]	Data [31:24]	Data [23:16]	Data [15:8]	Data [7:0]
Word	0					D3	D2	D1	D0
Word	4	D7	D6	D5	D4				
HalfWord	0							D3	D2
HalfWord	2					D1	D0		
HalfWord	4			D7	D6				
HalfWord	6	D5	D4						
Byte	0								D3
Byte	1							D2	
Byte	2						D1		
Byte	3					D0			
Byte	4				D7				
Byte	5			D6					
Byte	6		D5						
Byte	7	D4							

B.4 128-bit Data Bus Transfers

This section provides information about the little-endian, conventional big-endian, and byte-invariant bigendian mode accesses on 128-bit data bus for byte, HalfWord, Word, DWord, and QWord transfers. It contains the following sections:

- "Master Interface" on page 1406
  - "Little-Endian Data Transfer" on page 1406
  - "Conventional Big-Endian/ Byte-Invariant Big-Endian Data Transfer" on page 1406
- "Slave Interface" on page 1407
  - "Little-Endian Data Transfer" on page 1407
  - "Conventional Big-Endian Data Transfer" on page 1409
  - "Byte-Invariant Big-Endian Data Transfer" on page 1410

B.4.1 Master Interface

The AXI, AHB or native DMA master interface initiates only QWord transfers. Therefore, this section does not describe the transfer types such as HalfWord and Byte.

B.4.1.1 Little-Endian Data Transfer

Table B-13 provides information about the data transfers in the little-endian mode on the DWC_ether_qos master interface.

Table B-13 128-Bit Data Bus: Little-Endian Support for Master Interface

Transfer Size	Address Offset A[2:0]	Data															
		127:120	119:112	111:104	103:96	95:88	87:80	79:72	71:64	63:56	55:48	47:40	39:32	31:24	23:16	15:8	7:0
QWord	0	MSB	MSB - 1	MSB - 2					MSB - 1	LSB + 7						LSB + 1	LSB

B.4.1.2 Conventional Big-Endian/ Byte-Invariant Big-Endian Data Transfer

Table B-14 provides information about the data transfers in the conventional big-endian and byte-invariant big-endian mode on the DWC_ether_qos master interface. In byte-invariant big-endian mode for AXI, only QWord transfers are applicable.

Table B-14 128-Bit Data Bus: Big-Endian/ Byte-Invariant Support for Master Interface

Transfer Size	Address Offset A[2:0]	Data															
		127 : 120	119: 112	111 : 104	103: 96	95: 88	87: 80	79: 72	71: 64	63: 56	55: 48	47: 40	39: 32	31: 24	23: 16	15: 8	7:0
QWord	0	LSB	LSB + 1	LSB + 2					LSB + 7	MSB - 7						MSB - 1	MSB

B.4.2 Slave Interface

The Control and Status Register are 32-bit wide. Therefore, the DWord or QWord CSR access internally gets converted into two/four 32-bit CSR accesses as shown in Table B-15. The DWord or QWord access through slave interface is supported only with the EQOS-AXI slave interface.

Table B-15 128-Bit Data Bus: Reference CSR Register for Slave Interface

CSR Bits	31:24	23:16	15:18	7:0
First CSR Register	D3	D2	D 1	D0
Second CSR Register	D7	D6	D5	D4
Third CSR Register	D11	D10	D9	D8
Fourth CSR Register	D15	D14	D13	D12
Address in LE	A3	A2	A1	A0
	A7	A6	A5	A4
	AB	AA	A9	A8
	AF	AE	AD	AC
Address in CBE or BIBE	A0	A1	A2	A3
	A4	A5	A6	A7
	A8	A9	AA	AB
	AC	AD	AE	AF

B.4.2.1 Little-Endian Data Transfer

Table B-16 provides information about the data transfers in little-endian mode on the DWC_ether_qos slave interface. In Table B-16, the QWord and DWord accesses are applicable only for the AXI slave port.

Table B-16 128-Bit Data Bus: Little-Endian Support for Slave Interface

Transfer Size	Address Offset A[2:0]	Data															
		127:120	119:112	111:104	103:96	95:88	87:80	79:72	71:64	63:56	55:48	47:40	39:32	31:24	23:16	15:8	7:0
QWord	0	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
DWord	0									D7	D6	D5	D4	D3	D2	D1	D0
DWord	8	D15	D14	D13	D12	D11	D10	D9	D8								
Word	0													D3	D2	D1	D0
Word	4									D7	D6	D5	D4				
Word	8					D11	D10	D9	D8								
Word	0xC	D15	D14	D13	D12												
HalfWord	0															D1	D0
HalfWord	2													D3	D2		
HalfWord	4											D5	D4				
HalfWord	6									D7	D6						
HalfWord	8							D9	D8								
HalfWord	0xA					D11	D10										
HalfWord	0xC			D13	D12												
HalfWord	0xE	D15	D14														
Byte	0																D0
Byte	1															D1	
Byte	x										--	--	--	--	--		
Byte	7									D7							
Byte	8								D8								
Byte	x			--	--	--	--	--									
Byte	0xE		D14														
Byte	0xF	D15															

In Table B-16, the ellipses (...) represent Dx where x is the byte offset.

B.4.2.2 Conventional Big-Endian Data Transfer

Table B-17 provides information about the data transfers in conventional big-endian mode on the DWC_ether_qos slave interface. The QWord and DWord access is not supported on the AHB slave port and therefore, these are not described in Table B-17.

Table B-17 128-Bit Data Bus: Conventional Big-Endian Support for Slave Interface

Transfer Size	Address Offset A[2:0]	Data															
		127:120	119:112	111:104	103:96	95:88	87:80	79:72	71:64	63:56	55:48	47:40	39:32	31:24	23:16	15:8	7:0
Word	0	D3	D2	D1	D0												
Word	4					D7	D6	D5	D4								
Word	8									D11	D10	D9	D8				
Word	0xC													D15	D14	D13	D12
HalfWord	0	D3	D2														
HalfWord	2			D1	D0												
HalfWord	4					D7	D6										
HalfWord	6							D5	D4								
HalfWord	8									D11	D10						
HalfWord	0xA											D9	D8				
HalfWord	0xC													D9	D8		
HalfWord	0xE															D13	D12
Byte	0	D3															
Byte	1		D2														
Byte	x			--	--	--	--	--									
Byte	7								D4								
Byte	8									D11							
Byte	x										--	--	--	--	--		
Byte	0xE															D13	

Transfer Size	Address Offset A[2:0]	Data															
		127 : 120	119 : 112	111 : 104	103 : 96	95: 88	87: 80	79: 72	71: 64	63: 56	55: 48	47: 40	39: 32	31: 24	23: 16	15: 8	7:0
Byte	0xF																D12

In Table B-17, the ellipses (...) represent Dx where x is the byte offset.

B.4.2.3 Byte-Invariant Big-Endian Data Transfer

Table B-18 provides information about the data transfers in byte-invariant big-endian mode on the DWC_ether_qos slave interface. For a QWord access, Bits[127:96] get transferred to Bits[31:0] of Register 0 (at address offset 0) in the first internal cycle, and Bits[31:0] of the AXI bus get transferred to Bits[31:0] of Register 3 (at address offset 0xC) in the fourth internal cycle.

Table B-18 128-Bit Data Bus: Byte-Invariant Big-Endian Support for Slave Interface

Transfer Size	Address Offset A[2:0]	Data															
		127 : 120	119 : 112	111 : 104	103 : 96	95: 88	87: 80	79: 72	71: 64	63: 56	55: 48	47: 40	39: 32	31: 24	23: 16	15: 8	7:0
QWord	0	D3	D2	D1	D0	D7	D6	D5	D4	D11	D10	D9	D8	D15	D14	D13	D12
DWord	0									D3	D2	D1	D0	D7	D6	D5	D4
DWord	8	D11	D10	D9	D8	D15	D14	D13	D12								
Word	0													D3	D2	D1	D0
Word	4									D7	D6	D5	D4				
Word	8					D11	D10	D9	D8								
Word	0xC	D15	D14	D13	D12												
HalfWord	0															D3	D2
HalfWord	2													D1	D0		
HalfWord	4											D7	D6				
HalfWord	6									D5	D4						
HalfWord	8							D11	D10								

Transfer Size	Address Offset A[2:0]	Data															
		127 : 120	119 : 112	111 : 104	103 : 96	95: 88	87: 80	79: 72	71: 64	63: 56	55: 48	47: 40	39: 32	31: 24	23: 16	15: 8	7:0
HalfWord	0xA					D9	D8										
HalfWord	0xC			D15	D14												
HalfWord	0xE	D13	D12														
Byte	0																D3
Byte	1															D2	
Byte	x										--	--	--	--	--		
Byte	7									D4							
Byte	8								D11								
Byte	x			--	--	--	--	--									
Byte	0xE		D13														
Byte	0xF	D12															

In Table B-18, the ellipses (...) represent Dx where x is the byte offset.

C

Back-to-Back Register Support

This appendix provides the list of back-to-back registers supported by DWC_ether_qos:

Table C-1 List of Back-to-Back Registers

Register Name	Configurations
MAC_Configuration	
MAC_VLAN_Incl	DWC_EQOS_SA_VLAN_INS_CTRL_EN
MAC_Inner_VLAN_Incl	DWC_EQOS_SA_VLAN_INS_CTRL_EN && DWC_EQOS_DOUBLE_VLAN_EN
MAC_PTO_Control	DWC_EQOS_PTO_EN
MAC_Log_Message_Interval	
MAC_Ext_Configuration	
MAC_Packet_Filter	
MAC_VLAN_Tag	!DWC_EQOS_ERVFE
MAC_VLAN_Tag_Ctrl	DWC_EQOS_ERVFE
MAC_TxQ_Prty_Map0	DWC_EQOS_DCB_EN && DWC_EQOS_NUM_TXQ>1
MAC_TxQ_Prty_Map1	DWC_EQOS_DCB_EN && DWC_EQOS_NUM_TXQ>4
MAC_RxQ_Ctrl0	DWC_EQOS_NUM_RXQ>1
MAC_RxQ_Ctrl1	
MAC_RxQ_Ctrl2	
MAC_RxQ_Ctrl4	
MAC_RxQ_Ctrl3	DWC_EQOS_NUM_RXQ>4
MAC_L3_L4_Control(#i) (for i = 0; i <= DWC_EQOS_L3_L4_FILTER_NUM-1)	DWC_EQOS_L3_L4_FILTER_EN
MAC_Timestamp_Control	DWC_EQOS_TIME_STAMPING

Register Name	Configurations
MAC_Ext_Cfg1	DWC_EQOS_SPLIT_HDR_EN
MTL_TxQ(#i)_Operation_Mode (for i = 0; i < DWC_EQOS_NUM_TXQ)	!DWC_EQOS_CORE
MTL_RxQ_DMA_Map0	!DWC_EQOS_CORE && !DWC_EQOS_MTL_SUBSYS && DWC_EQOS_NUM_DMA_RX_CH>1
MTL_RxQ_DMA_Map1	!DWC_EQOS_CORE && !DWC_EQOS_MTL_SUBSYS && DWC_EQOS_NUM_DMA_RX_CH>1 && DWC_EQOS_NUM_RXQ>4
MTL_RxQ(#i)_Operation_Mode (for i = 0; i < DWC_EQOS_NUM_RXQ)	!DWC_EQOS_CORE
MTL_RxQ(#i)_Control (for i = 0; i < DWC_EQOS_NUM_RXQ)	!DWC_EQOS_CORE

D

DWC_ether_qos Automotive Safety

This appendix provides pointers to the Safety Features supported in DWC_ether_qos and the Automotive Safety Package available. You must have the corresponding license to use these features.

This appendix has the following sections:

D.1 Overview of Automotive Safety Features

To make an IP better suited for applications in the automotive space, functional safety needs to be considered throughout the IP development process. ISO 26262 addresses functional safety and requires the hardware components to be analyzed with respect to their ability to detect and/or control the effects of random hardware faults. Random hardware faults can occur at any point in time during the life-cycle of the hardware component. They can take the form of permanent faults, or transient faults, and require safety mechanisms to be in place to ensure that severe consequences do not occur as a result of those faults.

The DWC_ether_qos includes the internal safety mechanisms which assist in detecting and handling random hardware faults. These complement the mechanisms already defined in the DWC_ether_qos protocol.

"ASIL B Ready" certification for DWC_ether_qos is in the process. Fore more details on certification, contact Synopsys Support.

Reference Configuration

As the DWC_ether_qos is a highly configurable IP with several optional features, a reference configuration suited for automotive domain usage is used for the Safety Analysis. This configuration includes all the major features required by AVB, TSN, IEEE 1588, specifications and includes all the fault detection and reporting related safety logic in the controller. The details of the reference configuration parameters are provided in the Safety Manual.

The internal safety mechanisms monitor the IP logic and report any detected errors to the application. The application should take an action to handle the error conditions and ensure safe operation of the hardware component.

FMEDA

A Failure Mode, Effects, and Diagnostic Analysis (FMEDA) has also been performed for this reference configuration. The analysis relies on a per-transistor failure rate derived in accordance to IEC TR 62380. This, in conjunction with area information from design synthesis trials, allows to determine the failure rate for the IP which can then be distributed between the modules of the design according to their relative area.

The list of failure modes and their effects is derived for each module and each of the failure modes is assigned a portion of the failure rate of the module they belong to. If the failure mode has the potential to violate a top level safety requirement (error at a top level output pin) there is a search for a safety mechanism that can prevent that violation, totally or partially. If a safety mechanism exists, for example, ECC protection for RAMs, a diagnostic coverage (DC) value is assigned representing the percentage of the failure rate associated with the specific failure mode that is prevented from violating a top level safety requirement. The process is repeated for all failure modes and finally results are accumulated and IP safety metrics are collected in the FMEDA.

Interface DFMEA

The interface DFMEA focuses on the integration of DWC_ether_qos. The DFMEA report is an Excel file listing several items that must be accounted for, during the integration process. For each item in the list, the potential failure mode is identified with the corresponding effect of the failure, the severity of the failure (S), the probability of occurrence (O), the detection capability (D), the risk priority number (RPN=S*O*D) and the recommended actions to mitigate the risks of the failure mode.

D.2 Automotive Safety Package Documents

Table D-1 describes the documents included in the DWC_ether_qos safety package. After you have configured the controller in coreConsultant, you can access the automotive documents in your <workspace>/doc directory.

Table D-1 Description of Documents in Automotive Safety Package

File Name	Description	Contents
Safety_Manual_DWC_EQOS_5.10a.pdf	Safety Manual	Lists all Synopsys quality process documents; explains IP design flow and tools, verification flows (high-level), and code coverage concepts; describes FMEDA worksheet
SG_Quality_Manual.pdf	Quality Manual	Describes the internal Quality Management System and how its specific activities are related to the Synopsys quality process documents listed in the Safety Manual
FMEDA_DWC_EQOS_5.10a.xls	FMEDA worksheet	Calculates FIT rates for each module; enumerates diagnostic capabilities for failure modes.
DFMEA_DWC_EQOS_5.10a.xls	DFMEA worksheet	Interface DFMEA analysis

D.3 Configuring the Automotive Safety Features

When you have a valid automotive safety package license (DWC-Ethernet-QOS-ASP), you can select the safety features supported by DWC_ether_qos, from the **Automotive Safety Package** pane in the **Specify Configuration** activity dialog box, while configuring the controller in the coreConsultant GUI.

For details of the various automotive safety features supported by DWC_ether_qos, see "Automotive Safety Features" on page 397.