LogiCORE IP AXI Ethernet v6.1

Product Guide

Vivado Design Suite

PG138 April 2, 2014

XILINX
ALL PROGRAMMABLE™

Table of Contents

Appendix A: Migrating and Upgrading

Appendix B: Debugging

Appendix C: Additional Resources and Legal Notices

Introduction

This document provides the design specification for the Xilinx® LogiCORE™ IP AXI Ethernet core. This core implements a tri-mode (10/100/1000 Mb/s) Ethernet MAC or a 10/100 Mb/s Ethernet MAC. This core supports the use of MII, GMII, SGMII, RGMII, and 1000BaseX Interfaces to connect MAC to a PHY chip. It also provides on-chip PHY in case of SGMII and 1000BaseX modes. The MDIO interface is used to access PHY Management registers. This core optionally enables 'TCP/UDP full checksum offload', 'VLAN stripping, tagging, translation' and 'Extended filtering for multicast frames' features.

The AXI Ethernet core optionally supports Ethernet Audio Video Bridging (AVB) functions. This core provides a control interface to internal registers using a 32-bit AXI4-Lite interface subset. This AXI4-Lite slave interface supports single beat read and write data transfers (no burst transfers). The transmit and receive data interface is through the AXI4-Stream interface.

Features

- Support for MII, GMII, RGMII, SGMII, and 1000BaseX PHY interfaces.
- Support for SGMII over Select Input/Output (I/O) Low Voltage Differential Signaling (LVDS) in faster devices.
- Support for pause frames for flow control
- Media Independent Interface Management (also called as MDIO), is used for accessing the PHY registers.

See Feature Summary in Chapter 1 for more features.

LogiCORE IP Facts Table	
Core Specifics	
Supported Device Family(1)	Zynq®-7000 All Programmable SoC, Virtex®-7, Kintex®--7, Artix®-7 FPGAs
Supported User Interfaces	AXI4-Lite, AXI4-Stream
Resources	See Table 2-21, Table 2-22, Table 2-23, and Table 2-24.
Provided with Core	
Design Files	Encrypted RTL
Example Design	Not Provided
Test Bench	Not Provided
Constraints File	Not Provided
Simulation Model	N/A
Supported S/W Driver(2)	Standalone and Linux
Tested Design Flows(3)	
Design Entry	Vivado® Design Suite Vivado IP Integrator
Simulation	For supported simulators, see the Xilinx Design Tools: Release Notes Guide.
Synthesis	Vivado Synthesis
Support	
Provided by Xilinx @ www.xilinx.com/support	

Notes:

1. For a complete list of supported devices, see the Vivado IP catalog.
2. Standalone driver details can be found in the SDK directory (*<install_directory>*/doc/usenglish/xilinx_drivers.htm). Linux OS and driver support information is available from //wiki.xilinx.com.
3. For the supported versions of the tools, see the Xilinx Design Tools: Release Notes Guide.

Overview

The AXI Ethernet IP core can only be added to a Vivado® IP Integrator block design in the Vivado Design Suite. The AXI Ethernet IP core represents a hierarchical design block containing multiple LogiCORE™ IP instances (infrastructure cores) that become configured and connected during the system design session. Each of the infrastructure cores can also be added directly to a block design (outside of AXI Ethernet).

This core provides additional functionality and ease of use related to Ethernet. Based on the configuration, this IP creates interface ports, instantiates required helper cores, and also connects these cores. Helper cores for this IP are the Xilinx® LogiCORE IP Tri-Mode Ethernet MAC (TEMAC) and Xilinx LogiCORE Ethernet 1000Base-X PCS/PMA (Gigabit Ethernet PCS PMA). Additional functionality is provided using the axi_ethenet_buffer helper core. For detailed specifications, see Chapter 2, Product Specification. See the change log for this core for the core versions used with this design. All documents can be downloaded from the Xilinx website.

How To Use This Document

Some of the information in this document is identical or very similar for all modes of the AXI Ethernet core. The first sections of this document clarify these similarities. In the cases where slight differences occur for a particular mode, footnotes call attention to the variance. Other information in this document is specific to the type of PHY interface selected.

Feature Summary

Following is a list of more features. Also see Features on the IP Facts page.

- Full duplex support (Half duplex is not supported)
- IEEE 1588 support
- Supports optional 1588 hardware timestamping for one-step and two-step when enabled with 1000BASE-X PHY targeting 7 series FPGAs GTX transceivers
- The system timer provided to the core and the consequential timestamping taken from it are available in one of two formats which are selected during IP core generation.
  - Time-of-Day (ToD) format: IEEE1588-2008 format consisting of a 48-bit second field and a 32-bit nanosecond field.
  - Correction Field format: IEEE1588-2008 numerical format consisting of a 64-bit field representing nanoseconds multiplied by 216 (see IEEE1588 clause 13.3.2.7)
- Option to not to include I/O cells in GMII mode.
- Optional support for jumbo frames up to 16 KB
- Optional TX and RX Transmission Control Protocol/ User Datagram Protocol (TCP/UDP) partial checksum offload
- Optional IPv4 TX and RX TCP/UDP full checksum offload
- Support for VLAN frames
- Optional TX and RX VLAN tagging, stripping, and translation
- Independent 4K, 8K, 16K, or 32KB TX and RX frame buffer memory.
- Optional extended filtering for multicast frames
- Optional TX and RX statistics gathering
- Auto PAD and FCS field insertion or pass through on transmit
- Auto PAD and FCS field stripping or pass through on receive
- Ethernet Audio Video Bridging (AVB) at 100/1000 Mb/s (Additional license required)
- Filtering of bad receive frames
- Option to include or exclude shareable logic resources in the core
- Optional support for board-based I/O constraints generation

Licensing and Ordering Information

This Xilinx LogiCORE IP AXI Ethernet module is provided at no additional cost with the Xilinx Vivado Design Suite under the terms of the Xilinx End User License. To use AXI Ethernet, a AXI TEMAC license must be purchased governed under the terms of the Xilinx Core License Agreement. Also, to enable the Ethernet AVB feature, an additional license key is required.

For more information, visit the AXI Ethernet product web page.

Information about this and other Xilinx LogiCORE IP modules is available at the Xilinx Intellectual Property page.For information on pricing and availability of other Xilinx LogiCORE IP modules and tools, contact your local Xilinx sales representative.

IMPORTANT: *There is no special license for the AXI Ethernet core, but there is a license for the TEMAC core as well as the optional Ethernet AVB feature. More details related to licensing of TEMAC can be found in the LogiCORE IP Tri-Mode Ethernet MAC Product Guide (PG051) [Ref 1].*

Product Specification

Functional Description

A high-level block diagram of the AXI Ethernet IP core is shown in Figure 2-1.

The AXI Ethernet core provides an AXI4-Lite bus interface for a simple connection to the processor core to allow access to the registers. The 32-bit AXI4-Stream buses are provided for moving transmit and receive Ethernet data to and from the AXI Ethernet core. These buses are designed to be used with an AXI DMA IP core, AXI4-Stream Data FIFO, AXI4-Stream Data FIFO, or any other custom logic in any supported device. The AXI4-Stream buses are designed to provide support for TCP/UDP partial or full checksum offload in hardware if that function is required. The AXI4-Stream buses are described in Frame Transmission in Chapter 2.

The PHY side of the core is connected to an off-the-shelf Ethernet PHY device, which performs the BASE-T standard at 1 Gb/s, 100 Mb/s, and 10 Mb/s speeds. The PHY device can be connected using any of the following supported interfaces: GMII/MII, RGMII, or, by additionally using the "Ethernet 1000BASE-X PCS/PMA" module for SGMII.

A 1000BaseX PHY can be implemented directly in the FPGA itself using the 'Ethernet 1000BASE-X PCS/PMA' module configured in 1000BASE-X mode. This can be connected directly to an external optical module.

The supported physical interface types are:

- **GMII**. The Gigabit Media Independent Interface (GMII) is defined by the IEEE802.3 specification; it can provide support for Ethernet operation at 10 Mb/s, 100 Mb/s and 1 Gb/s speeds.
- **MII**. The Media Independent Interface (MII) is defined by the IEEE802.3 specification; it can provide support for Ethernet operation at 10 Mb/s and 100 Mb/s speeds.
- **RGMII**. The Reduced Gigabit Media Independent Interface (RGMII) is, effectively, a Double Data Rate version of GMII; it can provide support for Ethernet operation at 10 Mb/s, 100 Mb/s and 1 Gb/s speeds.
- **1000BASE-X**. Physical Coding Sublayer (PCS) and Physical Medium Attachment (PMA) operation, as defined in the IEEE 802.3-2008 standard.

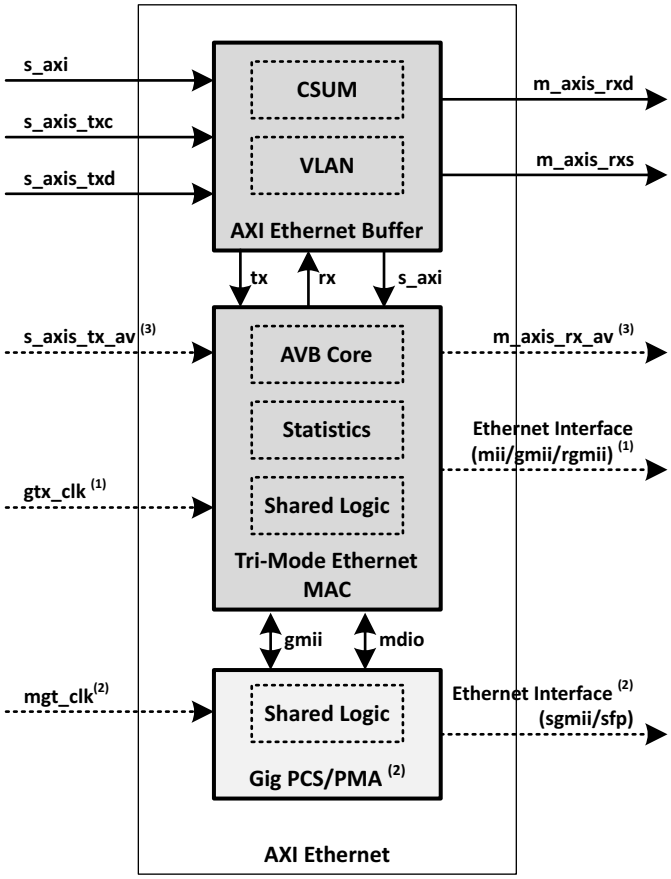
- **GMII to Serial-GMII (SGMII) bridge.** As defined in the Serial-GMII specification (ENG-46158).

In AVB mode, Ethernet AVB Endpoint is selected which is designed to meet the following IEEE specifications.

- **IEEE802.1AS**. Supports clock master functionality, clock slave functionality and the Best Master Clock Algorithm (BMCA)
- **IEEE802.1Qav**. Supports arbitration between different priority traffic and implements bandwidth policing.

The following helper cores are used by the AXI Ethernet AppCore.

- Tri Mode Ethernet MAC core.
- Gigabit Ethernet PCS/PMA core.
- AXI Ethernet Buffer. Features like 'TX and RX TCP/UDP Partial Checksum offload', 'IPv4 TX and RX TCP/UDP full checksum offload', 'TX and RX VLAN stripping, tagging' and 'Extended filtering for multicast frames' are provided using this helper core. The details related to these features are described in this document.
- Timer Synchronization core (Timer Sync): This core synchronizes the timer in the selected format from the system clock domain into the core clock domain.



(1) This interface is present only in MII, GMII or RGMII modes. It is not present in SGMII or 1000BaseX modes.
(2) This interface and GigPCS/PMA module are present only in SGMII or 1000BaseX modes.
(3) AVB Interfaces are present only when AVB mode is enabled.

Figure 2-1: **Block Diagram for the AXI Ethernet Core**

Partial TCP/UDP Checksum Offload in Hardware

When using TCP or UDP Ethernet protocols, data integrity is maintained by calculating and verifying checksum values over the TCP and UDP frame data. Normally this checksum functionality is handled by the protocol stack software which can be relatively slow and uses significant processor power for large frames at high Ethernet data rates. An alternative is to offload some of this transmit checksum generation and receive checksum verification in hardware. This is possible by including checksum off-loading in the AXI Ethernet core using the C_TXCSUM and C_RXCSUM parameters. Including the checksum offload functions are a trade-off between using more FPGA resources and getting higher Ethernet performance while freeing up processor use for other functions.

When using the TCP/UDP checksum offload function, checksum information is passed between the software and the AXI Ethernet core, using the AXI4-Stream Control and AXI4-Stream Status interfaces. Table 2-27, Table 2-29, Figure 2-22, Table 2-31, Table 2-32, Table 2-33, Table 2-34, Table 2-35, Table 2-36, and Figure 2-24 show the checksum offload fields.

The use of the TCP/UDP checksum offload function requires that the core is connected to the AXI Ethernet core through the AXI4-Stream Control and AXI4-Stream Data interfaces. See Mapping AXI DMA IP Buffer Descriptor Fields to AXI4-Stream Fields for more information.

TX_CSBEGIN is the beginning offset and points to the first byte that needs to be included in the checksum calculation. The first byte is indicated by a value of zero. The beginning position must be 16-bit aligned. With TCP and UDP, set this value to skip over the Ethernet frame header as well as the IP datagram header. This allows the checksum calculation to start in the correct place of the TCP or UDP segment. Operating systems might provide functions to calculate this value as it is normally based on the variable IP datagram header size. In all cases, the TX_CSBEGIN value must be 14 or larger to be valid.

TX_CSINSERT is the offset which points to the location where the checksum value should be written into the TCP or UDP segment header. This value must be 16-bit aligned and cannot be in the first 8 bytes of the frame. Again, operating systems might provide functions to calculate this value as it is normally variable based on the variable IP datagram header size.

TX_CSCNTRL is a 16-bit field; however, only the least significant bit is defined. This bit controls the insertion of the checksum into the frame data. If set to a 1, the checksum value is written into the transmit frame; otherwise, the frame is not modified.

TX_CSINIT is a 16-bit seed that can be used to insert the TCP or UDP pseudo header into the checksum calculation. In many cases the protocol stack calculates the pseudo header checksum value and places it in the header checksum field of the transmit frame. In those cases this field should be zeroed.

If the protocol stack does not provide the pseudo header checksum in the header checksum field location of the transmit frame, then that field should be zeroed and the pseudo header checksum value must be calculated and placed in the TX_CSINIT field of the buffer descriptor.

In order for the transmit checksum to be calculated correctly, the transmit Ethernet FCS must not be provided as part of the transmit data and the transmit FCS calculation and insertion must be enabled in the AXI Ethernet core.

There is a special case for checksums of UDP datagrams. From the UDP RFC 768.

> If the computed checksum is zero, it is transmitted as all ones (the equivalent in ones complement arithmetic). An all zero transmitted checksum value means that the transmitter generated no checksum (for debugging or for higher level protocols that do not care).

If the frame encapsulates a UDP datagram, and if the resulting checksum is zero, a value of all ones is used. This case does not exist for TCP because a checksum of zero is legal; *however*, the partial checksum logic does not have any way of knowing if the datagram is TCP or UDP.

For both cases, if the computed checksum is zero, a value of all ones is used instead. If a TCP datagram computed checksum is zero, this can result in the packet being dropped by the receiver.

RX_CSRAW is the raw receive checksum calculated over the entire Ethernet payload. It is calculated starting at byte 14 of the Ethernet frame with the count starting at zero, not one (the byte following the Type/Length field) and continues until the end of the Ethernet frame. If the receive Ethernet FCS stripping is not enabled in the AXI Ethernet core, the FCS is also included in the checksum. The application is required to calculate the checksum of the fields which should not have been included to subtract them from the RAW checksum value. In most cases, the protocol software that allows receive checksum offloading requires a pass or fail indication. The application has to compare the adjusted raw checksum value with the checksum field of the TCP or UDP header and provide the pass or fail indication.

Full TCP/UDP Checksum Offload in Hardware

When using TCP or UDP Ethernet protocols, data integrity is maintained by calculating and verifying checksum values over the TCP and UDP frame data. Normally this checksum functionality is handled by the protocol stack software which can be relatively slow and uses a significant amount of the processor for large frames at high Ethernet data rates.

An alternative is to offload the transmit checksum generation and receive checksum verification in hardware. This is possible by including full checksum offloading in the AXI Ethernet core using parameters. Including the full checksum offload functions are a trade-off between using more FPGA resources and getting higher Ethernet performance while freeing up the processor for other functions.

Full checksum offloading is supported for Ethernet II and Sub-Network Access Protocol (SNAP) frame formats. The frame formats must use the IPv4 Internet protocol and transport data through TCP or UDP. Each frame format can support a single 32-bit VLAN tag (the TPID must equal 0x8100). Example diagrams of these frame formats are shown in Figures 2-2 to 2-9. In these figures, the conditions shown in red are used by the hardware to determine if the TCP/UDP checksum and/or the IPv4 Header checksum is calculated.

It is possible for the IPv4 Header checksum to be calculated even though the TCP/UDP checksum is not calculated. This can occur if the frame is Ethernet II or SNAP with an IPv4 Header that is 5 words in length, the IP flags are set to 0, and the fragment offset is set to zero (the protocol field can be set to something other than TCP or UDP). However, for the TCP or UDP checksum to be calculated, the IPv4 Header checksum must be calculated with the protocol field set to 0x6 for TCP or 0x11 for UDP. Figure 2-2 shows an Ethernet II frame with IPv4 and TCP. The following conditions must be met for the IPv4 Header checksum and TCP checksum to be calculated:

- Type = 0x0800
- Version – 0x4
- Header Length = 0x5
- IP Flag MF = 0b0
- Fragment Offset = 0b0000000000000
- Protocol = 0x06

If Protocol /= 0x06, but the rest of the conditions are met, only the IPv4 Header checksum is calculated.

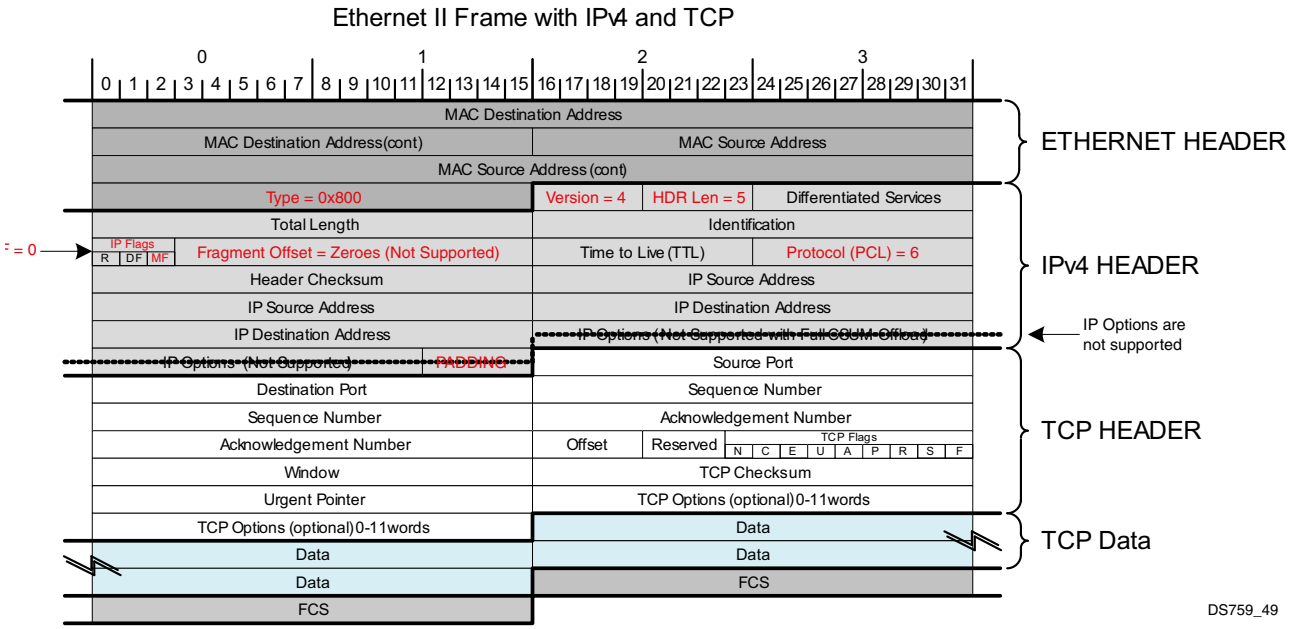


Figure 2-2: **Ethernet II Frame with IPv4 and TCP**

Figure 2-3 shows a VLAN Ethernet II frame with IPv4 and TCP. The following conditions must be met for the IPv4 Header checksum and TCP checksum to be calculated:

- VLAN TPID = 0x8100
- Type = 0x0800
- Version – 0x4
- Header Length = 0x5
- IP Flag MF = 0b0
- Fragment Offset = 0b0000000000000
- Protocol = 0x06

If Protocol /= 0x06, but the rest of the conditions are met, only the IPv4 Header checksum is calculated.

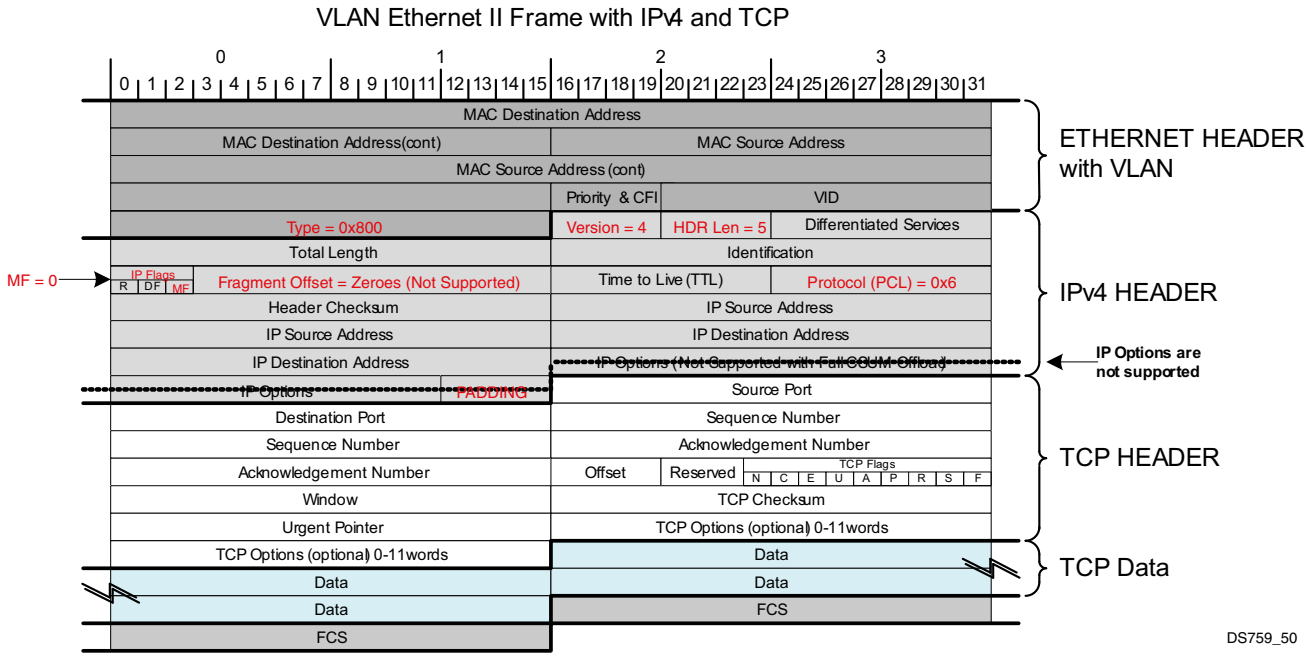


Figure 2-3: **VLAN Ethernet II Frame with IPv4 and TCP**

Figure 2-4 shows an Ethernet II frame with IPv4 and UDP. The following conditions must be met for the IPv4 Header checksum and UDP checksum to be calculated:

- Version – 0x4
- Header Length = 0x5
- IP Flag MF = 0b0
- Fragment Offset = 0b0000000000000
- Protocol = 0x11

If Protocol /= 0x11, but the rest of the conditions are met, only the IPv4 Header checksum is calculated.

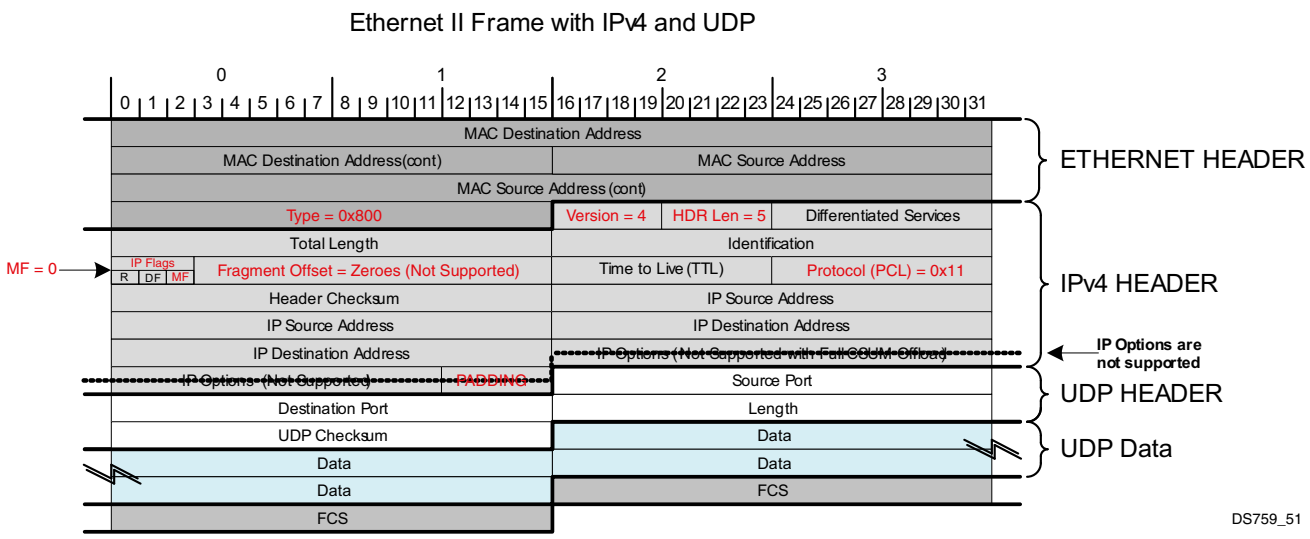


Figure 2-4: **Ethernet II Frame with IPv4 and UDP**

Figure 2-5 shows a VLAN Ethernet II frame with IPv4 and UDP. The following conditions must be met for the IPv4 Header checksum and UDP checksum to be calculated:

- VLAN TPID = 0x8100
- Version – 0x4
- Header Length = 0x5
- IP Flag MF = 0b0
- Fragment Offset = 0b0000000000000
- Protocol = 0x11

If Protocol /= 0x11, but the rest of the conditions are met, only the IPv4 Header checksum is calculated.

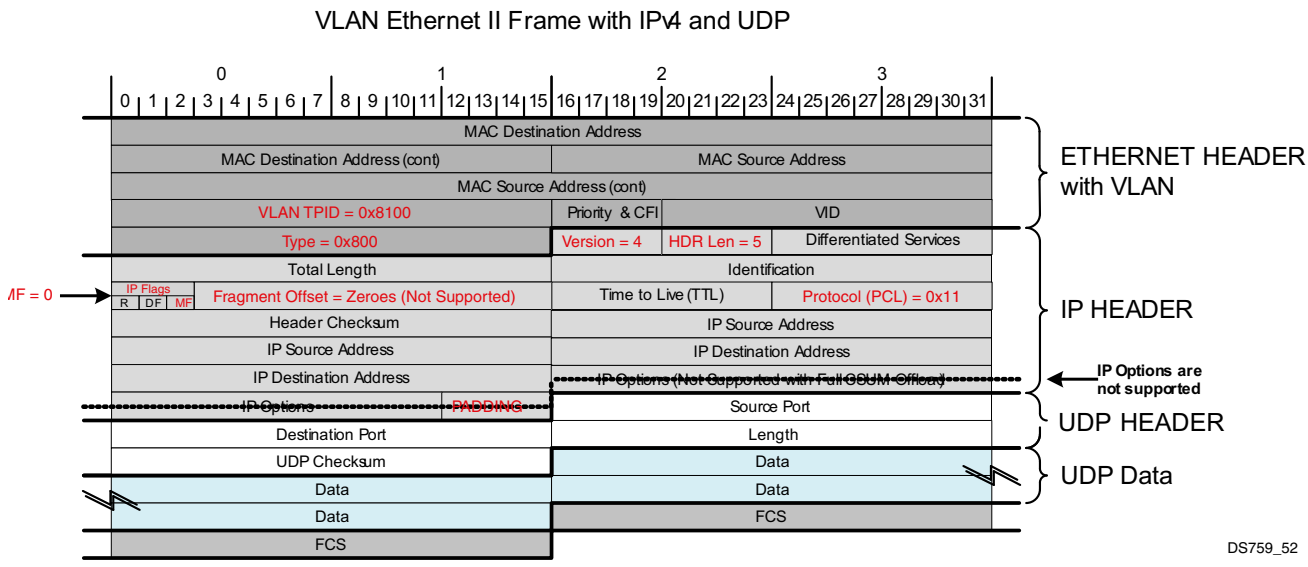


Figure 2-5: **VLAN Ethernet II Frame with IPv4 and UDP**

Figure 2-6 shows a SNAP frame with IPv4 and TCP. The following conditions must be met for the IPv4 Header checksum and TCP checksum to be calculated:

- Length <= 0x0600
- DSAP = 0xAA
- SSAP = 0xAA
- Control = 0x03
- OIU – 0x000000
- Type = 0x0800
- Version – 0x4
- Header Length = 0x5
- IP Flag MF = 0b0
- Fragment Offset = 0b0000000000000
- Protocol = 0x06

If Protocol /= 0x06, but the rest of the conditions are met, only the IPv4 Header checksum is calculated.

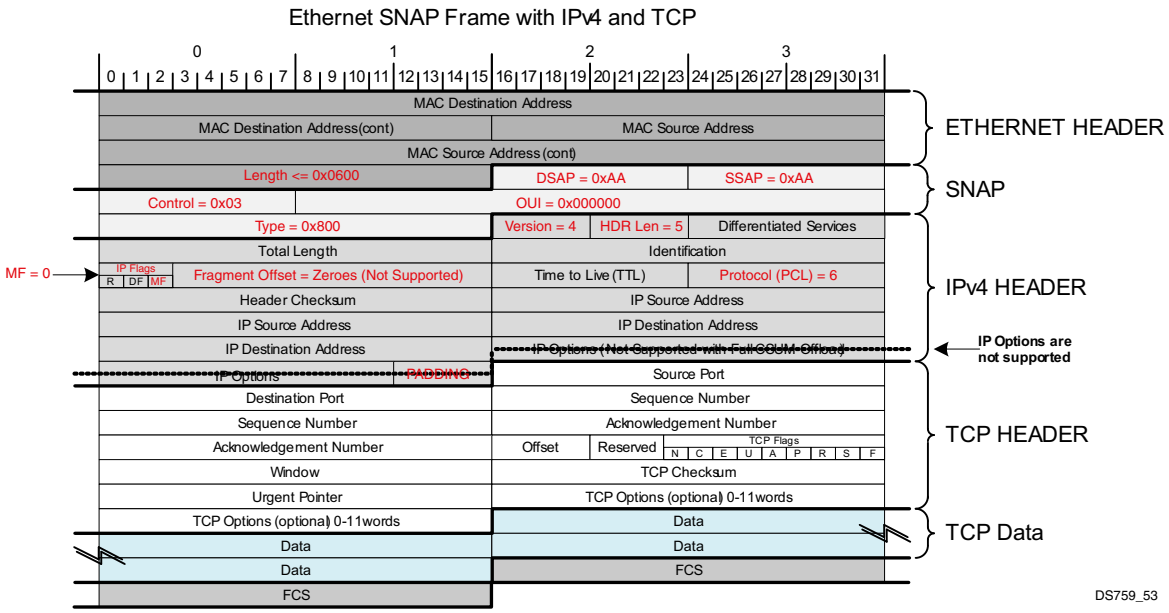


Figure 2-6: **SNAP Frame with IPv4 and TCP**

Figure 2-7 shows a VLAN SNAP frame with IPv4 and TCP. The following conditions must be met for the IPv4 Header checksum and TCP checksum to be calculated:

- VLAN TPID = 0x8100
- Length <= 0x0600
- DSAP = 0xAA
- SSAP = 0xAA
- Control = 0x03
- OIU – 0x000000
- Type = 0x0800
- Version – 0x4
- Header Length = 0x5
- IP Flag MF = 0b0
- Fragment Offset = 0b0000000000000
- Protocol = 0x06

If Protocol /= 0x06, but the rest of the conditions are met, only the IPv4 Header checksum is calculated.

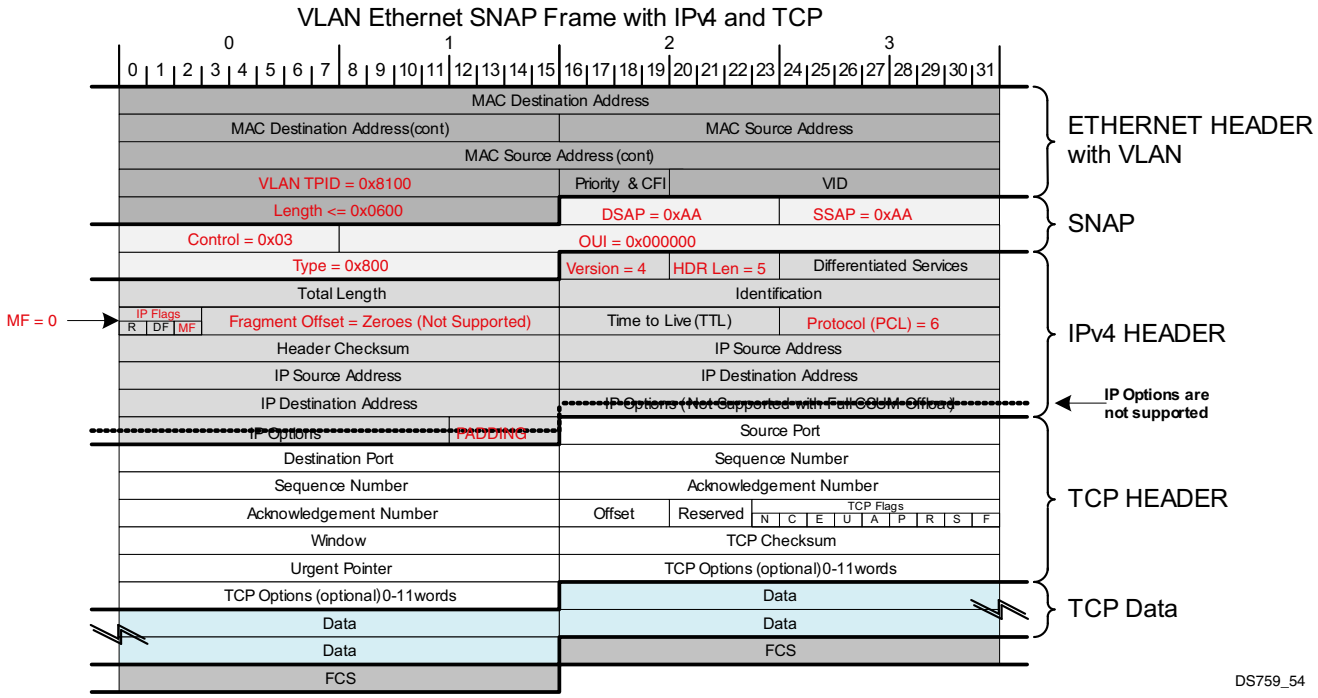


Figure 2-7: **VLAN SNAP Frame with IPv4 and TCP**

Figure 2-8 shows a SNAP frame with IPv4 and UDP. The following conditions must be met for the IPv4 Header checksum and UDP checksum to be calculated:

- Length <= 0x0600
- DSAP = 0xAA
- SSAP = 0xAA
- Control = 0x03
- OIU – 0x000000
- Type = 0x0800
- Version – 0x4
- Header Length = 0x5
- IP Flag MF =0b0
- Fragment Offset =0b0000000000000
- Protocol = 0x11

If Protocol /= 0x11, but the rest of the conditions are met, only the IPv4 Header checksum is calculated.

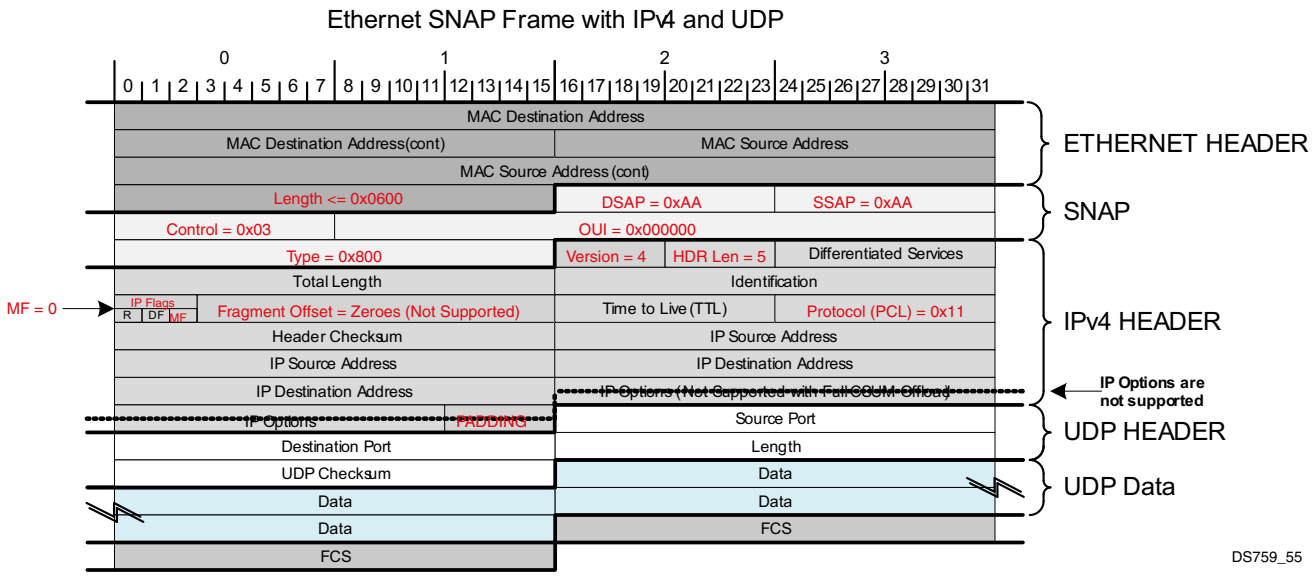


Figure 2-8: **SNAP Frame with IPv4 and UDP**

Figure 2-9 shows a VLAN SNAP frame with IPv4 and UDP. The following conditions must be met for the IPv4 Header checksum and UDP checksum to be calculated:

- VLAN TPID = 0x8100
- Length <= 0x0600
- DSAP = 0xAA
- SSAP = 0xAA
- Control = 0x03
- OIU – 0x000000
- Type = 0x0800
- Version – 0x4
- Header Length = 0x5
- IP Flag MF = 0b0
- Fragment Offset = 0b0000000000000
- Protocol = 0x11

If Protocol /= 0x11, but the rest of the conditions are met, only the IPv4 Header checksum is calculated.

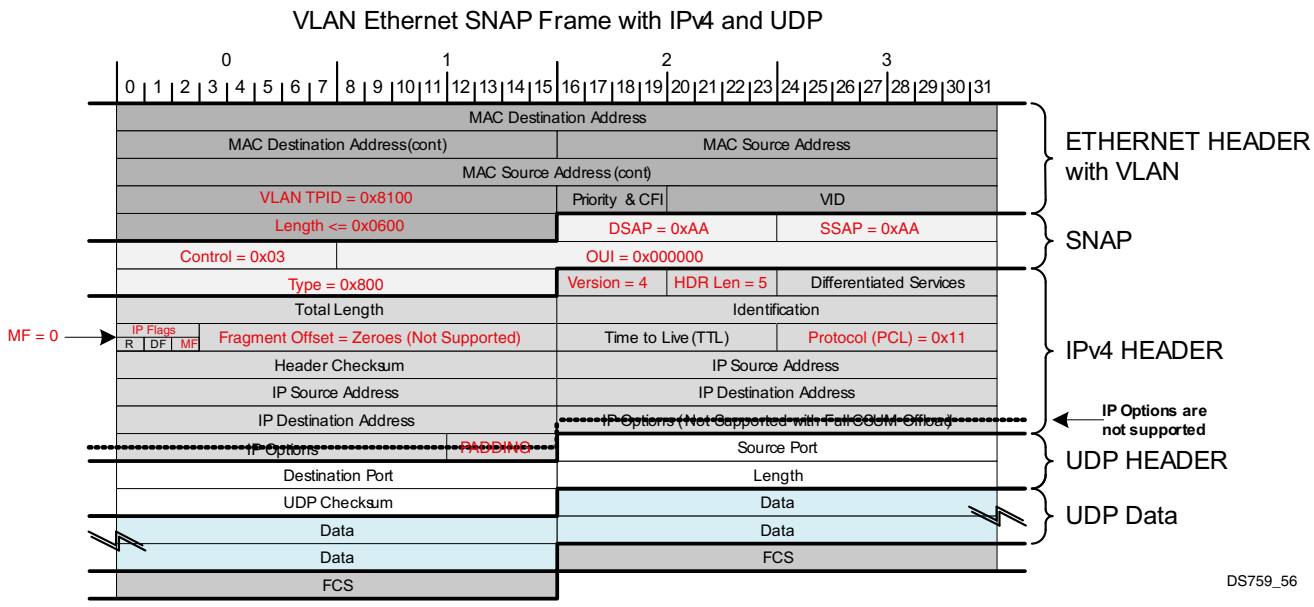


Figure 2-9: **VLAN SNAP Frame with IPv4 and UDP**

If an Ethernet II frame with protocol information is less than 60 bytes, the transmitter pads the frame with zeroes until it is 60 bytes. Because the Ethernet II frame populates the Type/Length field with Type information (0x0800), instead of a Length information, the AXI Ethernet core receive logic is incapable of stripping off any padded bytes. As a result, the receiver reports the length of all transmitter padded packets to be 60 bytes in length.

Frame Transmission

Padding

When fewer than 46 bytes of data are supplied to the AXI Ethernet core, the transmitter adds padding up to the minimum frame length. However, when you are providing the FCS field as part of the frame, the frame must already be padded if necessary to maintain the minimum frame length.

FCS Pass Through

The AXI Ethernet core can calculate and add the FCS field to each transmitted frame, or it can pass through a user-supplied FCS field with frame data. When a user-supplied FCS field is passed through, you must also supply padding as necessary to ensure that the frame meets the minimum frame length requirement. FCS insertion or pass through is controlled by the TC register bit 29 (Table 2-56).

Virtual LAN (VLAN) Frames

When transmitting VLAN frames (if enabled by the TC register bit 27 Table 2-56) without extended VLAN mode, you must supply the VLAN type tag 0x8100, as well as the two-byte tag control field along with the rest of the frame data. More information about the tag control field is available in the IEEE Std 802.3-2008 specification.

Maximum Frame Length and Jumbo Frames

The maximum length of a frame specified in the IEEE Std 802.3-2008 specification is 1518 bytes for non-VLAN tagged frames. VLAN tagged frames can be extended to 1522 bytes. When jumbo frame handling is disabled (TC register bit 30 Table 2-56) and you attempt to transmit a frame that exceeds the maximum legal length, the AXI Ethernet core inserts an error code to corrupt the current frame and the frame is truncated to the maximum legal length. When jumbo frame handling is enabled, frames longer than the legal maximum are transmitted error free. Jumbo frames are restricted by the AXI Ethernet design to less than 16 KB.

Frame Reception

Frame Reception with Errors

An unsuccessful frame reception (for example, a fragment frame or a frame with an incorrect FCS) is dropped and not passed to the system. A Receive Reject interrupt is activated (see bit 3 in Table 2-43).

FCS Pass Through or Stripping

If the Length/Type field has a length interpretation, the received frame can be padded to meet the minimum frame size specification. If FCS Pass Through is disabled (RCW1 register bit 29 Table 2-55) and Length/Type filed error checking is enabled (RCW1 register bit 25 Table 2-55), the padding is stripped along with the FCS field and is not passed to you. If FCS Pass Through is disabled (RCW1 register bit 29 Table 2-55) and Length/Type field error checking is also disabled, the padding is not stripped and is passed to you but the FCS field is stripped and is not passed to you.

If the FCS Pass Through is enabled, any padding is passed to you along with the FCS field. Even though the FCS is passed up to you, it is also verified and the frame is dropped if the FCS is incorrect. A Receive Reject interrupt is activated (see bit 3 in Table 2-43).

Table 2-1: **Receive Frame FCS Field and Pad Field Stripping or Pass Through**

	FCS Pass Through (RCW1 register bit 29 = 1)	FCS Strip (RCW1 register bit 29 = 0)
Length/Type field error check (RCW1 register bit 25 = 0)	FCS and padding (if present) fields passed to user for all accepted frames	FCS and padding (if present) fields stripped and not passed to user for all accepted frames
Length/Type field error ignore (RCW1 register bit 25 = 1)	FCS and padding (if present) fields passed to user for all accepted frames	FCS field stripped and not passed to user but padding (if present) passed to user for all accepted frames

Virtual LAN (VLAN) Frames

Received VLAN tagged frames are passed to you if VLAN frame reception is enabled (RCW1 register bit 27 Table 2-55). This is the basic native VLAN support provided by the TEMAC core. For more information about extended VLAN functions, see the following sections.

Maximum Frame Length and Jumbo Frames

The maximum length of a frame specified in the IEEE Std 802.3-2008 specification is 1518 bytes for non-VLAN tagged frames. VLAN tagged frames can be extended to 1522 bytes. When jumbo frame handling is disabled (RCW1 register bit 30 Table 2-55) and a received frame exceeds the maximum legal length, the frame is dropped and a Receive Reject interrupt is activated (see bit 3 in Table 2-43). When jumbo frame handling is enabled, frames longer then the legal maximum are received in the same way as shorter frames. Jumbo frames are restricted by the AXI Ethernet design to less than 16 KB.

Length/Type Field Error Checks

Length/Type field error checking is specified in IEEE Std 802.3. This functionality must be enabled (RCW1 register bit 25 Table 2-55) to comply with this specification. Disabling Length/Type checking is intended only for specific applications, such as when using over a proprietary backplane.

Enabled

When Length/Type error checking is enabled, the following checks are made on all frames received. If either of these checks fails, the frame is dropped and a Receive Reject interrupt is activated (see bit 28 in Table 2-43).

- A value greater than or equal to decimal 46 but less than decimal 1536 in the length/type field is checked against the actual data length received.
- A value less than decimal 46 in the length/type field is checked to ensure the data field is padded to exactly 46B. The resultant frame is now the minimum frame size: 64B total in length.

Additionally, if FCS passing is disabled, the length/type field is used to strip the FCS field *and* any padding that might exist. *Neither* is passed to you.

Disabled

When the length/type error checking is disabled, the length/type error checking is not performed and a frame that has only these errors is accepted. Additionally, if FCS passing is disabled, the length/type field is *not* used to determine padding that might exist and the FCS field *is* stripped but any padding that might exist in the frame *is not* stripped and *is* passed to you.

Address Filtering

Basic Mode

The receive address filtering function accepts or rejects received frames by examining the destination address field. Part of this function is carried out in the TEMAC component and part is carried out based on the bit settings in the Control register (Reset and Address Filter Register — Offset 0x0000_0000). Figure 2-11 shows the address filtering flow.

The decisions shown in white are made in the TEMAC component while the decisions shown in gray are made based on the Control register settings. The filtering functions includes:

TEMAC component functions

- Programmable unicast destination address matching
- Four programmable multicast address matching
- Broadcast address recognition (0xFFFF_FFFF_FFFF)
- Optional pass through mode with address filter disabled (promiscuous mode)
- Pause control frame address recognition (0x0100 00C2 8001)

Control Register enabled functions

- Enable or reject received multicast frames
- Enable or reject received broadcast frames

Receive address filtering eliminates the software overhead required to process frames that are not relevant to a particular Ethernet interface by checking the Destination Address (DA) field of the received frame.

The unicast address and multicast addresses are programmed in software through the AXI4-Lite bus as are the Address Filter enable bit, Multicast Address enable bit, and Broadcast Address enable bit. The pause frame address and broadcast address are predefined and do not need programming. See the footnote in Table 2-43 for a more detailed description on the conditions that can cause the receive reject interrupt to be set.

Using the Address Filters

There are four, 48-bit (6-byte) registers, that can be used for address filtering. The address filters can be accessed by first setting the Filter Mask Index in the Filter Mask Index register. While the Filter Mask Index is set, the Address Filter register can be set accessed (Figure 2-10).

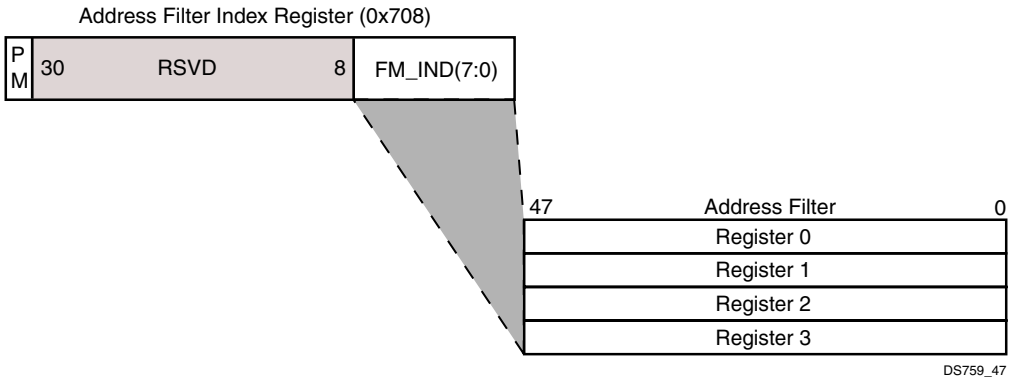


Figure 2-10: **Address Filter Access**

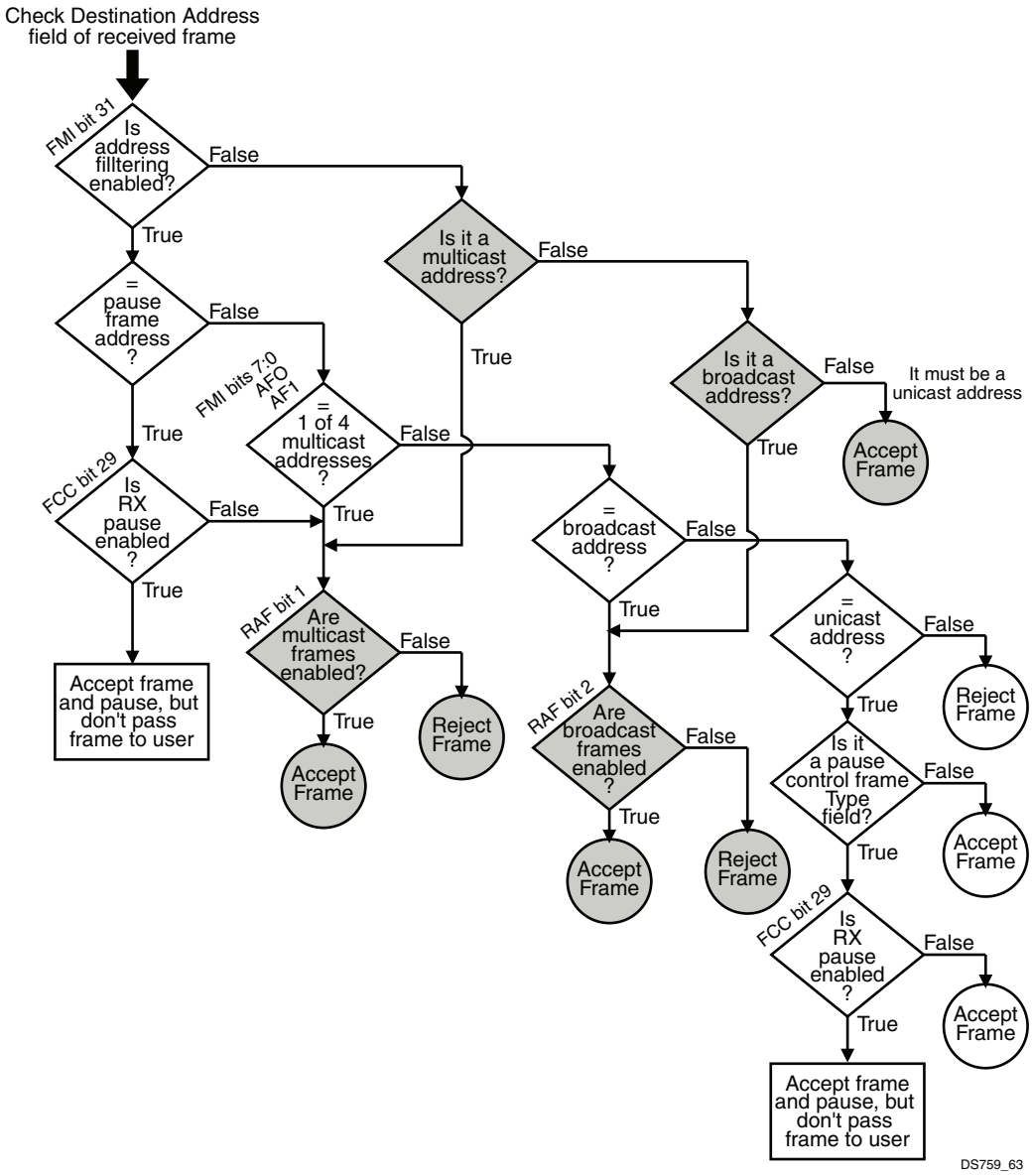


Figure 2-11: **Receive Address Basic Filtering Flow**

Extended Multicast Address Filtering Mode

General

Currently the TEMAC core provides up to four multicast addresses that can be specified for receive address validation (if an incoming multicast frame receive address matches one of the four specified addresses, it is accepted). You might want to use many more multicast address values to filter receive addresses. While this can be supported with promiscuous mode and software application filtering, some degree of hardware offloading is desired to reduce processor utilization.

Extended multicast filtering is included at build time by setting the C_MCAST_EXTEND parameter to 1. This provides additional logic for address filtering beyond what is built into the TEMAC core itself. The TEMAC core prevents receiving any multicast frames if they do not match one of the four entries in the built-in multicast address table. As a result, the TEMAC core has to be placed in promiscuous address mode to force it to pass all multicast frames through to the extended multicast address filtering logic.

TEMAC in promiscuous address mode will also pass through all unicast address frames. To avoid increasing the processor load for unicast address filtering, additional unicast address filtering has to be added to the extended multicast address filtering logic. You must make sure that the TEMAC core is in promiscuous receive address mode when using this extended multicast address filtering mode.

Implementation Details

Received multicast frames that meet all other hardware verification requirements receive a first level address filtering in hardware. Frames that pass this initial filtering are passed up to software drivers with information provided by hardware to assist the software drivers in providing the second level/final address filtering. If the frame does not pass hardware filtering, the frame is dropped and no action is required by the software drivers.

While a MAC multicast address is defined as any 48-bit MAC address that has bit 0 (LSB) set to 1 (for example 01:00:00:00:00:00), in most cases the MAC multicast address is created from a IP multicast address as shown in Figure 2-12. It is these IP multicast addresses that are a subset of MAC multicast addresses that are filtered by the extended multicast address filtering mode.

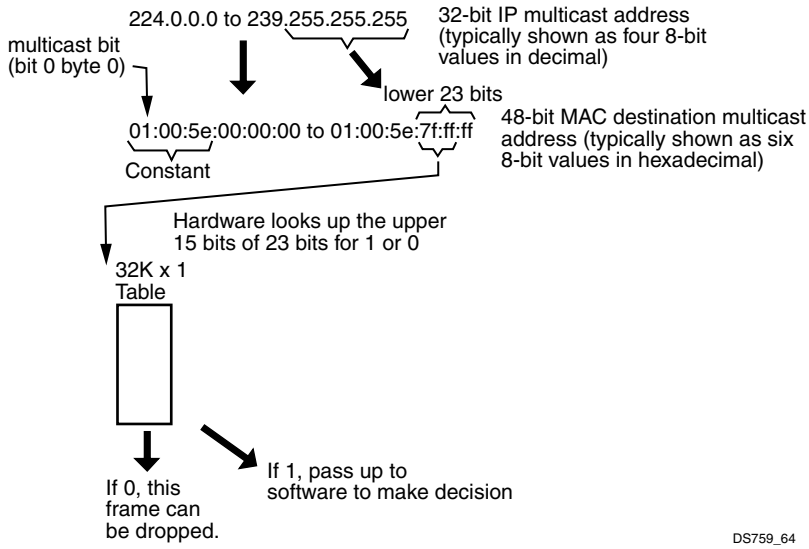


Figure 2-12: **Mapping IP Multicast Addresses to MAC Multicast Addresses**

When a multicast address frame is received while this extended multicast filtering is enabled, the AXI Ethernet core first verifies that the first 24 bits are 01:00:5E and then uses the upper 15 bits of the unique 23-bit MAC multicast address to index this memory. If the associated memory location contains a 1, the frame is accepted and passed up to software for a comparison on the full 23-bit address. If the memory location is a 0 or the upper 24 bits are not 01:00:5E then the frame is not accepted and it is dropped. The memory is 1-bit wide but is addressed on 32-bit word boundaries. The memory is 32K deep. This table must be initialized by software using the AXI4-Lite interface.

When using the extended multicast address filtering, the TEMAC core must be set to promiscuous mode so that all frames are available for filtering. In this mode the TEMAC core no longer checks for a unicast address match. Additional registers are available to provide unicast address filtering while in this mode. Table 2-47 shows the Receive VLAN Tag register bit definitions and Table 2-48 shows the Unicast Address Word Lower register bit definitions.

For builds that have the extended multicast address filtering enabled, promiscuous mode can be achieved by making sure that the TEMAC core is in promiscuous mode, and by clearing the EMultiFltrEnbl bit (bit 12) in the Reset and Address Filter Register — Offset 0x0000_0000 in Chapter 2.

When a received frame is accepted and passed up to software, additional information is provided in the receive AXI4-Stream Status words to help the software perform the additional address filtering with less overhead.

Receive AXI4-Stream Status words 0 and 1 include the destination address of the frame, Word 2 includes bits to indicate if the frame had a destination address that was the broadcast address, a MAC multicast address, or an IP multicast address. If none of those bits are set, it was a unicast address. See Mapping AXI DMA IP Buffer Descriptor Fields to AXI4-Stream Fields for more information.

This allows the Vivado® IDE to make decisions about the destination address without accessing the address from within the receive AXI4-Stream Data transfer. When using a Xilinx AXI DMA core, the information needed for filtering is in the buffer descriptor. A decision can then be made regarding accepting or rejecting the frame without accessing the data buffer itself, thus reducing memory access and buffer indexing overhead.

Flow Control

The flow control function is defined by IEEE Std 802.3-2008 Clause 31. The AXI Ethernet core can be configured to send pause frames and to act upon the pause frames received. These two behaviors can be configured independently (asymmetrically). To enable or disable transmit and receive flow control, see the FCC register (Table 2-57).

Flow control can be used to prevent data loss when an Ethernet interface is unable to process frames fast enough to keep up with the rate of frames provided by another Ethernet interface. When this occurs, the Ethernet interface that requires relief can transmit a pause control frame to the link partner to request it cease transmitting for a defined period of time.

Transmitting a Pause Control Frame

For the AXI Ethernet core, a pause frame transmission can be initiated by writing a pause value to the Transmit Pause Frame Register — Offset 0x0000_0004 in Chapter 2 while transmit pause processing is enabled (FCC register bit 30 in 1 Table 2-57).

Requesting the transmit of a pause frame does not interrupt a transmission in progress but the pause frame is transmitted after the frame in progress. A request to transmit a pause frame results in the transmission of a pause frame even if the transmitter itself is already paused due to the reception of a pause frame.

The destination address supplied with the transmitted pause control frame can be set by writing to the RCW0 and RCW1 registers (Table 2-55).

Receiving a Pause Control Frame

When an error free frame is received by AXI Ethernet core, it examines the following information:

1. The destination address field is compared to the pause control address and the configured unicast address.
2. The Length/Type field is compared against the control type code (0x8808).
3. The opcode field contents are matched against the pause control opcode (0x0001).

If compare step 2 or 3 fails or if flow control for the receiver is disabled (FCC register bit 29 is 0 Table 2-57), the frame is ignored by the flow control logic and is passed to you. If the frames pass all 3 compare steps and receive flow control is enabled, the pause parameter in the frame is used to inhibit transmitter operation for the time defined in the IEEE Std 802.3-2008 specification, a Receive Reject interrupt is activated (see bit 28 in Table 2-43), and the frame is not passed up to software. If the transmitter is paused and a second pause frame is received, the current pause value of the transmitter is replaced with the new pause value received, including a possible value of 0x0.

Extended VLAN Support

VLAN General Information

Virtual Local Area Network (VLAN) frames are used to segregate Ethernet traffic within a larger physical LAN. VLAN frames are created by inserting a 4-byte VLAN TAG field in an Ethernet frame where the 2-byte Type/Length field would normally occur thus extending the overall frame by 4 bytes. The VLAN TAG field is further broken down into additional fields as shown in Figure 2-13.

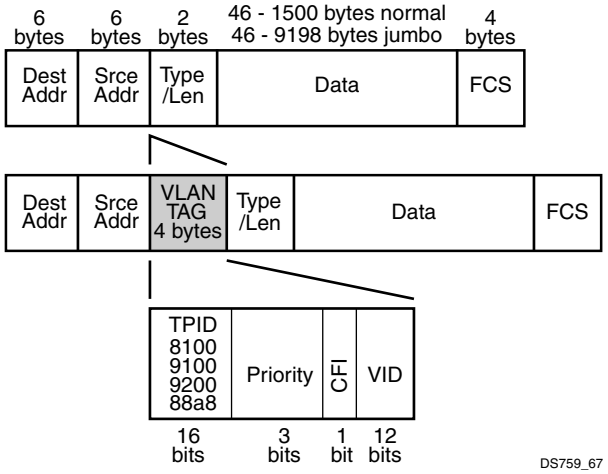


Figure 2-13: **VLAN Frame Showing VLAN Tag Field**

The TEMAC core provides basic VLAN support that can be enabled or disabled independently. This basic support recognizes VLAN frames that have a TPID value of 0x8100. When basic VLAN function is enabled, the TEMAC core allows good VLAN frames with this TPID value to be processed for validation and address filtering rather than being dropped. However, some applications require using a TPID value other than 0x8100, or have multiple VLAN tags within one frame (referred to as double tagging, triple tagging). Additionally, some common operations are performed on VLAN frames that can be off-loaded from software to hardware to reduce processor utilization. Some of these tasks, translation, stripping, and auto tagging, are available when the extended VLAN support is included in the core at build-time by setting the C_TXVLAN_* and C_RXVLAN_* parameters.

The extended VLAN functions are available individually and independently between the transmit and receive paths. To use the extended VLAN functions, the circuitry must be included at build time by setting the appropriate parameters and also the functions must be enabled at run time by setting the New Functions enable bit (bit 11) of the Reset and Address Filter Register — Offset 0x0000_0000 in Chapter 2.

VLAN Translation

VLAN translation enables the AXI Ethernet core to replace the VLAN ID (VID) value of the VLAN Tag field of a VLAN frame with a new VID as it passes through the AXI Ethernet core in either the transmit or receive direction.

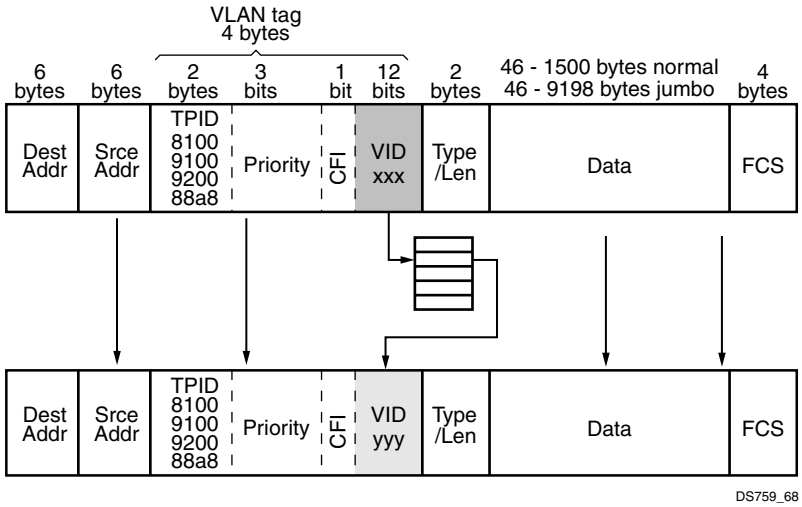


Figure 2-14: **VLAN VID Translation**

The TEMAC core does not recognize transmitting or receiving VLAN frames with a TPID other than 0x8100 when VLAN mode is enabled. If VLAN mode is disabled, then the maximum length of a normal frame is not extended from 1518 to 1522 bytes. Additionally, multiple tagging is also not supported because of the even larger frame sizes.

To support multiple VLAN tagging and the use of TPID values other than 0x8100 in the outer tag, jumbo frame mode must be used with basic VLAN mode disabled. Using this setting eliminates automatic invalidating (by the TEMAC core) of any frames that normally would be too large for *normal* frame sizes. You must enable jumbo frame mode and disable VLAN mode when needed for extended VLAN mode.

Transmit Path

When transmitting frames, the outgoing frame is detected as a VLAN frame by recognizing a VLAN Tag Protocol Identifier value (TPID) in the Type/Length field by comparing it against user defined values in the VLAN TPID Word 0 Register — Offset 0x0000_0028 in Chapter 2 and VLAN TPID Word 1 Register — Offset 0x0000_002C in Chapter 2. The TPID values are shared between the receive and transmit paths.

After a VLAN frame is identified, the 12-bit Unique VLAN Identifier (VID) is used to access the TEMAC Receive Configuration Word 0 Register – Offset 0x0000_0400 in Chapter 2 to supply a replacement VID value which is substituted into the outgoing frame.

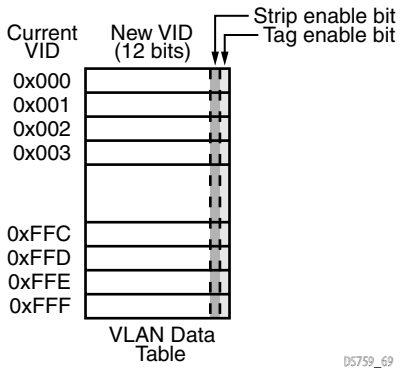


Figure 2-15: **VLAN Data Table**

Using transmit In-Band FCS mode of the TEMAC core is not allowed when using VLAN translation because the user-provided FCS field value is not correct for the new VID field. For double and triple-tagged VLAN frames, only the outer VID is translated. The following TPID values are commonly used to flag VLAN frames: 0x8100, 0x9100, 0x9200, and 0x88A8. However, the TPID values used to identify VLAN frames are programmable through the TPID registers. Transmit and receive VLAN translation can be enabled separately with their respective parameters. For VID values that do not need to be translated, the VLAN data table location associated with their value must be initialized to that same value.

Receive Path

The receive operates similarly to the transmit side. The frame first passes through address filtering and validation processing before being checked for a VLAN TPID. Receive FCS stripping in the TEMAC core is required when using VLAN translation because the FCS field that arrives with the frame is no longer valid with the new TPID value. Although receive stripping is enabled, any padding, if present, is not stripped due to the TYPE/LENGTH field of the receive frame containing a VLAN tag rather than a length value.

VLAN Tagging and Double Tagging

VLAN tagging, also referred to as stacking, allows the TEMAC to insert a pre-defined VLAN tag in select Ethernet frames as they pass through the core in either the transmit or receive direction.

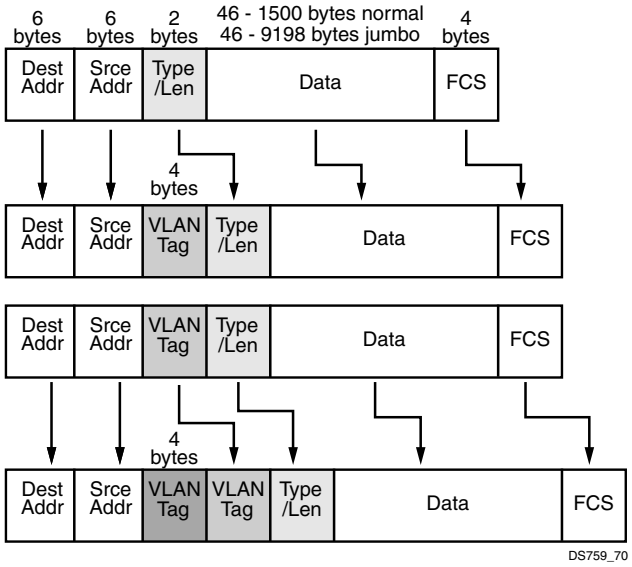


Figure 2-16: **VLAN Tagging**

One VLAN tag is added depending on mode of operation:

- Non-VLAN frames get one VLAN tag added to become single VLAN tagged frames.
- VLAN tagged frames receive another VLAN tag and no checking is performed to see how many VLAN tags the frame already has (if there were three tags, it now has four).

Therefore, in cases that require adding a VLAN tag, one VLAN tag is added to the existing frame.

The TEMAC core basic VLAN mode extends the maximum normal frame size validation by 4 bytes. This mode does not extend to multiple VLAN tagging. Multiple VLAN frames that exceed 1522 bytes would be discarded as being too long. As mentioned previously, this requires the use of jumbo frame mode which eliminates the automatic invalidation of frames that normally would be too large for normal frame sizes.

When VLAN tagging is enabled at build time with the appropriate parameter, a field in the Reset and Address Filter Register — Offset 0x0000_0000 in Chapter 2 is used to select one of four VLAN tagging modes and the Transmit VLAN Tag Register — Offset 0x0000_0018 in Chapter 2 and Receive VLAN Tag Register — Offset 0x0000_001C is used to hold the VLAN tag value which is inserted. The four VLAN tagging modes which are selectable at run time are:

- Do not add tags to any frames
- Add one tag to all frames
- Add one tag only to frames that are already VLAN tagged
- Add one tag only to select frames that are already VLAN tagged based on VID value

The fourth mode requires a method for specifying which tagged frames should receive an additional VLAN tag. The TEMAC Receive Configuration Word 0 Register – Offset 0x0000_0400 in Chapter 2 and Receive VLAN Data Table — Offset 0x0000_8000-0x0000_BFFF in Chapter 2 are used for this purpose. A "1" in the tag enable field for a TPID value indicates that frame should receive an additional tag. Again, transmit In-Band FCS mode is not allowed and receive FCS stripping is required when using VLAN tagging because FCS field value will not be correct for the frame with the additional VLAN tag. Although receive stripping is enabled, any padding, if present, is not stripped because the TYPE / LENGTH field of the receive frame contains a VLAN tag rather than a length value. However, the length field is still present.

VLAN Stripping

VLAN stripping allows the TEMAC to remove a VLAN tag in select Ethernet frames as they pass through the AXI Ethernet core in either the transmit or receive direction.

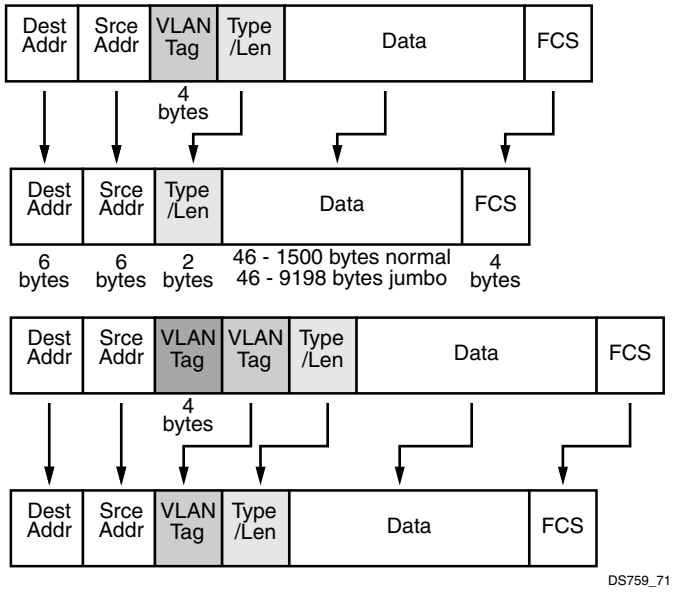


Figure 2-17: **VLAN Stripping**

One VLAN tag is removed:

- Non-VLAN frames are not changed.
- VLAN tagged frames have the outer VLAN tag removed, and the AXI Ethernet core does not check to see how many VLAN tags it already has (if there are four tags, AXI Ethernet core makes it three).

When VLAN stripping is enabled at build time with the appropriate parameter, a field in the Reset and Address Filter Register — Offset 0x0000_0000 in Chapter 2 is used to select one of three VLAN stripping modes.

- Do not strip tags from any frames
- Strip one tag from all VLAN tagged frames
- Strip one tag only from select VLAN tagged frames based on VID value

The third mode requires a method for specifying which tagged frames should be stripped. The TEMAC Receive Configuration Word 0 Register – Offset 0x0000_0400 and Receive VLAN Data Table — Offset 0x0000_8000-0x0000_BFFF in Chapter 2 are used for this purpose. A "1" in the strip enable field for a TPID value indicates that frame should have its VLAN tag stripped.

Again, transmit In-Band FCS mode is not allowed and receive FCS stripping is required when using VLAN stripping because FCS field value would not be correct for the frame with the VLAN tag removed. Although receive stripping is enabled, any padding, if present, is not stripped due to the TYPE/LENGTH field of the receive frame containing a VLAN tag rather than a length value.

Order of VLAN Functions When Combined

When multiple VLAN functions are combined, the order of processing for both transmit and receive is:

1. VLAN Stripping.
2. VLAN Translation.
3. VLAN Tagging.

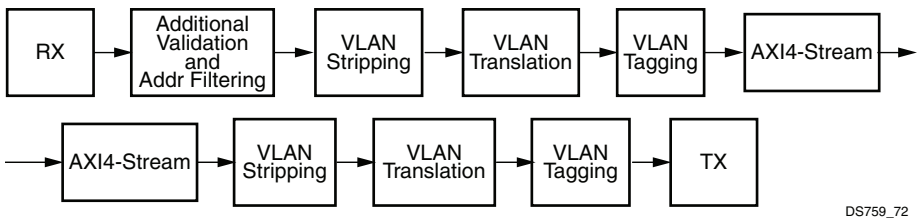


Figure 2-18: **Order of Extended VLAN Functions**

Using the MII Management to Access Internal or External PHY Registers

The MII Management interface is used to access PHY registers. These PHYs can either be external to the FPGA or, internal to the FPGA. In case of SGMII or 1000BaseX modes, one PHY is present internal to the FPGA. The details of PHY registers can be found in their respective documents. More details are added based on the IEEE standard. For 1000BASE-X PCS/PMA Management registers. See the *LogiCORE IP Ethernet 1000BASE-X PCS/PMA or SGMII Product Guide* (PG047) [Ref 2].

IMPORTANT: *Prior to any MII Management accesses, the MII Management Configuration register must be written with a valid CLOCK_DIVIDE value and the MDIOEN bit must be set.*

The value of the PHYAD and REGAD fields in the MII Management Control register determines which PHY registers are accessed. Each PHY, internal or external, should have a unique 5-bit PHY address excluding "00000" which is define as a broadcast PHY address. The MII Management interface is defined in IEEE Std 802.3, Clause 22 as a two-wire interface with a shared bidirectional serial data bus and a clock with a maximum permitted frequency of 2.5 MHz. As a result, MII Management access can take many AXI4-Lite clock cycles to complete.

To write to a PHY register, the data must be written to the MII Management Data Write register. The PHY address (PHYAD) and PHY register (REGAD) are written to the MII Management Control register. Setting the Initiate bit in the MII Management Control register starts the operation. The format of the PHYAD and REGAD in the MII Management Control register is shown in Figure 2-19.

To read from a PHY register, the PHY address and register number are written to the MII Management Control register. Setting the Initiate bit in the MII Management Control register starts the operation. When the operation completes, the PHY register value is available in the MII Management Read Data register. To access the internal SGMII or 1000BASE-X registers, the PHYAD should match that set by the parameter C_PHYADDR.

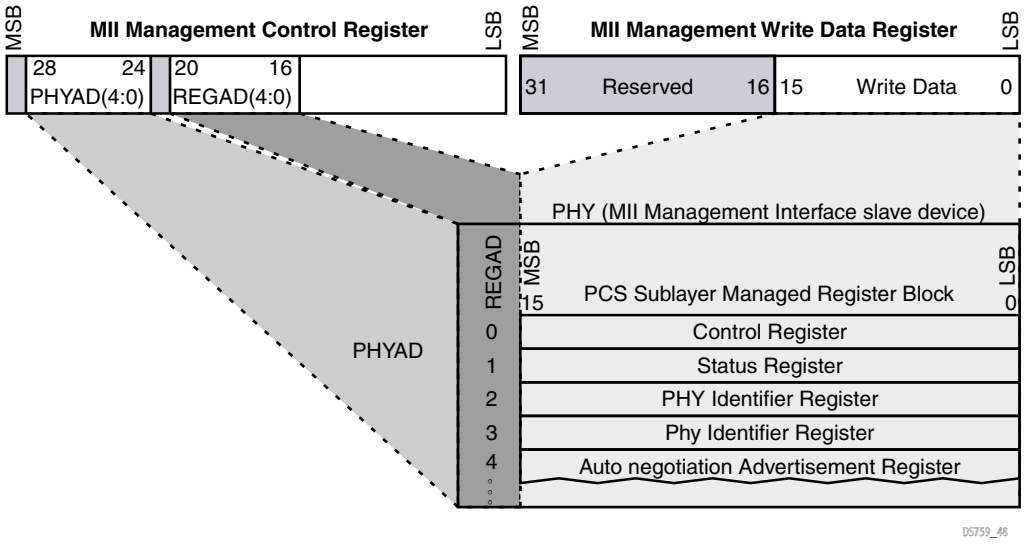


Figure 2-19: **MII Management Write Register Field Mapping**

Table 2-2 provides an example of a PHY register write through the MII Management Interface.

Table 2-2: **Example of a PHY Register Write via the MII Management Interface**

Register	Access	Value	Activity
MIIM Write Data register	Write	0x0000ABCD	Write the value that is written to the PHY register (0xABCD in this case).
MII Management Control register	Write	0x01024800	Initiate the write to the MII Management Control register by setting the PHYAD (00001), REGAD(00010), OP (01), and Initiate bit (1).
MII Management Control register	Read	0x01024880	Poll the MII Management Control register bit 7. When set to 1, the data has been written.

Table 2-3: **Example of a PHY Register Read via the MII Management Interface**

Register	Access	Value	Activity
MII Management Control register	Write	0x01028800	Initiate the write to the MII Management Control register by setting the PHYAD (00001), REGAD(00001), OP (10), and Initiate bit (1).
MII Management Control register	Read	0x01028880	Poll the MII Management Control register bit 7. When set to 1, the read data is available.
MII Management Read Data register	Read		Read data provided by PHY register.

After a transfer has been initiated on the MDIO interface, it is also possible to access a non-MDIO register in the memory space normally. The MDIO transfer has completed when the RDY bit in the MII Management Control register is 1. This bit can either be polled, or the interrupt can be monitored.

If the MII Management Control register is rewritten in an attempt to start a new transfer, the data is captured; however, the transfer does not take place until the current transaction completes. If the previous transaction was a read, the read data is valid when the first transaction completes. If the previous transaction was a write, the MII Management Write Data register can be written after the first transaction completes. The MII Management Control register should be checked to ensure all MDIO transactions have been completed before accessing the data or initiating a transfer.

Serial Gigabit Media Independent Interface

The Serial-GMII (SGMII) is an alternative interface to the GMII/MII that converts the parallel interface of the GMII into a serial format. This radically reduces the I/O count and is therefore often favored by PCB designers. This is achieved by using a serial transceiver. SGMII can carry Ethernet traffic at 10 Mb/s, 100 Mb/s, and 1 Gb/s.

The SGMII physical interface was defined by Cisco Systems, Inc. The data signals operate at a rate of 1.25 Gb/s. Differential pairs are used to provide signal integrity and minimize noise. The sideband clock signals defined in the specification are not implemented in AXI Ethernet core.

Instead, the transceiver is used to transmit and receive the differential data at the required rate using clock data recovery. For more information on SGMII, see the *Serial GMII Specification v1.7*.

Note: SGMII over LVDS is supported.

SGMII Auto-Negotiation

The external SGMII capable PHY device performs auto negotiation with its link partner on the PHY Link (Ethernet bus) resolving operational speed and duplex mode and then in turn performs a secondary auto negotiation with the transceiver across the SGMII Link. This transfers the results of the PHY with Link Partner auto negotiation across the SGMII to the AXI Ethernet core.

The results of the SGMII auto negotiation can be read from the SGMII Management Auto negotiation Link Partner Ability Base register (Table 2-19). The duplex mode and speed of AXI Ethernet core should then be set to match (see Chapter 2, Product Specification).

There are two methods that can be used for the completion of an auto negotiation cycle:

1. Polling the auto negotiation complete bit of SGMII Management Status register (Register 1, bit 5 Table 2-6).
2. Using the auto negotiation complete interrupt (Interrupt Status Register — Offset 0x0000_000C in Chapter 2 and SGMII Management Auto Negotiation Interrupt Control register, Table 2-15.)

When placed into loopback, data is routed from the transmitter to the receiver path at the last possible point in the PCS/PMA sublayer. This is immediately before the transceiver interface. When placed into loopback, a constant stream of Idle code groups is transmitted through the transceiver.

Loopback in this position allows test frames to be looped back within the system without allowing them to be received by the link partner (the device connected on the other end of the Ethernet. The transmission of Idles allows the link partner to remain in synchronization so that no fault is reported.

Gigabit Ethernet PCS/PMA Management Registers

Gigabit Ethernet PCS PMA has configuration registers as defined in IEEE 802.3. These have an address range from 0 to 15. These registers are configured using the MDIO interface. These registers are given here for quick reference. For more information related to these registers see the *LogiCORE IP Ethernet 1000BASE-X PCS/PMA or SGMII Product Guide* (PG047) [Ref 2].

These registers contain information relating to the operation of the 1000BASE-X PCS/PMA sublayer, including the status of the physical Ethernet link (PHY Link). Additionally, these registers are directly involved in the operation of the 1000BASE-X auto negotiation function which occurs between AXI Ethernet core and its link partner, the Ethernet device connected at the far end of the PHY Link. These registers are accessed through the MII Management interface (Using the Address Filters). These registers are only valid when using the 1000BASE-X PHY interface.

Table 2-4: **Gigabit Ethernet PCS PMA Internal Management Registers**

Register Name	Register Address (REGAD)
Control register (Register 0)	0
Status register (Register 1)	1
PHY Identifier (Register 2 and 3)	2,3
Auto Negotiation Advertisement register (Register 4)	4
Auto Negotiation Link Partner Ability Base register (Register 5)	5
Auto Negotiation Expansion register (Register 6)	6
Auto Negotiation Next Page Transmit register (Register 7)	7
Auto Negotiation Next Page Receive register (Register 8)	8
Extended Status register (Register 15)	15
Vendor Specific register: Auto Negotiation Interrupt Control register (Register 16)	16
Vendor Specific register: Loopback Control register (Register 17)	17

Table 2-5: **Control Register (Register 0)**

Bits	Name	Core Access	Reset Value	Description
15	Reset	Read/write Self clearing	0	1 = Core Reset 0 = Normal Operation
14	Loopback	Read/write	0	1 = Enable Loopback Mode 0 = Disable Loopback Mode When used with a device-specific transceiver, the core is placed in internal loopback mode. With the TBI version, Bit 1 is connected to ewrap. When set to '1,' indicates to the external PMA module to enter loopback mode.
13	Speed Selection (LSB)	Returns 0	0	Always returns a 0 for this bit. Together with bit 0.6, speed selection of 1000 Mb/s is identified
12	Auto-Negotiation Enable	Read/write	1	1 = Enable Auto-Negotiation Process 0 = Disable Auto-Negotiation Process
11	Power Down	Read/ write	0	1 = Power down 0 = Normal operation With the PMA option, when set to '1' the device-specific transceiver is placed in a low-power state. This bit requires a reset (see bit 0.15) to clear. With the TBI version this register bit has no effect.

Table 2-5: **Control Register (Register 0) *(Cont'd)***

Bits	Name	Core Access	Reset Value	Description
10	Isolate (1)	Read/write	1	1 = Electrically Isolate PHY from GMII 0 = Normal operation
9	Restart Auto-Negotiation	Read/write Self clearing	0	1 = Restart Auto-Negotiation Process 0 = Normal Operation
8	Duplex Mode	Returns 1	1	Always returns a '1' for this bit to signal Full-Duplex Mode.
7	Collision Test	Returns 0	0	Always returns a '0' for this bit to disable COL test.
6	Speed Selection (MSB)	Returns 1	1	Always returns a '1' for this bit. Together with bit 0.13, speed selection of 1000 Mb/s is identified.
5	Unidirectional Enable	Read/ write	0	Enable transmit regardless of whether a valid link has been established. This feature is only possible if Auto-Negotiation Enable bit 0.12 is disabled
4–0	Reserved	Returns 0s	00000	Always return 0s, writes ignored.

1. When using the 1000Base-X TEMAC core (C_TYPE = 1 and C_PHY_TYPE = 5), set the isolate bit to zero (Control register 0 bit 10). The core is not operational until this is completed.

Table 2-6 shows the Gigabit Ethernet PCS PMA Management Status register bit definitions.

Table 2-6: **Management Status Register (Register 1) Bit Definitions**

Bits	Name	Core Access	Reset Value	Description
15	100BASE-T4	Returns 0	0	Always returns a 0 for this bit because 100BASE-T4 is not supported.
14	100BASE-X Full Duplex	Returns 0	0	Always returns a 0 for this bit because 100BASE-X full duplex is not supported.
13	100BASE-X Half Duplex	Returns 0	0	Always returns a 0 for this bit because 100BASE-X half duplex is not supported.
12	10 Mb/s Full Duplex	Returns 0	0	Always returns a 0 for this bit because 10 Mb/s full duplex is not supported.
11	10 Mb/s Half Duplex	Returns 0	0	Always returns a 0 for this bit because 10 Mb/s half duplex is not supported.
10	100BASE-T2 Full Duplex	Returns 0	0	Always returns a 0 for this bit because 100BASE-T2 full duplex is not supported.
9	100BASE-T2 Half Duplex	Returns 0	0	Always returns a 0 for this bit because 100BASE-T2 half duplex is not supported.
8	Extended Status	Returns 1	1	Always returns a 1 for this bit indicating the presence of the extended register (register 15).
7	Unidirectional Ability	Returns 1	1	Always returns a 1.
6	MF Preamble Suppression	Returns 1	1	Always returns a 1 for this bit to indicate the support of management frame preamble suppression.
5	Auto Negotiation Complete	Read	0	0 – auto negotiation process not completed 1 – auto negotiation process complete
4	Remote Fault	Read only self clearing on read	0	0 – no remote fault condition detected 1 – remote fault condition detected
3	Auto Negotiation Ability	Returns 1	1	Always returns a 1 for this bit indicating that the PHY is capable of auto negotiation.
2	Link Status	Read only self clearing on read	0	0 – PHY Link is down 1 – PHY Link is up
1	Jabber Detect	Returns 0	0	Always returns a 0 for this bit because no jabber detect is supported.
0	Extended Capability	Returns 0	0	Always returns a 0 for this bit because no extended register set is supported.

Table 2-7 shows the first Management PHY Identifier register bit definitions.

Table 2-7: **Management PHY Identifier (Register 2) Bit Definitions**

Bits	Name	Core Access	Reset Value	Description
0–15	OUI	Read	0x0000	Organizationally Unique Identifier (OUI).

Table 2-8 shows the second Management PHY Identifier register bit definitions.

Table 2-8: **Management PHY Identifier (Register 3) Bit Definitions**

Bits	Name	Core Access	Reset Value	Description
10–15	OUI	Read	000000	Organizationally Unique Identifier (OUI).
4–9	MMN	Returns 0	000000	Manufacturer Model Number. Always returns 0s.
0–3	Revision	Returns 0	0000	Revision Number. Always returns 0s.

Table 2-18 shows the Management Auto Negotiation Advertisement register bit definitions.

Table 2-9: **Management Auto Negotiation Advertisement Register (Register 4) Bit Definitions**

Bits	Name	Core Access	Reset Value	Description
0–15	All Bits	Read	0x0001	SGMII defined value sent from the MAC to the PHY.

Table 2-10 shows the TEMAC Internal 1000BASE-X PCS/PMA Management Auto Negotiation Link Partner Ability Base register bit definitions.

Table 2-10: **Management Auto Negotiation Link Partner Ability Base Register (Register 5) Bit Definitions**

Bits	Name	Core Access	Reset Value	Description
15	PHY Link Status	Read	1	This refers to the link status of the PHY with its Link Partner across the medium. 0 – Link Down 1 – Link Up
14	Acknowledge	Read	0	Used by Auto-negotiation function to indicate reception of a link partner base or next page
13	Reserved	Returns 0	0	Always return zero.
12	Duplex Mode	Read	0	1 = Full Duplex 0 = Half Duplex
10–11	Speed	Read	00	00 – Reserved 01 – 1000 Mb/s 10 – 100 Mb/s 11 – 10 Mb/s
1–9	Reserved	Returns 0s	000000000	Always return zeros.
0	Reserved	Returns 1	1	Always return one.

Table 2-11 shows the Management Auto Negotiation Expansion register bit definitions.

Table 2-11: **Management Auto Negotiation Expansion Register (Register 6) Bit Definitions**

Bits	Name	Core Access	Reset Value	Description
3–15	Reserved	Returns 0s	0x0	Always return zeros.
2	Next Page Able	Returns 1	1	Always returns a 1 for this bit because the device is Next Page Able.
1	Page Received	Read self clearing on read	0	0 – a new page is not received 1 – a new page is received
0	Reserved	Returns 0s	0	Always return zeros.

Table 2-12 shows the Management Auto Negotiation Next Page Transmit register bit definitions.

Table 2-12: **Management Auto Negotiation Next Page Transmit Register (Register 7) Bit Definitions**

Bits	Name	Core Access	Reset Value	Description
15	Next Page	Read/Write	0	0 – last page 1 – additional next pages to follow
14	Reserved	Returns 0s	0	Always return zeros.
13	Message Page	Read/Write	1	0 – unformatted page 1 – message page
12	Acknowledge 2	Read/Write	0	0 – cannot comply with message 1 – complies with message
11	Toggle	Read	0	Value toggles between subsequent pages.
0–10	Message or unformatted Code Field	Read/Write	0x001 (null message code)	Message code field or unformatted page encoding as dictated by bit 13.

Table 2-13 shows the Management Auto Negotiation Next Page Receive register bit definitions.

Table 2-13: **Management Auto Negotiation Next Page Receive Register (Register 8) Bit Definitions**

Bits	Name	Core Access	Reset Value	Description
15	Next Page	Read	0	0 – last page 1 – additional next pages to follow
14	Acknowledge	Read	0	Used by auto negotiation function to indicate reception of a link partner base or next page.
13	Message Page	Read	0	0 – unformatted page 1 – message page
12	Acknowledge 2	Read	0	0 – cannot comply with message 1 – complies with message
11	Toggle	Read	0	Value toggles between subsequent pages.
0–10	Message or unformatted Code Field	Read	0x0 (null message code)	Message code field or unformatted page encoding as dictated by bit 13.

Table 2-14 shows the Management Extended Status register bit definitions.

Table 2-14: **Management Extended Status Register (Register 15) Bit Definitions**

Bits	Name	Core Access	Reset Value	Description
15	1000BASE-X Full Duplex	Returns 1	1	Always returns a 1 for this bit because 1000BASE-X full duplex is supported.
14	1000BASE-X Half Duplex	Returns 0	0	Always returns a 1 for this bit because 1000BASE-X half duplex is not supported.
13	1000BASE-T Full Duplex	Returns 0	0	Always returns a 1 for this bit because 1000BASE-T full duplex is not supported.
12	1000BASE-T Half Duplex	Returns 0	0	Always returns a 1 for this bit because 1000BASE-T half duplex is not supported.
0–11	Reserved	Returns 0s	0x0	Always return zeros.

Table 2-15 shows the Management Auto Negotiation Interrupt Control register bit definitions.

Table 2-15: **Management Auto Negotiation Interrupt Control Register (Register 16) Bit Definitions**

Bits	Name	Core Access	Reset Value	Description
2–15	Reserved	Returns 0s	0	Always return zeros.
1	Interrupt Status	Read/Write	0	If the interrupt is enabled, this bit is asserted upon the completion of an auto negotiation cycle; it is only cleared by writing 0 to this bit. If the interrupt is disabled, this bit is set to 0. This is the auto negotiation complete interrupt. 0 – interrupt is asserted 1 – interrupt is not asserted
0	Interrupt Enable	Read/Write	1	0 – interrupt is disabled 1 – interrupt is enabled

Table 2-16: **Management Loopback Control Register (Register 17) Bit Definitions**

Bits	Name	Core Access	Reset Value	Description
1–15	Reserved	Returns 0s	0	Always return zeros.
0	Loopback Position	Read/Write	0	Loopback is enabled or disabled using register 0 bit 14. 0 – loopback (when enabled) occurs directly before the interface to the GTX transceiver 1 – loopback (when enabled) occurs in the GTX transceiver

1000BASE-X PCS/PMA

PCS/PMA

The Physical Coding Sublayer (PCS) for 1000BASE-X operation is defined in IEEE 802.3 clause 36 and 37 and performs the following:

- Encoding (and decoding) of GMII data octets to form a sequence of ordered sets
- 8B/10B encoding (and decoding) of the sequence ordered sets
- 1000BASE-X Auto-Negotiation for information exchange with the link partner

The Physical Medium Attachment (PMA) for 1000BASE-X operation is defined in IEEE 802.3 clause 36 and performs the following:

- Serialization (and de serialization) of code-groups for transmission (and reception) on the underlying serial PMD sublayer

- Recovery of clock from the 8B/10B coded data supplied by the PMD sublayer

1000BASE-X PCS/PMA functionality is provided by connecting the TEMAC silicon component to a GTP transceiver.

PMD

The Physical Medium Dependent (PMD) sublayer is defined in IEEE 802.3 clause 38 for 1000BASE-LX and 1000BASE-SX (long and short wave laser). This type of PMD sublayer is provided by the external GBIC or SFP optical transceiver which should be connected directly to the ports of the GTX transceiver.

1000BASE-X Auto-Negotiation

1000BASE-X auto negotiation is described in IEEE Std 802.3, clause 37. This function allows a device to advertise the supported modes of operation to a device at the remote end of a link segment (the link partner on Ethernet), and detect corresponding operational modes advertised by the link partner. The results of the auto negotiation can be read from the 1000BASE-X Management Auto negotiation Link Partner Ability Base register (Table 2-10). The duplex mode and speed of the AXI Ethernet core should then be set to match (see Chapter 2, Product Specification).

There are two methods that can be used for the completion of an auto negotiation cycle:

1. By polling the auto negotiation complete bit of 1000BASE-X Management Status register (Register 1, bit 5 Table 2-6).
2. By using the auto negotiation complete interrupt (Interrupt Status Register — Offset 0x0000_000C and 1000BASE-X Management Auto Negotiation Interrupt Control register Table 2-15.)

When placed into loopback, data is routed from the transmitter to the receiver path at the last possible point in the PCS/PMA sublayer. This is immediately before the transceiver interface. When placed into loopback, a constant stream of Idle code groups is transmitted through the transceiver. Loopback in this position allows test frames to be looped back within the system without allowing them to be received by the link partner (the device connected on the other end of the Ethernet. The transmission of Idles allows the link partner to remain in synchronization so that no fault is reported.

Loopback can be enabled or disabled by writing to the 1000BASE-X Management Control register bit 14 (Table 2-6).

Gigabit Ethernet PCS PMA Management Registers

Gigabit Ethernet PCS PMA has configuration registers as defined in IEEE 802.3. These have an address range from 0 to 15. These registers are configured using the MDIO interface. These registers are given here for quick reference. for more information related to these registers see the *LogiCORE IP Ethernet 1000BASE-X PCS/PMA or SGMII Product Guide* (PG047) [Ref 2].

These registers contain information relating to the operation of the SGMII PCS sublayer, including the status of both the SGMII Link and the physical Ethernet link (PHY Link). Additionally, these registers are directly involved in the operation of the SGMII auto negotiation function which occurs between AXI Ethernet core and the external PHY device (typically a tri-speed BASE-T PHY). These registers are accessed through the MII Management interface (Using the Address Filters). These registers are only valid when using the SGMII PHY interface. When using 1000BASE-X, AXI Ethernet core is typically connected to an external optical transceiver device such as a GBIC or SFP transceiver.

Table 2-17: **Gigabit Ethernet PCS PMA Internal Management Registers**

Register Name	Register Address (REGAD)
Control register (Register 0)	0
Status register (Register 1)	1
PHY Identifier (Register 2 and 3)	2,3
Auto Negotiation Advertisement register (Register 4)	4
Auto Negotiation Link Partner Ability Base register (Register 5)	5
Auto Negotiation Expansion register (Register 6)	6
Auto Negotiation Next Page Transmit register (Register 7)	7
Auto Negotiation Next Page Receive register (Register 8)	8
Extended Status register (Register 15)	15
Vendor Specific register: Auto Negotiation Interrupt Control register (Register 16)	16
Vendor Specific register: Loopback Control register (Register 17)	17

Table 2-18: **Management Auto Negotiation Advertisement Register (Register 4) Bit Definitions**

Bits	Name	Core Access	Reset Value	Description
15	Next Page	Read/Write	0	0 – next page functionality is not advertised 1 – next page functionality is advertised
14	Reserved	Returns 0s	0	Always return zeros.
12–13	Remote Fault	Read/Write self clearing after auto negotiation	0x0	00 – no error 01 – offline 10 – link failure 11 – auto negotiation error
9–11	Reserved	Returns 0s	0x0	Always return zeros.
7–8	Pause	Read/Write	0x3	00 – No pause 01 – Symmetric pause 10 – Asymmetric pause towards link partner 11 – both symmetric pause and asymmetric pause towards link partner
6	Half Duplex	Returns 0s	0	Always return zeros because half duplex is not supported.
5	Full Duplex	Read/Write	1	0 – full duplex mode is not advertised 1 – full duplex mode is advertised
0–4	Reserved	Returns 0s	0x0	Always return zeros.

Table 2-19 shows the Management Auto Negotiation Link Partner Ability Base register bit definitions.

Table 2-19: **Management Auto Negotiation Link Partner Ability Base Register (Register 5) Bit Definitions**

Bits	Name	Core Access	Reset Value	Description
15	Next Page	Read	0	0 – next page functionality is not supported 1 – next page functionality is supported
14	Acknowledge	Read	0	Used by the auto negotiation function to indicate reception of a link partner base or next page.
12–13	Remote Fault	Read	0x0	00 – no error 01 – offline 10 – link failure 11 – auto negotiation error
9–11	Reserved	Returns 0s	0x0	Always return zeros.
7–8	Pause	Read	0x	00 – no pause 01 – asymmetric pause supported 10 – symmetric pause supported 11 – both symmetric pause and asymmetric pause supported
6	Half Duplex	Read	0	0 – half duplex mode is not supported 1 – half duplex mode is supported
5	Full Duplex	Read	0	0 – full duplex mode is not supported 1 – full duplex mode is supported
0–4	Reserved	Returns 0s	0x0	Always return zeros.

Details of remaining registers are given in Table 2-6, Table 2-18, Table 2-8, Table 2-19, Table 2-36, Table 2-12, Table 2-13, Table 2-14, Table 2-15 and Table 2-16. More details related to these registers are available in the *LogiCORE IP Ethernet 1000BASE-X PCS/PMA or SGMII Product Guide* (PG047) [Ref 2].

Standards

This section describes the standards that are supported by the AXI Ethernet IP core.

- GMII. The Gigabit Media Independent Interface (GMII) is defined by the IEEE802.3 specification; it can provide support for Ethernet operation at 10 Mb/s, 100 Mb/s and 1 Gb/s speeds.
- MII. The Media Independent Interface (MII) is defined by the IEEE802.3 specification; it can provide support for Ethernet operation at 10 Mb/s and 100 Mb/s speeds.
- 1000BASE–X Physical Coding Sublayer (PCS), Physical Medium Attachment (PMA), and RGMII operation, as defined in the IEEE 802.3-2008 standard
- GMII to Serial-GMII (SGMII) bridge or SGMII to GMII bridge, as defined in the Serial-GMII specification (ENG-46158)
- Reduced Gigabit Media Independent Interface (RGMII), version 2.0
- IEEE802.1AS Supports clock master functionality, clock slave functionality and the Best Master Clock Algorithm (BMCA)
- IEEE802.1Qav Supports arbitration between different priority traffic and implements bandwidth policing.

These standards are implemented by the respective helper cores used in this AXI Ethernet core. For more details related to these, see the *LogiCORE IP Tri-Mode Ethernet MAC Product Guide* (PG051) [Ref 1] and *LogiCORE IP Ethernet 1000BASE-X PCS/PMA or SGMII Product Guide* (PG047) [Ref 2].

Performance

To measure the system performance of the AXI Ethernet core, a system was built in which it was added to each of the supported device families as the Device Under Test (DUT) as shown in Figure 2-20; the AXI Ethernet core represents the DUT block in Figure 2-20.

Because the AXI Ethernet core is used with other design modules in the FPGA, the utilization and timing numbers reported in this section are estimates only. When this core is combined with other designs in the system, the utilization of FPGA resources and timing of the core design can vary from the results reported here.

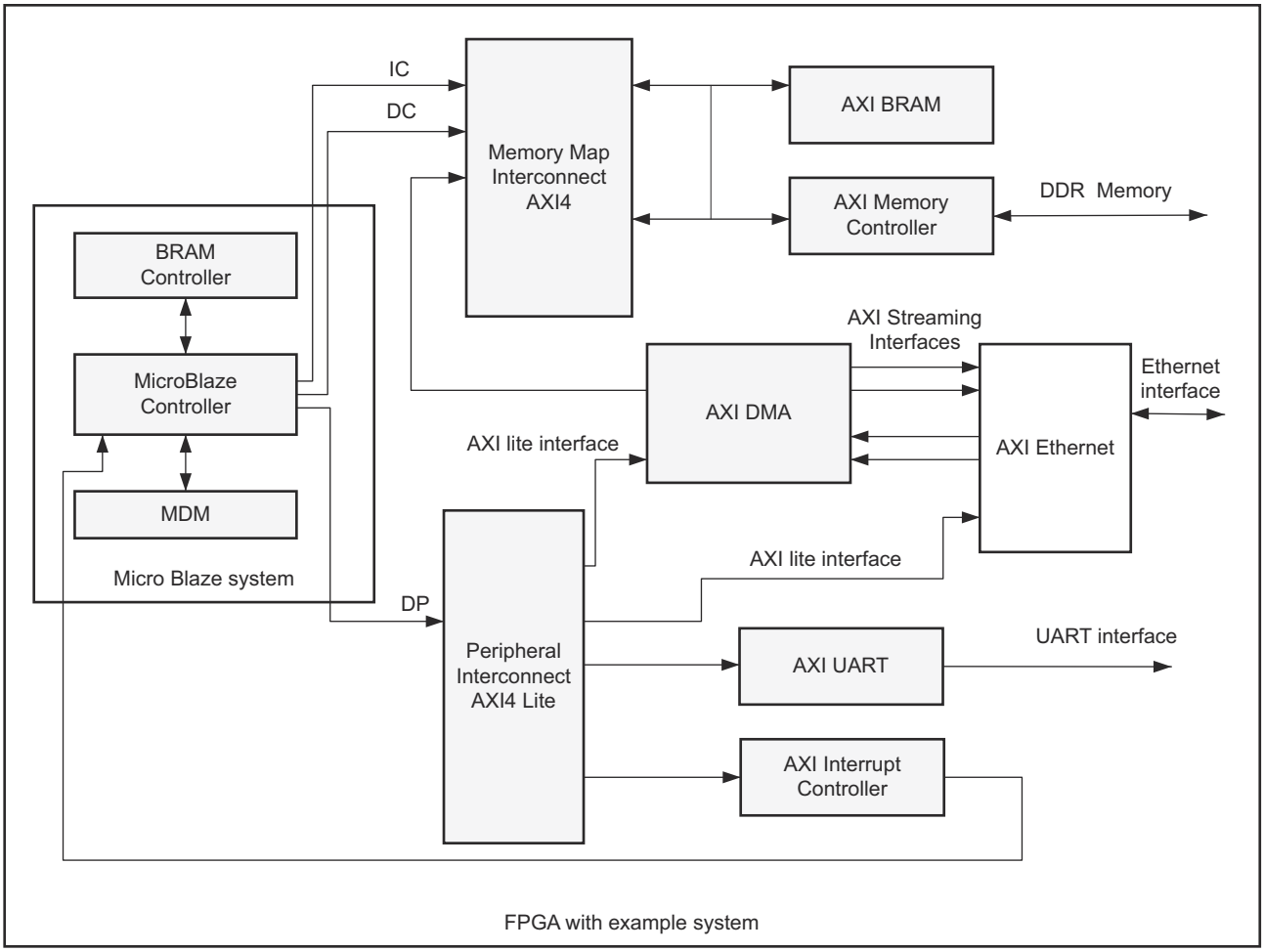


Figure 2-20: **System Configuration with the AXI Ethernet Core as the DUT for All Supported Device Families**

The target FPGA was then filled with logic to drive the LUT and block RAM utilization to approximately 70% and the I/O utilization to approximately 80%. Using the default tool options and the slowest speed grade for the target FPGA, the F_{TYP} numbers are shown in Table 2-20.

Table 2-20: **System Performance**

Target FPGA	AXI4-Lite	AXI4-Stream	MicroBlaze
Kintex-7 xc7k325tffg900-2	100	150	100
Virtex-7 xc7vx485tffg1761-2	100	150	100

The target F_{MAX} is influenced by the exact system. The F_{TYP} typical use case frequency numbers are provided here.

As the AXI Ethernet IP core represents a hierarchical design block containing multiple LogiCORE™ IP instances, the latency, max frequency, throughput and power are provided by the IP instances that are present in a given configuration.

Resource Utilization

These values were generated using the Vivado IDE. They are derived from post-synthesis reports, and might change during MAP and PAR.

Because the AXI Ethernet core is a module that is used with other designs in the FPGA, the utilization and timing numbers reported in this section are estimates and can vary from the results reported here. AXI Ethernet core benchmarks for different systems are shown in Table 2-21 for Zynq®-7000, Table 2-22 for Virtex®-7, Table 2-23 for Kintex®-7, and Table 2-24 for Artix®-7 devices. All the resource numbers are reported with shared logic in the core.

Table 2-21: **Zynq-7000 Device Performance and Resource Utilization Benchmarks (xc7z045ffg900-2)**

Configuration								Resources							
PHY_TYPE	ENABLE_AVB	TXCSUM RXCSUM	TXVLAN_TRAN RXVLAN_TRAN	TXVLAN_TAG RXVLAN_TAG	TXVLAN_STRP RXVLAN_STRP	MCAST_EXTEND	TX RX MEM	LUT as Logic	LUT as Distributed RAM	LUT as Shift Register	Register as Flip-Flop	Register as Latch	RAMB36:FIFO36	RAMB18:FIFO18	DSPs
1000Basex	FALSE	None	1	0	1	0	4k	4197	344	56	6266	0	6	2	0
1000Basex	FALSE	None	1	1	0	1	4k	4411	344	54	6483	0	7	2	0
1000Basex	FALSE	Full	0	0	0	0	4k	5037	344	51	6585	0	4	0	0
1000Basex	TRUE	Full	0	0	1	0	4k	5525	440	75	8518	0	6	2	1
GMII	FALSE	None	0	0	0	1	4k	3436	344	28	5316	0	5	0	0
GMII	FALSE	Full	0	0	0	0	4k	4437	344	30	5817	0	4	0	0
GMII	FALSE	None	1	0	0	0	4k	3505	344	33	5467	0	6	2	0
GMII	FALSE	None	0	1	0	0	4k	3603	344	33	5535	0	4	2	0
GMII	FALSE	None	0	0	1	0	4k	3518	344	32	5437	0	4	2	0
1000Basex	FALSE	None	0	0	0	1	4k	4048	344	49	6084	0	5	0	0
RGMII	FALSE	None	0	0	0	1	4k	4088	394	48	6286	0	5	0	0
RGMII	FALSE	Full	0	0	0	0	4k	5098	394	50	6787	0	4	0	0
RGMII	FALSE	None	1	0	1	0	4k	4267	394	55	6468	0	6	2	0
RGMII	FALSE	None	0	1	0	0	4k	4274	394	53	6505	0	4	2	0
GMII	TRUE	None	0	0	0	0	4k	5543	490	73	8604	0	6	0	1
SGMII	FALSE	None	0	0	0	1	4k	3840	394	33	6287	0	5	0	0
SGMII	FALSE	Full	0	0	0	0	4k	4770	394	34	6812	0	4	0	0
SGMII	FALSE	None	1	0	1	0	4k	4058	394	36	6483	0	6	2	0
SGMII	FALSE	None	0	1	0	0	4k	4117	394	37	6517	0	4	2	0
MII	FALSE	None	0	0	0	1	4k	3252	344	26	5287	0	5	0	0
MII	FALSE	Full	0	0	0	0	4k	4185	344	27	5812	0	4	0	0
MII	FALSE	None	1	0	0	0	4k	3497	344	29	5449	0	6	2	0
MII	FALSE	None	0	1	0	0	4k	3530	344	29	5517	0	4	2	0
MII	FALSE	None	0	0	1	0	4k	3400	344	29	5419	0	4	2	0
SGMII	TRUE	None	0	0	0	0	4k	5291	490	47	8595	0	6	0	1

Table 2-22: **Virtex-7 FPGA Performance and Resource Utilization Benchmarks (xc7vx485tffg1761-2)**

Configuration								Resources							
PHY_TYPE	ENABLE_AVB	TXCSUM RXCSUM	TXVLAN_TRAN RXVLAN_TRAN	TXVLAN_TAG RXVLAN_TAG	TXVLAN_STRP RXVLAN_STRP	MCAST_EXTEND	TX RX MEM	LUT as Logic	LUT as Distributed RAM	LUT as Shift Register	Register as Flip-Flop	Register as Latch	RAMB36:FIFO36	RAMB18:FIFO18	DSPs
1000Basex	FALSE	None	1	0	1	0	4k	4201	344	56	6266	0	6	2	0
1000Basex	FALSE	None	1	1	0	1	4k	4417	344	54	6483	0	7	2	0
1000Basex	FALSE	Full	0	0	0	0	4k	5041	344	51	6585	0	4	0	0
1000Basex	TRUE	Full	0	0	1	0	4k	5525	440	75	8518	0	6	2	1
RGMII	FALSE	None	0	0	0	1	4k	3464	344	28	5335	0	5	0	0
RGMII	FALSE	Full	0	0	0	0	4k	4463	344	30	5836	0	4	0	0
RGMII	FALSE	None	1	0	1	0	4k	3613	344	35	5517	0	6	2	0
RGMII	FALSE	None	0	1	0	0	4k	3627	344	33	5554	0	4	2	0
GMII	FALSE	None	0	0	0	1	4k	3440	344	28	5316	0	5	0	0
GMII	FALSE	Full	0	0	0	0	4k	4441	344	30	5817	0	4	0	0
GMII	FALSE	None	1	0	0	0	4k	3508	344	33	5467	0	6	2	0
GMII	FALSE	None	0	1	0	0	4k	3603	344	33	5535	0	4	2	0
GMII	FALSE	None	0	0	1	0	4k	3519	344	32	5437	0	4	2	0
GMII	TRUE	None	0	0	0	0	4k	4906	440	133	7984	0	6	0	1
1000Basex	FALSE	None	0	0	0	1	4k	4052	344	49	6084	0	5	0	0
SGMII	FALSE	None	0	0	0	1	4k	3844	394	33	6287	0	5	0	0
SGMII	FALSE	Full	0	0	0	0	4k	4772	394	35	6812	0	4	0	0
SGMII	FALSE	None	1	0	1	0	4k	4061	394	36	6483	0	6	2	0
SGMII	FALSE	None	0	1	0	0	4k	3253	344	26	5287	0	5	0	0
MII	FALSE	None	0	0	0	1	4k	4191	344	28	5812	0	4	0	0
MII	FALSE	Full	0	0	0	0	4k	3501	344	28	5449	0	6	2	0
MII	FALSE	None	1	0	0	0	4k	3539	344	30	5517	0	4	2	0
MII	FALSE	None	0	1	0	0	4k	3408	344	28	5419	0	4	2	0
MII	FALSE	None	0	0	1	0	4k	3408	344	28	5419	0	4	2	0
SGMII	TRUE	None	0	0	0	0	4k	5290	490	48	8595	0	6	0	1

Table 2-23: **Kintex-7 FPGA Performance and Resource Utilization Benchmarks (xc7k325tffg900-2)**

Configuration								Resources							
PHY_TYPE	ENABLE_AVB	TXCSUM RXCSUM	TXVLAN_TRAN RXVLAN_TRAN	TXVLAN_TAG RXVLAN_TAG	TXVLAN_STRP RXVLAN_STRP	MCAST_EXTEND	TX RX MEM	LUT as Logic	LUT as Distributed RAM	LUT as Shift Register	Register as Flip-Flop	Register as Latch	RAMB36:FIFO36	RAMB18:FIFO18	DSPs
1000Basex	FALSE	None	1	0	1	0	4k	4203	344	56	6266	0	6	2	0
1000Basex	FALSE	None	1	1	0	1	4k	4415	344	54	6483	0	7	2	0
1000Basex	FALSE	Full	0	0	0	0	4k	5038	344	51	6585	0	4	0	0
1000Basex	TRUE	Full	0	0	1	0	4k	5528	440	75	8518	0	6	2	1
RGMII	FALSE	None	0	0	0	1	4k	3478	344	28	5353	0	5	0	0
RGMII	FALSE	Full	0	0	0	0	4k	4479	344	30	5854	0	4	0	0
RGMII	FALSE	None	1	0	1	0	4k	3630	344	35	5535	0	6	2	0
RGMII	FALSE	None	0	1	0	0	4k	3644	344	33	5572	0	4	2	0
GMII	FALSE	None	0	0	0	1	4k	3441	344	28	5316	0	5	0	0
GMII	FALSE	Full	0	0	0	0	4k	4442	344	30	5817	0	4	0	0
GMII	FALSE	None	1	0	0	0	4k	3509	344	33	5467	0	6	2	0
GMII	FALSE	None	0	1	0	0	4k	3607	344	33	5535	0	4	2	0
GMII	FALSE	None	0	0	1	0	4k	3520	344	32	5437	0	4	2	0
GMII	TRUE	None	0	0	0	0	4k	4908	440	133	7984	0	6	0	1
1000Basex	FALSE	None	0	0	0	1	4k	4051	344	49	6084	0	5	0	0
SGMII	FALSE	None	0	0	0	1	4k	4773	394	35	6812	0	4	0	0
SGMII	FALSE	Full	0	0	0	0	4k	4773	394	35	6812	0	4	0	0
SGMII	FALSE	None	1	0	1	0	4k	4064	394	36	6483	0	6	2	0
SGMII	FALSE	None	0	1	0	0	4k	4124	394	36	6517	0	4	2	0
MII	FALSE	None	0	0	0	1	4k	3256	344	26	5287	0	5	0	0
MII	FALSE	Full	0	0	0	0	4k	4192	344	27	5812	0	4	0	0
MII	FALSE	None	1	0	0	0	4k	3494	344	29	5449	0	6	2	0
MII	FALSE	None	0	1	0	0	4k	3535	344	29	5517	0	4	2	0
MII	FALSE	None	0	0	1	0	4k	3407	344	29	5419	0	4	2	0
SGMII	TRUE	None	0	0	0	0	4k	5296	490	48	8595	0	6	0	1

Table 2-24: **Artix-7 FPGA Performance and Resource Utilization Benchmarks (xc7a200tffg1156-2)**

Configuration								Resources							
PHY_TYPE	ENABLE_AVB	TXCSUM RXCSUM	TXVLAN_TRAN RXVLAN_TRAN	TXVLAN_TAG RXVLAN_TAG	TXVLAN_STRP RXVLAN_STRP	MCAST_EXTEND	TX RX MEM	LUT as Logic	LUT as Distributed RAM	LUT as Shift Register	Register as Flip-Flop	Register as Latch	RAMB36:FIFO36	RAMB18:FIFO18	DSPs
1000Basex	FALSE	None	1	0	1	0	4k	4208	344	58	6276	0	6	2	0
1000Basex	FALSE	None	1	1	0	1	4k	4428	344	56	6493	0	7	2	0
1000Basex	FALSE	Full	0	0	0	0	4k	5042	344	53	6595	0	4	0	0
1000Basex	TRUE	Full	0	0	1	0	4k	5536	440	77	8528	0	6	2	1
RGMII	FALSE	None	0	0	0	1	4k	3478	344	28	5353	0	5	0	0
RGMII	FALSE	Full	0	0	0	0	4k	4471	344	30	5854	0	4	0	0
RGMII	FALSE	None	1	0	1	0	4k	3624	344	35	5535	0	6	2	0
RGMII	FALSE	None	0	1	0	0	4k	3644	344	33	5572	0	4	2	0
GMII	FALSE	None	0	0	0	1	4k	3438	344	28	5316	0	5	0	0
GMII	FALSE	Full	0	0	0	0	4k	4433	344	30	5817	0	4	0	0
GMII	FALSE	None	1	0	0	0	4k	3508	344	33	5467	0	6	2	0
GMII	FALSE	None	0	1	0	0	4k	3601	344	33	5535	0	4	2	0
GMII	FALSE	None	0	0	1	0	4k	3526	344	32	5437	0	4	2	0
GMII	TRUE	None	0	0	0	0	4k	4906	440	133	7984	0	6	0	1
1000Basex	FALSE	None	0	0	0	1	4k	4057	344	51	6094	0	5	0	0
SGMII	FALSE	None	0	0	0	1	4k	3866	394	33	6323	0	5	0	0
SGMII	FALSE	Full	0	0	0	0	4k	4801	394	35	6848	0	4	0	0
SGMII	FALSE	None	1	0	1	0	4k	4083	394	36	6519	0	6	2	0
SGMII	FALSE	None	0	1	0	0	4k	4147	394	36	6553	0	4	2	0
MII	FALSE	None	0	0	0	1	4k	3255	344	26	5287	0	5	0	0
MII	FALSE	Full	0	0	0	0	4k	4191	344	28	5812	0	4	0	0
MII	FALSE	None	1	0	0	0	4k	3498	344	29	5449	0	6	2	0
MII	FALSE	None	0	1	0	0	4k	3531	344	30	5517	0	4	2	0
MII	FALSE	None	0	0	1	0	4k	3405	344	28	5419	0	4	2	0
SGMII	TRUE	None	0	0	0	0	4k	5324	490	48	8631	0	6	0	1

Port Descriptions

The AXI Ethernet core creates the ports depending on the mode of operation. These ports are grouped into interfaces based on the functionality. These interfaces have associated clock and reset ports. The details of the interfaces are given in Table 2-25.

Table 2-25: **I/O Interfaces**

Interface Name	Description.
s_axi	AXI-Lite interface used to configure the AXI Ethernet and TEMAC
s_axis_txc	AXI Stream Transmit Control
s_axis_txd	AXI Stream Transmit Data
m_axis_rxd	AXI Stream Receive Data
m_axis_rxs	AXI Stream Receive Status
s_axis_tx_av	AXI Stream AVB Transmit Data. This interface is present only in AVB mode.
m_axis_rx_av	AXI Stream AVB Receive Data. This interface is present only in AVB mode.
mdio	MDIO interface to configure PHY. This interface is present in MII, GMII, RGMII and SGMII modes.
mii	Media Independent Interface. This interface is present only in MII mode
gmii	Gigabit Media Independent Interface. This interface is present only in GMII mode.
rgmii	Reduced Gigabit Media Independent Interface. This interface is present only in SGMII mode.
sgmii	Serial Gigabit Media Independent Interface. This interface is present only in SGMII mode.
sfp	In 1000BaseX mode, this interface connects to the SFP cage. This interface is present only in 1000BaseX mode.
mgt_clk	Differential clock input for the serial transceiver. This interface is present only in Shared Logic in Core configuration in SGMII or 1000BaseX modes.

The details of all the I/O signals that are included in this table and other individual signals are provided in the Table 2-26.

I/O Signal Descriptions

Table 2-26: **I/O Signal Descriptions**

Signal Name	Interface	Signal Type	Init Status	Description
AXI4-Lite Slave Signals				
s_axi_lite_clk	AXI4-Lite	I		Clock
s_axi_lite_resetn(1)	AXI4-Lite	I		Reset (active-Low)
s_axi_awaddr(31:0)	AXI4-Lite	I		Write address
s_axi_awvalid	AXI4-Lite	I		Write address valid: Indicates a valid write address and control information is available
s_axi_awready	AXI4-Lite	O		Write address ready: Slave is ready to accept address and control information
s_axi_wdata(31:0)	AXI4-Lite	I		AXI write data bus
s_axi_wstrb(3:0)	AXI4-Lite	I		Write strobes: Indicates which byte lanes have valid data. `s_axi_wstrb[n]` corresponds to `s_axi_wdata[(8xn)]`+7:(8xn)]
s_axi_wvalid	AXI4-Lite	I		Write valid: Indicated valid write data and strobes are available. 1 = write data and strobes available 0 = write data and strobes not available
s_axi_wready	AXI4-Lite	O		Write ready: Indicates the slave can accept the write data 1= slave ready 0 = slave not ready
s_axi_bresp(1:0)	AXI4-Lite	O		Write response: Indicates the status of the write transaction
s_axi_bvalid	AXI4-Lite	O		Write response valid: Indicates a valid write response is available 1= write response available 0 = write response not available
s_axi_bready	AXI4-Lite	I		Response ready: Indicates the master can accept the response information 1 = master ready 0 = master not ready
s_axi_araddr(31:0)	AXI4-Lite	I		Read address
s_axi_arvalid	AXI4-Lite	I		Read address valid: When High this signal indicates the read address and control information is valid and remain valid until `s_axi_arready` is High 1 = Address and control information valid 0= Address and control information not valid

Table 2-26: **I/O Signal Descriptions *(Cont'd)***

Signal Name	Interface	Signal Type	Init Status	Description
s_axi_arready	AXI4-Lite	O		Address ready: Indicates the slave is ready to accept an address and associated control signals
s_axi_rdata(31:0)	AXI4-Lite	O		Read data.
s_axi_rresp(1:0)	AXI4-Lite	O		Read response: Indicates the status of the read transaction.
s_axi_rvalid	AXI4-Lite	O		Read data valid: Indicates the read data is available and the read transfer can complete 1 = read data available 0 = read data not available
s_axi_rready	AXI4-Lite	I		Read ready: Indicates the master can accept the read data and response information 1 = master ready 0 = master not ready
AXI4-Stream Transmit Data Signals				
axis_clk	AXI4-Stream clock	I		AXI4-Stream clock for TXD RXD TXC and RXS interfaces.
axi_str_txd_aresetn(1)	AXI4-Stream TxD	I		AXI4-Stream Transmit Data Reset
axi_str_txd_tvalid	AXI4-Stream TxD	I		AXI4-Stream Transmit Data Valid
axi_str_txd_tready	AXI4-Stream TxD	O		AXI4-Stream Transmit Data Ready
axi_str_txd_tlast	AXI4-Stream TxD	I		AXI4-Stream Transmit Data Last Word
axi_str_txd_tkeep(3:0)	AXI4-Stream TxD	I		AXI4-Stream Transmit Data Valid Strobes
axi_str_txd_tdata(31:0)	AXI4-Stream TxD	I		AXI4-Stream Transmit Data bus
AXI4-Stream Transmit Control Signals				
axi_str_txc_aresetn(1)	AXI4-Stream TxC	I		AXI4-Stream Transmit Control Reset
axi_str_txc_tvalid	AXI4-Stream TxC	I		AXI4-Stream Transmit Control Valid
axi_str_txc_tready	AXI4-Stream TxC	O		AXI4-Stream Transmit Control Ready
axi_str_txc_tlast	AXI4-Stream TxC	I		AXI4-Stream Transmit Control Last Word
axi_str_txc_tkeep(3:0)	AXI4-Stream TxC	I		AXI4-Stream Transmit Control Valid Strobes
axi_str_txc_tdata(31:0)	AXI4-Stream TxC	I		AXI4-Stream Transmit Control bus

Table 2-26: **I/O Signal Descriptions *(Cont'd)***

Signal Name	Interface	Signal Type	Init Status	Description
AXI4-Stream Receive Data Signals				
axi_str_rxd_aresetn[(1)]	AXI4-Stream RxD	I		AXI4-Stream Receive Data Reset
axi_str_rxd_tvalid	AXI4-Stream RxD	O		AXI4-Stream Receive Data Valid
axi_str_rxd_tready	AXI4-Stream RxD	I		AXI4-Stream Receive Data Ready
axi_str_rxd_tlast	AXI4-Stream RxD	O		AXI4-Stream Receive Data Last Word
axi_str_rxd_tkeep(3:0)	AXI4-Stream RxD	O		AXI4-Stream Receive Data Valid Strobes
axi_str_rxd_tdata(31:0)	AXI4-Stream RxD	O		AXI4-Stream Receive Data bus
AXI4-Stream Receive Control Signals				
axi_str_rxs_aresetn[(1)]	AXI4-Stream RxC	I		AXI4-Stream Receive Control Reset
axi_str_rxs_tvalid	AXI4-Stream RxC	O		AXI4-Stream Receive Control Valid
axi_str_rxs_tready	AXI4-Stream RxC	I		AXI4-Stream Receive Control Ready
axi_str_rxs_tlast	AXI4-Stream RxC	O		AXI4-Stream Receive Control Last Word
axi_str_rxs_tkeep(3:0)	AXI4-Stream RxC	O		AXI4-Stream Receive Control Valid Strobes
axi_str_rxs_tdata(31:0)	AXI4-Stream RxC	O		AXI4-Stream Receive Control bus
AXI4-Stream Ethernet AVB Transmit Data Signals				
avb_tx_clk	AXI4-Stream AvTx	O		AXI4-Stream AVB Transmit Data Clock
axi_str_avb_tx_aresetn	AXI4-Stream AvTx	I		AXI4-Stream AVB Transmit Data Reset
axi_str_avb_tx_tvalid	AXI4-Stream AvTx	I		AXI4-Stream AVB Transmit Data Valid
axi_str_avb_tx_tready	AXI4-Stream AvTx	O		AXI4-Stream AVB Transmit Data Ready
axi_str_avb_tx_tlast	AXI4-Stream AvTx	I		AXI4-Stream AVB Transmit Data Last Word
axi_str_avb_tx_tdata(7:0)	AXI4-Stream AvTx	I		AXI4-Stream AVB Transmit Data bus
axi_str_avb_tx_tuser(0:0)	AXI4-Stream AvTx	I		AXI4-Stream AVB Transmit User defined signal
AXI4-Stream Ethernet AVB Receive Data Signals				
avb_rx_clk	AXI4-Stream AvRx	O		AXI4-Stream AVB Receive Data Clock
axi_str_avb_rx_aresetn	AXI4-Stream AvRx	I		AXI4-Stream AVB Receive Data Reset
axi_str_avb_rx_tvalid	AXI4-Stream AvRx	O		AXI4-Stream AVB Receive Data Valid
axi_str_avb_rx_tlast	AXI4-Stream AvRx	O		AXI4-Stream AVB Receive Data Last Word
axi_str_avb_rx_tdata(7:0)	AXI4-Stream AvRx	O		AXI4-Stream AVB Receive Data bus
axi_str_avb_rx_tuser(0:0)	AXI4-Stream AvRx	O		Receive channel User information used to indicate if the received frame is good (active-Low) or bad (active-High).

Table 2-26: **I/O Signal Descriptions *(Cont'd)***

Signal Name	Interface	Signal Type	Init Status	Description
Ethernet AVB Interrupt Signals				
interrupt_ptp_timer	AVB	O		This interrupt is asserted every 10 ms as a measure by the RTC. This is used as a timer for the PTP software algorithms.
interrupt_ptp_tx	AVB	O		This is asserted following the transmission of any PTP packet from the Tx PTP packet buffers. Following this interrupt, the software is required to record the Tx Frame Time Stamp.
interrupt_ptp_rx	AVB	O		This is asserted following the transmission of any PTP packet from the Rx PTP packet buffers. Following this interrupt, the software is required to record the Rx Frame Time Stamp.
Reference signals can be used for 1722 logic				
rtc_nanosec_field(31:0)	AVB	O		The synchronized nanosecond field from the RTC.
rtc_sec_field(47:0)	AVB	O		The synchronized second field from the RTC
clk8k	AVB	O		An 8 kHz clock which is derived from and is synchronized to the RTC. The period of this clock, 125us, marks the isochronous cycle.
System Signals				
interrupt	System	O	0	Interrupt indicator for core

Table 2-26: **I/O Signal Descriptions *(Cont'd)***

Signal Name	Interface	Signal Type	Init Status	Description
Ethernet System Signals				
phy_rst_n	Ethernet	O	0	TEMAC to PHY reset signal: This active-Low reset is held active for 10 ms after power is applied and during any reset. After the reset goes inactive, the PHY cannot be accessed for an additional 5 ms.
ref_clk	Ethernet	I		200 MHz input clock on global clock routing used for signal delay primitives for all GMII and RGMII PHY modes.
gtx_clk[(2)]	Ethernet	I		The 125 MHz clock used in all MII, GMII, RGMII, and SGMII configurations to control the PHY reset requirements. Also, it is a 125 MHz input clock on global clock routing used to derive the other transmit clocks for all GMII and RGMII PHY modes. For TEMAC MII PHY systems, this clock must be driven by some clock (does not need to be 125 MHz). The AXI4-Lite clock can be used in these cases; however, the use of a slower clock increases the PHY reset (10 ms @ 125 MHz) and the time required to wait after reset (5 ms @ 125 MHz) before accessing the PHY registers. This clock is also used when Ethernet Statistics are enabled with all supported device families.
mgt_clk_p	Ethernet	I		Positive polarity of differential clock used to drive GTX/GTP serial transceivers. Must be connected to an external, high-quality differential reference clock of frequency of 125 MHz.
mgt_clk_n	Ethernet	I		Negative polarity of differential clock used to drive GTX/GTP serial transceivers. Must be connected to an external, high-quality differential reference clock of frequency of 125 MHz.
signal_detect	Ethernet	I		This signal should be routed to an appropriate port on the optical SFP module. It is required to detect cable pull conditions cleanly. If not used, it needs to be tied up to 1.
Ethernet MII Signals				
mii_txd(3:0)	Ethernet bus MII	O	0	TEMAC to PHY transmit data
mii_tx_en	Ethernet bus MII	O	0	TEMAC to PHY transmit enable
mii_tx_er	Ethernet bus MII	O	0	TEMAC to PHY transmit Error enable
mii_rxd(3:0)	Ethernet bus MII	I		PHY to TEMAC receive data

Table 2-26: **I/O Signal Descriptions *(Cont'd)***

Signal Name	Interface	Signal Type	Init Status	Description
mii_rx_dv	Ethernet bus MII	I		PHY to TEMAC receive data valid indicator
mii_rx_er	Ethernet bus MII	I		PHY to TEMAC receive error indicator
mii_rx_clk	Ethernet bus MII	I		PHY to TEMAC receive clock
mii_tx_clk(2)	Ethernet bus MII	I		PHY to TEMAC transmit clock (also used for GMII/MII mode)
Ethernet GMII Signals				
gmii_txd(7:0)	Ethernet bus GMII	O	0	TEMAC to PHY transmit data
gmii_tx_en	Ethernet bus GMII	O	0	TEMAC to PHY transmit enable
gmii_tx_er	Ethernet bus GMII	O	0	TEMAC to PHY transmit Error enable
gmii_gtx_clk	Ethernet bus GMII	O	0	TEMAC to PHY transmit clock
gmii_rxd(7:0)	Ethernet bus GMII	I		PHY to TEMAC receive data
gmii_rx_dv	Ethernet bus GMII	I		PHY to TEMAC receive data valid indicator
gmii_rx_er	Ethernet bus GMII	I		PHY to TEMAC receive error indicator
gmii_rx_clk	Ethernet bus GMII	I		PHY to TEMAC receive clock
gmii_tx_clk	Ethernet bus GMII	I	–	PHY to TEMAC tx input clock in 10 and 100 Mb/s modes.
Ethernet SGMII and 1000Base-X Signals				
txp	Ethernet bus SGMII and 1000Base-X	O	0	TEMAC to PHY transmit data positive
txn	Ethernet bus SGMII and 1000Base-X	O	0	TEMAC to PHY transmit data negative
rxp	Ethernet bus SGMII and 1000Base-X	I		PHY to TEMAC receive data positive
rxn	Ethernet bus SGMII and 1000Base-X	I		PHY to TEMAC receive data negative

Table 2-26: **I/O Signal Descriptions *(Cont'd)***

Signal Name	Interface	Signal Type	Init Status	Description
Ethernet RGMII Signals				
rgmii_txd(3:0)	Ethernet bus RGMII	O	0	TEMAC to PHY transmit data
rgmii_tx_ctl	Ethernet bus RGMII	O	0	TEMAC to PHY transmit control
rgmii_txc	Ethernet bus RGMII	O	0	TEMAC to PHY transmit clock
rgmii_rxd(3:0)	Ethernet bus RGMII	I		PHY to TEMAC receive data
rgmii_rx_ctl	Ethernet bus RGMII	I		PHY to TEMAC receive control
rgmii_rxc	Ethernet bus RGMII	I		PHY to TEMAC receive clock
Ethernet MII Management Interface (MIIM) Signals				
mdc	Ethernet bus MIIM	O	0	TEMAC to PHY MII management bus clock
mdio(4)	Ethernet bus MIIM	I/O	1	Tri-stateable bidirectional MII Management data bus.

Notes:

1. See Resets in Chapter 3.
2. See Clocking in Chapter 3.
3. This core does not support half duplex operation.
4. The MDIO signal is required to be pulled High as per the PHY data sheet. If the MDIO interface is not used with the soft PCS PMA core (C_TYPE = 1 and C_PHY_TYPE = 4 or 5), the internal MDIO_I signal must be tied High to allow MDIO communication to the internal MAC. In SGMII and 1000BaseX modes, individual `mdio_i` (input) `mdio_o` (output) and `mdio_t` (3-state enable) signals are provided instead of the mdio bidirectional port.

AXI4-Stream Interface

The Ethernet frame data to be transmitted and the frame data that is received passes between the AXI Ethernet core and the rest of the embedded system through AXI4-Stream interfaces. In many cases, the other end of the AXI4-Stream interfaces are connected to a soft IP DMA controller implemented in FPGA logic. However, any custom logic can be used to connect to the AXI4-Stream interface as long as it meets the requirements of the AXI Ethernet core AXI4-Stream interface.

The AXI4-Stream interface is a high-performance, synchronous, point-to-point connection which, in its general use case, is described in its specification listed in the References in Appendix C. This section describes the specific 32-bit implementation used by the AXI Ethernet core to transfer transmit and receive Ethernet frame data with the rest of the embedded system. The AXI4-Stream model used is called *Packetized* and *Aligned Strobe* which is defined in the subsequent sections.

Packetized

Data is transferred in packets rather than as a continuous stream. The signals *_tlast are used to indicate the last 32-bit word of a packet being transferred.

Aligned Strobe

A write strobe is used for each byte (*_tkeep(3:0)) in the data bus (Four write strobes in our case). Null strobes are not allowed at the beginning, in the middle of a transfer, or at the end of a transfer. This means that the first word transferred and every additional word up until the last must contain a valid 32-bit value. The last word might be sparse which means it might contain 4, 3, 2, or 1 valid bytes aligned to the right and the write strobes are used to indicate which bytes are valid. *_tlast is used to indicated the last data of a frame. In some cases the write strobe signals can be tied to the active state (1) (see Transmit AXI4-Stream Interface).

Throttling

The driver of the AXI4-Stream uses the *_tvalid to throttle during a transfer. By taking *_tvalid inactive the current transfer is held until it is active again. The receiver of the AXI4-Stream uses the *_tready signal to throttle during a transfer. By taking *_tready inactive the current transfer is held until it is active again.

Dual Channel AXI4-Stream

The transmit AXI4-Stream interface uses two AXI4-Stream buses. The AXI4-Stream Data Bus is used for frame data only while the AXI4-Stream Control Bus contains control information. Similarly, the receive AXI4-Stream interface uses two AXI4-Stream buses. The AXI4-Stream Data Bus is used for frame data only, and the AXI4-Stream Status Bus provides status information. The control/status and data buses must have the same clock source but there is no synchronization between the two buses with regards to frame to frame data.

The transmit AXI4-Stream control bus can be configured to use one of two format types: Normal Transmit or Receive Status Transmit. This configuration is controlled by the first nibble (4 bits aligned left) of the first word transferred on the transmit control bus and the receive status bus. A value of 0xA identifies the transfer as a Normal Transmit control packet. A value of 0x5 identifies the transfer as a Receive Status packet. All other values are currently undefined and are ignored. These transfer types are defined in Normal Transmit AXI4-Stream Transfer – Flag=0xA and Receive Status Transmit AXI4-Stream Transfer – Flag=0x5. The receive AXI4-Stream interface only supports one format type.

Functional Description

The AXI4-Stream interface transfers data in one direction only. The Ethernet transmit interface uses an AXI4-Stream Data interface and an AXI4-Stream Control interface. The Ethernet receive interface uses an AXI4-Stream Data interface and an AXI4-Stream Status interface. The AXI4-Stream interfaces used in this implementation are 32-bits wide, have side-band control signals, and typically operate with a clock between 100 and 125 MHz. Data is transferred across the AXI4-Stream Data interfaces. Additional control information is transferred across the transmit AXI4-Stream Control interface and additional status information is transferred across the receive AXI4-Stream Status interface.

For the transmit datapath the *Source* is the embedded system, typically a DMA controller, and the *Destination* is the AXI Ethernet core. For the receive datapath the *Source* is the AXI Ethernet and the *Destination* is the embedded system, typically a DMA controller.

Control signals are used to mark the start and end of data across the AXI4-Stream interfaces as well as to signal the readiness of the Source and Destination and to indicate which bytes in the 32-bit path contain valid data. The destination uses the *_tready signal to indicate it is able to receive data, while the source uses *_tvalid to indicated when valid data is on the bus and the *_tlast signal to indicate the last 8, 16, 24, or 32 bits of data.

Transmit AXI4-Stream Interface

The transmit control block must maintain coherence between the data and control buses. Because data frames can vary from 1 byte to over 9 Kb in length and the control information for each frame is a constant six 32-bit words, care must be taken under conditions where the buffer for the frame data or control data fills up to prevent an out-of-sequence condition.

To maintain coherency, the AXI4-Stream data ready signal is held *not ready* until a AXI4-Stream control stream has been received. After this has occurred, the AXI4-Stream data ready signal is driven *ready* (as long as there is buffer space available) and the AXI4-Stream control ready signal is held *not ready* until the data stream transfer is complete (Figure 2-21 and Figure 2-23). The write strobe signals for the control and status buses are always in the active state (0xF).

The right-most write strobe signal for the data bus is always in the active state (0x1, 0x3, 0x7, or 0xF). These signals can be tied off rather than routing signals from the AXI4-Stream source to the destination. The AXI Ethernet core provides these ports to be compliant with the standard; however, there is not any logic based on these inputs which are considered constants. The transmit interface can encounter two AXI4-Stream transfer types: Normal Transmit or Receive Status Transmit, as described in the following sections.

Normal Transmit AXI4-Stream Transfer – Flag=0xA

The Normal Transmit transfer is used to connect the AXI Ethernet core to an external core.

Figure 2-21 illustrates the waveforms when connected to a core such as AXI_DMA or AXI_FIFO_MM_S. AXI_DMA supports advanced features such as partial CSUM offloading or extended VLAN; however, AXI_FIFO_MM_S does not support any of the advanced features.

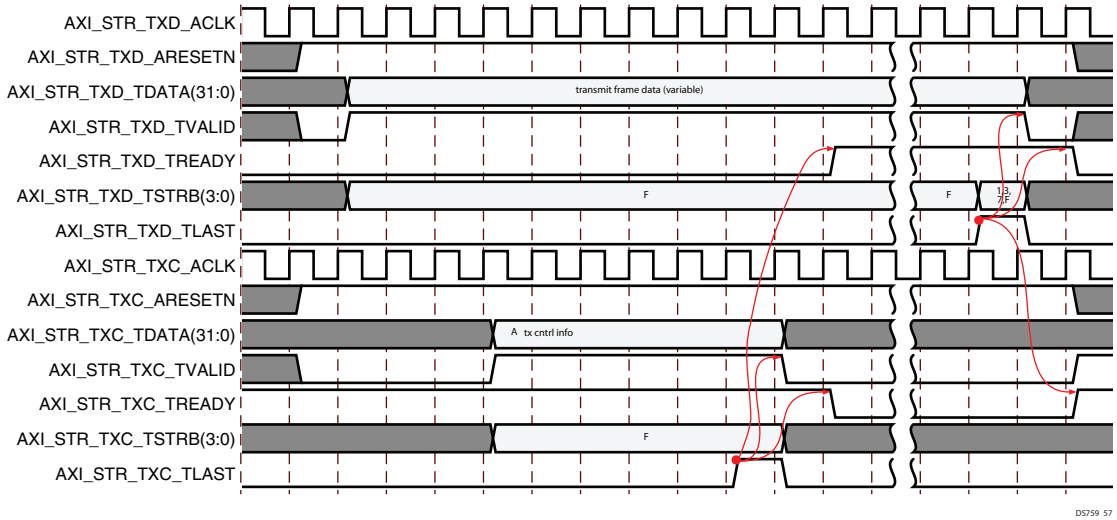


Figure 2-21: **Normal Transmit AXI4-Stream Waveform**

Normal Transmit AXI4-Stream Control Words

The Normal Transmit AXI4-Stream Control frame always contains six 32-bit control words (Words 0 to 5). Of these words, only control words 0, 1, 2, and 3 are used by the AXI Ethernet core. Figure 2-22, Table 2-27, Table 2-29, and Table 2-30 show the definitions of these words.

If the transmit AXI4-Stream control word bits 1:0 are 00 (TX_CSCNTRL is disabled) or if the parameter C_TXCSUM is 0 (the transmit checksum offload function is not included in build), then none of the transmit AXI4-Stream control words are used and no transmit checksum offload takes place. If the parameter C_TXCSUM is 1, transmit partial checksum offload can be controlled on a frame-by-frame basis by setting or clearing the transmit AXI4-Stream control word 1 bits 1:0 to 01 (TX_CSCNTRL). If the parameter C_TXCSUM is 2, the transmit full checksum offload can be controlled on a frame-by-frame basis by setting or clearing the transmit AXI4-Stream control word 1 bits 1:0 to 10 (TX_CSCNTRL). For more details about how the transmit AXI4-Stream control words are used for transmit checksum offload, see Partial TCP/UDP Checksum Offload in Hardware.

The transmit AXI4-Stream Data strobes are used to indicate how many bytes in the last 32-bit word of the payload are valid data. A "1" is used to indicate valid bytes. For example, `axi_str_txd_strb(3:0)` = "0001" would indicate that only the first byte of the last word of the payload [`axi_str_txd(7:0)`] is valid and the remaining three bytes are unused.

`axi_str_txd_strb(3:0)` = "0011" would indicate that the first two bytes of the last word of the payload [`axi_str_txd(15:0)`] are valid and the remaining two bytes are unused. See Figure 2-22 for the Transmit AXI4-Stream Control Word definition.

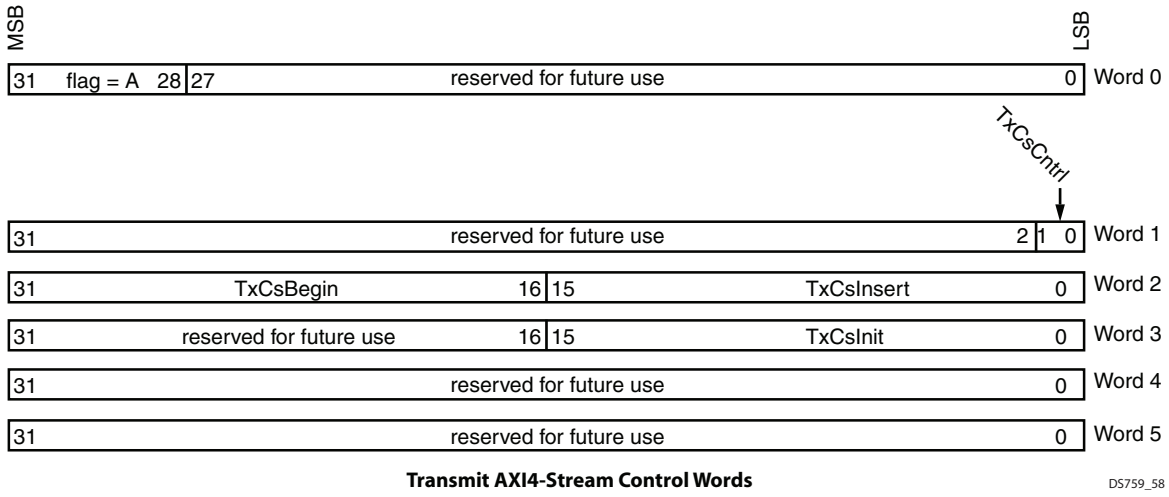


Figure 2-22: **Transmit AXI4-Stream Control Words**

Table 2-27: **Transmit AXI4-Stream Control Word 0 – TAG**

Bits	Name	Description
31–28	Flag	1010 = Normal Transmit Frame 0101= Receive Status Transmit Frame All other selections are reserved.
27–0	Reserved	Reserved for future use

Table 2-28: **Transmit AXI4-Stream Control Word 1 – APP0**

Bits	Name	Description
31–2	Reserved	Reserved for future use
1–0	TxCsumCntrl	Transmit Checksum Enable: • 00 = No transmit checksum offloading should be performed on this frame • 01 = Partial transmit checksum offloading should be performed on this frame based upon other data provided in the control words. For the partial checksum to be performed, C_TXCSUM must be set to 1. • 10 = Full transmit IP and TCP/UDP checksum offloading should be performed on this frame if it meets the requirements. For the full checksum to performed, C_TXCSUM must be set to 2. • 11 = Reserved

Table 2-29: **Transmit AXI4-Stream Control Word 2 – APP1**

Bits	Name	Description
31–16	TxCsBegin	**Transmit Checksum Calculation Starting Point:** This value is the offset to the location in the frame to the first byte that needs to be included in the checksum calculation. The first byte is indicated by a value of zero. The beginning position must be 16-bit aligned.
15–0	TxCsInsert	**Transmit Checksum Insertion Point:** This value is the offset to the location in the frame where the checksum value should be written into the TCP or UDP segment header. The value must be 16-bit aligned and cannot be in the first 8 bytes of the frame. It also should not contain a value that exceeds the length of the frame.

Table 2-30: **Transmit AXI4-Stream Control Word 3 – APP2**

Bits	Name	Description
31–16	Reserved	Undefined value.
15–0	TxCsInit	**Transmit Checksum Calculation Initial Value:** This value is a 16-bit seed that can be used to insert the TCP or UDP pseudo header into the checksum calculation. See Partial TCP/UDP Checksum Offload in Hardware for more information on using this field.

Receive Status Transmit AXI4-Stream Transfer – Flag=0x5

The Receive Status transfer is utilized when the receive AXI4-Stream interface is tied directly to the transmit AXI4-Stream interface for the purpose of looping back Ethernet receive data to the Ethernet transmit interface with no external intervention. In that case the status transfer enables the receive data frame to be presented to the TEMAC transmitter and the status content is ignored. No advanced checksum offload or VLAN functions is allowed for these operations even if they were included in the core at build time. Notice the different identification flag value in Figure 2-23.

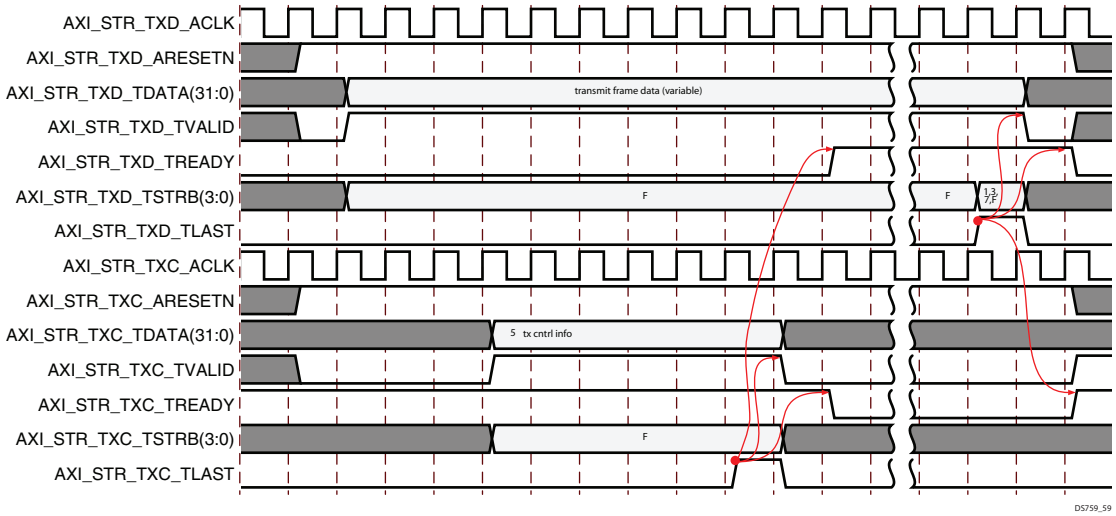


Figure 2-23: **Receive Status Transmit AXI4-Stream Waveform**

Receive AXI4-Stream Interface

Unlike the transmit AXI4-Stream Control interface, the receive AXI4-Stream Status interface has only one format type (TAG/FLAG).

The receive interface has been designed to allow throttling on both the AXI4-Stream Status bus and the AXI4-Stream Data bus. After receiving Ethernet data, the receive interface transfers the AXI4-Stream Status information before the AXI4-Stream Data information. See Figure 2-26 for a receive waveform diagram. This diagram shows what signals are required when connected to a core such as AXI_DMA. When connecting to a core such as AXI_FIFO_MM_S, a signal minimization can be made because the receive status bus information is not required.

In this case, the external core must actively drive the signals `axi_str_rxs_aclk` and `axi_str_rxs_aresetn`, `axi_str_rxs_tready` must be tied High, and all of the `axi_str_rxs*` inputs to the external core can be left open.

For the loopback of the receive AXI4-Stream to transmit AXI4-Stream to work, the receive AXI4-Stream Data bus is throttled by the transmit AXI4-Stream Data bus until the receive AXI4-Stream Status has been received by the transmit channel.

The receive AXI4-Stream Status frame always contain six 32-bit status words (words 0 to 5). Figure 2-24, Table 2-31, Table 2-32, Table 2-33, Table 2-34, Table 2-35, and Table 2-36 show the definitions of these words. Reserve fields do not have defined values. If the parameter C_RXCSUM is 0, the receive checksum offload function is not included in the build and receive AXI4-Stream Status word 4, bits 15-0 are always zero. If C_RXCSUM is 1, the raw checksum is calculated for every frame received and is placed in the receive AXI4-Stream Status word 4. For more information about using the receive raw checksum value, see Partial TCP/UDP Checksum Offload in Hardware.

Receive AXI4-Stream Status word 5, bits 15-0 always contains the number of bytes in length of the frame being sent across the receive AXI4-Stream Status interface.

The `axi_str_rxd_strb(3:0)` bus is used to indicate how many bytes in the last 32-bit word of the AXI4-Stream Data bus. A 1 is used to indicate valid bytes. For example, `axi_str_rxd_strb(3:0)` = "0001" indicates that only the first byte of the last word of the AXI4-Stream Data bus [`axi_str_rxd_data(7:0)`] is valid and the remaining three bytes are unused. `axi_str_rxd_strb(3:0)` = "0011" would indicate that the first two bytes of the last word of the payload [`axi_str_rxd_data(15:0)`] are valid and the remaining two bytes are unused.

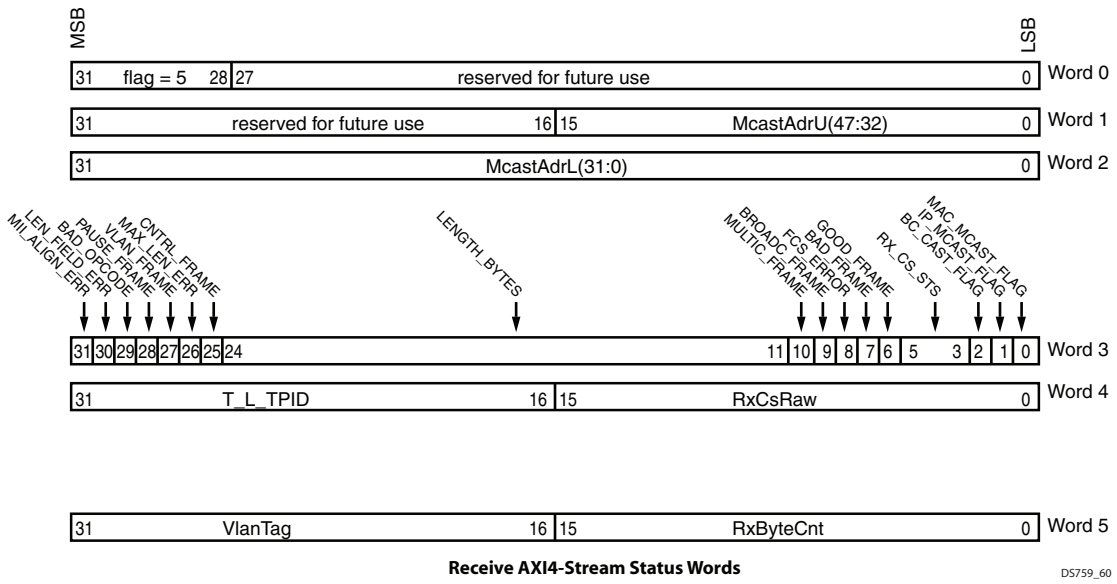


Figure 2-24: **Receive AXI4-Stream Status Words**

Table 2-31: **Receive AXI4-Stream Status Word 0 – TAG**

Bits	Name	Description
31–28	Flag	0x5 = Receive Status Frame
27–0	Reserved	Undefined value.

Table 2-32: **Receive AXI4-Stream Status Word 1 – APP0**

Bits	Name	Description
31–16	Reserved	Undefined value.
15–0	MCAST_ADR_U	**Multicast Address (47:32):** These are the upper 16 bits of the multicast destination address of this frame. This value is only valid if the AXI4-Stream Status word 2, bit 0 is a 1. The address is ordered so the first byte received is the lowest positioned byte in the register; for example, MAC address of AA-BB-CC-DD-EE-FF would be stored in UnicastAddr(47:0) as 0xFFEEDDCCBBAA. This word would be 0xFFEE.

Table 2-33: **Receive AXI4-Stream Status Word 2 – APP1**

Bits	Name	Description
31–0	MCAST_ADR_L	**Multicast Address (31:0):** These are the lower 32 bits of the multicast destination address of this frame. This value is only valid AXI4-Stream Status word 2, bit 0 is a 1. The address is ordered so the first byte received is the lowest positioned byte in the register; for example, MAC address of AA-BB-CC-DD-EE-FF would be stored in UnicastAddr(47:0) as 0xFFEEDDCCBBAA. This word would be 0xDDCCBBAA.

Table 2-34: **Receive AXI4-Stream Status Word 3 – APP2**

Bits	Name	Description
31	MII_ALIGN_ERR	**MII Alignment Error:** Used in 10/100 MII mode. Asserted if the previous frame received has an incorrect FCS value and a misalignment occurs when the 4-bit MII data bus is converted to the 8-bit GMII data bus.
30	LEN_FIELD_ERR	**Length Field Error:** Asserted if the LT field contains a length value that does not match the number of Ethernet MAC data bytes received. Also asserted High if the LT field indicates that the frame contains padding but the number of Ethernet MAC data bytes received is not equal to 64 bytes (minimum frame size). This bit is not defined when LT field error-checks are disabled or when received frames are less than the legal minimum length.
29	BAD_OPCODE	**Bad OP Code:** Asserted if the previous frame is error free. Contains the special control frame identifier in the LT field, but contains an OPCODE unsupported by the Ethernet MAC (any OPCODE other than PAUSE).
28	PAUSE_FRAME	**Pause Frame:** Asserted if the previous frame is error-free. Contains the special control frame identifier in the LT field. Contains a destination address matching either the Ethernet MAC control multicast address or the configured source address of the Ethernet MAC. Contains the supported PAUSE OPCODE and is acted upon by the Ethernet MAC.
27	VLAN_FRAME	**VLAN Frame:** Asserted if the previous frame contains a VLAN identifier in the LT field when receiver VLAN operation is enabled.
26	MAX_LEN_ERR	**Maximum Length Error:** Asserted if the previous frame exceeded the specified IEEE Std 802.3-2005 maximum legal length. This is only valid if jumbo frames are disabled.
25	CNTRL_FRAME	**Control Frame:** Asserted if the previous frame contains the special control frame identifier in the LT field.
24–11	LENGTH_BYTES	**Length Bytes:** The length of the previous frame in number of bytes. The count sticks at 16383 for any jumbo frames larger than this value.
10	MULTIC_FRAME	**Multicast Frame:** Asserted if the previous frame contains a multicast address in the destination address field.
9	BROADC_FRAME	**Broadcast Frame:** Asserted if the previous frame contained the broadcast address in the destination address field.
8	FCS_ERR	**FCS Error:** Asserted if the previous frame received has an incorrect FCS value or the Ethernet MAC detects error codes during frame reception.
7	BAD_FRAME	**Bad Frame:** Asserted if the previous frame received contains errors.
6	GOOD_FRAME	**Good Frame:** Asserted if the previous frame received is error-free.

Table 2-34: **Receive AXI4-Stream Status Word 3 – APP2 *(Cont'd)***

Bits	Name	Description
5–3	RX_CS_STS	**Receive CSUM Status:** • 000 = Neither the IP header nor the TCP/UDP checksums were checked • 001 = The IP header checksum was checked and was correct. The TCP/UDP checksum was not checked • 010 = Both the IP header checksum and the TCP checksum were checked and were correct • 011 = Both the IP header checksum and the UDP checksum were checked and were correct • 100 = Reserved • 101 = The IP header checksum was checked and was incorrect. The TCP/UDP checksum was not checked • 110 = The IP header checksum was checked and is correct but the TCP checksum was checked and was incorrect • 111 = The IP header checksum was checked and is correct but the UDP checksum was checked and was incorrect
2	BCAST_FLAG	**Broadcast Frame Flag:** This bit, when 1, indicates that the current frame is a Broadcast frame that has passed the hardware address filtering.
1	IP_MCAST_FLAG	**IP Multicast Frame Flag:** This bit, when 1, indicates that the current frame is a multicast frame that appears to be formed from an IP multicast frame (the first part of the destination address is 01:00:5E) that has passed the hardware multicast address filtering.
0	MAC_MCAST_FLAG	**MAC Multicast Frame Flag:** This bit, when 1, indicates that the current frame is a MAC multicast frame that has passed the hardware multicast address filtering.

Table 2-35: **Receive AXI4-Stream Status Word 4 – APP3**

Bits	Name	Description
31–16	T_L_TPID	**Type Length VLAN TPID:** This is the value of the 13th and 12th bytes of the frame (index starts at zero). If the frame is not VLAN type, this is the type/length field. If the frame is VLAN type, this is the value of the VLAN TPID field prior to any stripping, translation or tagging.
15–0	RX_CSRAW	**Receive Raw Checksum:** This value is the raw receive checksum calculated over the entire Ethernet frame starting at byte 14 (index starts at zero). If the receive FCS stripping is not enabled, the FCS is included in the checksum and must be removed by the application.

Table 2-36: **Receive AXI4-Stream Status Word 5 – APP4**

Bits	Name	Description
31–16	VLAN_TAG	**VLAN Priority CFI and VID:** This is the value of the 15th and 14th bytes of the frame (index starts at zero). If the frame is VLAN type, this is the value of the VLAN priority, CFI, and VID fields prior to any stripping, translation, or tagging. If the frame is not VLAN type, this is the first 2 bytes of the data field.
15–0	RX_BYTECNT	**Receive Frame Length (Bytes):** This value is the number of bytes in the Ethernet frame which is in the receive AXI4-Stream Data interface.

Mapping AXI DMA IP Buffer Descriptor Fields to AXI4-Stream Fields

The AXI Ethernet core requires that certain AXI4-Stream Control/Status words be used to support TCP/IP Checksum Offload. The AXI Ethernet core does not have any requirements on how the AXI4-Stream words are created or where the data comes from, only that the correct values are in each field. At the time that this document is written, Xilinx provides a core that can be used to provide the require AXI4-Stream functionality to implement TCP / IP Checksum Offload, the AXI_DMA IP core. See the change log for the version of AXI DMA to use.

The AXI DMA core is designed to operate with many AXI4-Stream cores in addition to the AXI Ethernet core. This guide contains information about the mapping between the AXI Ethernet AXI4-Stream fields and the AXI DMA Buffer Descriptor fields for the purposes of TCP / IP Checksum Offload.

The AXI DMA core uses registers to point to data areas in external memory called Buffer Descriptors. The Buffer Descriptors are five 32-bit words in external memory and contain AXI DMA operation control information, pointers to other areas of external memory which contain data to move which are called Data Buffers, and generic Application Defined words which map to AXI4-Stream Control and AXI4-Stream Status words.

Figure 2-25 shows the mapping between the AXI DMA Buffer Descriptor words in external memory and the fields in the transmit AXI4-Stream case and Figure 2-26 shows the mapping for the receive AXI4-Stream case. The first word in the AXI_STR_TXC_TDATA data contains the Flag information that is directly set by the AXI DMA core, and the first word in the AXI_STR_RXS_TDATA data contains the Flag information that is set by the AXI Ethernet core.

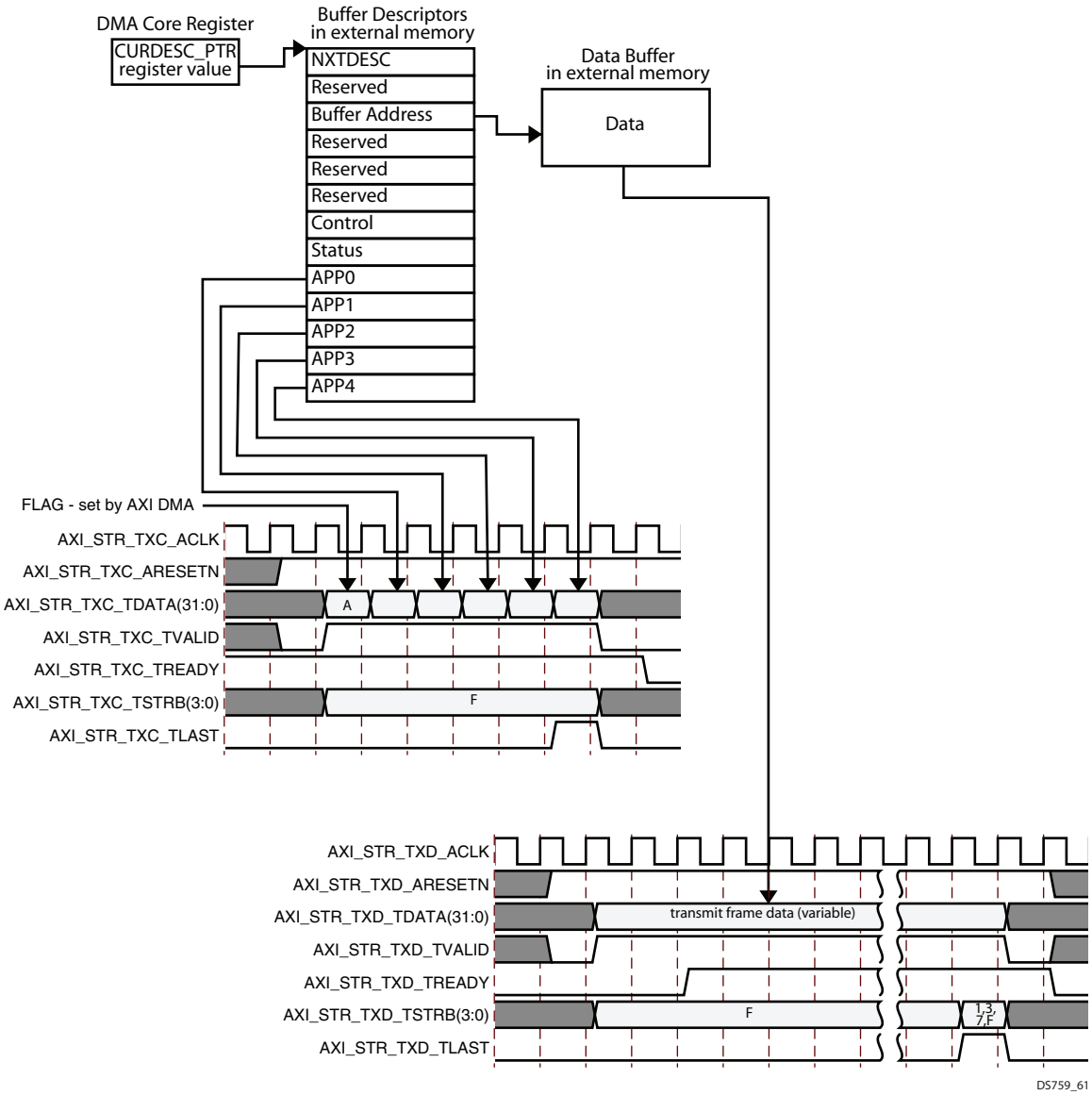


Figure 2-25: **Transmit AXI DMA Buffer Descriptor AXI4-Stream Field Mapping**

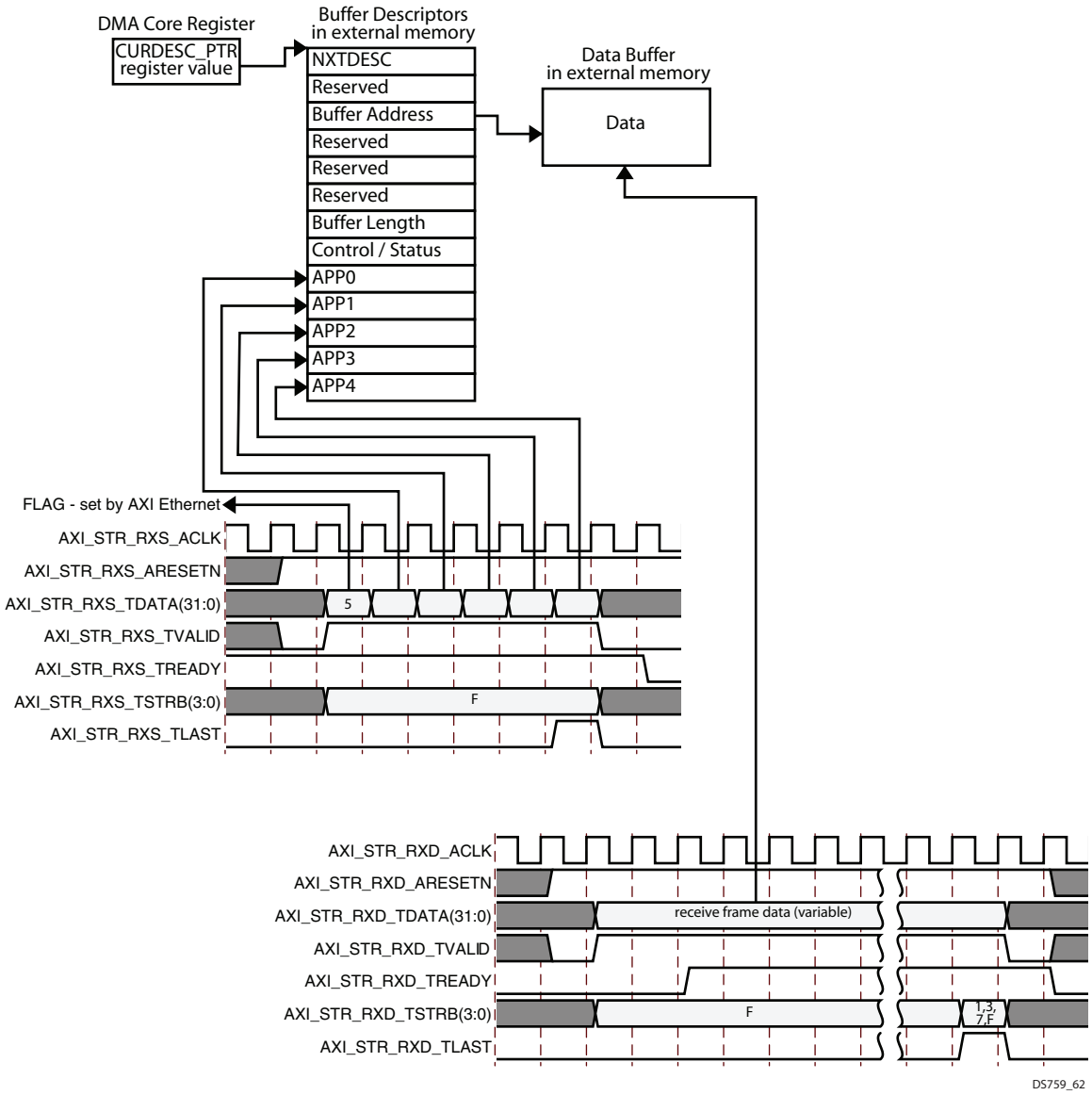


Figure 2-26: **Receive AXI DMA Buffer Descriptor AXI4-Stream Field Mapping**

Ethernet Audio Video Bridging

Ethernet Audio Video Bridging (AVB) functionality is supported by the AXI Ethernet core. This mode is enabled by selecting the Enable AVB check box in the Vivado Integrated Design Environment (IDE). AVB RX and TX streaming interfaces are created for this mode. These interfaces are directly connected to the TEMAC. A brief introduction is provided here and more information about the functionality can be found in the *LogiCORE IP Tri-Mode Ethernet MAC Product Guide* (PG051) [Ref 1].

AVB AXI4-Stream Interface

The AVB AXI4-Stream interface is a limited interface in that it does not allow throttling from the external connections. As a result, some of the standard AXI4-Stream signals are missing from the port list as well as the addition of the `tuser` signals. Both transmit and receive buses are 8-bits wide which eliminates the need of the write strobe signal bus. For both buses, the AXI Ethernet core provides the AXI4-Stream clocks and clock enables which are derived from the internal Ethernet clocks.

The AVB AXI4-Stream interface is connected to external logic that currently is not provided by Xilinx. Several third-party partners have created this logic, have tested it with the system core, and have integrated it into AVB systems. This external logic takes time-sensitive audio or video information and splits it into Ethernet frames using a protocol encoding that is similar to TCP/IP. This is all done in FPGA logic.

Ethernet AVB frames are passed back and forth to the Ethernet through the AVB AXI4-Stream interfaces. Internal to AXI Ethernet core, the AVB frames and the legacy frames are multiplexed and demultiplexed based on a prioritization and time slotting method. The AVB function in the AXI Ethernet core is responsible for helping to choose the most accurate AVB system clock in the Ethernet network and synchronizes to the clock so all AVB nodes are synchronized.

Transmit Interface

The AXI Ethernet core provides the signal `axi_str_avbtx_aclk` which is derived from the TEMAC transmit MAC interface clock. This clock operates at 125 MHz when operating at 1 Gb/s and is 25 MHz when operating at 100 Mb/s. During a transfer, the AXI Ethernet core uses the TEMAC transmit MAC interface clock enable to toggle `axi_str_avbtx_tready`. The clock enable is High for every clock cycle when operating at 1 Gb/s and toggles every other clock cycle for 100 Mb/s. When the AXI Ethernet core is ready to transmit an AVB frame, it drives the `axi_str_avbtx_tready` signal High. When the external logic is ready to transmit a frame, it drives the `axi_str_avbtx_tvalid` signal High and provides the first byte of data on the `axi_str_avbtx_tdata` bus. Now the external logic must provide a new byte of data on every clock cycle that the `axi_str_avbtx_tready` signal is High while `tvalid` is active until the end of the frame is reached.

The external logic cannot throttle the AVB transmit interface. The AXI Ethernet core accepts the first byte and then drives `axi_str_avbtx_tready` Low until the TEMAC has started the transmit, then it drives it back to High and continues to use it as a clock enable for the remainder of that frame.

On the last byte of the frame, the external logic drives the `axi_str_avbtx_tlast` signal High for one clock cycle with `axi_str_avbtx_tready`. If it does not have any additional frames to transmit, it removes the `tvalid` signal when it takes the `tlast` signal Low. However, if another frame is ready, the external logic leaves the `tvalid` signal High.

The `tuser` signal is intended to allow the external logic to indicate that the current frame in progress has an error such as an underflow and the frame should be aborted. It is intended that this be connected to the underflow input of the AVB to force the current frame to be aborted, but the current AVB core does not provide an AVB underflow input. Figure 2-27 shows a transmit AXI4-Stream waveform for 1 Gb/s mode where there are additional AVB frames available after the completion of the current frame. Figure 2-28 shows the TX client interface operating at 100 Mb/s.

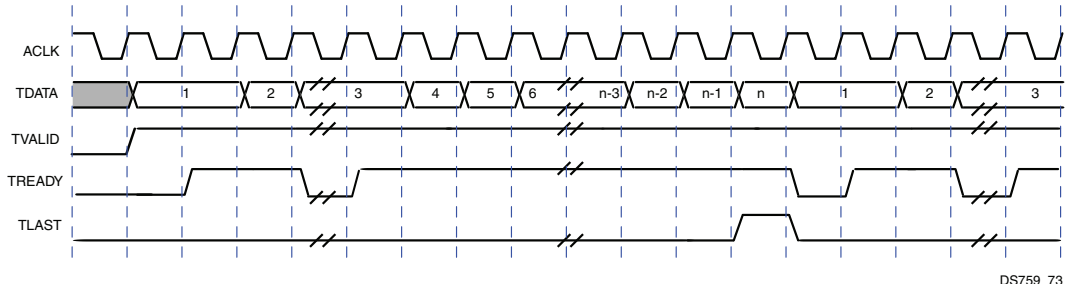


Figure 2-27: **1000 Mb/s Transmit AVB AXI4-Stream (back-to back)**

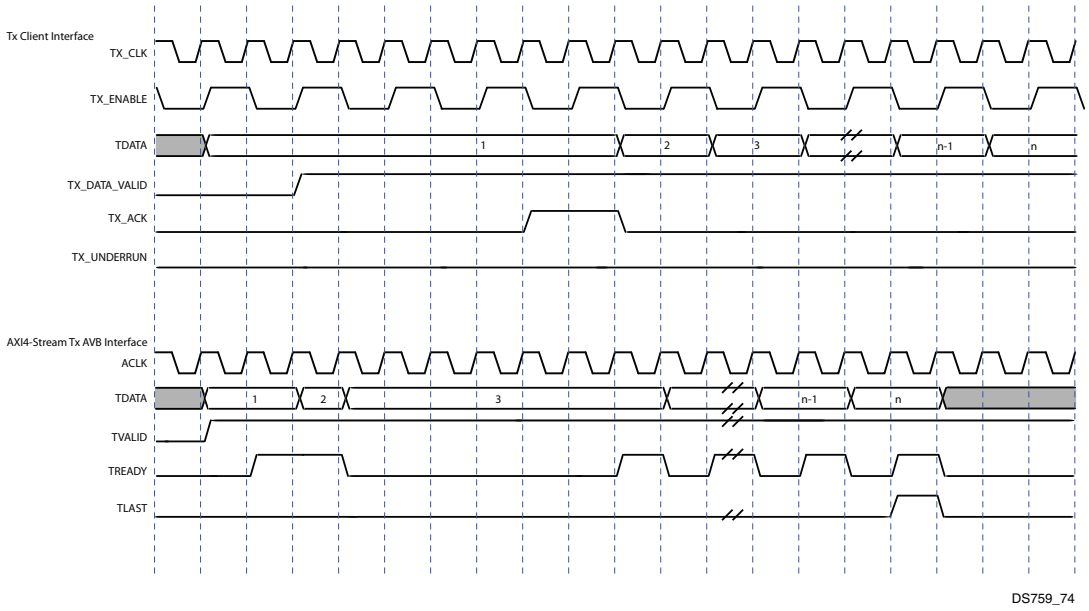


Figure 2-28: **100 Mb/s Transmit AVB AXI4-Stream**

Receive Interface

The receive interface provides an AXI4-Stream clock and which is derived from the TEMAC receive client interface and uses the `axi_str_avbrx_tvalid` signal as a clock enable derived from the receive client interface clock enable. These signals behave similarly to the transmit interface when the Ethernet bus speed changes.

When AXI Ethernet core has received an AVB frame to transfer over the AXI4-Stream to the external logic, it drives the `axi_str_avbrx_tvalid` signal High and provides a new `axi_str_avbrx_tdata` byte value on each clock cycle when `axi_str_avbrx_tvalid` signal is High. The destination cannot throttle and must always be ready to receive a frame. After AXI Ethernet core transfers the second to last byte, it drives the `axi_str_avbrx_tvalid` signal Low and wait until it gets a good or bad frame indication from the TEMAC before it finishes the frame. When it receives the good or bad frame indication, it drives the `axi_str_avbrx_tvalid` signal High again for one clock/`axi_str_avbrx_tvalid` cycle along with the last byte value. It drives the `axi_str_avbrx_tuser` signal High if the frame is bad. If the frame is good, it drives `axi_str_avbrx_tuser` signal Low while driving the `axi_str_avbrx_tlast` signal High.

All receive frames, good or bad, that meet the address filtering rules, appear on the receive AXI_STREAM interface with the only indication of good versus bad being the value of `axi_str_avbrx_tuser` during `axi_str_avbrx_tlast`. Figure 2-29 shows the receive waveforms for the AVB interface operating at 100 Mb/s.

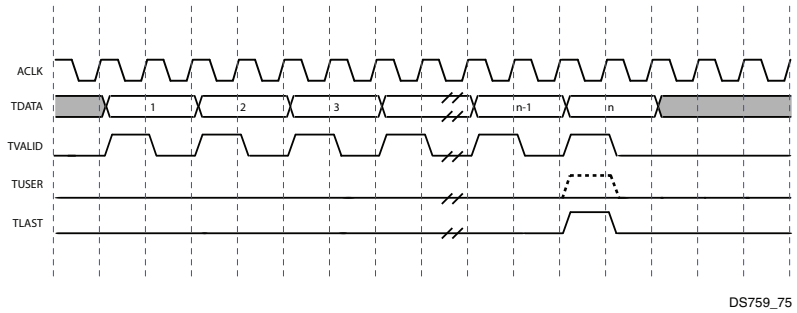


Figure 2-29: **100 Mb/s Receive AVB AXI4-Stream**

Statistics Vectors

Transmit Statistics Vector

The transmitter provides 32 bits of statistics for each frame transmitted and a signal which can be used to count the total number of bytes transmitted. Statistics information is provided using a 32-bit vector for one clock cycle, as shown in Figure 2-30. Table 2-37 shows the bit definition of the transmit statistics. Bits 28 to 20 are always driven to zero because half-duplex is not supported. The waveform in Figure 2-30 represents the statistics counter updates for the corresponding vector bits. The entire vector otherwise is not accessible through an addressable register or available on the external ports.

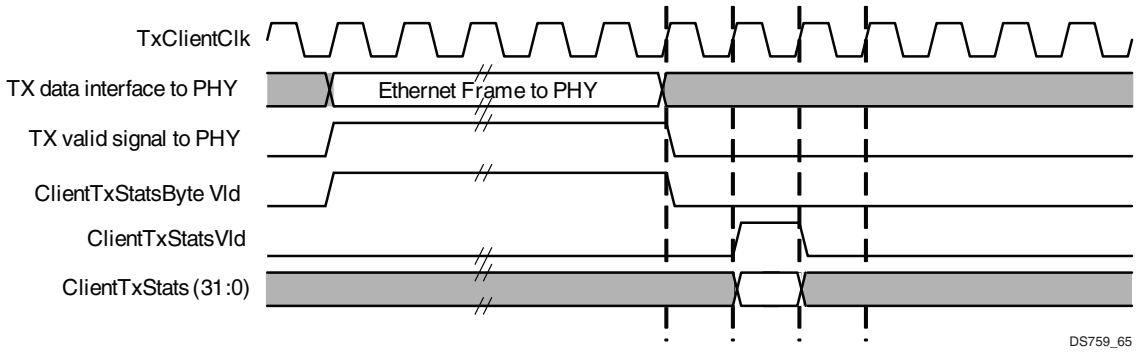


Figure 2-30: **TEMAC Transmit Statistics Waveforms**

Table 2-37: **Transmit Statistics Bit Definitions**

TX Statistics	Name	Description
31	PAUSE_FRAME_TRANSMITTED	Asserted if the previous frame was a pause frame initiated by writing to the TPF register.
30	BYTE_VALID	TEMAC: Asserted if a MAC frame byte (Destination Address to FCS inclusive) is in the process of being transmitted. This is valid on every clock cycle. Do not use this as an enable signal to indicate that data is present on the transmit data pins going to the PHY.
29	Reserved (driven to zero)	Returns 0.
28–25(1)	TX_ATTEMPTS(3:0)	Full Duplex: Returns 0s Half Duplex: The number of attempts that have been made to transmit the previous frame. This is a 4-bit number: 0 should be interpreted as 1 attempt; 1 as 2 attempts, up until 15 as 16 attempts. Only full-duplex is supported.
24(1)	Reserved (driven to zero)	Returns 0.
23(1)	EXCESSIVE COLLISION	Full Duplex: Returns 0s Half Duplex: Asserted if a collision has been detected on each of the last 16 attempts to transmit the previous frame. Only full-duplex is supported.

Table 2-37: **Transmit Statistics Bit Definitions *(Cont'd)***

TX Statistics	Name	Description
22[1]	LATE_COLLISION	Full Duplex: Returns 0s Half Duplex: Asserted if a late collision occurred during frame transmission. Only full-duplex is supported.
21[1]	EXCESSIVE_DEFERRAL	Full Duplex: Returns 0s Half Duplex: Asserted if the previous frame was deferred for an excessive amount of time as defined by the constant "maxDeferTime" in *IEEE 802.3-2005*. Only full-duplex is supported.
20[1]	TX_DEFERRED	Full Duplex: Returns 0s Half Duplex: Asserted if transmission of the frame was deferred. Only full-duplex is supported.
19	VLAN_FRAME	Asserted if the previous frame contains a VLAN identifier in the Length/Type field when transmitter VLAN operation is enabled
18–5	FRAME_LENGTH_COUNT	The length of the previous frame in number of bytes. The count sticks at 16838 for jumbo frames larger than this value.
4	CONTROL_FRAME	Asserted if the previous frame has the special Control type code 0x8808 in the Length/Type field
3	UNDERRUN_FRAME	Asserted if the previous frame contains an underrun error.
2	MULTICAST_FRAME	Asserted if the previous frame contains a multicast address in the destination address field.
1	BROADCAST_FRAME	Asserted if the previous frame contains a broadcast address in the destination address field.
0	SUCCESSFUL_FRAME	Asserted if the previous frame is transmitted without error.

Notes:

1. Bits 28:20 are for Half-Duplex only. These bits return zero in Full Duplex mode.

Receive Statistics Vector

The receiver provides 28 bits of statistics for each frame transmitted and a valid signal which can be used to count the total number of bytes transmitted. Statistics information is provided using a 28-bit vector for one clock cycle as shown in Figure 2-31. The waveform in Figure 2-31 represents the statistics counter updates for the corresponding vector bits. The entire vector otherwise is not accessible through an addressable register or available on the external ports. Table 2-38 shows the bit definition of the receive statistics.

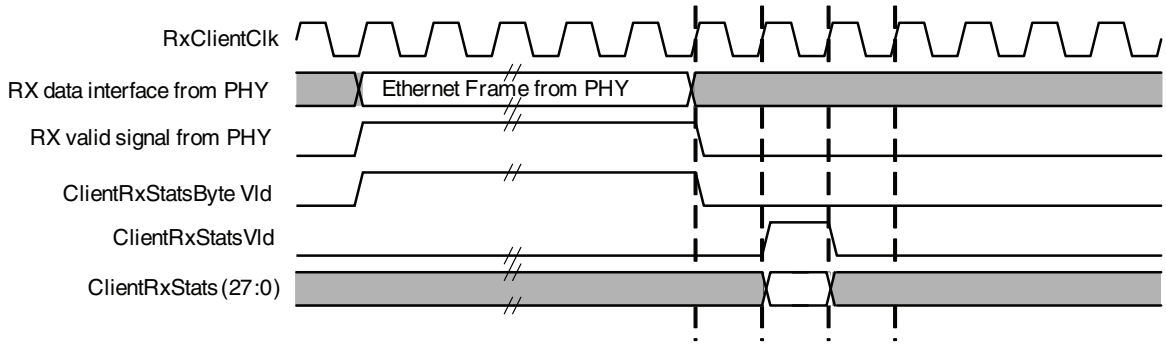


Figure 2-31: **Receive Statistics Waveforms**

Table 2-38: **Receive Statistics Bit Definitions**

RX Statistics	Name	Description
27	ADDRESS MATCH	When the Ethernet MAC is configured in Address Filtering mode, asserted if the previous frame successfully passed the address filter. When the Ethernet MAC is configured in promiscuous mode, this bit is always asserted.
26	ALIGNMENT_ERROR	Used in 10/100 MII mode. Asserted if the previous frame received has an incorrect FCS value and a misalignment occurs when the 4-bit MII data bus is converted to the 8-bit GMII data bus.
25	Length/Type Out Of Range	Asserted if the Length/Type field contains a length value that does not match the number of Ethernet MAC data bytes received. Also asserted High if the Length/Type field indicates that the frame contains padding but the number of Ethernet MAC data bytes received is not equal to 64 bytes (minimum frame size). This bit is not defined when Length/Type field error-checks are disabled or when received frames are less than the legal minimum length.
24	BAD_OPCODE	Asserted if the previous frame is error free. Contains the special control frame identifier in the LT field, but contains an OPCODE unsupported by the Ethernet MAC (any OPCODE other than PAUSE).
23	FLOW_CONTROL_FRAME	Asserted if the previous frame is error-free. Contains the special control frame identifier in the LT field. Contains a destination address matching either the Ethernet MAC control multicast address or the configured source address of the Ethernet MAC. Contains the supported PAUSE OPCODE and is acted upon by the Ethernet MAC.
22	BYTE_VALID	TEMAC: Asserted if a MAC frame byte (Destination Address to FCS inclusive) is in the process of being received. This is valid on every clock cycle. Do not use this as an enable signal to indicate that data is present on the receive data pins going to the receive MAC interface.
21	VLAN_FRAME	Asserted if the previous frame contains a VLAN identifier in the Length/Type field when the receiver VLAN operation is enabled.
20	OUT_OF_BOUNDS	Asserted if the previous frame exceeded the specified IEEE Std 802.3-2005 maximum legal length. This is only valid if jumbo frames are disabled.

Table 2-38: **Receive Statistics Bit Definitions *(Cont'd)***

RX Statistics	Name	Description
19	CONTROL_FRAME	Asserted if the previous frame contains the special control frame identifier in the Length/Type field.
18–5	FRAME_LENGTH_COUNT	The length of the previous frame in number of bytes. The count sticks at 16383 for any jumbo frames larger than this value.
4	MULTICAST_FRAME	Asserted if the previous frame contains a multicast address in the destination field.
3	BROADCAST_FRAME	Asserted if the previous frame contains the broadcast address in the destination field.
2	FCS_ERROR	Asserted if the previous frame received has an incorrect FCS value or the Ethernet MAC detects error codes during frame reception.
1	BAD_FRAME(1)	Asserted if the previous frame received contains errors.
0	GOOD_FRAME(1)	Asserted if the previous frame received is error free.

Notes:

1. If the length/type field error checks are disabled, then a frame containing this type of error is marked as a GOOD_FRAME, providing no additional errors were detected.

Register Space

The AXI Ethernet core contains memory and addressable registers for read and write operations as shown in Table 2-39. All register are directly accessible using a single AXI4-Lite interface. The base address is computed in the Vivado IP Integrator system during creation of the system. All the registers addresses mentioned here are the offset from the base address. The address space from 0x34 to 0x3FFF belongs to the TEMAC. A few registers from the TEMAC are briefly mentioned here for ease of use. See the *LogiCORE IP Tri-Mode Ethernet MAC Product Guide* (PG051) [Ref 1] for more information related to these registers. All reserved address spaces indicated in Table 2-39 return zeros when read.

Table 2-39: **AXI4-Lite Addressable Memory and Soft Registers**

Register Name	AXI4-Lite Address (offset from C_BASEADDR)	Access
Reset and Address Filter register TEMAC (RAF)	0x00000000	Read/Write
Transmit Pause Frame TEMAC (TPF)	0x00000004	Read/Write
Transmit Inter Frame Gap Adjustment TEMAC (IFGP)	0x00000008	Read/Write
Interrupt Status register TEMAC (IS)	0x0000000C	Read/Write
Interrupt Pending register TEMAC (IP)	0x00000010	Read
Interrupt Enable register TEMAC (IE)	0x00000014	Read/Write
Transmit VLAN Tag TEMAC (TTAG)	0x00000018	Read/Write
Receive VLAN Tag TEMAC (RTAG)	0x0000001C	Read/Write
Unicast Address Word Lower TEMAC (UAWL)	0x00000020	Read/Write
Unicast Address Word Upper TEMAC (UAWU)	0x00000024	Read/Write
VLAN TPID TEMAC Word 0 (TPID0)	0x00000028	Read/Write
VLAN TPID TEMAC Word 1 (TPID1)	0x0000002C	Read/Write
PCS PMA TEMAC Status register (PPST)	0x00000030	Read
Reserved	0x00000034 – 0x000001FC	Reserved
Statistics Counters	0x00000200 – 0x000003FC	Read
TEMAC Receive Configuration Word 0 Register (RCW)	0x00000400	Read/Write
TEMAC Receive Configuration Word 1 Register (RCW)	0x00000404	Read/Write
TEMAC Transmitter Configuration register (TC)	0x00000408	Read/Write
TEMAC Flow Control Configuration register (FCC)	0x0000040C	Read/Write
TEMAC Ethernet MAC Mode Configuration register (EMMC)	0x00000410	Read except bits 30 and 31 which are Read/Write
Rx Max Frame Configuration	0x00000414	Read/Write

Table 2-39: **AXI4-Lite Addressable Memory and Soft Registers *(Cont'd)***

Register Name	AXI4-Lite Address (offset from C_BASEADDR)	Access
Tx Max Frame Configuration	0x00000418	Read/Write
RGMII/SGMII Configuration	0x00000420	Read
Reserved	0x0000041C–0x000004F4	Reserved
Identification register	0x000004F8	Read
Ability register	0x000004FC	Read
MII Management Configuration register	0x00000500	Read/Write
MII Management Control	0x00000504	Read/Write
MII Management Write Data	0x00000508	Read/Write
MII Management Read Data	0x0000050C	Read
Reserved	0x00000510–0x000005FC	Reserved
MDIO Interrupt Status register (MIS)	0x00000600	Read/Write
Reserved	0x00000604–0x0000061C	Reserved
MDIO Interrupt Pending register (MIP)	0x00000620	Read
Reserved	0x00000624–0x0000063C	Reserved
MDIO Interrupt Enable register (MIE)	0x00000640	Read/Write
Reserved	0x00000644–0x0000065C	Reserved
MDIO Interrupt Clear register (MIC)	0x00000660	Read/Write
Reserved	0x00000664–0x000006FC	Reserved
TEMAC Unicast Address Word 0 register (UAW0)	0x00000700	Read/Write
TEMAC Unicast Address Word 1 register (UAW1)	0x00000704	Read/Write
Filter Mask Index (FMI)	0x00000708	Read/Write
Reserved	0x0000070C	Reserved
Address Filter (31:0)	0x00000710	Read/Write
Address Filter (47:32)	0x00000714	Read/Write
Reserved	0x00000718–0x0000078C	Read/Write
Reserved	0x00000790–0x00000FFC	Reserved
Reserved	0x00001000–0x00003FFC	Reserved
Transmit VLAN Data Table TEMAC	0x00004000–0x00007FFC	Read/Write
Receive VLAN Data Table TEMAC	0x00008000–0x0000BFFC	Read/Write
Reserved	0x0000C000–0x0000FFFC	Reserved
Ethernet AVB	0x00010000–0x00013FFC	Read/Write
Reserved	0x00014000–0x0001FFFC	Reserved
Multicast Address Table TEMAC	0x00020000–0x0003FFFC	Read/Write

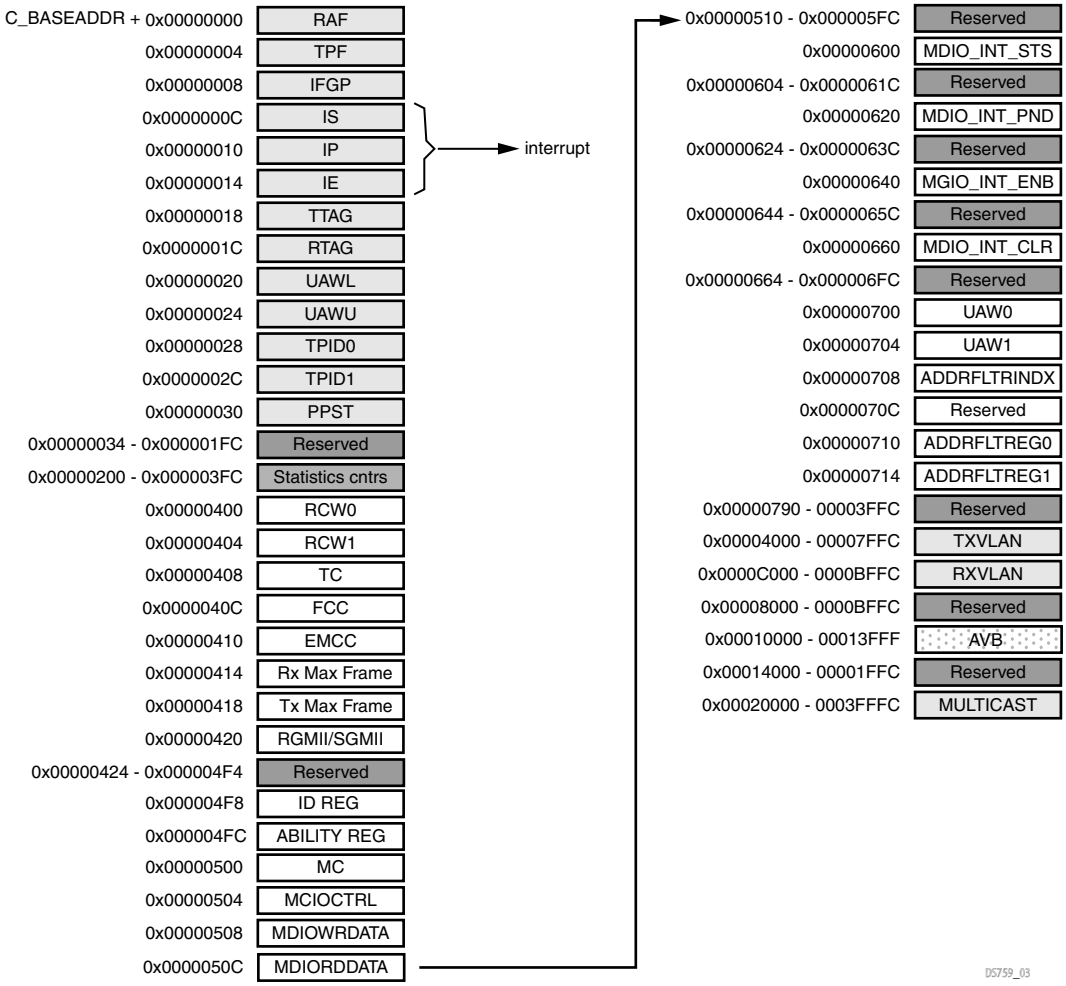


Figure 2-32: **Address Mapping Diagram**

Reset and Address Filter Register — Offset 0x0000_0000

The Reset and Address Filter (RAF) register is shown in Figure 2-33. This register allows the software to block receive multicast and broadcast Ethernet frames. Additional receive address filtering is provided with the registers in Table 2-74 and Table 2-75. The multicast reject bit provides a means of blocking receive multicast Ethernet frames without having to clear out any multicast address values stored in the multicast address table. It also provides a means for allowing more than four multicast addresses to be received (the limit of the multicast address table). To accept more than four multicast addresses, the FMI register would be set to promiscuous mode and the multicast reject bit of this register set to allow multicast frames. See Extended Multicast Address Filtering Mode for more information. Software might also need to filter out additional receive frames with other addresses. The broadcast reject bit provides the only means for rejecting receive broadcast Ethernet frames.

As additional functionality was added to the core, this register became the convenient location for new bits to control those new functions. Care has been taken to minimize the effect of these new bits on existing applications by ensuring that the default values of these bits disable new functionality. This setting ensures that when applications do not use the new bits, the core operates the way it did previously.

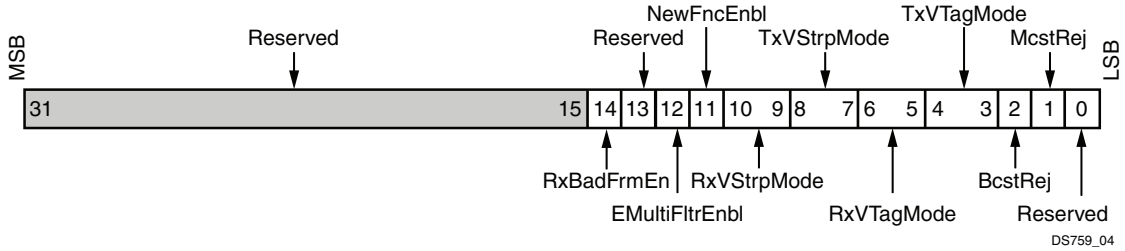


Figure 2-33: **Reset and Address Filter Register (offset 0x0000_0000)**

Table 2-40 shows the Reset and Address Filter register bit definitions.

Table 2-40: **Reset and Address Filter Register Bit Definitions**

Bits	Name	Core Access	Reset Value	Description
31–15	Reserved	Read	0x0	**Reserved:** These bits are reserved for future use and always return zero.
14	RxBadFrmEn	Read/Write	0	**Receive Bad Frame Enable**. This bit provides a means for allowing bad receive frames to be accepted and passed to the RX AXI4-Stream interface as if they were good frames. 0 – Normal operation, bad frames are rejected. 1 – Bad frames are accepted.
13	Reserved	Read	0	**Reserved**: These bits are reserved for future use and always return zero.
12	EMultiFltrEnbl	Read/Write	0	**Enhanced Multicast Filter Enable**: This bit provides a simple way to disable the new enhanced multicast filtering if present. This is necessary if promiscuous address reception mode is desired or if use of the built-in 4 TEMAC multicast address registers is required when the core includes the enhanced multicast address filtering function enabled at build time by the C_MCAST_EXTEND parameters. See Extended Multicast Address Filtering Mode for more details. 0 – Disable enhanced multicast address filtering mode. 1 – Enable enhanced multicast address filtering mode if present.

Table 2-40: **Reset and Address Filter Register Bit Definitions *(Cont'd)***

Bits	Name	Core Access	Reset Value	Description
11	NewFncEnbl	Read/Write	0	**New Functions Enable**: This bit provides a simple way to disable new functions that have been added in this version. This includes the VLAN tagging, VLAN stripping, VLAN translation, and extended multicast filtering. Enabling the new functions only affect operation if the functions have been added to the design using the appropriate parameters at build-time. 0 – Disable new functions. 1 – Enable new functions if present.
10–9	RxVStrpMode	Read/Write	00	**Receive VLAN Strip Mode**: These bits select the operation mode for receive VLAN stripping and are only used when C_RXVLAN_STRP = 1. Valid VLAN TPID values must be initialized in the TPID0 and TPID1 registers. For mode 11, the Receive VLAN data table must be initialized. See Extended VLAN Support for more details. 00 – No VLAN tags are stripped from receive frames. 01 – One VLAN tag are stripped from all receive frames that have VLAN tags. 10 – Reserved. 11 – One VLAN tag is stripped from select receive frames that already have VLAN tags.
8–7	TxVStrpMode	Read/Write	00	**Transmit VLAN Strip Mode**: These bits select the operation mode for transmit VLAN stripping and are only used when C_TXVLAN_STRP = 1. Valid VLAN TPID values must be initialized in the TPID0 and TPID1 registers. For mode 11, the Transmit VLAN data table must be initialized. See Extended VLAN Support for more details. 00 – No VLAN tags are stripped from transmit frames. 01 – One VLAN tag is stripped from all transmit frames that have VLAN tags. 10 – Reserved. 11 – One VLAN tag is stripped from select transmit frames that already have VLAN tags.
6–5	RxVTagMode	Read/Write	00	**Receive VLAN Tag Mode**: These bits select the operation mode for receive VLAN tagging and are only used when C_RXVLAN_TAG = 1. The VLAN tag that is added is from the RTAG register. Valid VLAN TPID values must be initialized in the TPID0 and TPID1 registers. For mode 11, the Receive VLAN data table must be initialized. See Extended VLAN Support for more details. 00 – No VLAN tags are added to receive frames. 01 – VLAN tags are added to all receive frames. 10 – VLAN tags are added to all receive frames that already have a VLAN tag. 11 – VLAN tags are added to select receive frames that already have VLAN tags.

Table 2-40: **Reset and Address Filter Register Bit Definitions *(Cont'd)***

Bits	Name	Core Access	Reset Value	Description
4–3	TxVTagMode	Read/Write	00	**Transmit VLAN Tag Mode**: These bits select the operation mode for transmit VLAN tagging and are only used when C_TXVLAN_TAG = 1. The VLAN tag that is added is from the TTAG register. Valid VLAN TPID values must be initialized in the TPID0 and TPID1 registers. For mode 11, the Transmit VLAN data table must be initialized. See Extended VLAN Support for more details. 00 – No VLAN tags are added to transmit frames. 01 – VLAN tags are added to all transmit frames. 10 – VLAN tags are added to all transmit frames that already have a VLAN tag. 11 – VLAN tags are added to select transmit frames that already have VLAN tags.
2	BcstRej	Read/Write	0	**Reject Receive Broadcast Destination Address:** This bit provides a means for accepting or rejecting broadcast Ethernet frames. 0 – Accept receive broadcast destination address Ethernet frames. 1 – Reject all receive broadcast destination address Ethernet frames. This is the only method available for blocking broadcast Ethernet frames.
1	McstRej	Read/Write	0	**Reject Receive Multicast Destination Address:** This bit provides a means for accepting or rejecting multicast Ethernet frames. 0 – Accept receive multicast destination address Ethernet frames that meet address filtering specified in FMI register and/or the multicast address table. 1 – Reject all receive multicast destination address Ethernet frames regardless of FMI register and multicast address table.
0	Reserved	Read	0	**Reserved:** These bits are reserved for future definition and always return zero.

Transmit Pause Frame Register — Offset 0x0000_0004

The Transmit Pause Frame (TPF) TEMAC register is shown in Figure 2-34. This register provides a value of pause when enabled by the FCC register (TEMAC Flow Control Configuration Register — Offset 0x0000_040C). When enabled, the Ethernet transmits a pause frame whenever this register is written. Pause values are defined in units of pause quanta which are defined as 512 bit times for the current transmission speed. Therefore, pause times can have values ranging from 0 to 65,535 * 512 bit times.

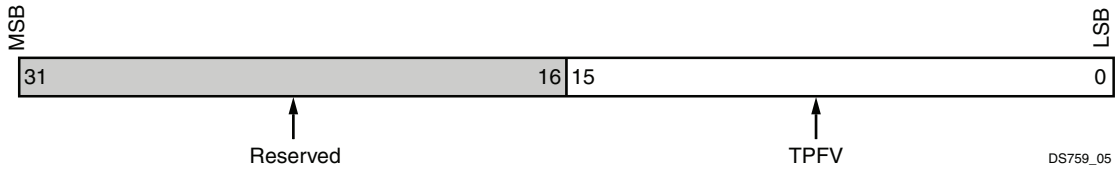


Figure 2-34: **Transmit Pause Frame Register (offset 0x0000_0004)**

Table 2-41 shows the Transmit Pause Frame register bit definitions.

Table 2-41: **Transmit Pause Frame Register Bit Definitions**

Bits	Name	Core Access	Reset Value	Description
31–16	Reserved	Read	0x0	**Reserved**: These bits are reserved for future use and always return zero.
15–0	TPFV	Read/Write	0x0	**Transmit Pause Frame Value**: These bits denote the value of the transmit pause frame pause time in units of 512 bit times. If enabled by the FCC register, writing a value into this register initiates the transmission of a single pause frame with the pause value defined in this field.

Transmit Inter Frame Gap Adjustment Register — Offset 0x0000_0008

The Transmit Inter Frame Gap Adjustment (IFGP) register is shown in Figure 2-35. This register provides a duration value of Inter Frame Gap when enabled by the TC register (TEMAC Transmit Configuration Register — Offset 0x0000_0408). When enabled, the TEMAC uses the value of this register to extend the Inter Frame Gap beyond the minimum of 12 idle cycles which is 96-bit times on the Ethernet Interface.

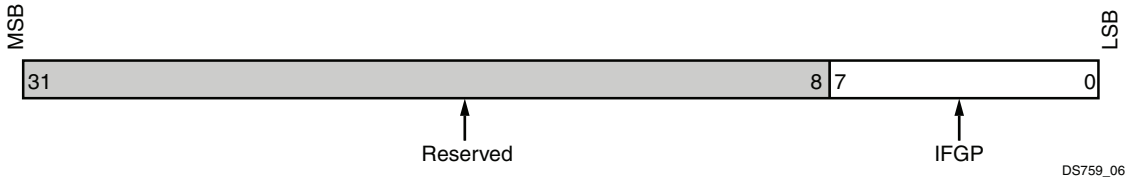


Figure 2-35: **Transmit Inter Frame Gap Adjustment Register (offset 0x0000_0008)**

Table 2-42 shows the Transmit Inter Frame Gap Adjustment register bit definitions.

Table 2-42: **Transmit Inter Frame Gap Adjustment Register Bit Definitions**

Bits	Name	Core Access	Reset Value	Description
31–8	Reserved	Read	0x0	**Reserved**: These bits are reserved for future use and always return zero.
7–0	IFGP0	Read/Write	0x0	**Transmit Inter Frame Gap Adjustment Value**: This 8-bit value can be used along with the Inter Frame Gap Adjustment Enable bit of the Transmit Configuration register (TEMAC Transmit Configuration Register — Offset 0x0000_0408) to increase the Transmit Inter Frame Gap. This value is the width of the IFG in idle cycles. Each idle cycle is 8 bit times on the Ethernet interface. The minimum IFG time is 12 idle cycles which is 96 bit-times. If this field value is less than 12 or if IFGP adjustment is disabled in the Transmit Configuration register, an IFGP of 12 idle cycles (96-bit times) is used.

Interrupt Status Register — Offset 0x0000_000C

The Interrupt Status (IS) register is shown in Figure 2-36. This register combined with the IE, IP, MIS, and MIE registers define the interrupt interface of the AXI Ethernet core. The Interrupt Status register uses one bit to represent each AXI Ethernet core internal interruptible condition. One of these interruptible conditions, register Access Complete (HardAcsCmplt), comes from the TEMAC component and is further defined and enabled by the MIS and MIE registers which are described in MDIO Interrupt Status Register — Offset 0x00000600 and MDIO Interrupt Enable Register — Offset 0x00000640.

When an interruptible condition occurs, it is captured in this register (represented as the corresponding bit being set to 1) even if the condition goes away. The latched interruptible condition is cleared by writing a 1 to that bit location. Writing a 1 to a bit location that is 0 has no effect. Likewise, writing a 0 to a bit location that is 1 has no effect. Multiple bits can be cleared in a single write.

IMPORTANT: *For any bit set in the Interrupt Status register, a corresponding bit must be set in the Interrupt Enable register for the same bit position to be set in the Interrupt Pending register.*

Whenever any bits are set in the Interrupt Pending register, the `interrupt` signal is driven active-High out of the AXI Ethernet core. Figure 2-37 shows the structure of the interrupt register.

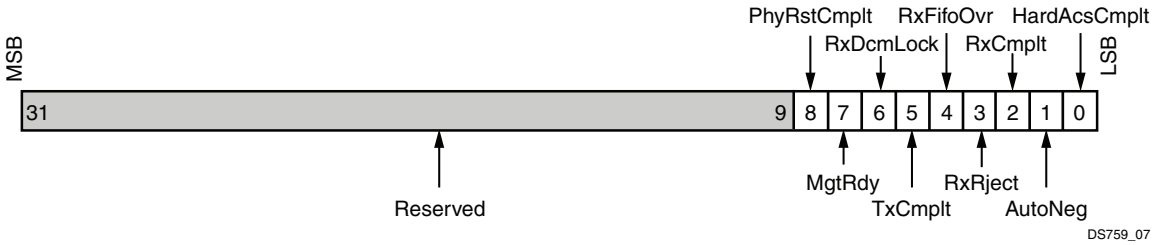


Figure 2-36: **Interrupt Status Register (offset 0x0000_000C)**

Table 2-43 shows the Interrupt Status register bit definitions.

Table 2-43: **Interrupt Status Register Bit Definitions**

Bits	Name	Core Access	Reset Value	Description
31–9	Reserved	Read	0x0	**Reserved:** These bits are reserved for future use and always return zero.
8	PhyRstCmplt	Read/Write	0	**PHY Reset Complete.** When set to 1, this bit indicates the PHY can be accessed. This signal does not transition to 1 for 5 ms after PHY_RST_N transitions to 1. 0 – PHY not ready 1 – PHY ready
7	MgtRdy(1)	Read/Write	0 / 1	**Serial Transceiver Ready:** This bit indicates if the TEMAC is out of reset and ready for use. In systems that use a serial transceiver, this bit goes to 1 when the serial transceiver is ready to use. Prior to that time, access of TEMAC registers does not complete and the core does not operate. In systems that do not use an serial transceiver, this signal goes to 1 immediately after reset. 0 – serial transceiver / TEMAC not ready 1 – serial transceiver / TEMAC ready
6	RxDcmLock	Read/Write	1	**Receive DCM Lock:** No longer used, but reserved for future use. This bit is always one. 0 – Rx DCM not locked 1 – Rx DCM Locked
5	TxCmplt	Read/Write	0	**Transmit Complete:** This bit indicates that a frame was successfully transmitted. 0 – no frame transmitted 1 – frame transmitted
4	RxMemOvr	Read/Write	0	**Receive Memory Overrun:** This bit indicates that the receive Memory overflowed while receiving an Ethernet frame. 0 – normal operation, no overflow occurred 1 – receive Memory overflow occurred and data was lost
3	RxRject(2)	Read/Write	0	**Receive Frame Rejected:** This bit indicates that a receive frame was rejected. 0 – no receive frame rejected 1 – receive frame was rejected
2	RxCmplt	Read/Write	0	**Receive Complete:** This bit indicates that a packet was successfully received. 0 – no frame received 1 – frame received
1	AutoNeg	Read/Write	0	**Auto Negotiation Complete:** This bit indicates that auto negotiation of the SGMII or 1000 Base-X interface has completed. 0 – auto negotiation not complete 1 – auto negotiation complete

Table 2-43: **Interrupt Status Register Bit Definitions *(Cont'd)***

Bits	Name	Core Access	Reset Value	Description
0	HardAcsCmplt	Read/Write	0	**Hard register Access Complete:** This bit indicates that an access of the TEMAC component has completed. 0 – Hard register access is not complete 1 – Hard register access is complete

Notes:

1. This bit resets to 0 but can change to 1 immediately after reset is removed. This bit can remain at 0 for some time in systems that are using serial transceivers when the serial transceivers are not yet ready for use.
2. See Figure 2-37 for conditions that cause the receive frame reject interrupt to occur. The receive frame reject interrupt occurs for any of the following reasons:
   A. The frame does not meet the Ethernet frame requirements (bad FCS, bad length, etc).
   B. In addition to the frame being good but not meeting the destination address filtering, the frame also does not match one of the 4 multicast table entries, it is not a broadcast frame, it does not match the unicast address register.
   C. The core was built to support extended multicast address filtering (C_MCAST_EXTEND=1).
   D. The frame is good and meets the destination address filtering but it is a multicast frame and the multicast reject bit is set in the soft RAF register.
   E. The frame is good and meets the destination address filtering but it is a broadcast frame and the broadcast reject bit is set in the soft RAF register.

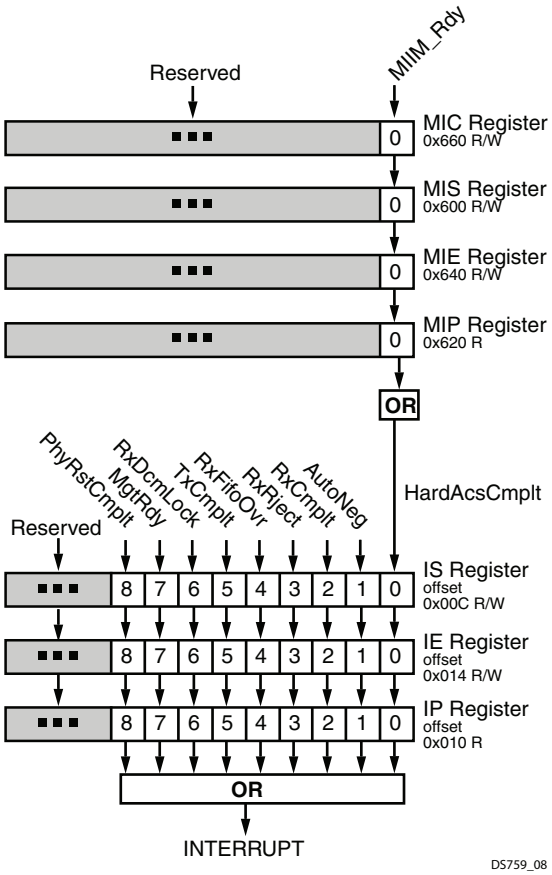


Figure 2-37: **AXI Ethernet Core Interrupt Structure**

Interrupt Pending Register — Offset 0x0000_0010

The Interrupt Pending (IP) register is shown in Figure 2-38. This register combined with the IS, IE, MIS, and MIE registers define the interrupt interface of the AXI Ethernet core. The Interrupt Pending register uses one bit to represent each AXI Ethernet core internal interruptible condition that is represented in the Interrupt Status register.

If one or more interrupt is latched in the Interrupt Status register and corresponding enable bits are set in the Interrupt Enable register, the corresponding bit is set in the Interrupt Pending register. If one or more bits is set in the Interrupt Pending register, the `interrupt` signal is driven active-High out of the AXI Ethernet core.

The Interrupt Pending register always represents the state of the Interrupt Status register bitwise ANDed with the IE register. The Interrupt Pending register is read only. To clear a bit in the Interrupt Pending register, either the corresponding bit must be cleared in either the Interrupt Status register or in the Interrupt Enable register.

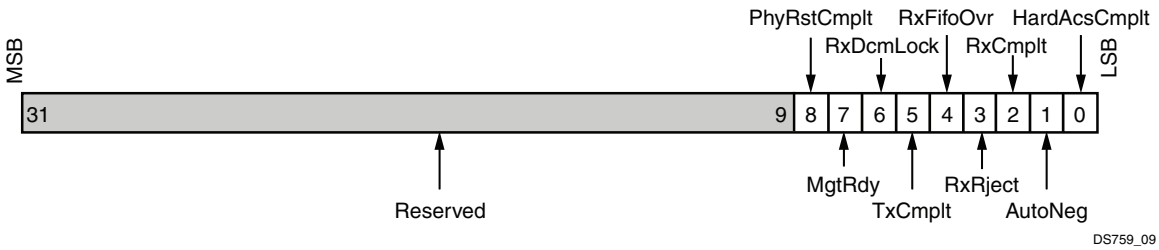


Figure 2-38: **Interrupt Pending Register (offset 0x0000_0010)**

Table 2-44 shows the Interrupt Pending register bit definitions.

Table 2-44: **Interrupt Pending Register Bit Definitions**

Bits	Name	Core Access	Reset Value	Description
31–9	Reserved	Read	0x0	**Reserved:** These bits are reserved for future use and always return zero.
8	PhyRstCmplt	Read/Write	0	**PHY Reset Complete:** When set to 1, this bit indicates the PHY can be accessed. This signal does not transition to 1 for 5 ms after PHY_RST_N transitions to 1. 0 – PHY not ready 1 – PHY ready
7	MgtRdy	Read/Write	0	**MGT Ready:** This bit indicates if the TEMAC is out of reset and ready for use. In systems that use an serial transceiver, this bit goes to 1 when the serial transceiver is ready to use. Prior to that time, access of TEMAC registers does not complete and the core does not operate. In systems that do not use an serial transceiver, this signal goes to 1 immediately after reset. 0 – Serial Transceiver / TEMAC not ready 1 – Serial Transceiver / TEMAC ready

Table 2-44: **Interrupt Pending Register Bit Definitions *(Cont'd)***

Bits	Name	Core Access	Reset Value	Description
6	RxDcmLock	Read/Write	0	**Receive DCM Lock:** No longer used, but reserved for future use. This bit is always one. 0 – Rx DCM not locked 1 – Rx DCM Locked
5	TxCmplt	Read/Write	0	**Transmit Complete:** This bit indicates that a frame was successfully transmitted. 0 – no frame transmitted 1 – frame transmitted
4	RxMemOvr	Read/Write	0	**Receive Memory Overrun:** This bit indicates that the receive Memory overflowed while receiving an Ethernet frame. 0 – normal operation, no overflow occurred 1 – receive Memory overflow occurred and data was lost
3	RxRject	Read/Write	0	**Receive Frame Rejected:** This bit indicates that a receive frame was rejected. 0 – no receive frame rejected 1 – receive frame was rejected
2	RxCmplt	Read/Write	0	**Receive Complete:** This bit indicates that a packet was successfully received. 0 – no frame received 1 – frame received
1	AutoNeg	Read/Write	0	**Auto Negotiation Complete:** This bit indicates that auto negotiation of the SGMII or 1000 Base-X interface has completed. 0 – auto negotiation not complete 1 – auto negotiation complete
0	HardAcsCmplt	Read/Write	0	**Hard register Access Complete:** This bit indicates that an access of the TEMAC component has completed. 0 – Hard register access is not complete 1 – Hard register access is complete

Interrupt Enable Register — Offset 0x0000_0014

The Interrupt Enable (IE) register is shown in Figure 2-39. This register, combined with the IS, IP, MIS, and MIE registers, define the interrupt interface of the AXI Ethernet core. The Interrupt Enable register uses one bit to represent each AXI Ethernet core internal interruptible condition represented in the Interrupt Status register. Each bit set in the Interrupt Enable register allows an interruptible condition bit in the Interrupt Status register to pass through to the Interrupt Pending register.

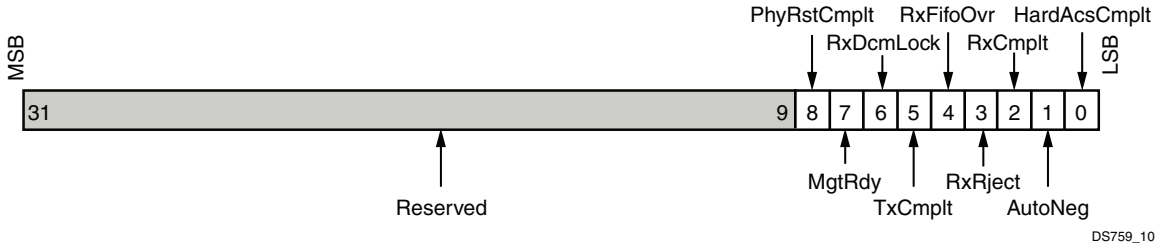


Figure 2-39: **Interrupt Enable Register (offset 0x0000_0014)**

Table 2-45 shows the Interrupt Enable register bit definitions.

Table 2-45: **Interrupt Enable Register Bit Definitions**

Bits	Name	Core Access	Reset Value	Description
31–9	Reserved	Read	0x0	**Reserved:** These bits are reserved for future definition and always return zero.
8	PhyRstCmplt	Read/Write	0	**PHY Reset Complete:** Bit used to enable interrupt. 0 – Interrupt Disabled 1 – Interrupt Enabled
7	MgtRdy	Read/Write	0	**MGT Ready:** Bit used to enable interrupt. 0 – Interrupt Disabled 1 – Interrupt Enabled
6	RxDcmLock	Read/Write	0	**Receive DCM Lock:** Bit used to enable interrupt. 0 – Interrupt Disabled 1 – Interrupt Enabled
5	TxCmplt	Read/Write	0	**Transmit Complete:** Bit used to enable interrupt. 0 – Interrupt Disabled 1 – Interrupt Enabled
4	RxMemOvr	Read/Write	0	**Receive Memory Overrun:** Bit used to enable interrupt. 0 – Interrupt Disabled 1 – Interrupt Enabled
3	RxRject	Read/Write	0	**Receive Frame Rejected:** Bit used to enable interrupt. 0 – Interrupt Disabled 1 – Interrupt Enabled
2	RxCmplt	Read/Write	0	**Receive Complete:** Bit used to enable interrupt. 0 – Interrupt Disabled 1 – Interrupt Enabled
1	AutoNeg	Read/Write	0	**Auto Negotiation Complete:** Bit used to enable interrupt. 0 – Interrupt Disabled 1 – Interrupt Enabled
0	HardAcsCmplt	Read/Write	0	**Hard Register Access Complete:** Bit used to enable interrupt. 0 – Interrupt Disabled 1 – Interrupt Enabled

Transmit VLAN Tag Register — Offset 0x0000_0018

The Transmit VLAN Tag (TTAG) register is shown in Figure 2-40. This register is only used when the VLAN tagging is included in the core at build-time (C_TXVLAN_TAG = 1). When a VLAN tag is added to a transmit frame, this is the value that is added to the frame right after the source address field. See Extended VLAN Supportfor more information about how VLAN tagging is performed.

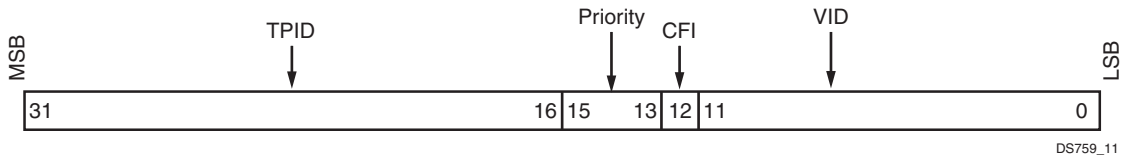


Figure 2-40: **Transmit VLAN Tag Register (offset 0x0000_0018)**

Table 2-46 shows the Transmit VLAN Tag register bit definitions.

Table 2-46: **Transmit VLAN Tag register Bit Definitions**

Bits	Name	Core Access	Reset Value	Description
31–16	TPID	Read/Write	0x0	Tag Protocol Identifier.
15–13	Priority	Read/Write	0x0	User Priority.
12	CFI	Read/Write	0	Canonical Format Indicator.
11–0	VID	Read/Write	0x0	**VLAN identifier:** Uniquely identifies the VLAN to which the frame belongs.

Receive VLAN Tag Register — Offset 0x0000_001C

The Receive VLAN Tag (RTAG) register is shown in Figure 2-41. This register is only used when the VLAN tagging is included in the core at build-time (C_RXVLAN_TAG = 1). When a VLAN tag is added to a receive frame, this is the value that is added to the frame right after the source address field. See Extended VLAN Support for more information about how VLAN tagging is performed.

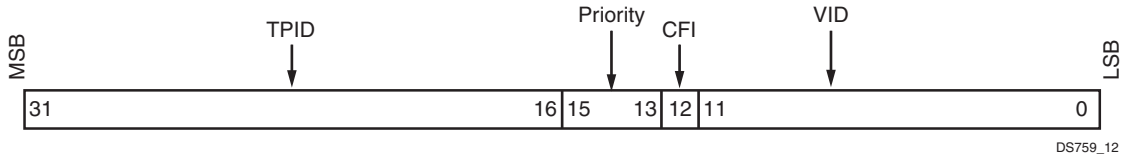


Figure 2-41: **Receive VLAN Tag Register (offset 0x0000_001C**

Table 2-47 shows the Receive VLAN Tag register bit definitions.

Table 2-47: **Receive VLAN Tag Register Bit Definitions**

Bits	Name	Core Access	Reset Value	Description
31–16	TPID	Read/Write	0x0	Tag Protocol Identifier.
15–13	Priority	Read/Write	0x0	User Priority.
12	CFI	Read/Write	0	Canonical Format Indicator.
11–0	VID	Read/Write	0x0	**VLAN identifier:** Uniquely identifies the VLAN to which the frame belongs

Unicast Address Word Lower Register — Offset 0x0000_0020

The Unicast Address Word Lower (UAWL) register is shown in Figure 2-42. This register and the Unicast Address Word Upper (UAWU) register are *only used when extended multicast filtering is included* in the core at build-time (C_MCAST_EXTEND = 1) and is enabled. These registers should not be confused with the UAW0 and UAW1 registers which are registers inside the TEMAC core which are *only used when extended multicast filtering is excluded in the core at build-time or is disabled*.

IMPORTANT: *When using extended multicast filtering, the TEMAC core must be placed in promiscuous address filtering mode.*

This register allows filtering of unicast frames not matching the address stored in these registers. See Extended Multicast Address Filtering Mode for more information.

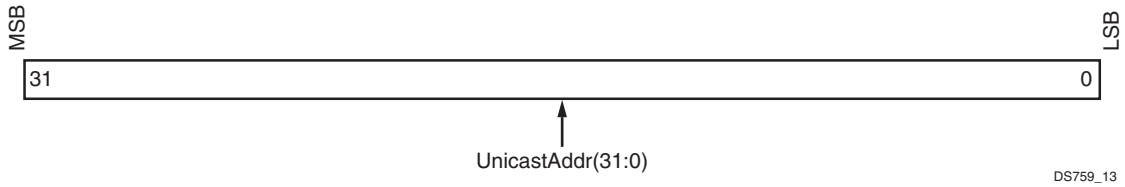


Figure 2-42: **Unicast Address Word Lower Register (offset 0x020)**

Table 2-48 shows the Unicast Address Word Lower register bit definitions.

Table 2-48: **Unicast Address Word Lower Register Bit Definitions**

Bits	Name	Core Access	Reset Value	Description
31–0	UnicastAddr	Read/Write	0x00000000	**Unicast Address (31:0):** This address is used to match against the destination address of any received frames. The address is ordered so the first byte transmitted/received is the lowest positioned byte in the register; for example, a MAC address of AA-BB-CC-DD-EE-FF would be stored in UnicastAddr(47:0) as 0xFFEEDDCCBBAA.

Unicast Address Word Upper Register — Offset 0x0000_0024

The Unicast Address Word Upper (UAWU) register is shown in Figure 2-43. This register and the register are only used when extended multicast filtering is included in the core at build-time (C_MCAST_EXTEND = 1).

IMPORTANT: *When using extended multicast filtering, the TEMAC core must be placed in promiscuous address filtering mode.*

This register allows filtering of unicast frames not matching the address stored in these registers. See Extended Multicast Address Filtering Mode for more information.

Figure 2-43: **Unicast Address Word Upper Register (offset 0x024)**

Table 2-49 shows the Unicast Address Word Upper register bit definitions.

Table 2-49: **Unicast Address Word Upper Register Bit Definitions**

Bits	Name	Core Access	Reset Value	Description
31–16	Reserved	Read	0x0	**Reserved:** These bits are reserved for future use and always return zero.
15–0	UnicastAddr	Read/Write	0x00000000	**Unicast Address (47:32):** This address is used to match against the destination address of any received frames. The address is ordered so the first byte transmitted/received is the lowest positioned byte in the register; for example, a MAC address of AA-BB-CC-DD-EE-FF would be stored in UnicastAddr(47:0) as 0xFFEEDDCCBBAA.

VLAN TPID Word 0 Register — Offset 0x0000_0028

The VLAN TPID Word 0 (TPID0) register is shown in Figure 2-44. This register is only used when transmit and/or receive VLAN functions are included in the core at build-time (C_TXVLAN_TAG = 1 and/or C_RXVLAN_TAG = 1 and/or C_TXVLAN_STRP= 1 and/or C_RXVLAN_STRP= 1 and/or C_TXVLAN_TRAN = 1 and/or C_RXVLAN_TRAN = 1). This register and the following register allow 4 TPID values be specified for recognizing VLAN frames for both the transmit and receive paths. The most common values for VLAN TPID are 0x8100, 0x9100, 0x9200, 0x88A8. See Extended VLAN Support for more information about extended VLAN functions.

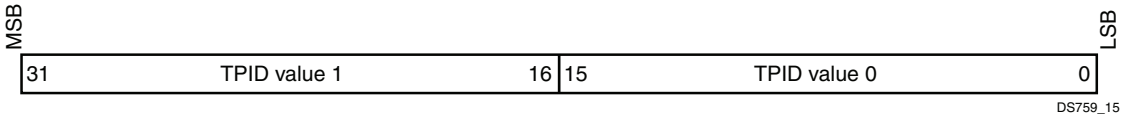


Figure 2-44: **VLAN TPID Word 0 Register (offset 0x0000_0028)**

Table 2-50 shows the VLAN TPID Word 0 register bit definitions.

Table 2-50: **VLAN TPID Word 0 Register Bit Definitions**

Bits	Name	Core Access	Reset Value	Description
31–16	TPID value 1	Read/Write	0x0	**TPID Value 1:** These bits represent one TPID value that is used for recognizing VLAN frames for both the transmit and receive paths.
15–0	TPID value 0	Read/Write	0x0	**TPID Value 0:** These bits represent one TPID value that is used for recognizing VLAN frames for both the transmit and receive paths.

VLAN TPID Word 1 Register — Offset 0x0000_002C

The VLAN TPID Word 1 (TPID1) register is shown in Figure 2-45. This register is only used when transmit and/or receive VLAN functions are included in the core at build-time (C_TXVLAN_TAG = 1 and/or C_RXVLAN_TAG = 1 and/or C_TXVLAN_STRP= 1 and/or C_RXVLAN_STRP= 1 and/or C_TXVLAN_TRAN = 1 and/or C_RXVLAN_TRAN = 1). This register and the previous register allow 4 TPID values be specified for recognizing VLAN frames for both the transmit and receive paths. The most common values for VLAN TPID are 0x8100, 0x9100, 0x9200, 0x88A8. See Extended VLAN Support for more information.

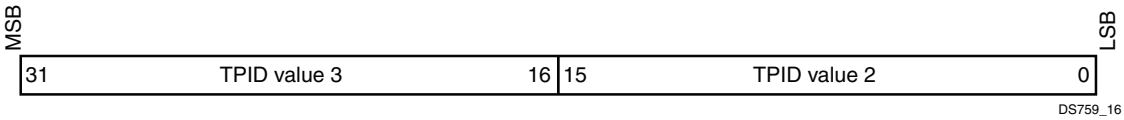


Figure 2-45: **VLAN TPID Word 1 Register (offset 0x0000_002C)**

Table 2-51 shows the VLAN TPID Word 1 register bit definitions.

Table 2-51: **VLAN TPID Word 1 Register Bit Definitions**

Bits	Name	Core Access	Reset Value	Description
31–16	TPID value 3	Read/Write	0x0	**TPID Value 3:** These bits represent one TPID value that is used for recognizing VLAN frames for both the transmit and receive paths.
15–0	TPID value 2	Read/Write	0x0	**TPID Value 2:** These bits represent one TPID value that is used for recognizing VLAN frames for both the transmit and receive paths.

PCS PMA TEMAC Status Register — Offset 0x0000_0030

The PCS PMA TEMAC Status (PPST) register is shown in Figure 2-46. This register reports valid information when AXI Ethernet core is configured for SGMII or 1000Base-X with the TEMAC operating at 10/100/1000 Mb/s (C_TYPE = 1 and C_PHY_TYPE = 4 or 5). It provides additional information about the serial interface status. For all other configurations, this register returns zeroes.

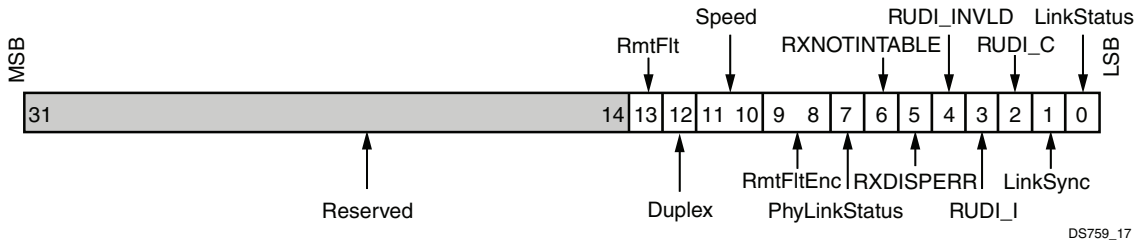


Figure 2-46: **PCS PMA TEMAC Status Register (offset 0x0000_0030)**

Table 2-52 shows the PCS PMA TEMAC Status register bit definitions.

Table 2-52: **PCS PMA TEMAC Status Register Bit Definitions**

Bits	Name	Core Access	Reset Value	Description
31–14	Reserved	Read	0x000000	Reserved
13	RmtFlt	Read	0	**RmtFlt:** Remote Fault (1000Base-X only) When this bit is logic one, it indicates that a remote fault is detected and the type of remote fault is indicated by bits[9:8]. ***Note:*** This bit is only deasserted when an MDIO read is made to status register (register 1 in Table 2-6). This signal has no significance in SGMII PHY mode.
12	Duplex	Read	0	Duplex: Duplex Mode This bit indicates the Duplex mode negotiated with the link partner 1 = Full Duplex 0 = Half Duplex ***Note:*** Half Duplex is not supported
11–10	Speed	Read	00	Speed: Speed This signal indicates the speed negotiated and is only valid when Auto-Negotiation is enabled. The signal encoding is: 11 = Reserved 10 = 1000 Mb/s 01 = 100 Mb/s 00 = 10 Mb/s

Table 2-52: **PCS PMA TEMAC Status Register Bit Definitions *(Cont'd)***

Bits	Name	Core Access	Reset Value	Description
9–8	RmtFltEnc	Read	00	RmtFltEnc: Remote Fault Encoding (1000Base-X only) This signal indicates the remote fault encoding (IEEE 802.3-2008 table 37-3). This signal is validated by bit 13, RmtFlt, and is only valid when Auto-Negotiation is enabled.
7	PhyLinkStatus	Read	0	**PhyLinkStatus:** PHY Link Status (SGMII only) When operating in SGMII mode, this bit represents the link status of the external PHY device attached to the other end of the SGMII link (High indicates that the PHY has obtained a link with its link partner; Low indicates that is has not linked with its link partner). When operating in 1000BASE-X mode this bit remains Low and should be ignored.
6	RXNOTINTABLE	Read	0	**RXNOTINTABLE:** Receive Not In Table. The core has received a code group which is not recognized from the 8B/10B coding tables.
5	RXDISPERR	Read	0	**RXDISPERR:** Receive Disparity Error.The core has received a running disparity error during the 8B/10B decoding function.
4	RUDI_INVLD	Read	0	**RUDI_INVLD:** RUDI(/INVALID/). The core has received invalid data while receiving/C/ or/I/ ordered set.
3	RUDI_I	Read	0	**RUDI_I:** RUDI(/I/). The core is receiving /I/ ordered sets (Idles)
2	RUDI_C	Read	0	**RUDI_C:** RUDI(/C/). The core is receiving /C/ ordered sets (Auto-Negotiation Configuration sequences).
1	LinkSync	Read	0	**LinkSynch:** Link Synchronization. This signal indicates the state of the synchronization state machine (IEEE802.3 figure 36-9) which is based on the reception of valid 8B/10B code groups. This signal is similar to Bit[0] (Link Status), but is NOT qualified with Auto-Negotiation. When High, link synchronization has been obtained and in the synchronization state machine, sync_status = OK. When Low, synchronization has failed.
0	LinkStatus	Read	0	**LinkStatus:** Link Status. This signal indicates the status of the link. When High, the link is valid: synchronization of the link has been obtained *and* Auto-Negotiation (if present and enabled) has successfully completed. When Low, a valid link has not been established. Either link synchronization has failed or Auto-Negotiation (if present and enabled) has failed to complete. When auto-negotiation is enabled this signal is identical to Bit[1].

Statistics Counters — Offset 0x0000_0200-0x0000_03FF

The set of 64-bit counters are only present when selected at build-time. The counters keep track of statistics for the transmit and receive Ethernet traffic and are defined in Table 2-53. The Half-Duplex counters have been omitted because this core does not support Half-Duplex.

Table 2-53: **Statistics Counter Locations**

C_BASEADDR + Offset	Name	Description
0x200	Received bytes (lower 32 bits)	(RXBL) A count of bytes of frames received (destination address to frame check sequence inclusive).
0x204	Received bytes (upper 32 bits)	(RXBU) A count of bytes of frames received (destination address to frame check sequence inclusive).
0x208	Transmitted bytes (lower 32 bits)	(TXBL) A count of bytes of frames transmitted (destination address to frame check sequence inclusive).
0x20C	Transmitted bytes (upper 32 bits)	(TXBU) A count of bytes of frames transmitted (destination address to frame check sequence inclusive).
0x210	Undersize frames received (lower 32 bits)	(RXUNDRL) A count of the number of frames received (less than 64 bytes in length) but otherwise well formed.
0x214	Undersize frames received (upper 32 bits)	(RXUNDRU) A count of the number of frames received (less than 64 bytes in length) but otherwise well formed.
0x218	Fragment frames received (lower 32 bits)	(RXFRAGL) A count of the number of frames received (less than 64 bytes in length) with a bad frame check sequence field.
0x21C	Fragment frames received (upper 32 bits)	(RXFRAGU) A count of the number of frames received (less than 64 bytes in length) with a bad frame check sequence field.
0x220	64 byte Frames Received OK (lower 32 bits)	(RX64BL) A count of error-free frames received that were 64 bytes in length.
0x224	64 byte Frames Received OK (upper 32 bits)	(RX64BU) A count of error-free frames received that were 64 bytes in length.
0x228	65–127 byte Frames Received OK (lower 32 bits)	(RX65B127L) A count of error-free frames received that were between 65 and 127 bytes in length.
0x22C	65–127 byte Frames Received OK (upper 32 bits)	(RX65B127U) A count of error-free frames received that were between 65 and 127 bytes in length.
0x230	128–255 byte Frames Received OK (lower 32 bits)	(RX128B255L) A count of error-free frames received that were between 128 and 255 bytes in length.
0x234	128–255 byte Frames Received OK (upper 32 bits)	(RX128B255U) A count of error-free frames received that were between 128 and 255 bytes in length.
0x238	256–511 byte Frames Received OK (lower 32 bits)	(RX256B511L) A count of error-free frames received that were between 256 and 511 bytes in length.
0x23C	256–511 byte Frames Received OK (upper 32 bits)	(RX256B511U) A count of error-free frames received that were between 256 and 511 bytes in length.
0x240	512–1023 byte Frames Received OK (lower 32 bits)	(RX512B1023L) A count of error-free frames received that were between 512 and 1023 bytes in length.
0x244	512–1023 byte Frames Received OK (upper 32 bits)	(RX512B1023U) A count of error-free frames received that were between 512 and 1023 bytes in length.
0x248	1024–MaxFrameSize byte Frames Received OK (lower 32 bits)	(RX1024BL) A count of error-free frames received that were between 1024 bytes and the specified IEEE 802.3-2002 maximum legal length.

Table 2-53: **Statistics Counter Locations *(Cont'd)***

C_BASEADDR + Offset	Name	Description
0x24C	1024–MaxFrameSize byte Frames Received OK (upper 32 bits)	(RX1024BU) A count of error-free frames received that were between 1024 bytes and the specified IEEE 802.3-2002 maximum legal length.
0x250	Oversize Frames Received OK (lower 32 bits)	(RXOVRL) A count of otherwise error-free frames received that exceeded the maximum legal frame length specified in IEEE 802.3-2002.
0x254	Oversize Frames Received OK (upper 32 bits)	(RXOVRU) A count of otherwise error-free frames received that exceeded the maximum legal frame length specified in IEEE 802.3-2002.
0x258	64 byte Frames Transmitted OK (lower 32 bits)	(TX64BL) A count of error-free frames transmitted that were 64 bytes in length.
0x25C	64 byte Frames Transmitted OK (upper 32 bits)	(TX64BU) A count of error-free frames transmitted that were 64 bytes in length.
0x260	65–127 byte Frames Transmitted OK (lower 32 bits)	(TX65B127L) A count of error-free frames transmitted that were between 65 and 127 bytes in length.
0x264	65–127 byte Frames Transmitted OK (upper 32 bits)	(TX65B127U) A count of error-free frames transmitted that were between 65 and 127 bytes in length.
0x268	128–255 byte Frames Transmitted OK (lower 32 bits)	(TX128B255L) A count of error-free frames transmitted that were between 128 and 255 bytes in length.
0x26C	128–255 byte Frames Transmitted OK (upper 32 bits)	(TX128B255U) A count of error-free frames transmitted that were between 128 and 255 bytes in length.
0x270	256–511 byte Frames Transmitted OK (lower 32 bits)	(TX256B511L) A count of error-free frames transmitted that were between 256 and 511 bytes in length.
0x274	256–511 byte Frames Transmitted OK (upper 32 bits)	(TX256B511U) A count of error-free frames transmitted that were between 256 and 511 bytes in length.
0x278	512–1023 byte Frames Transmitted OK (lower 32 bits)	(TX512B1023L) A count of error-free frames transmitted that were between 512 and 1023 bytes in length.
0x27C	512–1023 byte Frames Transmitted OK (upper 32 bits)	(TX512B1023U) A count of error-free frames transmitted that were between 512 and 1023 bytes in length.
0x280	1024–MaxFrameSize byte Frames Transmitted OK (lower 32 bits)	(TX1024BL) A count of error-free frames transmitted that were between 1024 and the specified IEEE 802.3-2002 maximum legal length.
0x284	1024–MaxFrameSize byte Frames Transmitted OK (upper 32 bits)	TX1025BU) A count of error-free frames transmitted that were between 1024 and the specified IEEE 802.3-2002 maximum legal length.
0x288	Oversize Frames Transmitted OK (lower 32 bits)	(TXOVRL) A count of otherwise error-free frames transmitted that exceeded the maximum legal frame length specified in IEEE 802.3-2002.
0x28C	Oversize Frames Transmitted OK (upper 32 bits)	(TXOVRU) A count of otherwise error-free frames transmitted that exceeded the maximum legal frame length specified in IEEE 802.3-2002.

Table 2-53: **Statistics Counter Locations *(Cont'd)***

C_BASEADDR + Offset	Name	Description
0x290	Frames Received OK (lower 32 bits)	(RXFL) A count of error-free frames received.
0x294	Frames Received OK (upper 32 bits)	(RXFU) A count of error-free frames received.
0x298	Frame Check Sequence Errors (lower 32 bits)	(RXFCSERL) A count of received frames that failed the CRC check and were at least 64 bytes in length.
0x29C	Frame Check Sequence Errors (upper 32 bits)	(RXFCSERU) A count of received frames that failed the CRC check and were at least 64 bytes in length.
0x2A0	Broadcast Frames Received OK (lower 32 bits)	(RXBCSTFL) A count of frames that were successfully received and were directed to the broadcast group address.
0x2A4	Broadcast Frames Received OK (upper 32 bits)	(RXBCSTFU) A count of frames that were successfully received and were directed to the broadcast group address.
0x2A8	Multicast Frames Received OK (lower 32 bits)	(RXMCSTFL) A count of frames that were successfully received and were directed to a non broadcast group address.
0x2AC	Multicast Frames Received OK (upper 32 bits)	(RXMCSTFU) A count of frames that were successfully received and were directed to a non broadcast group address.
0x2B0	Control Frames Received OK (lower 32 bits)	(RXCTRFL) A count of error-free frames received that contained the special Control Frame identifier in the length/type field.
0x2B4	Control Frames Received OK (upper 32 bits)	(RXCTRFU) A count of error-free frames received that contained the special Control Frame identifier in the length/type field.
0x2B8	Length/Type Out of Range (lower 32 bits)	(RXLTERL) A count of frames received that were at least 64 bytes in length where the length/type field contained a length value that did not match the number of MAC data bytes received. The counter also increments for frames in which the length/type field indicated that the frame contained padding, but where the number of MAC data bytes received was greater than 64 bytes (minimum frame size). The exception to the this is when the Length/Type Error Checks are disabled in the chosen MAC, in which case this counter does not increment.
0x2BC	Length/Type Out of Range (upper 32 bits)	(RXLTERU) A count of frames received that were at least 64 bytes in length where the length/type field contained a length value that did not match the number of MAC data bytes received. The counter also increments for frames in which the length/type field indicated that the frame contained padding, but where the number of MAC data bytes received was greater than 64 bytes (minimum frame size). The exception to the this is when the Length/Type Error Checks are disabled in the chosen MAC, which case this counter does not increment.

Table 2-53: **Statistics Counter Locations *(Cont'd)***

C_BASEADDR + Offset	Name	Description
0x2C0	VLAN Tagged Frames Received OK (lower 32 bits)	(RXVLANFL) A count of error-free VLAN frames received. This counter only increments when the receiver is configured for VLAN operation.
0x2C4	VLAN Tagged Frames Received OK (upper 32 bits)	(RXVLANFU) A count of error-free VLAN frames received. This counter only increments when the receiver is configured for VLAN operation.
0x2C8	Pause Frames Received OK (lower 32 bits)	(RXPFL) A count of error-free frames received that: • Contained the MAC Control type identifier 88-08 in the length/type field • Contained a destination address that matched either the MAC Control multicast address or the configured source address of the MAC • Contained the PAUSE opcode • Were acted upon by the MAC
0x2CC	Pause Frames Received OK (upper 32 bits)	(RXPFU) A count of error-free frames received that: • Contained the MAC Control type identifier 88-08 in the length/type field • Contained a destination address that matched either the MAC Control multicast address or the configured source address of the MAC • Contained the PAUSE opcode • Were acted upon by the MAC
0x2D0	Control Frames Received with Unsupported Opcode (lower 32 bits)	(RXUOPFL) A count of error-free frames received that contained the MAC Control type identifier 88- 08 in the length/type field but were received with an opcode other than the PAUSE opcode.
0x2D4	Control Frames Received with Unsupported Opcode (upper 32 bits)	(RXUOPFU) A count of error-free frames received that contained the MAC Control type identifier 88- 08 in the length/type field but were received with an opcode other than the PAUSE opcode.
0x2D8	Frames Transmitted OK (lower 32 bits)	(TXFL) A count of error-free frames transmitted.
0x2DC	Frames Transmitted OK (upper 32 bits)	(TXFU) A count of error-free frames transmitted.
0x2E0	Broadcast Frames Transmitted OK (lower 32 bits)	(TXBCSTFL) A count of error-free frames that were transmitted to the broadcast address.
0x2E4	Broadcast Frames Transmitted OK (upper 32 bits)	(TXBCSTFU) A count of error-free frames that were transmitted to the broadcast address.
0x2E8	Multicast Frames Transmitted OK (lower 32 bits)	(TXMCSTFL) A count of error-free frames that were transmitted to a group destination address other than broadcast.
0x2EC	Multicast Frames Transmitted OK (upper 32 bits)	(TXMCSTFU) A count of error-free frames that were transmitted to a group destination address other than broadcast.

Table 2-53: **Statistics Counter Locations *(Cont'd)***

C_BASEADDR + Offset	Name	Description
0x2F0	Underrun Errors (lower 32 bits)	(TXUNDRERL) A count of frames that would otherwise be transmitted by the core but could not be completed due to the assertion of TX_UNDERRUN during the frame transmission.
0x2F4	Underrun Errors (upper 32 bits)	(TXUNDRERU) A count of frames that would otherwise be transmitted by the core but could not be completed due to the assertion of TX_UNDERRUN during the frame transmission.
0x2F8	Control Frames Transmitted OK (lower 32 bits)	(TXCTRFL) A count of error-free frames transmitted that contained the MAC Control Frame type identifier 88–08 in the length/type field.
0x2FC	Control Frames Transmitted OK (upper 32 bits)	(TXCTRFU) A count of error-free frames transmitted that contained the MAC Control Frame type identifier 88–08 in the length/type field.
0x300	VLAN Tagged Frames Transmitted OK (lower 32 bits)	(TXVLANFL) A count of error-free VLAN frames transmitted. This counter only increments when the transmitter is configured for VLAN operation.
0x304	VLAN Tagged Frames Transmitted OK (upper 32 bits)	(TXVLANFU) A count of error-free VLAN frames transmitted. This counter only increments when the transmitter is configured for VLAN operation.
0x308	Pause Frames Transmitted OK (lower 32 bits)	(TXPFL) A count of error-free PAUSE frames generated and transmitted by the MAC in response to an assertion of pause_req.
0x30C	Pause Frames Transmitted OK (upper 32 bits)	(TXPFU) A count of error-free PAUSE frames generated and transmitted by the MAC in response to an assertion of pause_req.

TEMAC Receive Configuration Word 0 Register – Offset 0x0000_0400

The TEMAC Receive Configuration Word 0 (RCW0) register is shown in Figure 2-47. This register can be written at any time but the receiver logic only applies the configuration changes during Inter-Frame gaps.

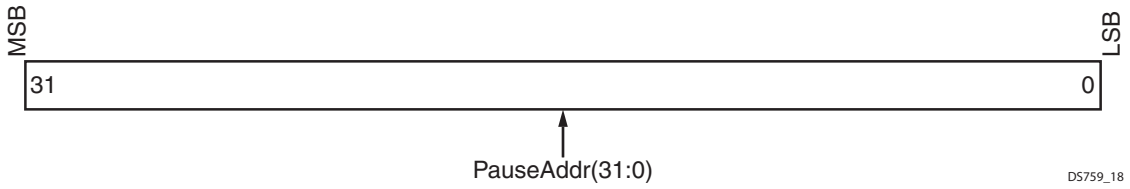


Figure 2-47: **TEMAC Receive Configuration Word 0 (RCW0) Register (offset 0x400)**

Table 2-54 shows the TEMAC Receive Configuration Word 0 register bit definitions.

Table 2-54: **TEMAC Receive Configuration Word 0 (RCW0) Register Bit Definitions**

Bits	Name	Core Access	Reset Value	Description
31–0	PauseAddr	Read/Write	0xDDC CBBAA	**Pause Frame Ethernet MAC Address (31:0):** This address is used to match the destination address of any received flow control frames. It is also used as the source address for any transmitted flow control frames. This address is ordered so that the first byte transmitted/ received is the lowest position byte in the register. For example, a MAC address of AA-BB-CC-DD-EE-FF would be stored in the PauseAddr(47:0) as 0xFFEED-DCCBBAA.

TEMAC Receive Configuration Word 1 Register — Offset 0x0000_0404

The TEMAC Receive Configuration Word 1 (RCW1) register is shown in Figure 2-48. This register can be written at any time but the receiver logic only applies the configuration changes during Inter Frame gaps. The exception to this is the Reset bit which is effective immediately.

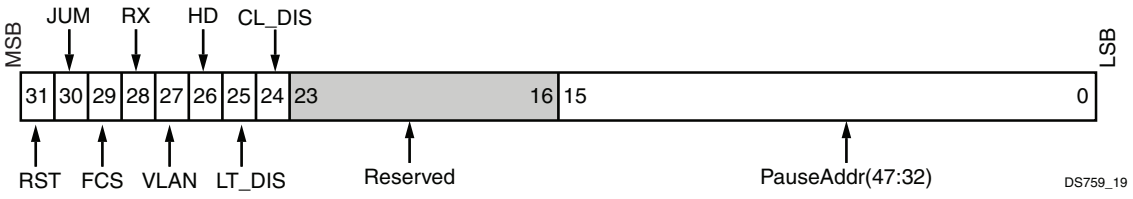


Figure 2-48: **TEMAC Receive Configuration Word 1 (RCW1) Register (offset 0x404)**

Table 2-55 shows the TEMAC Receive Configuration Word1 register bit definitions.

Table 2-55: **TEMAC Receive Configuration Word1 (RCW1) Register Bit Definitions**

Bits	Name	Core Access	Reset Value	Description
31	RST	Read/Write	0	**Reset:** When this bit is 1, the receiver is reset. The bit automatically resets to 0. The reset also sets all of the receiver configuration registers to their default values. Resetting the receiver without resetting AXI Ethernet core could cause the core to be in an unknown state. 0 – no reset 1 – initiate a receiver reset
30	JUM(1)	Read/Write	1	**Jumbo Frame Enable:** When this bit is 1 the receiver accepts frames over the maximum length specified in IEEE Std 802.3-2002 specification. 0 – receive jumbo frames disabled 1 – receive jumbo frames enabled
29	FCS	Read/Write	1	**In-Band FCS Enable:** When this bit is 1, the receiver provides the FCS field with the rest of the frame data. When this bit is 0 the FCS field is stripped from the receive frame data. In either case the FCS field is verified. 0 – strip the FCS field from the receive frame data 1 – provide the FCS field with the receive frame data
28	RX	Read/Write	0	**Receive Enable:** When this bit is 1, the receiver logic is enabled to operate. When this bit is 0, the receiver ignores activity on the receive interface. 0 – receive disabled 1 – receive enabled
27	VLAN(2)	Read/Write	1	**VLAN Frame Enable:** When this bit is 1, the receiver accepts VLAN tagged frames. The maximum payload length increases by four bytes. 0 – receive of VLAN frames disabled 1 – receive of VLAN frames enabled
26	HD	Read/Write	0	**Half-Duplex Mode:** When this bit is 1, the receive operates in half-duplex mode. When this bit is 0, the receiver operates in full-duplex mode. Only full-duplex is supported so this bit should always be set to 0. 0 – full-duplex receive 1 – half-duplex receive
25	LT_DIS	Read/Write	0	**Length/Type Field Valid Check Disable:** When this bit is 1, it disables the Length/Type field check on the receive frame. 0 – perform Length/Type field check 1 – do not perform Length/Type field check
24	CL_DIS	Read/Write	0x0	**Control Frame Length Check Disable:** When this bit is 1, control frames larger than the minimum frame length can be accepted
23–16	Reserved	Read	0x0	**Reserved:** These bits are reserved for future use and always return zero.

Table 2-55: **TEMAC Receive Configuration Word1 (RCW1) Register Bit Definitions *(Cont'd)***

Bits	Name	Core Access	Reset Value	Description
15–0	PauseAddr	Read/Write	0xFFEE	**Pause Frame Ethernet MAC Address (47:32):** This address is used to match the destination address of any received flow control frames. It is also used as the source address for any transmitted flow control frames. This address is ordered so that the first byte transmitted/ received is the lowest position byte in the register. For example, a MAC address of AA-BB-CC-DD-EE-FF would be stored in the PauseAddr(47:0) as 0xFFEEDDCCBBAA.

Notes:

1. Extended VLAN function require that jumbo frames be enabled.
2. This bit enables basic VLAN operation that is native to the TEMAC core. The TEMAC core recognizes VLAN frames when the Type/Length field contains a VLAN TAG with a TPID value of 0x8100. No other TPID values are recognized. Extended VLAN mode described later allow programmable TPID values. This bit must be 0 (disabled) when using extended VLAN mode.

TEMAC Transmit Configuration Register — Offset 0x0000_0408

The TEMAC Transmit Configuration (TC) register is shown in Figure 2-49. This register can be written at any time but the transmitter logic only applies the configuration changes during Inter-Frame gaps. The exception to this is the Reset bit which is effective immediately.

Figure 2-49: **TEMAC Transmit Configuration Register (offset 0x408)**

Table 2-56 shows the TEMAC Transmit Configuration register bit definitions.

Table 2-56: **TEMAC Transmit Configuration Register Bit Definitions**

Bits	Name	Core Access	Reset Value	Description
31	RST	Read/Write	0	**Reset:** When this bit is 1, the transmitter is reset. The bit automatically resets to 0. The reset also sets all of the transmitter configuration registers to their default values. Resetting the transmitter without resetting AXI Ethernet core could cause the core to be in an unknown state. 0 – no reset 1 – initiate a transmitter reset
30	JUM(1)	Read/Write	1	**Jumbo Frame Enable:** When this bit is 1 the transmitter sends frames over the maximum length specified in IEEE Std 802.3-2002 specification. 0 – send jumbo frames disabled 1 – send jumbo frames enabled
29	FCS	Read/Write	0	**In-Band FCS Enable:** When this bit is 1, the transmitter accepts the FCS field with the rest of the frame data. When this bit is 0 the FCS field is calculated and supplied by the transmitter. In either case the FCS field is verified. 0 – transmitter calculates and sends FCS field 1 – FCS field is provided with transmit frame data
28	TX	Read/Write	0	**Transmit Enable:** When this bit is 1, the transmit logic is enabled to operate. 0 – transmit disabled 1 – transmit enabled
27	VLAN(2)	Read/Write	1	**VLAN Frame Enable:** When this bit is 1, the transmitter allows transmission of VLAN tagged frames. 0 – transmit of VLAN frames disabled 1 – transmit of VLAN frames enabled
26	HD	Read/Write	0	**Half-Duplex Mode:** When this bit is 1, the transmitter operates in half-duplex mode. When this bit is 0, the transmitter operates in full-duplex mode. Only full-duplex is supported so this bit should always be set to 0. 0 – full-duplex transmit 1 – half-duplex transmit
25	IFG	Read/Write	1	**Inter Frame Gap Adjustment Enable:** When this bit is 1, the transmitter uses the value of the IFGP register (Figure 2-35) to extend the transmit Inter Frame Gap beyond the minimum of 12 idle cycles (96-bit times on the Ethernet Interface). 0 – no IFGP adjustment enabled 1 – IFGP adjusted based on IFGP register
24– 0	Reserved	Read	0x0	**Reserved:** These bits are reserved for future use and always return zero.

Notes:

1. Extended VLAN function require that jumbo frames be enabled.
2. This bit enables basic VLAN operation that is native to the TEMAC core. The TEMAC core recognizes VLAN frames when the Type/Length field contains a VLAN TAG with a TPID value of 0x8100. No other TPID values are recognized. Extended VLAN mode described later allow programmable TPID values. This bit must be 0 (disabled) when using extended VLAN mode.

TEMAC Flow Control Configuration Register — Offset 0x0000_040C

The TEMAC Flow Control Configuration (FCC) register is shown in Figure 2-50. This register can be written at any time but the flow control logic only applies the configuration changes during Inter-Frame gaps.

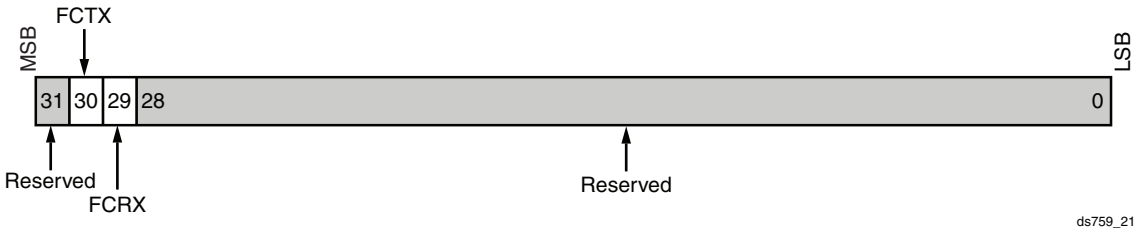


Figure 2-50: **TEMAC Flow Control Configuration Register (offset 0x40C)**

Table 2-57 shows the TEMAC Flow Control Configuration register bit definitions.

Table 2-57: **TEMAC Flow Control Configuration Register Bit Definitions**

Bits	Name	Core Access	Reset Value	Description
31	Reserved	Read	0	**Reserved:** These bits are reserved for future use and always return zero.
30	FCTX	Read/Write	1	**Transmit Flow Control Enable:** When this bit is 1, the transmitter sends a flow control frame when a value is written to the Transmit Pause Frame Register — Offset 0x0000_0004. 0 – transmit flow control frame disabled 1 – transmit flow control frame enabled
29	FCRX	Read/Write	1	**Receive Flow Control Enable:** When this bit is 1, the receive flow control frames inhibit transmitter operation. When this bit is 0, the flow control frames are passed through with other receive frames. 0 – receive flow control disabled 1 – receive flow control enabled
28–0	Reserved	Read	0x0	**Reserved:** These bits are reserved for future use and always return zero.

TEMAC Ethernet MAC Mode Configuration Register — Offset 0x0000_0410

The TEMAC Ethernet MAC Mode Configuration (EMMC) register is shown in Figure 2-51. This register can be written at any time but the Ethernet interface only applies the configuration changes during Inter Frame gaps. This register is slightly different for implementations using the TEMAC (C_TYPE = 0 or C_TYPE = 1).

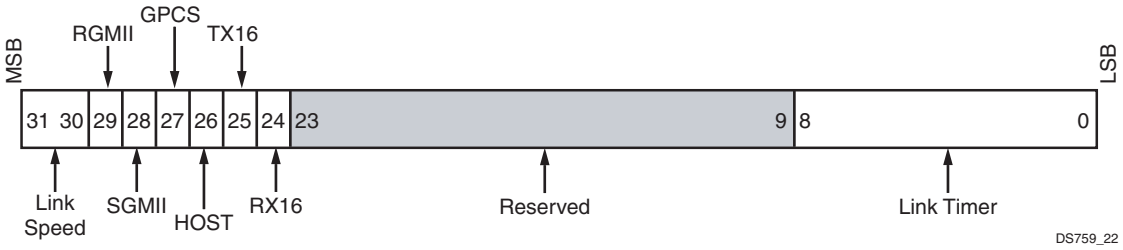


Figure 2-51: **TEMAC Ethernet MAC Mode Configuration Register (offset 0x410)**

Table 2-58 shows the TEMAC Ethernet MAC Mode Configuration register bit definitions.

Table 2-58: **TEMAC Ethernet MAC Mode Configuration Register Bit Definitions**

Bits	Name	Core Access	Reset Value	Description
31–30	Link Speed	Read/Write(1)(2)	10 / 01(3)	**Link Speed Selection:** The speed of the Ethernet interface is defined by the following values. 10 – 1000 Mb/s 01 – 100 Mb/s 00 – 10 Mb/s 11 – N/A
29	RGMII	Read/Write(1)(2)	0	**RGMII Mode Enable:** When this bit is 1, the Ethernet interface is configured in RGMII mode. 0 – not configured in RGMII mode 1 – configured in RGMII mode
28	SGMII	Read/Write(1)(2)	0	**SGMII Mode Enable:** When this bit is 1, the Ethernet interface is configured in SGMII mode. 0 – not configured in SGMII mode 1 – configured in SGMII mode
27	GPCS	Read Read/Write(1)(2)	0	**1000BASE-X Mode Enable:** When this bit is 1, the Ethernet interface is configured in 1000BASE-X mode. 0 – not configured in 1000BASE-X mode 1 – configured in 1000BASE-X mode
26	HOST	Read Read/Write(1)(2)	1 / 0(4)	**Host Interface Enable:** When this bit is 1, the host interface is enabled. 0 – host interface disabled 1 – host interface is enabled
25	TX16	Read Read/Write(1)(2)	0	**Transmit 16-bit Data Interface Enable:** When this bit is 1 and 1000BASE-X is being used, the transmit data interface is 16 bits wide. When this bit is 0, the transmit data interface is 8-bits wide. The 16-bit interface is not supported so this bit should always return 0. 0 – 8-bit transmit data interface 1 – 16-bit transmit data interface

Table 2-58: **TEMAC Ethernet MAC Mode Configuration Register Bit Definitions *(Cont'd)***

Bits	Name	Core Access	Reset Value	Description
24	RX16	Read Read/Write(1)(2)	0	**Receive 16-bit Data Interface Enable:** When this bit is 1 and 1000BASE-X is being used, the receive data interface is 16 bits wide. When this bit is 0, the receive data interface is 8-bits wide. The 16-bit interface is not supported so this bit should always return 0. 0 – 8-bit receive data interface 1 – 16-bit receive data interface
23–9	Reserved	Read	0x0	**Reserved:** These bits are reserved for future use and always return zero.
8–0	Link Timer	Read/Write(1)(2)		**Link Timer:** Sets the programmable link timer value, for operation with 1000BASE-X or SGMII modes

Notes:

1. Only bits 31–30 are Read/Write accessible in the TEMAC configuration
2. Only bits 31–30 are used with TEMAC. All other bits are reserved.
3. The Reset Value for LINK SPEED is "10" or 1000 Mb/s for all PHY interfaces except for MII which is not capable of that speed. The Reset Value for LINK SPEED for the MII interface is "01" or 100 Mb/s.
4. The use of the Host interface is hidden from you and is of no concern. However, this register returns a different reset value for different TEMAC implementations. The TEMAC implementation returns a 0.

Receive Max Frame Configuration Register — Offset 0x00000414

The Receive Max Frame Configuration (RXFC) register is shown in Figure 2-52. This register applies only to the TEMAC.

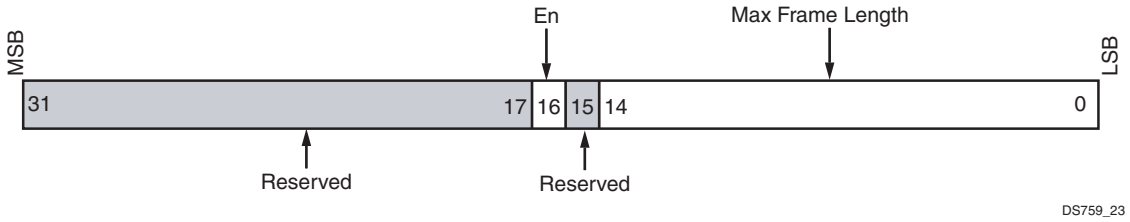


Figure 2-52: **Receive Max Frame Configuration Register (offset 0x00000414)**

Table 2-59 shows the Receive Max Frame Configuration register bit definitions. This register applies only to the TEMAC.

Table 2-59: **Receive Max Frame Configuration Register Bit Definitions**

Bits	Name	Core Access	Reset Value	Description
31–17	Reserved	Read	0	**Reserved.** These bits are reserved for future use and always return zero.
16	Enable	Read/Write	0	**RX Max Frame Enable.** When Low the MAC assumes use of the standard 1518/1522 depending upon the setting of VLAN Frame Enable in Table 2-55. When High the MAC allows frames up to RX Max Frame Length irrespective of the value of VLAN Frame Enable. If Jumbo Frame Enable is set in Table 2-55 then this register has no effect.
15	Reserved	Read	0	**Reserved.** These bits are reserved for future use and always return zero.
14–0	Max Frame Length	Read/Write	0x7D0	**RX Max Frame Length:** Set to preferred MAX frame length.

Transmit Max Frame Configuration Register — Offset 0x00000418

The Transmit Max Frame Configuration (TXFC) register is shown in is shown in Figure 2-53. This register applies only to the TEMAC.

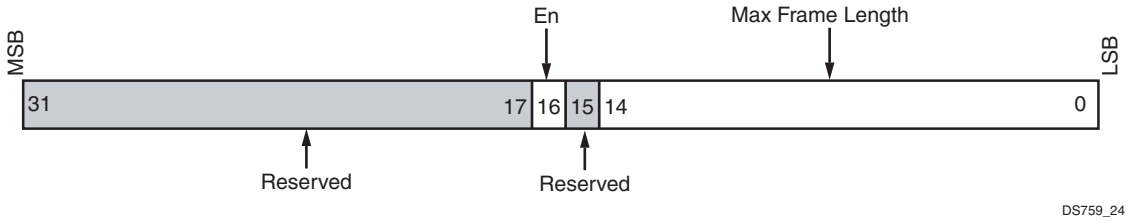


Figure 2-53: **Transmit Max Frame Configuration Register (offset 0x00000418)**

Table 2-60 shows the Transmit Max Frame Configuration register bit definitions. This register applies only to the TEMAC.

Table 2-60: **Transmit Max Frame Configuration Register Bit Definitions**

Bits	Name	Core Access	Reset Value	Description
31–17	Reserved	Read	0	**Reserved.** These bits are reserved for future use and always return zero.
16	Enable	Read/Write	0	**TX Max Frame Enable.** When Low the MAC assumes use of the standard 1518/1522 depending upon the setting of VLAN Frame Enable in Table 2-56. When High the MAC allows frames up to TX Max Frame Length irrespective of the value of VLAN Frame Enable. If Jumbo Frame Enable is set in Table 2-56 then this register has no effect.
15	Reserved	Read	0	**Reserved.** These bits are reserved for future use and always return zero.
14–0	Max Frame Length	Read/Write	0x7D0	**TX Max Frame Length:** Set to preferred MAX frame length.

Identification Register — Offset 0x000004F8

The Identification (ID) register is shown in Figure 2-54. The TEMAC return different values.

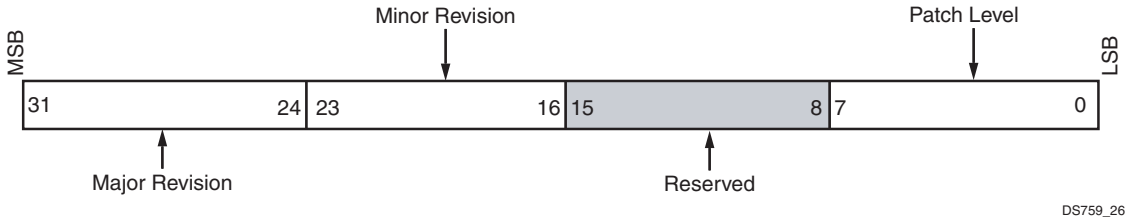


Figure 2-54: **Identification Register (offset 0x000004F8)**

Table 2-61 shows the Identification register bit definitions for the TEMAC.

Table 2-61: **Identification Register Bit Definitions**

Bits	Name	Core Access	Reset Value	Description
31–24	Major Revision	Read	0x05	**Major Revision.** These bits indicate the major revision of the TEMAC.
23–16	Minor Revision	Read	0x01	**Minor Revision.** These bits indicate the minor revision of the TEMAC.
15–8	Reserved	Read	0xFF	**Reserved.** These bits are reserved for future use and always return 0xFF.
7–0	Patch Level	Read	0x00	**Patch Level.** These bits indicate if a patch has been implemented. 0x00 = No Patch 0x01 = Rev 1 0x02 = Rev 2 ... 0xFF = Rev 255

Ability Register — Offset 0x000004FC

The Ability (AR) register is shown in Figure 2-55.

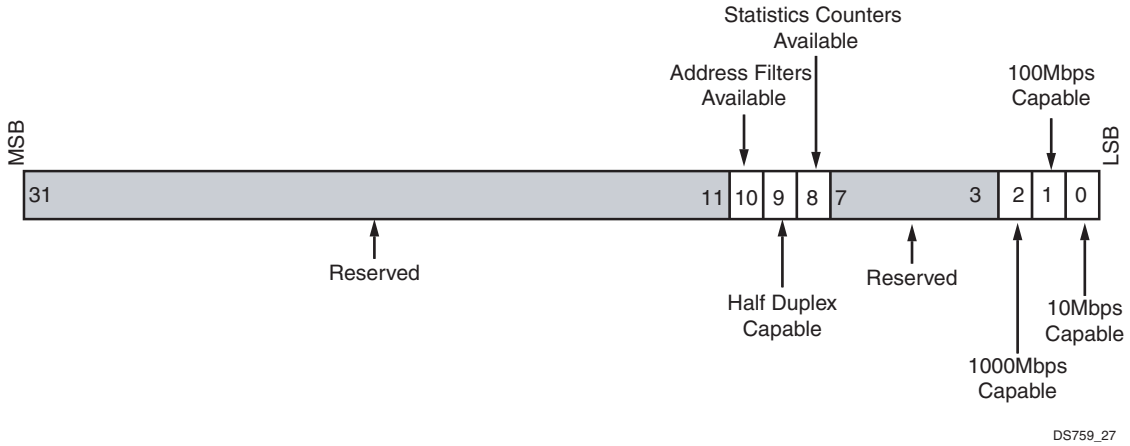


Figure 2-55: **Ability Register (offset 0x000004FC)**

Table 2-62 shows the Ability register bit definitions.

Table 2-62: **Ability Register Bit Definitions**

Bits	Name	Core Access	Reset Value	Description
31–11	Reserved	Read	0	**Reserved.** These bits are reserved for future definition.
10	Address Filters Available	Read	0x1	**Address Filters Available**.
9	Reserved	Read		Reserved.
8	Statistics Counters Available	Read	0x1(2)	Statistics Counters Available.
7–3	Reserved	Read	0	**Reserved.** These bits are reserved for future definition.
2	1000 Mb/s Capable	Read	0x1(2)	1000 Mb/s Ability
1	100 Mb/s Capable	Read	0x1(2)	100 Mb/s Ability
0	10 Mb/s Capable	Read	0x1(2)	10 Mb/s Ability

Notes:

1. This bit returns '1' for TEMAC, Half Duplex is not supported with the AXI Ethernet core.
2. Depends of parameter selection at build time.

MII Management Configuration Register — Offset 0x0000_0500

The MII Management (MDIO) Configuration (MC) register is shown in Figure 2-56. This register provides control for the TEMAC PHY MII management (MDIO) interface. The MDIO interface supplies a clock to the external device, MDC. This clock is derived from the `hostclk` input signal using the value in the Clock Divide[5:0]. The frequency of the MDIO clock is given by the following equation:

$$f_{MDC} = \frac{f_{HOSTCLK}}{(1 + \text{Clock Divide}[5:0]) \times 2}$$

To comply with the IEEE 802.3-2002 specification for this interface, the frequency of the MDC should not exceed 2.5 MHz. To prevent MDC from being out of specification, the Clock Divide[5:0] value powers up at 000000. While this value is in the register, it is impossible to enable the MDIO interface. Even if the MDIO interface is enabled by setting bit 6 of this register, the MDIO port is still disabled until a non-zero value has been written into the clock divide field.

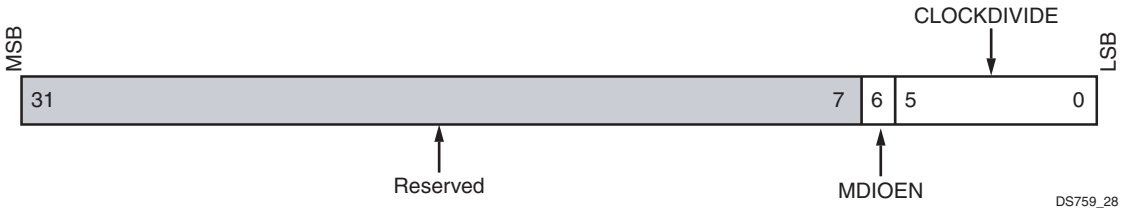


Figure 2-56: **MII Management (MDIO) Configuration Register (offset 0x500)**

Table 2-63 shows the MII Management Configuration register bit definitions.

Table 2-63: **MDIO MC Register Bit Definitions**

Bits	Name	Core Access	Reset Value	Description
31–7	Reserved	Read	0x0	**Reserved:** These bits are reserved for future use and always return zero.
6	MDIOEN	Read/Write	0	**MDIO Enable:** When this bit is 1, the MDIO (MII Management) interface is used to access the PHY. 0 – MDIO disabled 1 – MDIO enabled
5–0	CLOCK DIVIDE	Read/Write	0x0	**Clock Divide:** This value is used to derive the MDC (MII Management interface clock) signal. The maximum permitted frequency is 2.5 MHz.

MII Management Control Register — Offset 0x0000_0504

The MII Management (MDIO) Control register (MCR) is shown in Figure 2-57. This register can be written at any time but the Ethernet interface only applies the configuration changes during Inter Frame gaps. This register is slightly different for implementations using the TEMAC (C_TYPE = 0 or C_TYPE = 1).

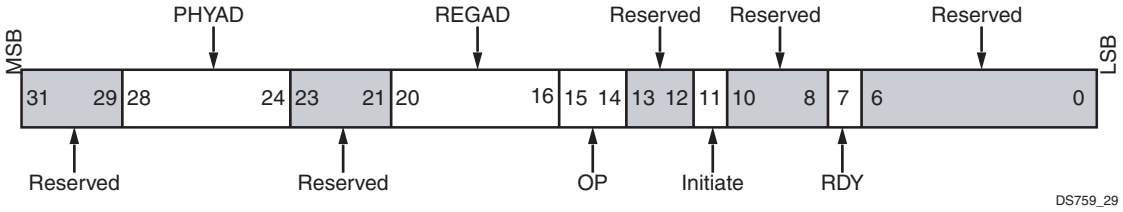


Figure 2-57: **MII Management (MDIO) Control Register (offset 0x504)**

Table 2-64 shows the MII Management Control register bit definitions.

Table 2-64: **MDIO MCR Bit Definitions**

Bits	Name	Core Access	Reset Value	Description
31–29	Reserved	Reserved	000	Reserved
28–24	PHYAD	Read/Write	00000	**PHY Physical Address:** The physical address of the PHY.
23–21	Reserved	Reserved	000	Reserved
20–16	REGAD	Read/Write(1)	00000	**PHY Register Address:** These bits represent the register to be accessed in a particular PHY, as indicated in the PHYAD field.
15–14	OP	Read/Write	00	**Operation Code.** These bits determine if a read or write is going to be performed. 01 – Write 10 – Read
13–12	Reserved	Read/Write	00	Reserved
11	Initiate	Read/Write(2)	0	**Initiate:** This bit must be set to 1 to initiate a MDIO transaction. 0 = Do not start an MDIO transaction 1 = Initiate an MDIO transaction
10–8	Reserved	Read/Write	000	Reserved
7	RDY	Read	1	Ready: This bit indicates if the MII Management interface is ready to accept a new transaction. 0 = Cannot accept a new transaction 1 = Ready to accept a new transaction
6–0	Reserved	Reserved	0000000	Reserved

Notes:

1. The first 16 registers (0–15) are defined by IEEE Std 802.3-2005. The remaining 16 registers (16–31) are reserved for PHY vendors' own register definition.
2. This bit clears upon a write.

MII Management Write Data Register — Offset 0x0000_0508

The MII Management (MDIO) Write Data register (MWD) is shown in Figure 2-58. This register is a temporary storage location for data to be written to a PHY register (internal or external) through the MDIO interface. A MDIO write is initiated by writing to the MII Management Control register with the physical PHY address (PHYAD), the PHY register to be accessed (REGAD), the access type, and setting the Initiate bit after providing the data to this register.

This register is only used for writing to PHY registers. When reading from PHY registers, the data is stored in the MII Management Read Data register. For more information on using the MDIO interface for accessing PHY registers, see Using the Address Filters.

Figure 2-58: **MII Management (MDIO) Write Data Register (offset 0x508)**

Table 2-65 shows the MII Management Write Data register bit definitions.

Table 2-65: **MWD Register Bit Definitions**

Bits	Name	Core Access	Reset Value	Description
31–16	Reserved	Read	0x0	**Reserved:** These bits are reserved for future use and always return zero.
15–0	Write Data	Read/Write	0x0	**MII Management Write Data:** This field temporarily holds data to be written to a PHY register.

MII Management Read Data Register — Offset 0x0000_050C

The MII Management (MDIO) Read Data register (MRD) is shown in Figure 2-59. This register is a temporary storage location for data to be read from a PHY register (internal or external) through the MDIO interface. A MDIO read is initiated by writing to the MII Management Control register with the physical PHY address (PHYAD), the PHY register to be accessed (REGAD), the access type, and setting the Initiate bit. This register is only used for temporarily storing the data read from the PHY registers. For more information on using the MDIO interface for accessing PHY registers, see Using the Address Filters.

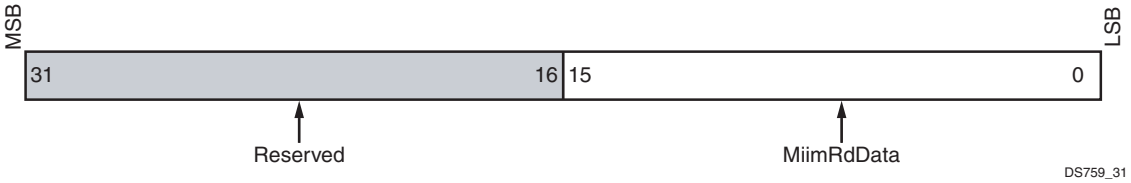


Figure 2-59: **MII Management (MDIO) Read Data Register (offset 0x50C)**

Table 2-58 shows the MDIO Ethernet MAC Mode Configuration register bit definitions.

Table 2-66: **MRD Register Bit Definitions**

Bits	Name	Core Access	Reset Value	Description
31–16	Reserved	Read/Write	0x0	**Reserved:** These bits are reserved for future use and always return zero.
15–0	Read Data	Read/Write	0x0	**MII Management Read Data:** This field temporarily holds data to be read from a PHY register.

MDIO Interrupt Status Register — Offset 0x00000600

The MDIO Interrupt Status (MIS) register is shown in Figure 2-60.

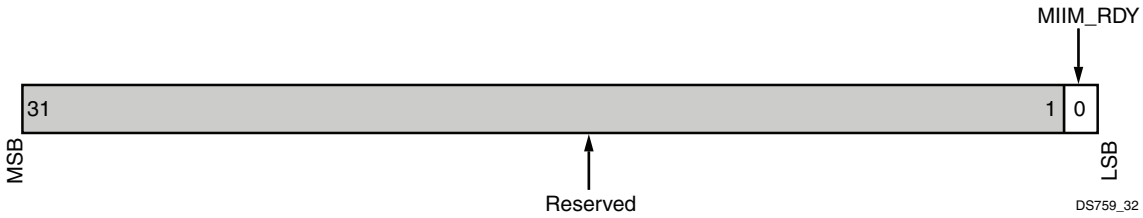


Figure 2-60: **MDIO Interrupt Status Register — Offset 0x00000600**

Table 2-67 shows the MDIO Interrupt Status register bit definitions.

Table 2-67: **MIS Register Bit Definitions**

Bits	Name	Core Access	Reset Value	Description
31–1	Reserved	Read	0	Reserved
0	MIIM_RDY	Read/Write	0	**MII Management Interrupt Status:** This bit is set by either the MIIM interface or by writing a '1' to it. In either case, it indicates an interrupt is pending. The interrupt is cleared by writing a '0' to this bit. 0 – no interrupt pending/clear interrupt 1 – interrupt pending/set interrupt

MDIO Interrupt Pending Register — Offset 0x00000620

The MDIO Interrupt Enable (MIP) register is shown in Figure 2-61. See Figure 2-41 for the structure of the interrupt register.

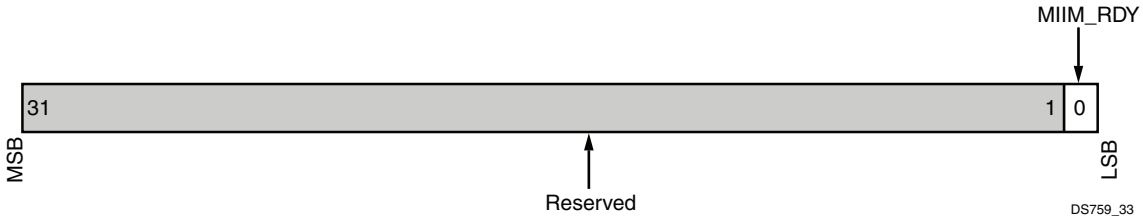


Figure 2-61: **MDIO Interrupt Pending Register — Offset 0x00000620**

Table 2-68 shows the MDIO Interrupt Pending register bit definitions.

Table 2-68: **MIP Register Bit Definitions**

Bits	Name	Core Access	Reset Value	Description
31–1	Reserved	Read	0	Reserved
0	MIIM_RDY	Read	0	**MII Management Interrupt Pending:** This bit indicates an interrupt is pending if the corresponding bit in the MIS and MIE are set. 0 – no interrupt pending 1 – interrupt pending

MDIO Interrupt Enable Register — Offset 0x00000640

The MDIO Interrupt Enable (MIE) register is shown in Figure 2-62. See Figure 2-37 for the structure of the interrupt register.

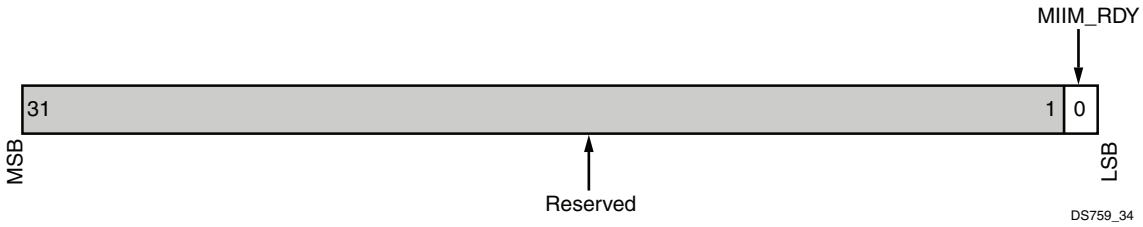


Figure 2-62: **MDIO Interrupt Enable Register — Offset 0x00000640**

Table 2-69 shows the MDIO Interrupt Enable register bit definitions.

Table 2-69: **MIE Register Bit Definitions**

Bits	Name	Core Access	Reset Value	Description
31–1	Reserved	Read	0	Reserved
0	MIIM_RDY	Read/Write	0	**MII Management Interrupt Enable:** When set, this bit allows an interrupt in the MIS register to generate an interrupt in the MIP register. 0 – Disable interrupt 1 – Enable interrupt

MDIO Interrupt Clear Register — Offset 0x00000660

The MDIO Interrupt Clear (MIC) register is shown in Figure 2-63. See Figure 2-37 for the structure of the interrupt register.

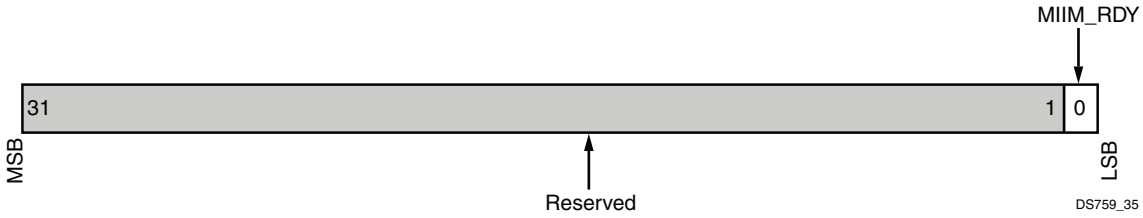


Figure 2-63: **MDIO Interrupt Clear Register — Offset 0x00000660**

Table 2-70 shows the MDIO Interrupt Clear register bit definitions.

Table 2-70: **MIC Register Bit Definitions**

Bits	Name	Core Access	Reset Value	Description
31–1	Reserved	Read	0	Reserved
0	MIIM_RDY	Read/Write	0	**MII Management Clear Enable:** Writing a '1' to this bit clears the corresponding interrupt in the MIS register. This bit is self clearing. 0 – Do not clear Interrupt 1 – Clear Interrupt

TEMAC Unicast Address Word 0 Register — Offset 0x00000700

The TEMAC Unicast Address Word 0 (UAW0) register is shown in Figure 2-64.

The Unicast Addresses register combine to provide a 48 bit ethernet station address. Word 0 provides the low order 32 bits of the address while word 1 provides the high order 16 bits.

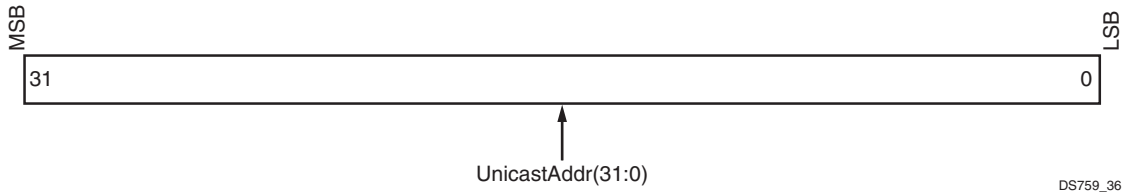


Figure 2-64: **TEMAC Unicast Address Word 0 Register (offset 0x00000700)**

Table 2-71 shows the TEMAC Unicast Address Word 0 register bit definitions.

Table 2-71: **TEMAC UAW0 Register Bit Definitions**

Bits	Name	Core Access	Reset Value	Description
31–0	UnicastAddr	Read/Write	0xFFFFFFFF	**Unicast Address (31:0):** This address is used to match against the destination address of any received frames. The address is ordered so the first byte transmitted/received is the lowest positioned byte in the register; for example, a MAC address of AA-BB-CC-DD-EE-FF would be stored in UnicastAddr(47:0) as 0xFFEEDDCCBBAA.

TEMAC Unicast Address Word 1 register — Offset 0x00000704

The TEMAC Unicast Address Word 1 (UAW1) register is shown in Figure 2-65.

The Unicast Addresses register combine to provide a 48 bit Ethernet station address. Word 0 provides the low order 32 bits of the address while word 1 provides the high order 16 bits.

Figure 2-65: **TEMAC Unicast Address Word 1 Register (offset 0x00000704)**

Table 2-72 shows the TEMAC Unicast Address Word 1 register bit definitions.

Table 2-72: **TEMAC UAW1 Register Bit Definitions**

Bits	Name	Core Access	Reset Value	Description
31–16	Reserved	Read	0x0	**Reserved:** These bits are reserved for future use and always return zero.
15–0	UnicastAddr	Read/Write	0x0000FFFF	**Unicast Address (47:32):** This address is used to match against the destination address of any received frames. The address is ordered so the first byte transmitted/received is the lowest positioned byte in the register; for example, a MAC address of AA-BB-CC-DD-EE-FF would be stored in UnicastAddr(47:0) as 0xFFEEDDCCBBAA.

Filter Mask Index Register — Offset 0x00000708

The TEMAC Address Filter Mode (FMI) register is shown in Figure 2-66.

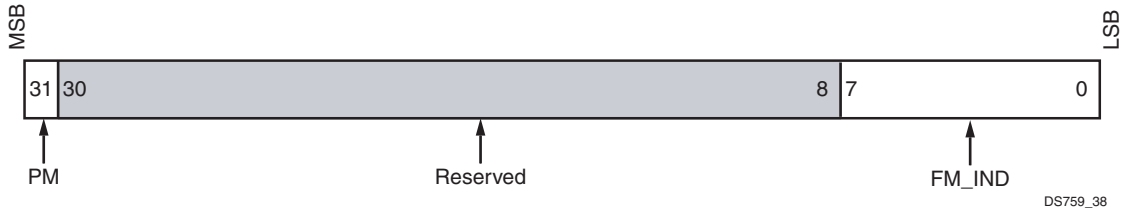


Figure 2-66: **Filter Mask Index Register (offset 0x00000708)**

Table 2-73 shows the TEMAC Address Filter Mask register bit definitions.

Table 2-73: **FMI Register Bit Definitions**

Bits	Name	Core Access	Reset Value	Description
31	PM(1)	Read/Write	1	**Promiscuous Receive Address Mode Enable:** When this bit is 1, the receive address filtering is disabled and all destination addresses are accepted. 0 – address filtering enabled 1 – address filtering disabled (all addresses accepted)
30–8	Reserved	Read	0x0	**Reserved:** These bits are reserved for future definition and always return zero.
7–0	FM_IND	Read/Write	0x00	**Filter Mask Index:** Provides the address index for the filter table. When set, it does not need to be changed until another address filter entry needs accessed. 0x00 = Filter 0 0x01 = Filter 1 0x02 = Filter 2 0x03 = Filter 3

Notes:

1. Extended Multicast Filtering requires that the promiscuous mode be enabled/ address filtering is disabled.

Address Filter Register 0 (31:0) — Offset 0x00000710

This register can be used to filter any address type, not just multicast addresses. Before accessing this register, set the Filter Mask Index (FM_IND) to the appropriate setting. See Using the Address Filters for more information.

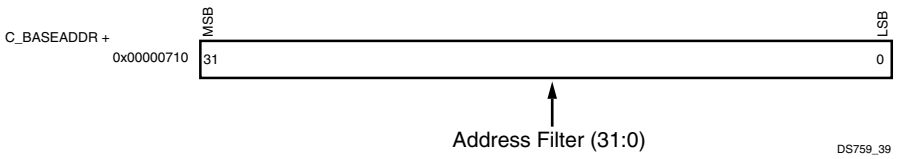


Figure 2-67: **Address Filter Register 0**

Table 2-74 shows the Address Filter 0 (AF0) register bit definitions.

Table 2-74: **AF0 Bit Definitions**

Bits	Name	Core Access	Reset Value	Description
31–0	Address Filter (31:0)	Read/Write	0xFFFFFFFF	**Address Filter (31:0):** This address is used to match against the destination address of any received frames. The address is ordered so the first byte transmitted/received is the lowest positioned byte in the register; for example, a MAC address of AA-BB-CC-DD-EE-FF would be stored in Addr(47:0) as 0xFFEEDDCCBBAA.

Address Filter Register 1 (47:32) — Offset 0x00000714

This register can be used to filter any address type, not just multicast addresses. Before accessing this register, set the Filter Mask Index (FM_IND) to the appropriate setting. See Using the Address Filters for more information.

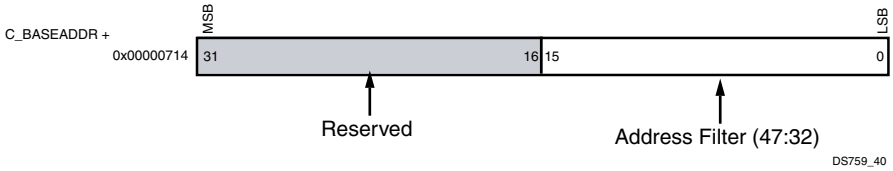


Figure 2-68: **Address Filter Register 1**

Table 2-75 shows the Address Filter 1 (AF1) register bit definitions.

Table 2-75: **AF1 Register Bit Definitions**

Bits	Name	Core Access	Reset Value	Description
31–16	Reserved	Read	0x0000	**Reserved:** These bits are reserved for future definition and always return zero.
15–0	Address Filter (47:32)	Read/Write	0xFFFF	**Address Filter(47:32):** This address is used to match against the destination address of any received frames. The address is ordered so the first byte transmitted/received is the lowest positioned byte in the register; for example, a MAC address of AA-BB-CC-DD-EE-FF would be stored in Addr(47:0) as 0xFFEEDDCCBBAA.

To update an address filter, first select the address filter to be accessed (bottom bits of the Filter Mask Index register (the MSB of this register provides the Promiscuous mode control), then access the associated register.

Transmit VLAN Data Table — Offset 0x0000_4000-0x0000_7FFF

This table is used for data to support transmit VLAN tagging, VLAN stripping, and VLAN translation. The table is always 4K entries deep but the width depends on how many of the VLAN functions are included at build time. VLAN translation requires 12 bits at each location while VLAN stripping and VLAN tagging require 1 bit each at each location. When all transmit VLAN functions are included, the table is 14 bits wide. If VLAN functions are not included, the bits for those functions are not present and writes to those bits have no effect while reads return zero.

IMPORTANT: *The table can be either 1-bit, 2-bits, 12-bits, 13-bits, or 14-bits wide depending on which features are present. The table must be initialized by software through the AXI4-Lite and is addressed on 32-bit word boundaries.*

The transmit VLAN Table entry with all VLAN functions present is shown in Figure 2-69 while Figure 2-70 shows the transmit VLAN Table entry with only the translation field. The bit locations for the functions do not change even when some functions are not used in the build. See Extended VLAN Support for more details.

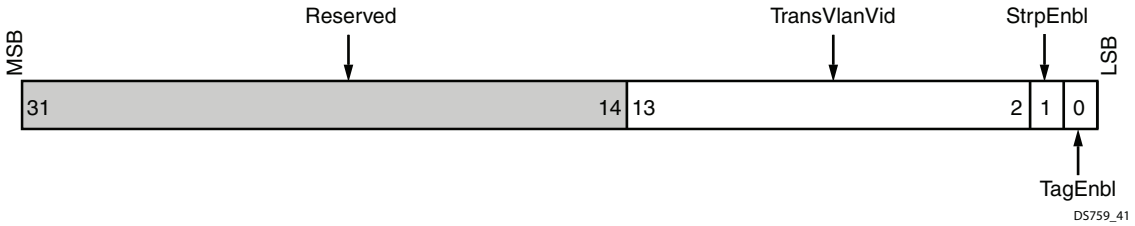


Figure 2-69: **Transmit VLAN Table Entry with all Fields (offset 0x0000_4000-0x0000_7FFF)**

Figure 2-70: **Transmit VLAN Table Entry with One Field (offset 0x0000_4000-0x0000_7FFF)**

Receive VLAN Data Table — Offset 0x0000_8000-0x0000_BFFF

This table is used for data to support receive VLAN tagging, VLAN stripping, and VLAN translation. The table is always 4K entries deep but the width depends on how many of the VLAN functions are included at build time. VLAN translation requires 12 bits at each location while VLAN stripping and VLAN tagging require 1 bit each at each location. When all receive VLAN functions are included, the table is 14 bits wide. If VLAN functions are not included, the bits for those functions are not present and writes to those bits have no effect while reads return zero.

IMPORTANT: *The table can be either 1-bit, 2-bits, 12-bits, 13-bits, or 14-bits wide depending on which features are present. The table must be initialized by software through the AXI4-Lite interface and is addressed on 32-bit word boundaries.*

The receive VLAN Table entry with all VLAN functions present is shown in Figure 2-71 while Figure 2-72 shows the receive VLAN table entry with only the translation field. The bit locations for the functions do not change even when some functions are not used in the build. See Extended VLAN Support for more details.

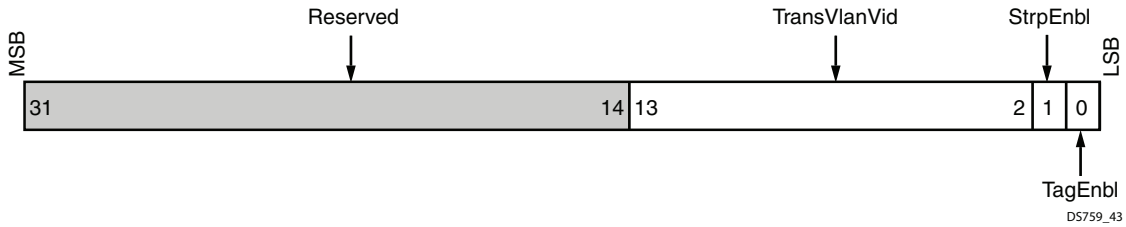


Figure 2-71: **Receive VLAN Table Entry with all Fields (offset 0x0000_8000-0x0000_BFFF)**

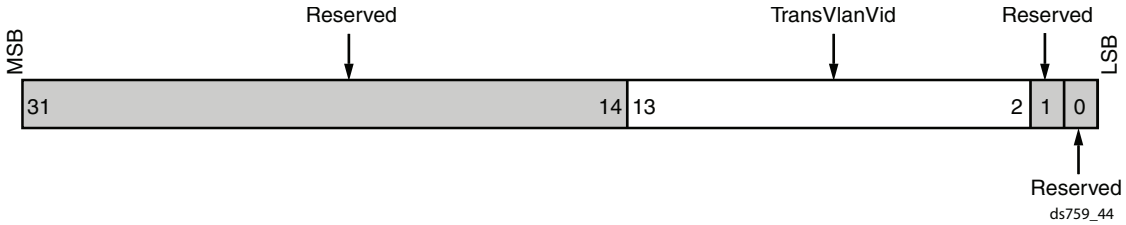


Figure 2-72: **Receive VLAN Table Entry with One Field (offset 0x0000_8000-0x0000_BFFF)**

AVB Addressing

Receive PTP Packet Buffer — Offset 0x00010000 – 0x00010FFF

The Receive PTP Packet Buffer is 4 Kb. See the version of the Tri-Mode Ethernet MAC core listed in the change log for more information.

Transmit PTP Packet Buffer — Offset 0x00011000 – 0x000117FF

The Transmit PTP Packet Buffer is divided into eight identical buffer sections with each section containing 256 bytes. See the version of the Tri-Mode Ethernet MAC core listed in the change log for more information.

AVB Tx/Rx Configuration — Offset 0x00012000 – 0x0001201B

See the version of the Tri-Mode Ethernet MAC core listed in the change log for more information.

AVB RTC Configuration — Offset 0x00012800 – 0x000128FF

See the version of the Tri-Mode Ethernet MAC core listed in the change log for more information.

Multicast Address Table — Offset 0x0002_0000-0x0003_FFFF

The Multicast Address Table entry is shown in Figure 2-74. The multicast address table is only present when extended multicast address filtering is selected at build-time (C_MCAST_EXTEND = 1). The purpose of the table is to allow the AXI Ethernet core to support reception of frames addressed to many multicast addresses while providing some of the filtering in hardware to offload some of the overhead required for filtering in software.

While a MAC multicast address is defined as any 48-bit MAC address that has bit 0 (LSB) set to 1 (for example 01:00:00:00:00:00), in most cases the MAC multicast address is created from a IP multicast address, as shown in Figure 2-73.

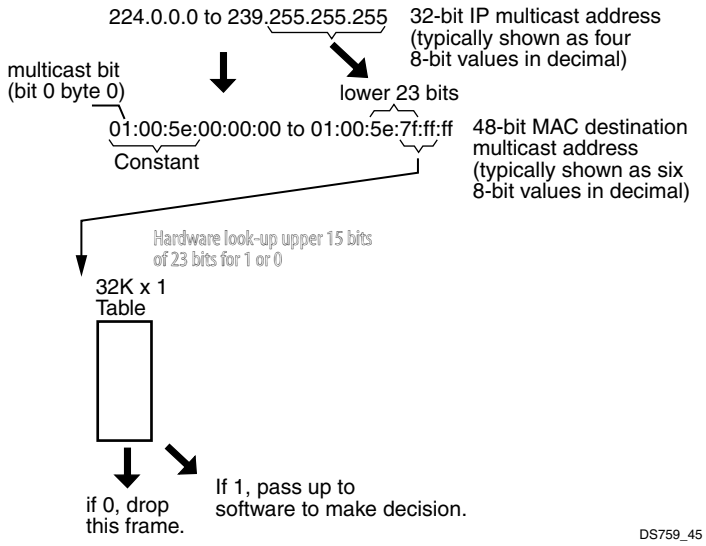


Figure 2-73: **Mapping IP Multicast Addresses to MAC Multicast Addresses**

When a multicast address frame is received while this extended multicast filtering is enabled, the AXI Ethernet core first verifies that the first 24 bits are 01:00:5E and then uses the upper 15 bits of the unique 23 bit MAC multicast address to index this memory. If the associated memory location contains a 1 then the frame is accepted and passed up to software for a comparison on the full 23-bit address. If the memory location is a 0 or the upper 24 bits are not 01:00:5E then the frame is not accepted and it is dropped.

The memory is 1-bit wide but is addressed on 32-bit word boundaries. The memory is 32K deep. This table must be initialized by software through the AXI4-Lite interface.

IMPORTANT: *When using the extended multicast address filtering, the TEMAC must be set to promiscuous mode so that all frames are available for filtering. When doing this the TEMAC no longer checks for a unicast address match. Additional registers (UAWL and UAWU) are available to provide unicast address filtering while in this mode.*

For builds that have the extended multicast address filtering enabled, promiscuous mode can be achieved by making sure that the TEMAC is in promiscuous mode and by clearing the EMultiFltrEnbl bit (bit 19) in the Reset and Address Filter register (RAF). See Extended Multicast Address Filtering Mode.

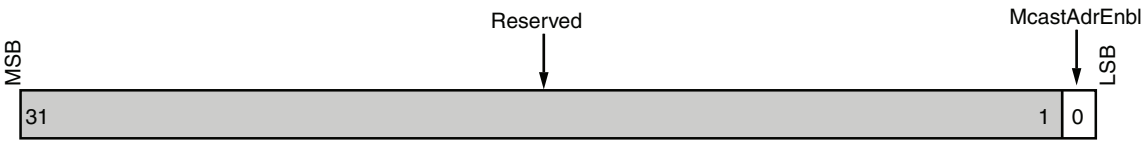


Figure 2-74: **Multicast Address Table Entry (offset 0x0002_0000-0x0003_FFFF)**

Table 2-76 shows the Multicast Address Table bit definitions.

Table 2-76: **Multicast Address Table Bit Definitions**

Bits	Name	Core Access	Reset Value	Description
31–1	Reserved	Read	0x0	**Reserved:** These bits are reserved for future use and always return zero.
0	McastAdrEnbl	Read/Write	0	**Multicast Address Enable:** This bit indicates that the received multicast frame with this upper 15 bits of the unique 23-bit MAC multicast address field should be accepted or rejected. 0 – Drop this frame 1 – Accept this frame

Designing with the Core

This chapter includes guidelines and additional information to facilitate designing with the core.

Design Guidelines

Chapter 1, Overview introduces the features and Chapter 2, Product Specification introduces the interfaces and registers space. This chapter provides design guidelines that need to be considered while creating the system using the AXI Ethernet core.

It is important to understand the features and interfaces provided by the core netlist, and as described in the following sections.

Customize and Generate the Core

Generate the core with your desired options using the IP catalog as described in Chapter 4, Customizing and Generating the Core.

Examine the Generated Design for this Hierarchical Block

- HDL is generated for this hierarchical block at locations as explained in Output Generation in Chapter 4.
- Examine this HDL for the I/O and interfaces generated. In many cases the required logic is generated automatically. Cross-check for the MDIO IOBUF instantiation as per the design requirement.
- Examine the Address Space that is allocated for AXI Ethernet in the IP Integrator **Address editor** tab. AXI Ethernet requires 256K Address Range.
- Examine the clocks required for the mode of operation and provide the clocks as required.
- If using a development board, check the board for the LOC constraints provided in the "Master Constraints" file delivered along with the board. Also check for the specific location constraints for transceivers.
- Synthesize and Implement the entire design.
- After implementation is complete you can also create a bitstream that can be downloaded to a Xilinx device.

Keep it Registered

To simplify timing and to increase system performance in an FPGA design, keep all inputs and outputs registered between the user application and the core. All inputs and outputs from the user application should come from, or connect to, a flip-flop. While registering signals might not be possible for all paths, it simplifies timing analysis and makes it easier for the Xilinx tools to place and route the design.

Recognize Timing Critical Signals

The constraints provided with the Infrastructure cores identify the critical signals and the timing constraints that should be applied. See Chapter 5, Constraining the Core. Also see the infrastructure core documentation for more information.

Make Only Allowed Modifications

The AXI Ethernet IP core should not be modified. Modifications can have adverse effects on system timing and protocol compliance. Supported user configurations of this IP can only be made by selecting the options from within the Vivado® design tools when the core is generated. See Vivado Integrated Design Environment in Chapter 4. Do not directly modify the infrastructure cores.

Clocking

When targeting a GMII design, a BUFGMUX is used to switch between the MII_TX_CLK and the GTX_CLK clocks. This allows the design to support data rates of 10/100 Mb/s and 1000 Mb/s. The FPGA pins for these clocks must be selected such that they are located in the same clock region and they are both on clock dedicated pins. The GMII status, control, and data pins must be chosen to be in the same clock region as these clocks. See the *7 Series FPGAs Clocking Resources User Guide* (UG472) [Ref 3] for the targeted FPGA family for more information.

IMPORTANT: *Pay special attention to clocking conflicts. Failure to adhere to these rules can cause build errors and data integrity errors.*

Resets

The TEMAC components are reset using the following AXI4 reset signals: `axi_str_txd_aresetn`, `axi_str_txc_aresetn`, `axi_str_rxd_aresetn`, `axi_str_rxs_aresetn`, or `s_axi_aresetn`. All resets must pass through reset detection circuits that detect and synchronize the resets to the different clock domains.

As a result, any time the AXI Ethernet core is reset, sufficient time must elapse for a reset to propagate through the reset circuits and logic. The amount of time required is dependent upon the slowest AXI Ethernet clock. Allow 30 clock cycles of the slowest AXI Ethernet clock, to elapse before accessing the core. Failure to comply causes unpredictable behavior.

In a system in which Ethernet operates at 10 Mb/s, the MAC interface operates at 2.5 MHz. If the AXI4-Lite interface operates 100 MHz and the AXI4-Stream interface operates at 125 MHz, the time that must elapse before AXI Ethernet is accessed is 12 us (400 ns * 30 clock cycles).

If the system uses DMA for data transfer on the streaming interfaces, it is advised to connect the reset outputs from the DMA to the AXI Ethernet streaming reset inputs.

RECOMMENDED: *As a general design guideline, Xilinx recommends asserting system* `aresetn` *signals for a minimum of 30 clock cycles (of the slowest* `aclk`*), as that is known to satisfy the preceding reset requirements.*

Design Parameters

To allow you to generate an AXI Ethernet core that is uniquely tailored to your system, These parameters can be configured through the Vivado® IDE. In addition to the parameters listed in Table 3-1, EDK tools infer a few other parameters. These parameters control the behavior of the software generated. All parameters inferred by the EDK tools are listed in Table 3-2.

Table 3-1: **Design Parameters**

Parameter	Default Value	Description
USE_BOARD_FLOW	FALSE	Generate Board based I/O Constraints
PHY_TYPE	GMII	Select the PHY Interface type
Enable_1588	FALSE	Enable 1588
Enable_1588_1step	TRUE	True: 1-step or false is to enable 2-step support
TIMER_CLK_PERIOD	4000	1588 System Timer reference clock period in picoseconds
SupportLevel	1	This selects if the Shared Logic is in Core or not. Default value is to include shared logic in core.
ENABLE_LVDS	FALSE	Enable standard I/O (LVDS) for SGMII instead of a transceiver
ENABLE_AVB	FALSE	Enable AVB
TXVLAN_TRAN	FALSE	Enable TX VLAN translation
RXVLAN_TRAN	FALSE	Enable Rx VLAN translation
TXVLAN_TAG	FALSE	Enable TX VLAN tagging
RXVLAN_TAG	FALSE	Enable Rx VLAN tagging
TXVLAN_STRP	FALSE	Enable TX VLAN stripping
RXVLAN_STRP	FALSE	Enable Rx VLAN stripping
MCAST_EXTEND	FALSE	Enable Rx extended multicast address filtering
Frame_Filter	FALSE	Frame Filter
Statistics_Reset	FALSE	Allow Statistics to be reset
Statistics_Counters	FALSE	Enable statistics counters
Number_of_Table_Entries	4	Number of Table Entries
PHYADDR	1	MDIO PHY Address, range 1 to 31
Statistics_Width	64 bit	Statistics Counter Width
RXMEM	4k	RX Memory Size
TXMEM	4k	TX Memory Size
TXCSUM	FALSE	TX Checksum Offload
RXCSUM	FALSE	RX Checksum Offload
SIMULATION_MODE	0	Enable Simulation Mode helps reducing the simulation time.
Include_IO	TRUE	Include the I/O elements. If this is TRUE, I/O elements are included. If this is FALSE, I/O elements are not included in the core. This is available only in GMII mode. When this is FALSE, TEMAC is configured in internal mode.

Table 3-2: **AXI Ethernet Core Design Parameters**

Feature/Description	Parameter Name	Allowable Values	Default Values	VHDL Type
User Specified TEMAC Implementation Parameters				
PHY Interface Type	C_PHY_TYPE	0 = MII 1 = GMII/MII 2 = Reserved 3 = RGMII v2.0 4 = SGMII 5 = 1000Base-X	1	integer
PHY Address for Gigabit Ethernet PCS PMA	C_PHYADDR(1)	00001 – 11111	00001	std_logic_vector
Transmit block RAM depth in bytes for TEMAC	C_TXMEM	4096, 8192, 16384, 32,768	4096	integer
Receive block RAM depth in bytes for TEMAC	C_RXMEM	4096, 8192, 16384, 32768	4096	integer
Transmit TCP/UDP Checksum offload	C_TXCSUM	0 = Tx CSUM unused 1 = Partial Tx CSUM used 2 = Full Tx CSUM used	0	integer
Receive TCP/UDP Checksum offload	C_RXCSUM	0 = Rx CSUM unused 1 = Partial Rx CSUM used 2 = Full Rx CSUM used	0	integer
Transmit VLAN tagging	C_TXVLAN_TAG	1 = Tx VLAN tagging used 0 = Tx VLAN tagging unused	0	integer
Receive VLAN tagging	C_RXVLAN_TAG	1 = Rx VLAN tagging used 0 = Rx VLAN tagging unused	0	integer
Transmit VLAN translation	C_TXVLAN_TRAN	1 = Tx VLAN translation used 0 = Tx VLAN translation unused	0	integer
Receive VLAN translation	C_RXVLAN_TRAN	1 = Rx VLAN translation used 0 = Rx VLAN translation unused	0	integer
Transmit VLAN stripping	C_TXVLAN_STRP	1 = Tx VLAN stripping used 0 = Tx VLAN stripping unused	0	integer
Receive VLAN stripping	C_RXVLAN_STRP	1 = Rx VLAN stripping used 0 = Rx VLAN stripping unused	0	integer
Extended Multicast address filtering for RX	C_MCAST_EXTEND	1 = Extended multicast filtering used 0 = Extended multicast filtering unused	0	integer
Statistics gathering	C_STATS	1 = Statistics gathering used 0 = Statistics gathering unused	0	integer
Statistics width	C_STATS_WIDTH	64 = 64-bit wide statistic vectors	64	integer
Ethernet Audio Video Bridging (AVB) mode	C_AVB	1 = Ethernet AVB mode used 0 = Ethernet AVB mode unused	0	integer

Table 3-2: **AXI Ethernet Core Design Parameters *(Cont'd)***

Feature/Description	Parameter Name	Allowable Values	Default Values	VHDL Type
Enable SGMII over LVDS	C_ENABLE_LVDS	1= Enable SGMII over LVDS 0= SGMII over LVDS is not enabled	0	integer

Notes:

1. The value "00000" is a broadcast PHY address and should not be used to avoid contention between the internal TEMAC PHYs and the external PHY(s)

Allowable Parameter Combinations

Support for many PHY interfaces is included and is selected with parameters at build time. The PHY interface supported does not vary based on the Ethernet MAC type selected. See Table 3-3 through Table 3-6 for the supported PHY interface with TEMAC.

Table 3-3: **Supported PHY Speeds Based on PHY Modes**

TEMAC			
PHY Interface	Full Duplex		
	10Mb/s	100Mb/s	1000Mb/s
MII	Yes	Yes	No
GMI	Yes	Yes	Yes
RGMII v2.0	Yes	Yes	Yes
SGMII	Yes	Yes	Yes
1000Base-X	No	No	Yes

Table 3-4: **Artix-7 FPGA PHY Support Based on I/O Voltage**

Artix®-7 FPGA									
Parallel PHY Interface				Serial PHY Interface		Miscellaneous PHY Signals[5]			
PHY Interface	Voltage Level Supported			PHY Interface	Interface Supported	Signal	Voltage Level Supported		
	3.3 V	2.5 V	1.8 V	SGMII	Yes		3.3 V	2.5 V	1.8 V

Table 3-4: **Artix-7 FPGA PHY Support Based on I/O Voltage *(Cont'd)***

Artix®-7 FPGA									
MII	Yes(1)	Yes(1)	Yes(2)	1000 Base-X	Yes	MDIO	Yes(1)	Yes(1)	Yes(2)
GMII	Yes(1)	Yes(1)	Yes(2)			MDC			
RGMII	No(3)	Yes(1)	Yes(4)			Reset			
						Interrupt			

Notes:

1. Requires the use of High Range (HR) I/O.
2. Because no PHY devices support MII, GMII/MII, and other miscellaneous PHY signals at 1.8 V, external voltage level shifting logic is required.
3. High Range (HR) I/O duty cycle distortion exceeds RGMII specification.
4. There are limited 1.8 V RGMII-only PHY devices available. If one of these devices is not used, external voltage level shifting logic is required.
5. The miscellaneous PHY signals include, but are not limited to the ones listed. Signal names can vary.

Table 3-5: **Virtex-7 FPGA PHY Support Based on I/O Voltage**

Virtex®-7 FPGA									
Parallel PHY Interface				**Serial PHY Interface**		**Miscellaneous PHY Signals(5)**			
PHY Interface	**Voltage Level Supported**			**PHY Interface**	**Interface Supported**	**Signal**	**Voltage Level Supported**		
	3.3 V	2.5 V	1.8 V	SGMII	Yes		3.3 V	2.5 V	1.8 V
MII	Yes(1)	Yes(1)	Yes(2)	1000 Base-X	Yes	MDIO	Yes(1)	Yes(1)	Yes(2)
GMII	Yes(1)	Yes(1)	Yes(2)			MDC			
RGMII	No(3)	No(3)	Yes(4)			Reset			
						Interrupt			

Notes:

1. Supported on the XC7V585T-FFG1761, XC7VX330T-FFG1761 devices when using the High Range (HR). Other devices do not contain HR I/O; therefore these voltages are not supported.
2. Because no PHY devices support MII, GMII/MII, and other miscellaneous PHY signals at 1.8 V, external voltage level shifting logic is required.
3. High Range (HR) I/O duty cycle distortion exceeds RGMII specification.
4. There are limited 1.8 V RGMII-only PHY devices available. If one of these devices is not used, external voltage level shifting logic is required.
5. The miscellaneous PHY signals include, but are not limited to the ones listed. Signal names can vary.

Table 3-6: **Kintex-7 FPGA PHY Support Based on I/O Voltage**

Kintex®-7 FPGA								
Parallel PHY Interface				Serial PHY Interface		Miscellaneous PHY Signals[5]		
PHY Interface	Voltage Level Supported			PHY Interface	Interface Supported	Signal	Voltage Level Supported	
	3.3 V	2.5 V	1.8 V	SGMII	Yes		3.3 V	2.5 V	1.8 V
MII	Yes[1]	Yes[1]	Yes[2]	1000 Base-X	Yes	MDIO	Yes[1]	Yes[1]	Yes[2]
GMII	Yes[1]	Yes[1]	Yes[2]			MDC			
RGMII	No[3]	Yes[1]	Yes[4]			Reset			
						Interrupt			

Notes:

1. Requires the use of High Range (HR) I/O.
2. Because no PHY devices support MII, GMII/MII, and other miscellaneous PHY signals at 1.8 V, external voltage level shifting logic is required.
3. High Range (HR) I/O duty cycle distortion exceeds RGMII specification.
4. There are limited 1.8 V RGMII-only PHY devices available. If one of these devices is not used, external voltage level shifting logic is required.
5. The miscellaneous PHY signals include, but are not limited to the ones listed. Signal names can vary.

Some of the optional functions provided by AXI Ethernet core are not compatible with other optional functions. Figure 3-1 shows which of these functions are compatible with each other. In Figure 3-1, Tx/Rx CSUM Offload refers to both Partial Checksum Offloading and Full Checksum Offloading.

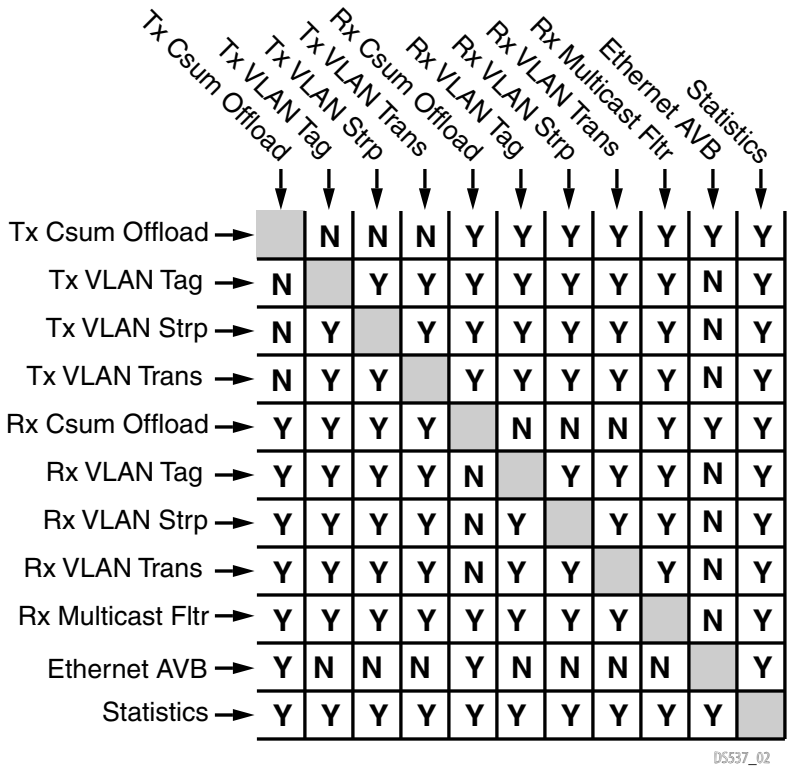


Figure 3-1: **Option Function Compatibility**

The AXI Ethernet core provides one Ethernet interface. Access to external PHY registers is provided using a standard MII Management bus. When using the SGMII or 1000 Base-X PHY interfaces, the AXI Ethernet core provides some PHY functionality and also includes PHY registers which are also accessible through the MII Management bus. These registers are described in Using the MII Management to Access Internal or External PHY Registers.

This core includes optional logic to calculate TCP/UDP checksums for transmit and verify TCP/UDP checksums for receive. Using this logic can significantly increase the maximum Ethernet bus data rate while reducing utilization of the processor for Ethernet tasks. Including the checksum offload function increases the amount of FPGA resources used for this core. The checksum information is included with each Ethernet frame passing over the AXI4-Stream interface. The checksum offload functionality cannot be used at the same time as the extended VLAN functionality.

The AXI Ethernet core provides memory buffering of transmit and receive Ethernet frames, thereby allowing more optimal transfer to and from the core with DMA. The number of frames that can be buffered in each direction is based on the size of each frame and the size of the memory buffer which are selected by parameters at build time. If the AXI Ethernet core transmit memory buffer becomes full, it throttles the transmit AXI4-Stream Data interface until more room is available for Ethernet frames. If the receive memory buffer becomes full, frames are dropped until more memory buffer room is available. Receive frames that do not meet Ethernet format rules or do not satisfy receive address qualification are always dropped.

Optional logic can be included to facilitate handling of VLAN type frames. Auto insertion, stripping, or translation of VLAN frames can be performed on transmit or receive with several options for choosing which frames are to be altered. Additional logic can be selected to provide additional filtering of receive frames with multicast destination addresses. The AXI Ethernet core provides native support for up to four (4) multicast addresses.

Logic can be selected to gather statistics on transmit and receive frames. This logic provides 64-bit counters for many statistics about the frames passing through the TEMAC core. Ethernet AVB support is available with an additional license and is supported at 100 Mb/s or 1000 Mb/s implementations.

Design Flow Steps

This chapter describes customizing and generating the core, constraining the core, and the simulation, synthesis and implementation steps that are specific to this IP core. More detailed information about the standard Vivado® design flows in the IP Integrator can be found in the following Vivado Design Suite user guides:

- *Vivado Design Suite User Guide: Designing IP Subsystems using IP Integrator* (UG994) [Ref 4]
- *Vivado Design Suite User Guide: Designing with IP* (UG896) [Ref 5]
- *Vivado Design Suite User Guide: Logic Simulation* (UG900) [Ref 6]
- *Vivado Design Suite User Guide: Getting Started* (UG910) [Ref 7]

Customizing and Generating the Core

This section includes information about using Xilinx tools to customize and generate the core in the Vivado Design Suite.

Vivado Integrated Design Environment

This core can be customized and generated in the core in the Vivado IP Integrator. For detailed information related to use of IP Integrator, see the *Vivado Design Suite User Guide: Designing IP Subsystems using IP Integrator* (UG994) [Ref 4]. You can customize the IP core for use in your design by specifying values for the various parameters associated with the IP core with the following steps:

1. Create a block design if not created already by selecting the option **create block design** in the **IP Integrator** tab.
2. Add AXI Ethernet to the block design.
3. Double-click the IP core to open up a configuration dialog box in Vivado IDE. The details of this configuration dialog box are provided in a later part of this section.
4. You cannot overwrite automatic parameters. In addition, you cannot edit sub-cores that are instantiated inside this AXI Ethernet. If you want to edit advanced parameters of the sub-cores, you need to build the system by directly using that core. You cannot modify internal connections of AXI Ethernet.

Configurations that are not allowed or not valid in a particular mode are greyed out.

Note: Figures in this chapter are illustrations of the Vivado IDE. This layout might vary from the current version.

Vivado IDE might auto-compute certain configuration values when validating or generating the design, as noted in this section. You can view the parameter value after successful completion of the `validate_bd_design` command.

Customizing AXI Ethernet Using Dialog Boxes in Vivado IDE

When the **Customize Block** option is selected, a window will display showing different available configurations. These are organized in various tabs for better readability and configuration purposes.

The **Board** tab: The Board tab (Figure 4-1) is visible only when a board is selected in the current project. In this tab, options related to the board-based I/O constraints are provided. More details onboard support packages are available in the *Vivado Design Suite User Guide: Designing IP Subsystems using IP Integrator* (UG994) [Ref 4].

- **Use board flow**. This option needs to be selected to make use of the board flow. If you do not want to make use of board flow, this option can be left unchecked. If the tool is able to generate the physical constraints for a given configuration, only those options are active; otherwise, they are grayed out. When this option is selected the following three interfaces can be enabled.
  - Ethernet: This option enables to select the type of Ethernet I/O interface available in board.
  - MDIO: This option enables to select the mdio interface available in board.
  - MGT_DIFF_CLK: This option allows you to select the serial transceiver clock source if that source is available on the board.

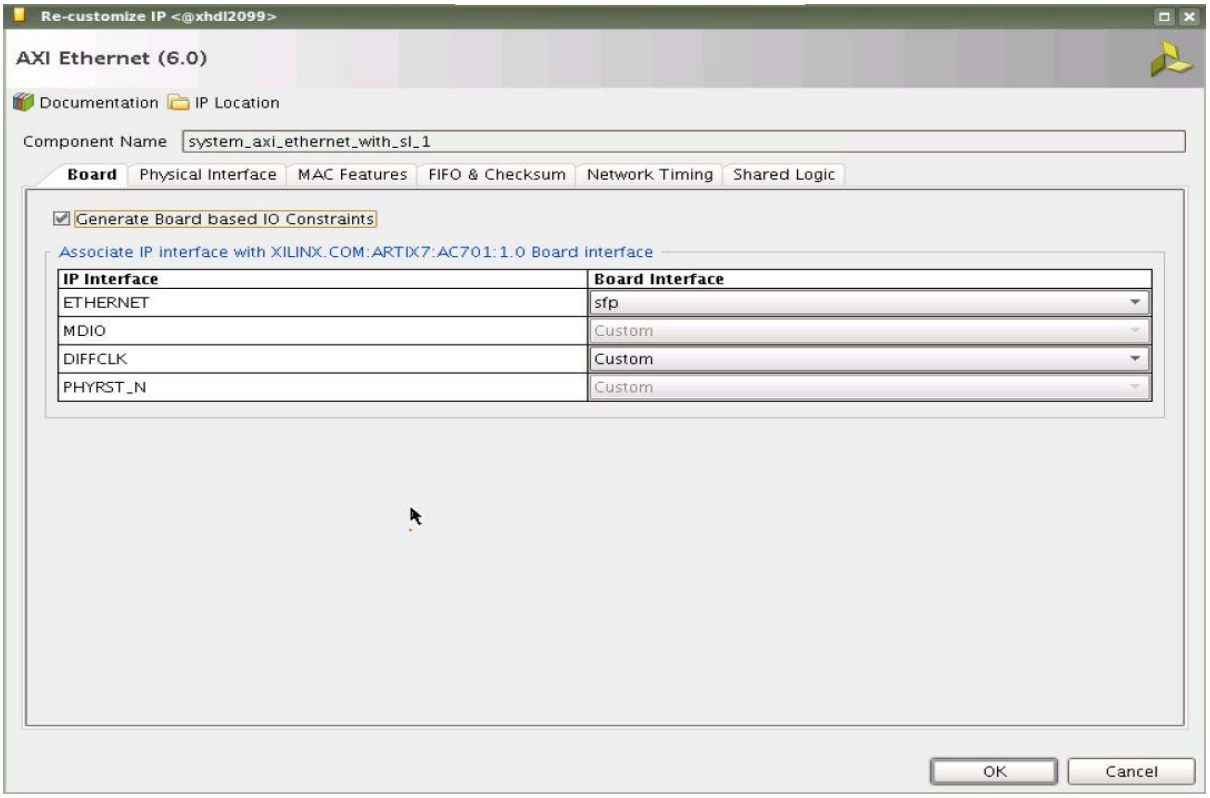


Figure 4-1: **Board Tab of AXI Ethernet Configuration Dialog Box**

The **Physical Interface** tab: This tab (Figure 4-2) is related to the physical interfaces that are available. The following options are available in this tab.

- Physical Interface Type: The physical interface type is selected using this option. The PHY types supported are MII, GMII, RGMII, SGMII, and 1000BaseX.
- PHY address is the address for the MDIO interface for the Gigabit Ethernet PCS PMA Phy address. This can be configured using this option.
- The LVDS mode is enabled only in the SGMII mode of operation.

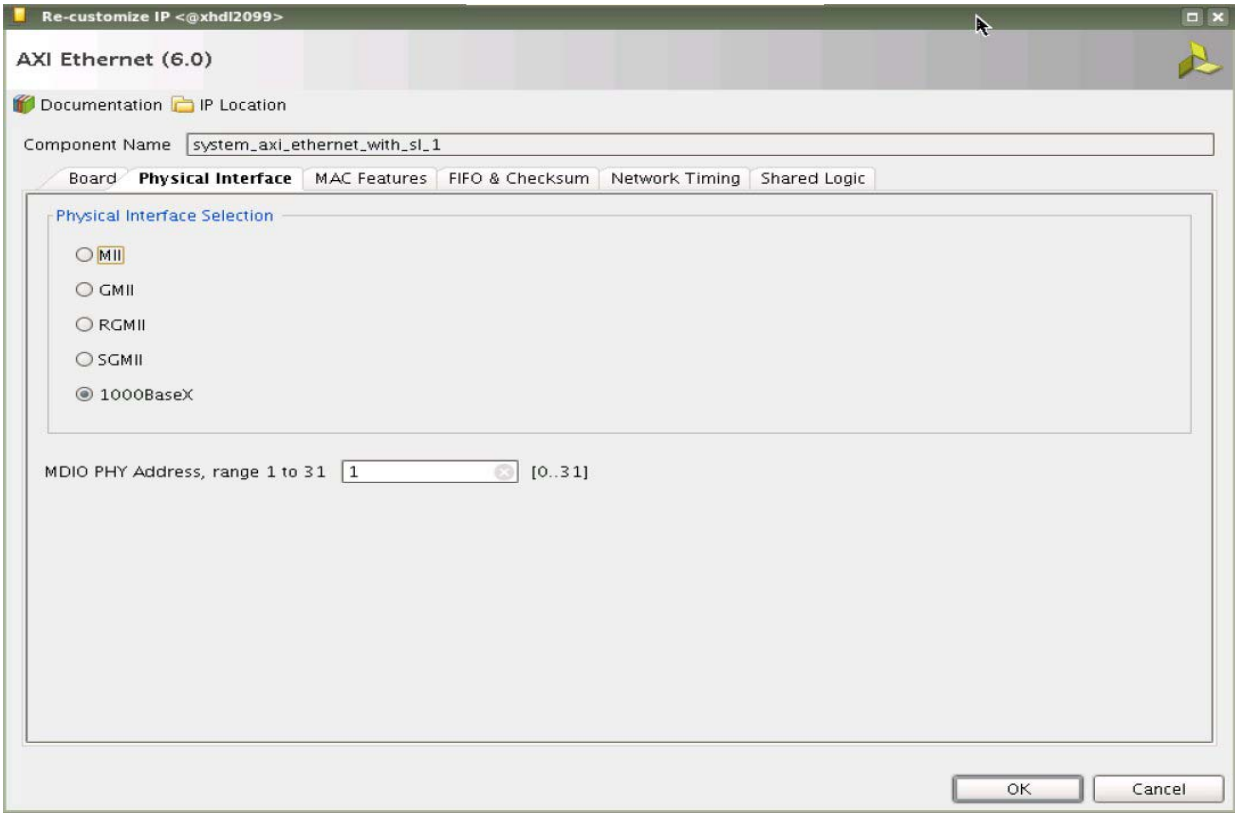


Figure 4-2: **Physical Interface Tab of AXI Ethernet Configuration Dialog Box**

The **MAC Features** tab: This MAC Features tab (Figure 4-3) has options related to the MAC functionality. The following options are available in this tab.

- Statistics Counter Options: Use this option to enable the statistic counters. **Enable Statistics Counters** should be selected for this purpose. By selecting the **Statistics Reset** option, the statistics reset capability can be enabled. The width of the statistics counter can be selected using the **Statistics Width** option.
- RX extended multicast filtering can be enabled using this **Enable RX extended multicast address filtering** option
- Advanced VLAN options for TX and RX data streams for VLAN tagging, VLAN stripping, and VLAN translation are selected using the respective buttons.

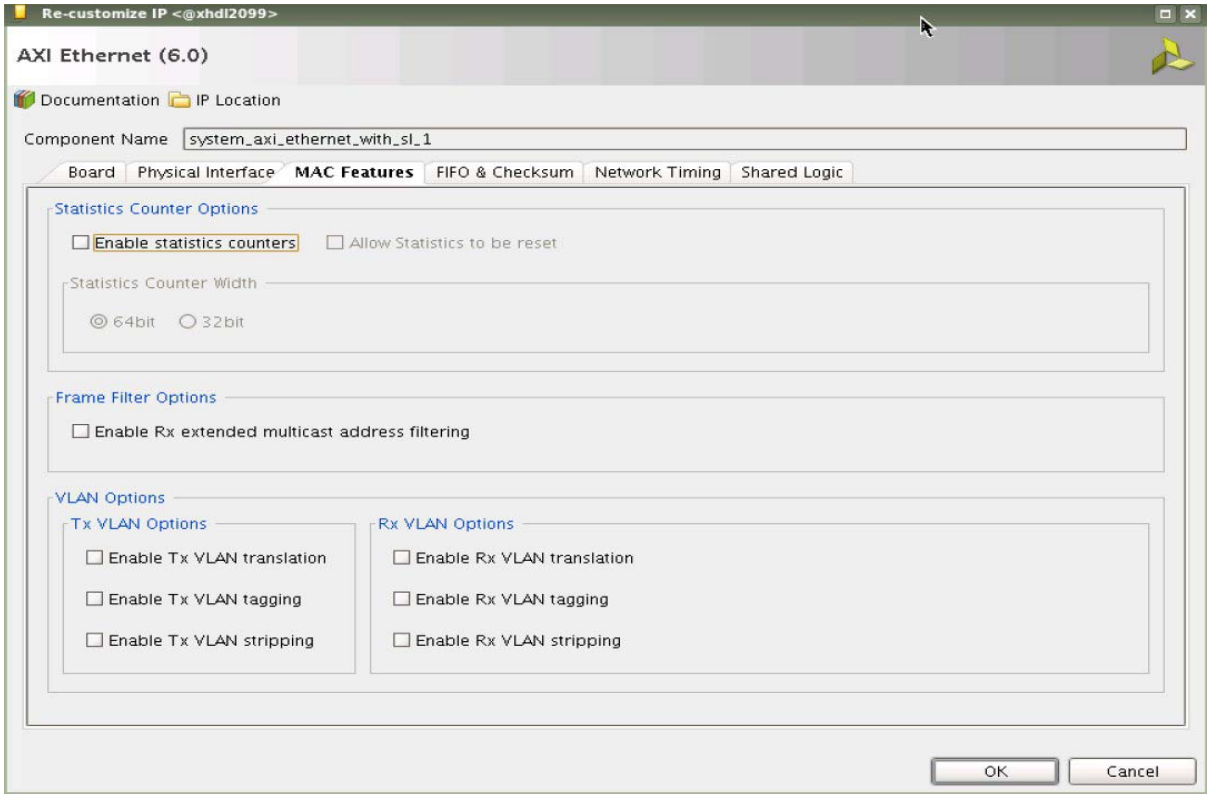


Figure 4-3: **MAC Features Tab of AXI Ethernet Configuration Dialog Box**

The **FIFO and Checksum** tab: The FIFO and Checksum tab (Figure 4-4) is related to the FIFO settings and checksum calculation methods. The following options are available in this tab.

- TX and RX memory sizes can be selected using the TX memory and RX memory size options,
- TX and RX checksum offload capability can be selected using the RX checksum offload option and TX checksum offload option.

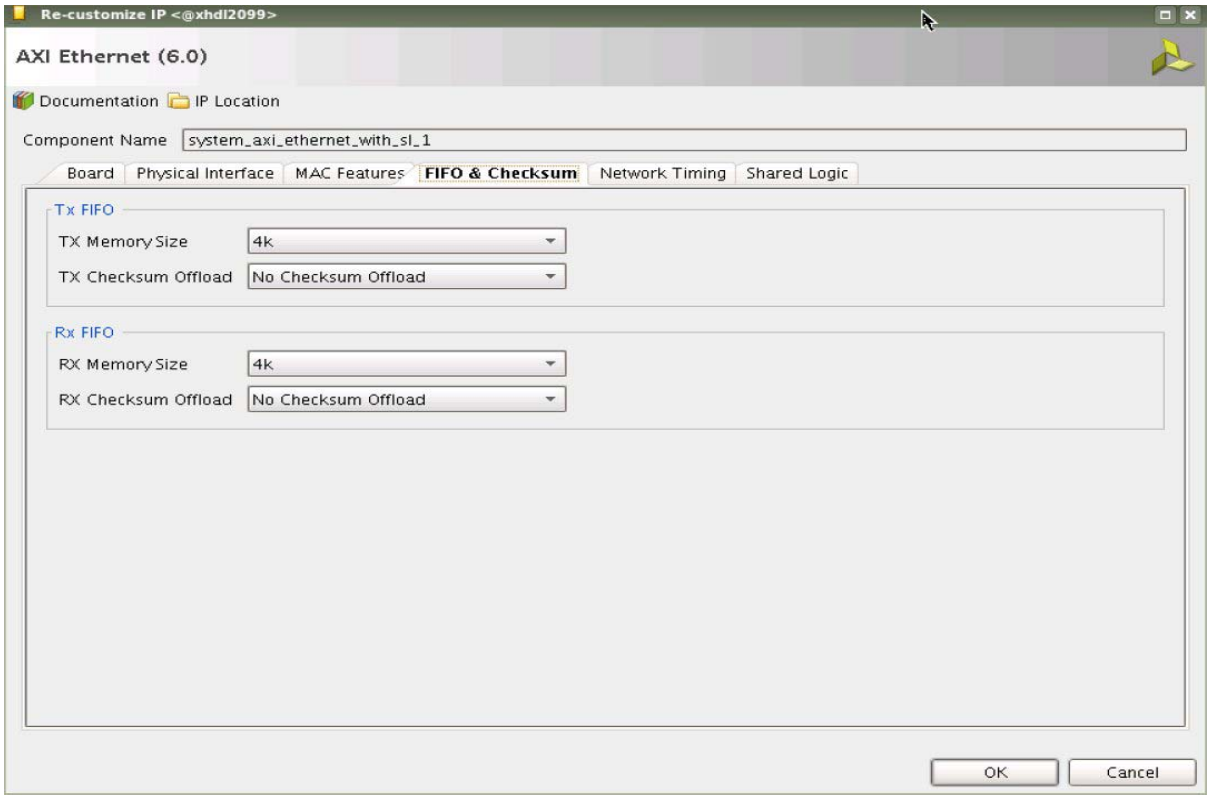


Figure 4-4: **FIFO and Checksum Tab of AXI Ethernet Configuration Dialog Box**

The **Network Timing** tab: The Network Timing tab is shown in Figure 4-5. The 1588 options are enabled only in 1000BaseX mode. This is used to select the 1588 mode and AVB mode. The following options are available in this tab.

- Enable 1588: This enables the 1588 mode of operation. The following sub-options are enabled only when Enable 1588 is enabled.
  - 1-step or 2-step Support: This enables the 1-step or 2-step operation method of 1588. This is enabled only when Enable 1588 is selected.
  - Selection of the Time-of-Day (ToD) timer and timestamp format or the Correction Field timer and timestamp format.
  - 1588 System reference Time period: This is the 1588 system reference clock period in picoseconds.
- Enable AVB: This would enable Audio Video Bridging functions. This can be enabled only when 1588 mode is disabled.

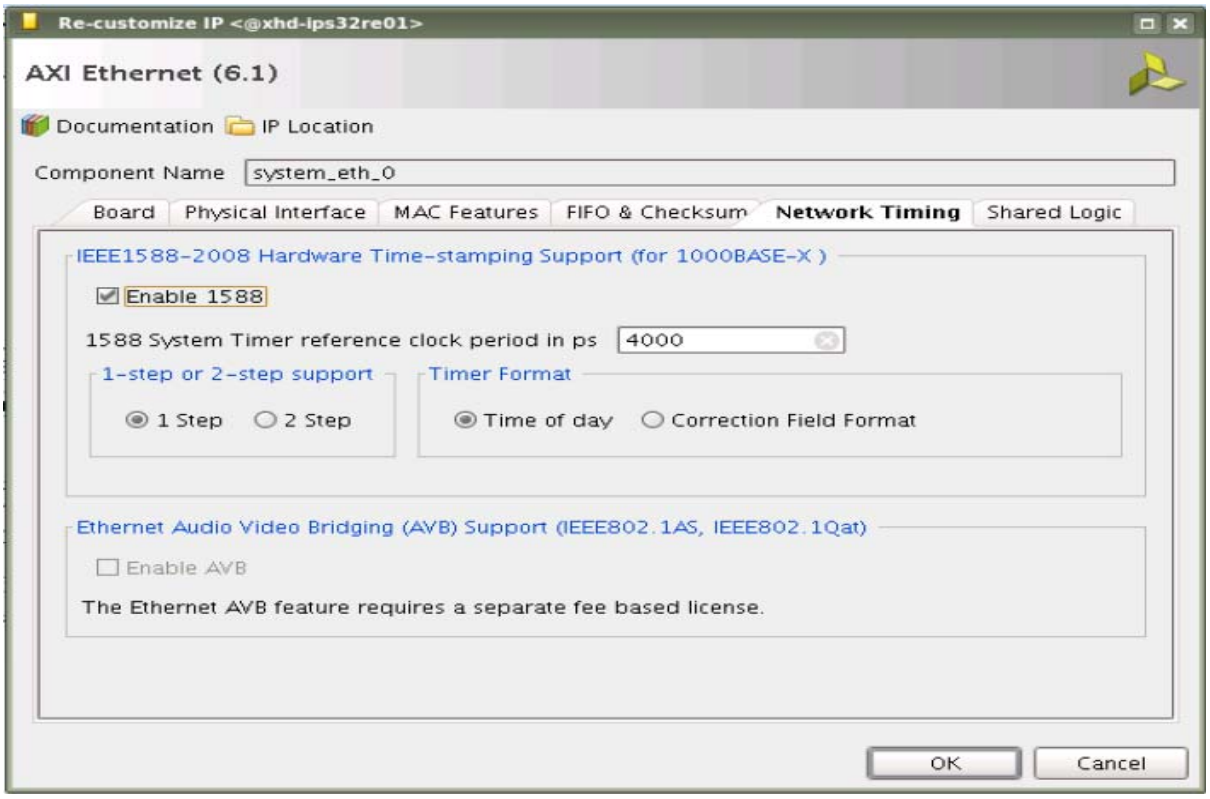


Figure 4-5: **Network Timing Tab of AXI Ethernet Configuration Dialog Box**

The **Shared Logic** tab: The Shared Logic tab (Figure 4-6) selects whether shared logic is in the core or not. Shared logic is different for different configurations. In GMII mode IDELAYCTRL is the shared element. In RGMII mode, IDELAYCTRL and the TX MMCM with its associated clock buffers for Artix®-7 or Kintex®-7 devices are shared logic. There is no shareable logic in MII mode. In SGMII using transceiver mode or 1000Basex Mode, the transceiver differential reference clock buffer, MMCM, and clock buffer are shared elements. In SGMII over LVDS, the transceiver differential reference clock buffer, MMCM, IDELAYCTRL and clock buffer are shared elements. The options for **Shared Logic** select whether to include or not include shared logic in the core.

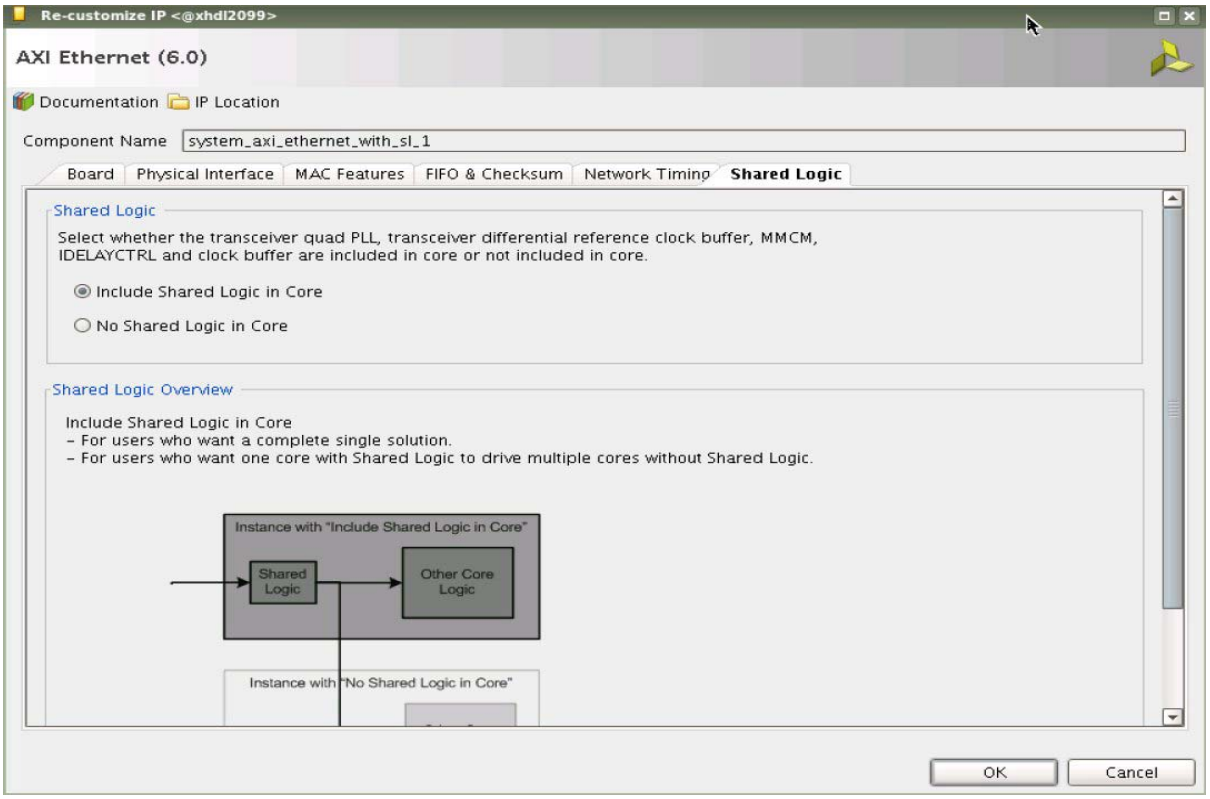


Figure 4-6: **Shared Logic Tab of AXI Ethernet Configuration Dialog Box**

Customizing AXI Ethernet to Share Resources across Multiple Instances

When using multiple instances of the AXI Ethernet core, based on the mode certain types of resources can be shared across these instances. Among these instances one instance should be configured in **Use Shared Logic in Core** and the rest in **No Shared Logic in Core**. Figure 4-7 shows that the outputs from a core configured in **Use Shared Logic in Core** are connected to the inputs of the core configured in **No Shared Logic in Core**.

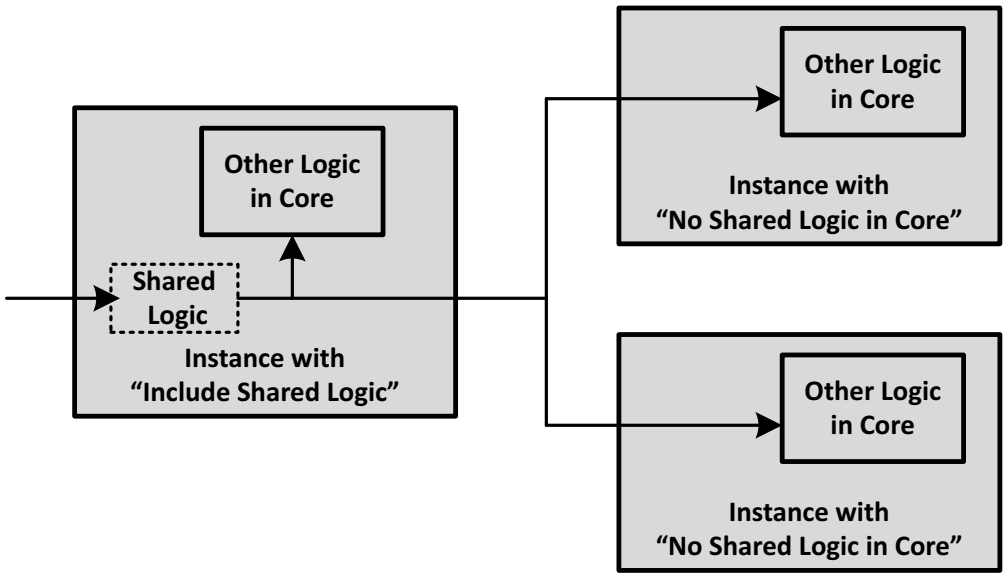


Figure 4-7: **AXI Ethernet Sharing Resources among Multiple Instances Using Shared Logic**

Following are two examples: one for RGMII mode and the other for 1000BaseX mode. Connections for other modes can be done similarly.

Example 1: Sharing IDELAYCTRL in RGMII Mode among Multiple AXI Ethernet Instances

When multiple AXI Ethernet instances are targeted for I/O in the same bank, IDELAYCTRL needs to be shared. The `gtx_clock` and `gtx_clk_out` of the instance configured in **Use Shared Logic in Core** need to be connected to inputs of instance configured in **No Shared Logic in Core** mode. This is shown in the Figure 4-8. The connections shown in brown are specific to connections for sharing logic. The rest are regular connections.

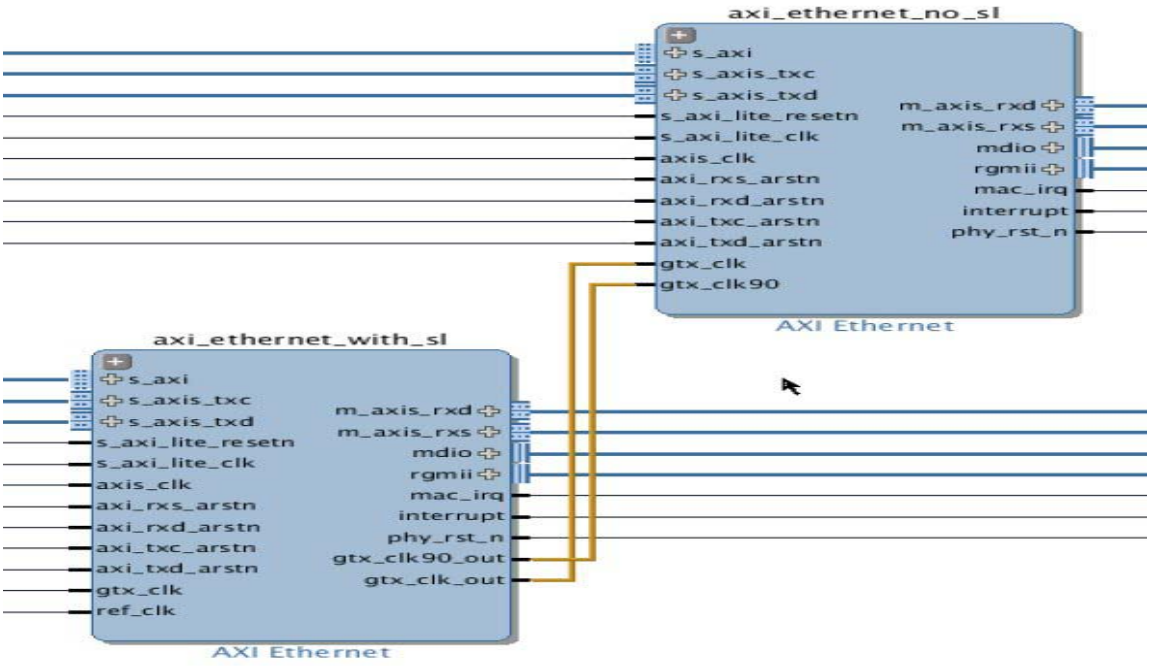


Figure 4-8: **Connections to Enable Sharing Logic among Two Instances**

Example 2: Sharing GT COMMON in 1000BaseX Mode among Multiple AXI Ethernet Instances

When multiple AXI Ethernet instances are targeted for GTs in the same quad, the GTCOMMON and IBUFDS need to be shared. The GT outputs of the instance configured in **Use Shared Logic in Core** needs to be connected to inputs of an instance configured in **No Shared Logic in Core** mode. An example where the RX clock is synchronous between both the instances is shown in Figure 4-9. The connections shown in brown are specific to connections for sharing logic. The rest are regular connections.

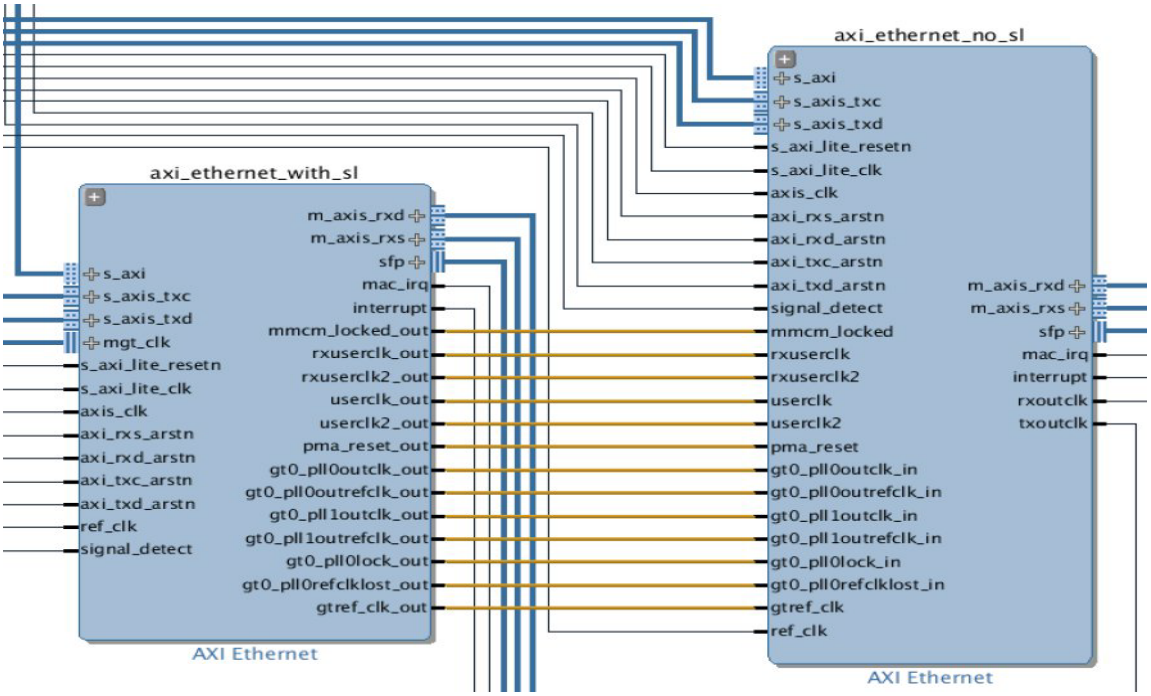


Figure 4-9: **Connections to Enable Sharing of GTCOMMON among Two Instances**

Using Designer Assistance for AXI Ethernet

After the AXI Ethernet core is added to the block design canvas, design assistance is enabled and **Run Block Automation** and **Run Connection Automation** commands are enabled. See Figure 4-10.

The **Run Block Automation** command will connect streaming interfaces of AXI Ethernet to a DMA or a FIFO as selected in the Vivado IDE. You cannot configure the AXI Ethernet core using the block automation Vivado IDE; therefore, it is recommended first to configure AXI Ethernet and then run block automation. `gtx_clk` and `ref_clk` are connected to respective clock sources in case the clock source is already present. The `axis_clk` signal should be connected to the same clock source as the streaming interfaces are connected. In the case of FIFO, `axis_clk` should be connected to the AXI_lite clock of the FIFO. In the case of the DMA this needs to be connected to the same clock source as `m_axi_mm2s_aclk` and `m_axi_s2mm_aclk` of DMA.

The **Run Connection Automation** command will connect the AXI4-Lite interface of the AXI Ethernet core to the peripheral interface of the processor. This will also connect the AXI4-Lite clock and the AXI4-Lite reset to the respective sources. When the project is set for a board that has the related interfaces, the **Run Connection Automation** command will also connect the I/Os to external I/O ports by creating them and providing the LOC constraints.

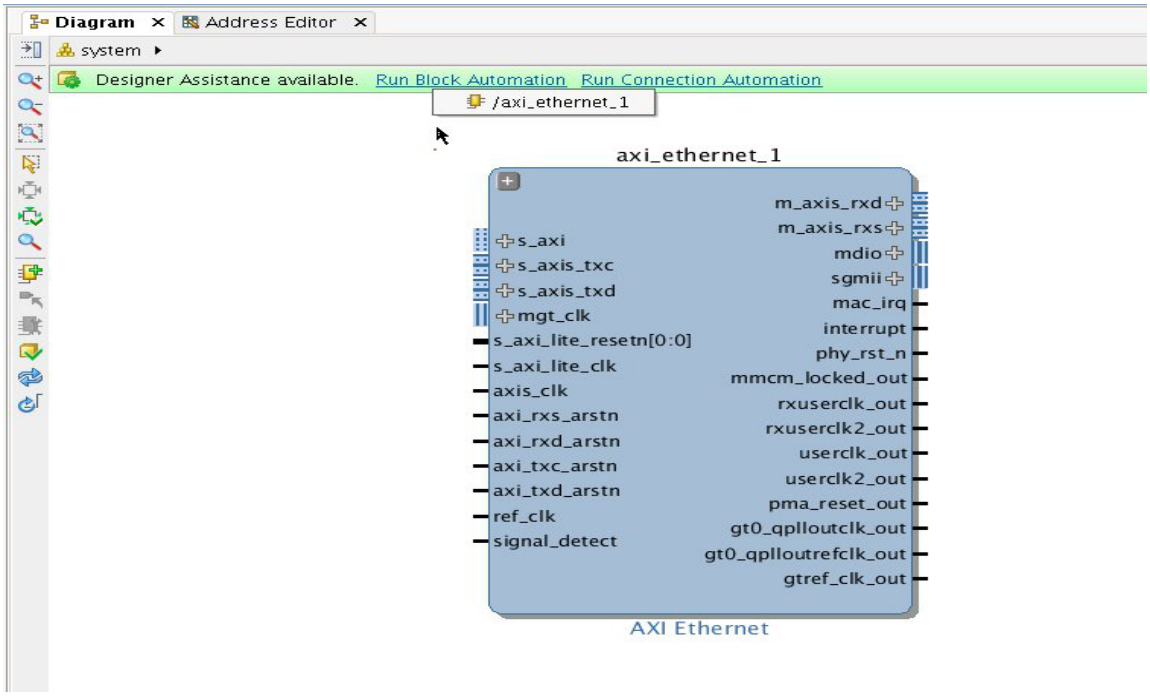


Figure 4-10: **Designer Assistance for AXI Ethernet**

Output Generation

For details, see the *Vivado Design Suite User Guide: Designing with IP* (UG896) [Ref 5].

Constraining the Core

This section contains information about constraining the core in the Vivado® Design Suite.

Required Constraints

Because the AXI Ethernet core is a hierarchical core, it enables the use of timing constraints from the infrastructure cores. The core automatically picks up the constraints from the sub cores. The timing constraints related to MII, GMII, RGMII interfaces are provided by Tri-Mode Ethernet Mac. The timing constraints related to VLAN and others are provided by AXI Ethernet Buffer. The timing constraints related transceiver, elastic buffers are provided by Gigabit Ethernet PCS PMA.

In addition to sub-cores constraints, the placement constraints should be provided at the top.

When a project is targeted for a board, you can use **Board Based IO Constraints** generation. Upon selection of the board interface, the constraints for that interface are generated automatically. The AXI Ethernet core will propagate the required inputs to sub cores and these will generate the constraints in their file list. These constraints are usually available in a file that has file name ending with `_board.xdc`.

Device, Package, and Speed Grade Selections

The selections need to be done in accordance with the requirements of the helper cores. See the *LogiCORE™ IP Ethernet 1000BASE-X PCS/PMA or SGMII Product Guide* (PG047) [Ref 2] and *LogiCORE IP Tri-Mode Ethernet MAC Product Guide* (PG051) [Ref 1] for more details.

Clock Frequencies

This section is not applicable for this IP core.

Clock Management

This section is not applicable for this IP core.

Clock Placement

The clock buffer needs to be placed in accordance with the requirements of the helper cores. See the *LogiCORE IP Ethernet 1000BASE-X PCS/PMA or SGMII Product Guide* (PG047) [Ref 2] and *LogiCORE IP Tri-Mode Ethernet MAC Product Guide* (PG051) [Ref 1] for more details.

Banking

There is no information currently provided for this core.

Transceiver Placement

The transceiver placements should be done in accordance with the requirement of Gigabit Ethernet PCS PMA. See the *LogiCORE IP Ethernet 1000BASE-X PCS/PMA or SGMII Product Guide* (PG047) [Ref 2].

I/O Standard and Placement

The I/O standards and placements needs to be placed in accordance with the requirements of the helper cores. See the *LogiCORE IP Ethernet 1000BASE-X PCS/PMA or SGMII Product Guide* (PG047) [Ref 2] and *LogiCORE IP Tri-Mode Ethernet MAC Product Guide* (PG051) [Ref 1] for more details.

Simulation

This section contains information about simulating IP in the Vivado Design Suite. For comprehensive information about Vivado simulation components, as well as information about using supported third-party tools, see the *Vivado Design Suite User Guide: Logic Simulation* (UG900) [Ref 6].

Synthesis and Implementation

This section contains information about synthesis and implementation in the Vivado Design Suite. For details about synthesis and implementation, see the *Vivado Design Suite User Guide: Designing with IP* (UG896) [Ref 5].

IEEE 1588 Product Specification

Generating the AXI Ethernet Core for 1588 Operation

See Chapter 4, Customizing and Generating the Core. To configure the core with IEEE 1588 hardware timestamping support, open the AXI Ethernet Vivado® IDE and perform the following steps:

1. From the Physical Interface tab, select the **1000BaseX** option.
2. From the Network Timing tab, select the **Enable 1588** option.
3. For 1-step and 2-step support, ensure that the **1 Step** option is selected; if only 2-step functionality is required, then select the **2 Step** option to save on logic resources.
4. Select whether to include logic that supports either the Time-of-Day (ToD) timer and timestamp format, or alternatively to support the Correction Field timer and timestamp format.
5. Accurately enter the clock period, in picoseconds, of the 1588 System Timer reference clock period. This is the clock provided to the system timer clock port of Table 5-2.
6. Click **OK** to generate the core.

Functional Description

IEEE 1588 hardware timestamping support has been added as an option to the AXI Ethernet core. When this option is selected, the AXI Ethernet Buffer helper core (see Figure 2-1 in Chapter 2, Product Specification) is not included.

The available feature set of the core, therefore, is built on the existing functionality provided by the following two cores.

- Tri-Mode Ethernet MAC (TEMAC): See the *LogiCORE IP Tri-Mode Ethernet MAC Product Guide* (PG051) [Ref 1]
- Ethernet 1000BASE-X PCS/PMA or SGMII: see the *LogiCORE IP Ethernet 1000BASE-X PCS/PMA or SGMII Product Guide* (PG047) [Ref 2].

Refer, in particular, to the *LogiCORE IP Tri-Mode Ethernet MAC Product Guide* (PG051) [Ref 1] for using the AXI4-Stream interfaces for the datapath, and to the AXI4-Lite interface for configuration and status of the TEMAC core. The additional ports described in this chapter are additional to the user interface ports described therein.

Supported Features

Devices and Physical Interface

IEEE 1588 hardware timestamping support is available only for the 1000BASE-X physical interface, supporting 7 series GTX transceivers.

7 series GTX transceivers are available in the Kintex®-7 family and in selected parts of the Virtex®-7 family; see the *7 Series FPGAs Overview* (DS180) [Ref 8].

IEEE 1588 Supported Features

- Hardware timestamping at full Ethernet line rate on both transmit and receive paths. Timestamp accuracy will be better than ± 10 ns under all operating conditions.
- 1-step and 2-step support for Time of Day (ToD) timestamps (IEEE1588-2008 format consisting of a 48-bit seconds field and a 32-bit nanoseconds field).
- On receive, all frames are timestamped with a captured 80-bit ToD timestamp. The full 80-bit timestamp is provided to the client logic out of band using additional ports defined in Table 5-4. In addition, an optional 64-bit timestamp can be provided in line with the received frame. This 64-bit timestamp consists of the lower 32 bits from the 1588 timers seconds field, plus all 32 bits of the nanoseconds field. For the Correction Field format, the full 64-bit timestamp is provided to the client logic out of band using additional ports defined in Table 5-3. In addition, the 64-bit timestamp can optionally be provided in line with the received frame. All PTP frame types are supported on receive.
- On transmit:
  - A command field is provided by the client to the TEMAC either in line with the frame sent for transmission, or in parallel with the frame sent for transmission. This indicates, on a frame-by-frame basis, the 1588 function to be performed (no-operation, 1-step or 2-step) and also indicates, for 1-step frames, whether there is a UDP checksum field that requires updating.
  - For 1-step and 2-step operation, the full 80-bit captured ToD timestamp is returned to the client logic using the additional ports defined in Table 5-5.
  - For 1-step operation, the full 80-bit ToD captured timestamp is inserted into the frame.

- For 1-step UDP frame types, the UDP checksum is updated in accordance with IETF RFC 1624. (In order for this update function to work correctly, the original checksum value for the frame sent for transmission should be calculated using a zero value for the timestamp data).
- For all 1-step frames, the Ethernet Frame Check Sequence (FCS) field is calculated after all frame modifications have been completed.

- Supported 1588 PTP frame types are:
  - For transmit 1-step:
    - Raw Ethernet frames
    - UDP IPv4 frames
    - UDP IPv6 frames
  - For transmit 2-step, all Precise Timing Protocol (PTP) frame formats can be supported.
  - For receive, all PTP frame formats can be supported.

Architecture Overview

Transmitter

Figure 5-1 shows the Transmitter portions of the Ethernet MAC and 1000BASE-X PHY enhanced with 1588 support.

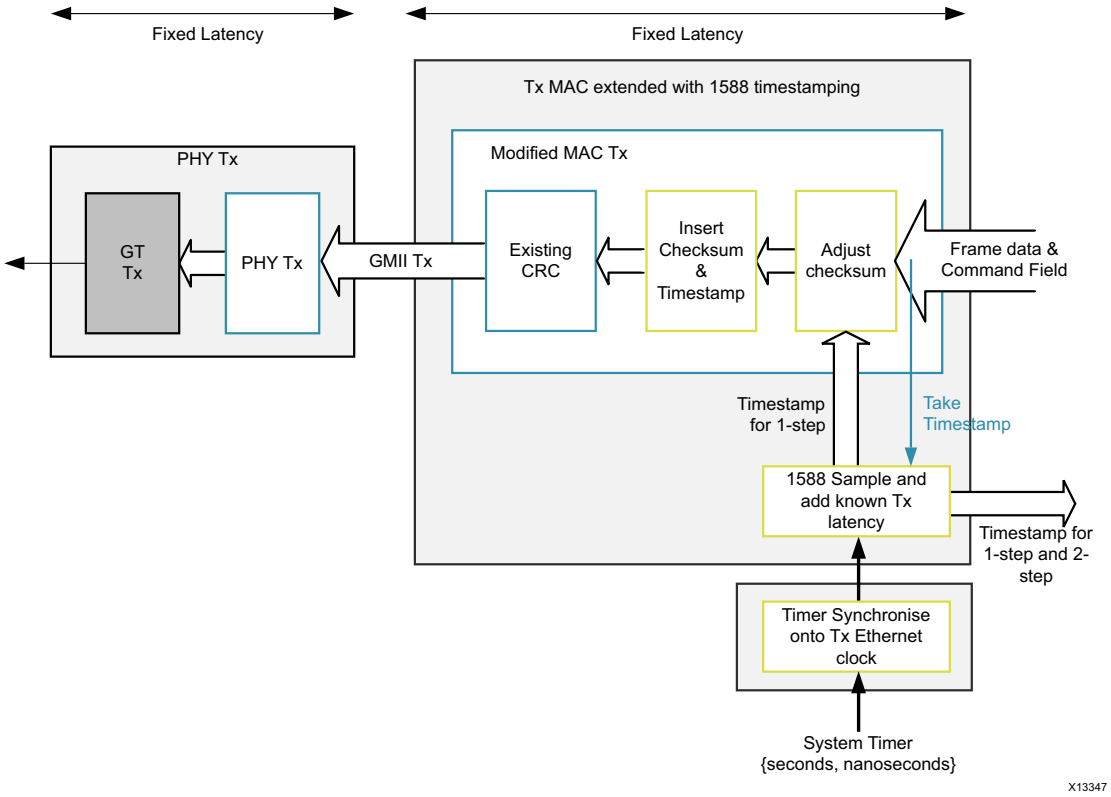


Figure 5-1: **Transmitter Block Level Diagram**

1588 Frame-by-Frame Timestamp Operation

The Ethernet frame sent to the MAC contains a command field. The format of the command field is defined in the following list. The information contained within the command field indicates one of the following on a frame-by-frame basis.

- No operation: the frame is not a PTP frame and no timestamp action should be taken.
- 2-step operation is required and a tag value (user-sequence ID) is provided as part of the command field; the frame should be timestamped, and the timestamp made available to the client logic, along with the provided tag value for the frame. The additional MAC transmitter ports (defined in Table 5-5) provide this function.
- 1-step operation is required
  - For the ToD timer and timestamp format, a timestamp offset value is provided as part of the command field; the frame should be timestamped, and the timestamp should be inserted into the frame at the provided offset (number of bytes) into the frame.
  - For the Correction Field format, a Correction Field offset value is provided as part of the command field; the frame should be time-stamped, and the captured 64-bit Timestamp is summed with the existing Correction Field contained within the frame and the summed result is overwritten into original Correction Field of the frame.

- For 1-step operation, in addition to the timestamp value insertion into the frame the CRC value of the frame should also be updated/recalculated. For UDP IPv4 and IPv6 PTP formatted frames, the checksum value in the header of the frame needs to updated/recalculated.
- For 1-step UDP frame types, the UDP checksum is updated in accordance with IETF RFC 1624.
  - If using the ToD format, in order for this update function to work correctly, the original checksum value for the frame sent for transmission should be calculated using a zero value for the timestamp data. This particular restriction does not apply when using the Correction Field format.
  - If using the Correction Field format then a different restriction does apply: the separation between the UDP Checksum field and the Correction Field within the 1588 PTP frame header is a fixed interval of bytes, supporting the 1588 PTP frame definition. This is a requirement to minimize the latency through the MAC because both the checksum and the correction field must both be fully contained in the MAC pipeline in order for the checksum to be correctly updated. This particular restriction does not apply to the ToD format because the original timestamp data is calculated as a zero value; consequently the checksum and timestamp position can be independently located within the frame.

Table 5-1 provides a definition of the command field.

Table 5-1: **Command Field Definition**

Bits	Name	Description
[1:0]	1588 operation	2'b00 – No operation: no timestamp is taken and the frame is not modified. 2'b01 – 1-step: a timestamp should be taken and inserted into the frame. 2'b10 – 2-step: a timestamp should be taken and returned to the client. The frame itself is not modified. 2'b11 – Reserved: act as No operation.
[7:2]	Reserved	Reserved for future use. Values are ignored.
[8]	Update Checksum	The usage of this field is dependent on the 1588 operation. For No operation or 2-step, this bit are ignored. For 1-step: 1'b0: the PTP frame does not contain a UDP checksum. 1'b1: the PTP frame does contain a UDP checksum which the core is required to recalculate.
[15:9]	Reserved	Reserved for future use. Values are ignored.
[31:16]	Tag Field	The usage of this field is dependent on the 1588 operation. For No operation, this field are ignored. For 1-step and 2-step this field is a tag field. This tag value is returned to the client with the timestamp for the current frame. This tag value can be used by software to ensure that the timestamp can be matched with the PTP frame that it sent for transmission.

Table 5-1: **Command Field Definition *(Cont'd)***

Bits	Name	Description
[47:32]	Timestamp Offset	The usage of this field is dependent on the 1588 operation. For No operation or 2-step this field is ignored. For 1-step, • In ToD format, this field is a numeric value indicating the number of bytes into the frame to where the first byte of the timestamp to be inserted is located (where a value of 0 represents the first byte of the Destination Address, etc). • In Correction Field format, this field is a numeric value indicating the number of bytes into the frame to where the first byte of the Correction Field to be modified is located (where a value of 0 represents the first byte of the Destination Address, etc). Only even values of offset are currently supported.
[63:48]	Checksum Offset	The usage of this field is dependent on the "1588 operation" and on the "Update Checksum" bit. For No operation, for 2-step or for 1-step when Update Checksum is set to 1'b0, this field is ignored. If using the Correction Field format, this field is completely ignored because the Checksum location is a fixed number of bytes prior to the position of the Correction Field Offset (fully supporting the IEEE1588 PTP frame formats). For 1-step when Update Checksum is set to 1'b1, this field is a numeric value indicating the number of bytes into the frame to where the first byte of the checksum is located (where a value of 0 represents the first byte of the Destination Address, etc).

Transmitter Latency and Timestamp Adjustment

Figure 5-1 illustrates 1**588 Sample and add known Tx latency** block. The timestamp is sampled when the Ethernet Start of Frame Delimiter (SFD) is observed at the very beginning of the TEMAC transmitter pipeline. This is required to update the UDP Checksum and FCS fields with the timestamp value that is to be inserted in the frame for 1-step operation. For this timestamp to provide reliable system behavior, the following conditions apply.

- The TEMAC contains a fixed latency from the timestamp position onwards through its pipeline.
- The 1000BASE-X core provides a fixed latency for the transmitter path.
- The 7 series FPGA GTX transceiver provides a fixed and deterministic latency through its transmitter path. This is achieved by using the GTX transceiver in TX buffer Bypass mode.

The logic is also then capable of adjusting the ToD timestamp value taken by adding a configurable duration (see the 1588 configuration registers described in IEEE1588 Configuration Registers). This value is user adjustable, but its default is initialized with the entire transmitter path latency (through TEMAC, 1000BASE-X, and transceiver).

This results in the returned timestamp default value representing the time at which the SFD can be first observed on the GTX serial transmit output. This latency adjust functionality can be applied to either of the ToD or Correction Field formats.

Receiver

Figure 5-2 shows the Receiver portions of the Ethernet MAC and 1000BASE-X PHY enhanced with 1588 support.

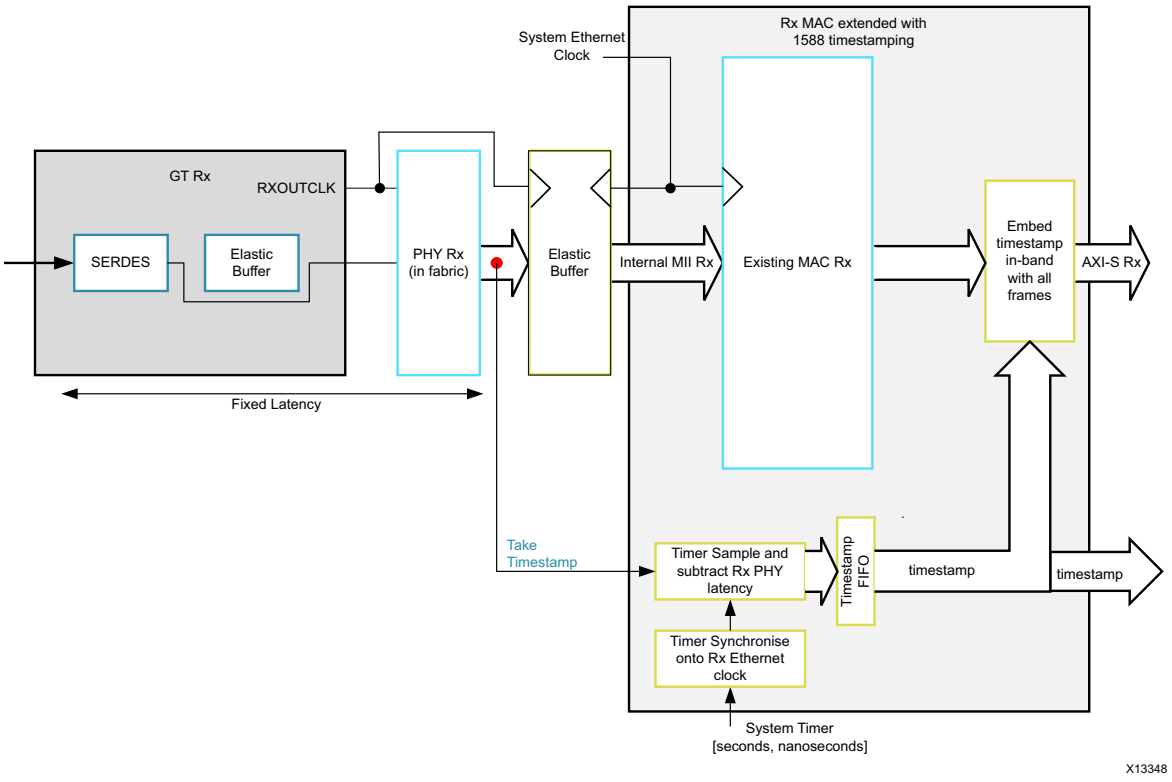


Figure 5-2: **Receiver Block Level Diagram**

1588 Frame-by-Frame Timestamp Operation

All received Ethernet frames are timestamped with a captured 80-bit ToD timestamp. The full 80-bit timestamp is provided to the client logic out of band using additional ports defined in Table 5-4. In addition, an optional 64-bit timestamp can be provided in line with the received frame. This 64-bit timestamp consists of the lower 32 bits from the 1588 timers seconds field, plus all 32 bits of the nanoseconds field. For the Correction Field format, the full 64-bit timestamp is provided to the client logic out of band using additional ports defined in Table 5-3. In addition, the 64-bit timestamp can optionally be provided in line with the received frame. All PTP frame types are supported on receive.

Receiver Latency and Timestamp Adjustment

Figure 5-2 illustrates a block called **Timer Sample and subtract known Rx PHY latency**. This illustrates the timestamp point in the receiver pipeline when the SFD is observed. This timestamp is performed in the 100BASE-X PHY block, prior to any variable length latency logic.

- The 7 series FPGA GTX transceiver provides a fixed and deterministic latency through its receiver path. This is achieved by using the GTX transceiver in RX buffer Bypass mode.
- The 1000BASE-X core provides a fixed latency for the receiver path up until the timestamp point.

The logic is also then capable of adjusting the ToD timestamp value taken by subtracting a configurable duration (see the 1588 configuration registers described in IEEE1588 Configuration Registers). This value is user adjustable, but its default is initialized with the receiver path latency (transceiver and 1000BASE-X logic) prior to the timestamping position. This results in the returned timestamp value representing the time at which the Start codegroup appeared on the transceiver serial input. This latency adjust functionality can be applied to either of the ToD or Correction Field formats. The Correction Field value is provided to the core in "1588 Correction Field Mode" using a 64-bits port. This Correction Field value is in a numerical format as defined in IEEE1588 clause 13.3.2.7. These details are provided in Table 5-3.

1588 Port Descriptions and Port Operation

Use the following port descriptions along with the ports described in the *LogiCORE IP Tri-Mode Ethernet MAC Product Guide* (PG051) [Ref 1].

IEEE 1588 System Timer Ports

The 1588 system-wide Time-of-Day (ToD) timer are provided to the core using the ports defined in Table 5-2.

Table 5-2: **1588 System Timer Ports**

Name	Clock domain	Description
systemtimer_clk	–	System reference clock for the system timer provided to the core.
systemtimer_s_field[47:0]	systemtimer_clk	The 48-bit seconds field of the 1588-2008 system timer. This increments by 1 every time the `systemtimer_ns_field` is reset back to zero.
systemtimer_ns_field[31:0]	systemtimer_clk	The 32-bit nanoseconds field of the 1588-2008 system timer. This counts from 0 up to (1x10^9)-1 [1 second], and then resets back to zero.

Table 5-3: **1588 Correction Field Ports**

systemtimer_clk	Clock Domain	Description
systemtimer_clk		Clock for the system timer provided to the core.
correction_timer[63:0]	systemtimer_clk	Bits [63:16] represent a 48-bit ns field. Bits[15:0] represents a fractional ns field • Bit 15 represents a half ns • Bit 14 represents a quarter ns • Bit 13 represents one eighth ns, etc.

Received Timestamp Ports (Out of Band)

The captured timestamp will always be presented out-of-band with TEMAC frame reception using a dedicated AXI4-Stream interface. The signal definition for this is defined in Table 5-4.

A timing diagram showing the operation of this interface follows the table. To summarize, the timestamp is valid on the same clock cycle as the first data word of frame data. This AXI4-Stream interface is synchronous to the TEMAC receive clock.

Table 5-4: **AXI4-Stream Interface Ports - Receive Timestamp**

Name	Direction	Description
m_axis_rx_ts_data[127:0]	OUT	AXI4-Stream Receive Timestamp from the TEMAC. • ToD Timestamp Format Bits[127:80] – Reserved Bits [79:32] – Captured Timestamp Seconds field Bits [31:0] – Captured Timestamp Nanoseconds field • Correction Field Timestamp Format Bits[127:64]: Reserved for future use (all bits should be ignored). Bits[63:0]: Transmit Timestamp from the TEMAC.
m_axis_rx_ts_tvalid	OUT	AXI4-Stream Receive Timestamp Data Valid from the MAC

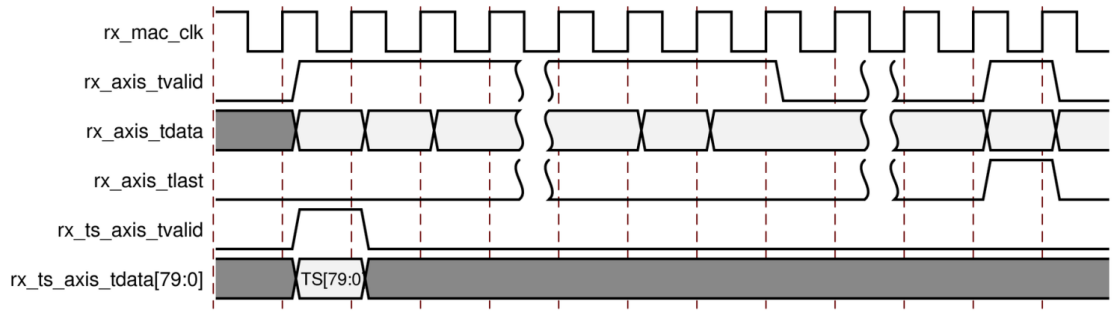


Figure 5-3: **AXI4-Stream Interface Timing – Receive Timestamp**

Received Frame Timestamp In-line with Frame Reception

The captured timestamp can optionally be provided in-line with the received frame using the TEMAC existing AXI4-Stream Interface – Receive (see *LogiCORE IP Tri-Mode Ethernet MAC Product Guide* (PG051) [Ref 1].

Enabling this mode of operation is through an AXI4-Lite addressable configuration bit (see Table 5-9, bit 22).

When enabled, a 64-bit timestamp is passed to the client immediately before the start of the frame reception (in place of the Preamble field). This 64-bit value includes the lower 32-bits of the seconds field, plus the entire nanoseconds field. See Figure 5-4.

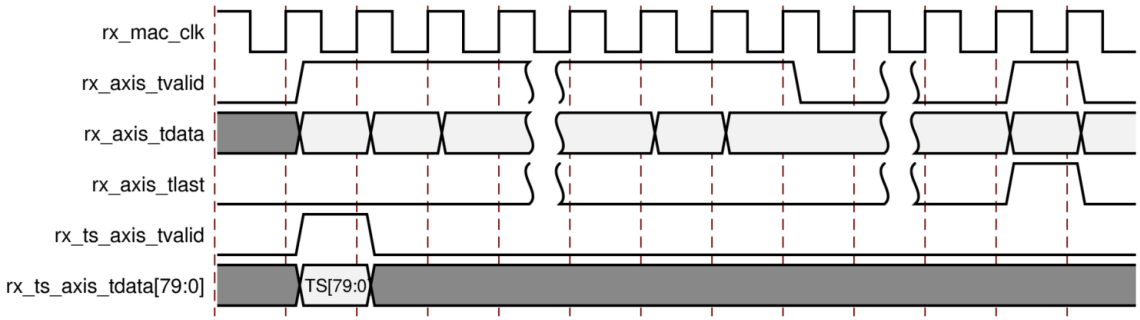


Figure 5-4: **Timing Diagram of the In-line Timestamp and Received Frame**

When this in-line timestamp mode is not enabled, the AXI4-Stream Interface - Receive is unchanged from the current operation as described in the *LogiCORE IP Tri-Mode Ethernet MAC Product Guide* (PG051) [Ref 1].

Transmit Timestamp Ports

The captured timestamp is always presented out-of-band for 1-step and 2-step frame transmission, using a dedicated AXI4-Stream interface. The signal is defined in Table 5-5.

A timing diagram showing the operation of this interface follows Table 5-5.

Table 5-5: **AXI4-Stream Interface Ports - Transmit 2-Step Timestamp**

Name	Direction	Description
m_axis_tx_ts_data[127:0]	OUT	Bits[31:0] – Captured Timestamp Nanoseconds field Bits[79:32] – Captured Timestamp Seconds field Bits[95:80] – Original 16-bit Tag Field for the frame (from the Tag Field of the Command Field for the frame sent for transmission). Bits[127:96] – Reserved
m_axis_tx_ts_tvalid	OUT	AXI4-Stream Transmit Timestamp Data Valid from the MAC

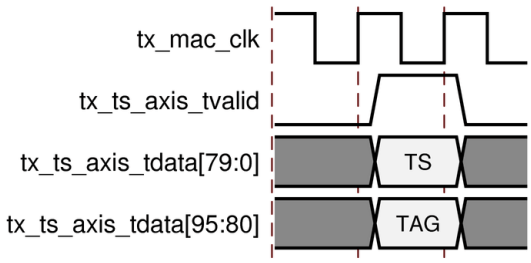


Figure 5-5: **Transmit 2-step Timestamp**

Providing the Command Field Out-of-Band

When not electing to provide the Command Field In-line with the frame sent for transmission, it must instead be provided out-of-band. This is achieved by expanding the size of the `tx_axis_tuser` signal from that already defined in the current AXI4-Stream Interface – Transmit (see the *LogiCORE IP Tri-Mode Ethernet MAC Product Guide* (PG051) [Ref 1]).

Selecting this mode of operation is through an AXI4-Lite addressable configuration bit (see Table 5-6).

The signal definition for this expanded `tx_axis_tuser` is defined in Table 5-6. A timing diagram showing the operation of this signal for normal frame transmission follows Table 5-6. To summarize, the Command Field bits of `tx_axis_tuser` must be valid on the same clock cycle when the first data word of the frame is sent for transmission.

Table 5-6: **tx_axis_tuser bitfield Definition**

Bits	Name	Description
tx_axis_tuser [0]	Underrun	AXI4-Stream user signal used to signal explicit underrun. This is defined in the *LogiCORE IP Ethernet 1000BASE-X PCS/PMA or SGMII Product Guide* (PG047) [Ref 2] and *LogiCORE IP Tri-Mode Ethernet MAC Product Guide* (PG051) [Ref 1].
tx_axis_tuser[63:1]	Reserved	Reserved for future use (all bits are ignored).
tx_axis_tuser[127:64]	Command Field	A 64-bit field as per the Command Field definition of Table 5-1. This field is only valid when you have elected not to use the In-Line option of Providing the Command Field In-line. (Otherwise all of these bits are ignored).

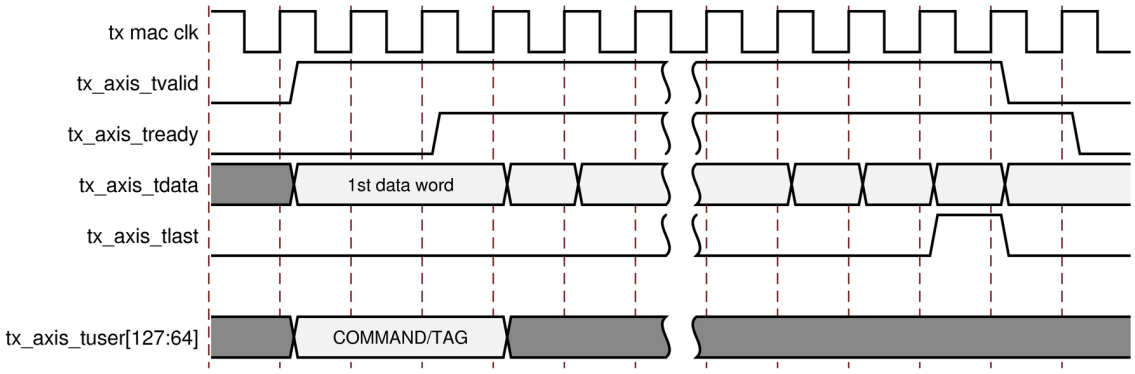


Figure 5-6: **AXI4-Stream Interface Timing – Out-of-band Command Field**

Providing the Command Field In-line

The Command Field can optionally be provided in-line with the frame sent for transmission using the TEMAC existing AXI4-Stream Interface – Transmit (see the *LogiCORE IP Tri-Mode Ethernet MAC Product Guide* (PG051) [Ref 1]).

Enabling this mode of operation is through an AXI4-Lite addressable configuration bit (see Table 5-8, bit 22).

When enabled, the 64-bit Command Field is passed to the MAC immediately before the start of the frame (in place of the Preamble field). See Figure 5-7.

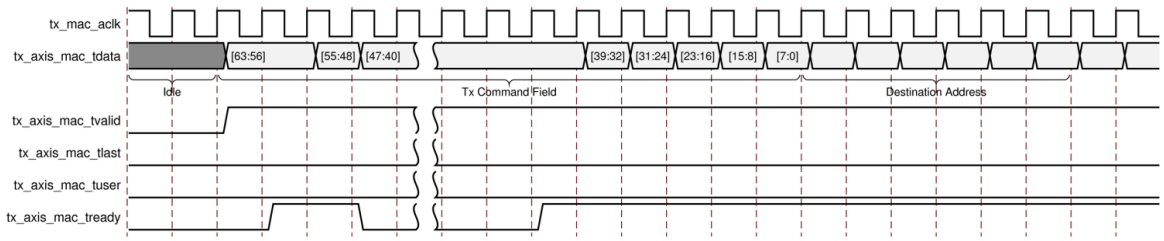


Figure 5-7: **Timing Diagram of the In-line Command Field and Frame Data for Transmission**

When this in-line Command Field mode is not enabled, the AXI4-Stream Interface – Transmit is unchanged from the current operation, and the Out Of Band Command Field method described previously must be used instead.

IEEE 1588 Mode Interfaces

Ports with common functionality are grouped as interfaces. The interfaces that are present in this mode are provided in Table 5-7.

Table 5-7: **Interfaces in 1588 Mode**

Interface Name	Old Interface name before version 6.0	Mode	Description
s_axi	s_axi	Slave	Interface used to configure the TEMAC
m_axis_rx	rx_axis_mac	Master	AXI Streaming interface for RX Data
s_axis_tx	tx_axis_mac	Slave	AXI Streaming interface for TX Data
m_axis_rx_ts	rx_axis_ts	Master	AXI Streaming interface for time stamping streaming packet
m_axis_tx_ts	tx_axis_ts	Master	AXI Streaming interface for time stamping streaming packet

IEEE1588 Configuration Registers

Enhancements to the Tri-Mode Ethernet MAC (TEMAC) Configuration/Status Registers

The following configuration registers/register bits have been added to the definitions in *LogiCORE IP Tri-Mode Ethernet MAC Product Guide* (PG051) [Ref 1].

Enhancement to the Transmitter Configuration Word (address 0x408)

Bit 22, previously reserved, is now defined in Table 5-8. All other bit definitions remain unchanged.

Table 5-8: **Transmitter Configuration Word (address 0x408)**

Bits	Default Value	Type	Description
21:0	N/A	RO	Reserved
22	0	RW	Inband 1588 Command Field Enable. When 0, the Command Field is provided out-of-band. When 1, the Command Field is provided in-Line. When the TEMAC does not include 1588 functionality, this bit is ignored because no Command Field is present.
24:23	N/A	RO	Reserved
25	0	RW	Interframe Gap Adjust Enable
26	0	RW	Half Duplex Enable
27	0	RW	VLAN Enable
28	1	RW	Transmit Enable
29	0	RW	In band FCS Enable
30	0	RW	Jumbo Frame Enable
31	0	RW	Reset

Enhancement to the Receiver Configuration Word 1 (address 0x404)

Bit 22, previously reserved, is now defined in Table 5-9. All other bit definitions remain unchanged.

Table 5-9: **Receiver Configuration Word 1 (address 0x404)**

Bits	Default Value	Type	Description
15:0	0	RW	Pause frame MAC Source Address[47:32]
21:16	N/A	RO	Reserved
22	0	RW	Inband 1588 Timestamp Enable. When 0, Timestamp is only provided out-of-band. When 1, the Timestamp is provided in-Line in addition to out-of-band. When the TEMAC does not include 1588 functionality, this bit is ignored because no timestamp is present.
24	0	RW	Control Frame Length Check Disable
25	0	RW	Length/Type Error Check Disable
26	0	RW	Half Duplex Enable
27	0	RW	VLAN Enable
28	1	RW	Receiver Enable
29	0	RW	In band FCS Enable
30	0	RW	Jumbo Frame Enable
31	0	RW	Reset

Transmitter Timestamp Adjust Control Register (address 0x41C)

This is an additional register, previously in reserved address space, present only when the TEMAC includes 1588 functionality. See Table 5-10.

Table 5-10: **Transmitter Timestamp Adjust Control Register (address 0x41C)**

Bits	Default Value	Type	Description
15:0	0xD8 (216 ns)	RW	Tx latency adjust value. This value is in units of nanoseconds and is initialized to reflect the delay following the timestamping position through the MAC (72 ns), 1000BASE-X FPGA logic (32 ns), and GTX transceiver (112 ns in its configured mode) components.
16	0	RW	Transmitter Timestamp Correction Enable. When 0, the transmitter timestamp is not adjusted. When 1, the transmitter timestamp is adjusted by the “Tx latency adjust value”
31:17	N/A	RO	Reserved

Enhancements to the Ethernet 1000BASE-X PCS/PMA or Core MDIO Configuration/Status Registers

The following vendor Specific Registers have be added to the MDIO PCS Address space when configured for 1000BASE-X operation. See the *LogiCORE IP Ethernet 1000BASE-X PCS/PMA or SGMII Product Guide* (PG047) [Ref 2].

Table 5-11: **1588 Control: Vendor Specific Register 19**

Bits	Default Value	Type	Description
15:4	N/A	RO	Reserved
3	1	RW	Timestamp correction enable. When 1, the Rx timestamp is adjusted to compensate for enabled PHY fixed and variable latencies. When 0, no adjustment is made to the timestamp.
2	1	RW	Fixed Rx PHY latency correction enable. When 1, the Rx timestamp is adjusted to compensate for fixed PHY latency by using the correction value specified in Table 5-12. When 0, no adjustment is made to compensate for fixed known latencies.
1	0	RO	Reserved
0	1	RW	Variable Rx transceiver latency correction enable. When 1, the Rx timestamp is adjusted to compensate for measurable variable transceiver latency (for 1000BASE-X this is the barrel shift position of the serial-to-parallel converter in the GTX transceiver PMA). This only varies when the core is initialized following a power-on, reset, or recovery from loss of synchronization; it will then remain constant for normal operation. When 0, no adjustment is made to compensate for measurable variable known latencies.

Table 5-12: **Rx PHY Fixed Latency: Vendor Specific Register 20**

Bits	Default Value	Type	Description
15:0	0xC8 (200 ns)	RW	Rx 1000BASE-X Fixed Delay in ns. This value is initialized to the known Rx latency from the serial wire input into the FPGA, through the transceiver fixed latency components (168 ns) and 1000BASE-X FPGA logic (32 ns) prior to the timestamping position.

Table 5-13: **Rx PHY Variable Latency: Vendor Specific Register 21**

Bits	Default Value	Type	Description
15:0	N/A	RO	Rx 1000BASE-X variable Rx Delay in UI. This value is measured within the core following Rx synchronization (for 1000BASE-X this is the barrel shift position of the serial-to-parallel converted in the transceiver PMA). This only varies when the core is initialized following a power-on, reset, or recovery from loss of synchronization; it then remains constant for normal operation.

Logic Utilization

Table 5-14 shows the logic utilization for the IEEE 1588 hardware timestamping solutions, inclusive of the TEMAC and 1000BASE-X helper cores. These numbers do not include the Statistic Counters option.

Table 5-14: **Logic Utilization for the IEEE1588 Hardware Timestamping Solutions**

Option	LUT as logic	LUT as Distributed RAM	LUT as shift register	Register as flip-flop	Register as Latch	RAMB36	RAMB18	DSPs	MMCMs	BUFGs
1-Step and 2-Step support	2630	330	50	3840	0	0	0	0	1	2
2-Step only support	2300	330	150	3340	0	0	0	0	1	2

Migrating and Upgrading

This appendix contains information about migrating a design from ISE® to the Vivado® Design Suite, and for upgrading to a more recent version of the IP core. For customers upgrading in the Vivado Design Suite, important details (where applicable) about any port changes and other impact to user logic are included.

Migrating to the Vivado Design Suite

For information on migrating from ISE tools to the Vivado Design Suite, see the *ISE to Vivado Design Suite Migration Guide* (UG911) [Ref 9].

Upgrading in the Vivado Design Suite

This section provides information about any changes to the user logic or port designations that take place when you upgrade to a more current version of this IP core in the Vivado Design Suite.

This AXI Ethernet IP core can be upgraded from an older version to the latest version. When the AXI Ethernet core is upgraded, there are a few ports renamed and a few parameters added. Table A-1 summarizes updated ports and any action to be taken. Parameters are automatically taken care of.

Table A-1: **New Port/Interface Name Changes**

New Port/ Interface Name	Reference to Older Name	Description	What to do
s_axis_txc	axi_str_txc	AXI4-Stream Transmit Control	Connect to DMA/FIFO Transmit Control interface.
s_axis_txd	axi_str_txd	AXI4-Stream Transmit Data	Connect to DMA/FIFO Transmit Data interface
m_axis_rxd	axi_str_rxd	AXI4-Stream Receive Data	Connect to DMA/FIFO Receive Data Interface
m_axis_rxs	axi_str_rxs	AXI4-Stream Receive Status	Connect to DMA Receive Status Interface

Table A-1: **New Port/Interface Name Changes *(Cont'd)***

New Port/ Interface Name	Reference to Older Name	Description	What to do
s_axis_tx_av	axi_str_avb_tx	AXI4-Stream AVB Transmit Data. This interface is present only in AVB mode.	Connect to AVB streaming TX master interface
m_axis_rx_av	axi_str_avb_rx	AXI4-Stream AVB Receive Data. This interface is present only in AVB mode.	Connect to AVB streaming RX slave interface
mdio	mdio or mdio_i, mdio_t, and mdio_o ports, and mdc port	MDIO interface to configure PHY.	Connect this to external PHY if required.
sgmii	txp, txn, rxp, rxn ports	Serial Gigabit Media Independent Interface.	Make this as an external interface.
sfp	txp, txn, rxp, rxn ports	In 1000BaseX mode, this interface connects to the SFP cage.	Make this as an external interface.
mgt_clk	mgt_clk_p, mgt_clk_n ports	Differential clock input for the serial transceiver.	Make this as an external interface.
ref_clk	refclk	Reference clock to the delayctrl in the case of RGMII mode and it is the independent clock source in case of 1000BaseX and SGMII.	Reconnect this to reference clock source.

Debugging

This appendix includes details about resources available on the Xilinx Support website and debugging tools.

Finding Help on Xilinx.com

To help in the design and debug process when using the AXI Ethernet core, the Xilinx Support web page (www.xilinx.com/support) contains key resources such as product documentation, release notes, answer records, information about known issues, and links for obtaining further product support.

Documentation

This product guide is the main document associated with the AXI Ethernet core. This guide, along with documentation related to all products that aid in the design process, can be found on the Xilinx Support web page (www.xilinx.com/support) or by using the Xilinx Documentation Navigator.

Download the Xilinx Documentation Navigator from the Design Tools tab on the Downloads page (www.xilinx.com/download). For more information about this tool and the features available, open the online help after installation.

Solution Centers

See the Xilinx Solution Centers for support on devices, software tools, and intellectual property at all stages of the design cycle. Topics include design assistance, advisories, and troubleshooting tips.

The Solution Center specific to the AXI Ethernet core is Xilinx Ethernet IP Solution Center.

Answer Records

Answer Records include information about commonly encountered problems, helpful information on how to resolve these problems, and any known issues with a Xilinx product. Answer Records are created and maintained daily ensuring that users have access to the most accurate information available.

Answer Records for this core can be located by using the Search Support box on the main Xilinx support web page. To maximize your search results, use proper keywords such as

- Product name
- Tool message(s)
- Summary of the issue encountered

A filter search is available after results are returned to further target the results.

Master Answer Record for the AXI Ethernet core

AR 54688

Contacting Technical Support

Xilinx provides technical support at www.xilinx.com/support for this LogiCORE™ IP product when used as described in the product documentation. Xilinx cannot guarantee timing, functionality, or support of product if implemented in devices that are not defined in the documentation, if customized beyond that allowed in the product documentation, or if changes are made to any section of the design labeled DO NOT MODIFY.

To contact Xilinx Technical Support:

1. Navigate to www.xilinx.com/support.
2. Open a WebCase by selecting the selecting the WebCase link located under Additional Resources.

When opening a WebCase, include:

- Target FPGA including package and speed grade.
- All applicable Xilinx Design Tools and simulator software versions.
- Additional files based on the specific issue might also be required. See the relevant sections in this debug guide for guidelines about which file(s) to include with the WebCase.

Note: Access to WebCase is not available in all cases. Log in to the WebCase tool to see your specific support options.

Debug Tools

There are many tools available to address AXI Ethernet core design issues. It is important to know which tools are useful for debugging various situations.

Vivado Lab Tools

Vivado® lab tools insert logic analyzer and virtual I/O cores directly into your design. Vivado lab tools also allow you to set trigger conditions to capture application and integrated block port signals in hardware. Captured signals can then be analyzed. This feature in the Vivado IDE is used for logic debugging and validation of a design running in Xilinx devices.

The Vivado logic analyzer is used with the logic debug IP cores, including:

- ILA 2.0 (and later versions)
- VIO 2.0 (and later versions)

See the *Vivado Design Suite User Guide: Programming and Debugging* (UG908) [Ref 10].

Reference Boards

Various Xilinx development boards support the AXI Ethernet core. The KC705 7 series FPGA evaluation board can be used to prototype designs and establish that the core can communicate with the system.

Hardware Debug

Hardware issues can range from link bring-up to problems seen after hours of testing. This section provides debug steps for common issues. The Vivado lab tools are a valuable resource to use in hardware debug. The signal names mentioned in the following individual sections can be probed using the Vivado lab tools for debugging the specific problems.

Ensure that all the timing constraints for the core were properly incorporated from the example design and that all constraints were met during implementation. Following is a list of some general checks.

- Does it work in post-place and route timing simulation? If problems are seen in hardware but not in timing simulation, this could indicate a PCB issue. Ensure that all clock sources are active and clean.
- If using MMCMs in the design, ensure that all MMCMs have obtained lock by monitoring the `locked` port.

- If your outputs go to 0, check your licensing.
- Different PHYs have different reset polarity. Check the reset polarity.

Interface Debug

AXI4-Lite Interfaces

Read from a register that does not have all 0s as a default to verify that the interface is functional. Output `s_axi_arready` asserts when the read address is valid, and output `s_axi_rvalid` asserts when the read data/response is valid.

If the interface is unresponsive, ensure that the following conditions are met:

- The `s_axi_aclk and aclk` inputs are connected and toggling.
- The interface is not being held in reset, and `s_axi_areset` is an active-Low reset.
- The interface is enabled, and `s_axi_aclken` is active-High (if used).
- The main core clocks are toggling and that the enables are also asserted.
- If the simulation has been run, verify in simulation and/or the Vivado lab tools capture that the waveform is correct for accessing the AXI4-Lite interface.

AXI4-Stream Interfaces

If data is not being transmitted or received, check the following conditions:

- If transmit `<interface_name>_tready` is stuck Low following the `<interface_name>_tvalid` input being asserted, the core cannot send data.
- If the receive `<interface_name>_tvalid` is stuck Low, the core is not receiving data.
- Check that the `aclk` inputs are connected and toggling.
- Check that the AXI4-Stream waveforms are being followed. See Figure 2-21, Figure 2-23, Figure 2-25, and Figure 2-26.
- Check core configuration.
- Add appropriate core specific checks.

Appendix C

Additional Resources and Legal Notices

Xilinx Resources

For support resources such as Answers, Documentation, Downloads, and Forums, see Xilinx Support.

For a glossary of technical terms used in Xilinx documentation, see the Xilinx Glossary.

References

Reference the change log for a list of the cores utilized in this design. The change log also identifies the version of the cores used and referenced throughout this document.

1. *LogiCORE™ IP Tri-Mode Ethernet MAC Product Guide* (PG051)
2. *LogiCORE IP Ethernet 1000BASE-X PCS/PMA or SGMII Product Guide* (PG047)
3. *7 Series FPGAs Clocking Resources User Guide* (UG472)
4. *Vivado® Design Suite User Guide: Designing IP Subsystems Using IP Integrator* (UG994)
5. *Vivado Design Suite User Guide: Designing with IP* (UG896)
6. *Vivado Design Suite User Guide: Logic Simulation* (UG900)
7. *Vivado Design Suite User Guide: Getting Started* (UG910)
8. *7 Series FPGAs Overview* (DS180)
9. *ISE® to Vivado Design Suite Migration Guide* (UG911)
10. *Vivado Design Suite User Guide: Programming and Debugging* (UG908)
11. *ARM® AMBA® AXI Protocol v2.0 Specification* (ARM IHI 0022C)
12. *ARM AMBA 4 AXI4-Stream Protocol v1.0 Specification* (ARM IHI 0051A)
13. *LogiCORE IP AXI Interconnect Product Guide* (PG059)
14. *Xilinx Constraints Guide* (UG625)
15. *Vivado Design Suite User Guide: Designing IP Subsystems using IP Integrator* (UG994)

Revision History

The following table shows the revision history for this document.

Date	Version	Revision
04/02/2014	6.1	• Added Correction Field update in 1588 functionality. • Updated Figure 4-5.
10/02/2013	6.0	• Added the Board tab to the Vivado IDE. • Updated the screen captures in Chapter 4. • Updated the Migrating and Upgrading appendix. • Added Shared Logic and Designer Assistance information throughout. • Modified Figure 2-1. • Added I/O Interfaces table to Chapter 2. • Added design parameters table to Chapter 3. • Updated most of the information in Chapter 4, Customizing and Generating the Core. Updated all screen captures.
06/19/2013	5.0	• Revision number advanced to 5.0 to align with core version number 5.0. • Added optional 1588 functionality. • Added support for SGMII over LVDS.
03/20/2013	1.0	Initial Xilinx release of this product guide. It replaces ds759.

Please Read: Important Legal Notices