AXI 1G/2.5G Ethernet Subsystem v7.2

Product Guide

Vivado Design Suite

PG138 (v7.2) June 5, 2024

Table of Contents

Chapter 1

Introduction

The AMD AXI Ethernet Subsystem 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 1000BASE-X interfaces to connect a media access control (MAC) to a physical-side interface (PHY) chip. It also provides an on-chip PHY for 1G/2.5G SGMII and 1000/2500 BASE-X modes. The MDIO interface is used to access PHY Management registers. This subsystem optionally enables TCP/UDP full checksum offload, VLAN stripping, tagging, translation, and extended filtering for multicast frames features.

Features

- Support for MII, GMII, RGMII, SGMII, and 1000BASE-X PHY interfaces Support for 1000BASE-X and SGMII over Select Input/Output (I/O) Low Voltage Differential Signaling (LVDS) for AMD UltraScale™/AMD UltraScale+™. For AMD Versal™ Adaptive SoC, the Async LVDS support is integrated with Advanced Input/Output (I/O) Wizard
- Support for pause frames for flow control
- Media Independent Interface Management (also called MII), is used for accessing the PHY registers
- Ethernet Audio Video Bridging (AVB) support
- AXI4-Stream transmit/receive interface
- Support for 2.5G Ethernet. This feature is enabled for the following devices:
- AMD Kintex™ 7, AMD Virtex™ 7 with GTH and GTX transceivers
- AMD Artix™ 7 devices with GTP and speed grade -2, -2L, and -3
- UltraScale, UltraScale+ devices with GTH and GTY transceivers, Versal Adaptive SoC with GTY/GTYP transceivers
- IEEE Standard 1588 support
- AXI4-Lite register interface

IP Facts

AMD LogiCORE™ IP Facts Table	
Subsystem Specifics	
Supported Device Family[1]	AMD Versal™ Adaptive SoC, AMD UltraScale+™, AMD Kintex™ UltraScale™, AMD Virtex™ UltraScale™, AMD Zynq™ 7000 SoC, 7 series FPGAs
Supported User Interfaces	AXI4-Lite, AXI4-Stream
Resources	Performance and Resource Utilization
Provided with Subsystem	
Design Files	Encrypted register transfer level (RTL)
Example Design	Verilog
Test Bench	Verilog
Constraints File	Not Provided
Simulation Model	Verilog and VHDL source HDL Model
Supported S/W Driver[2]	Standalone and Linux
Tested Design Flows[3]	
Design Entry	AMD Vivado™ Design Suite
Simulation	For supported simulators, see the *Vivado Design Suite User Guide: Release Notes, Installation, and Licensing* (UG973).
Synthesis	Vivado Synthesis
Support	
Release Notes and Known Issues	Master Answer Record: 54668
All Vivado IP Change Logs	Master Vivado IP Change Logs: 72775
Support web page	

Notes:

1. For a complete list of supported devices, see the AMD Vivado™ IP catalog.
2. Standalone driver details can be found on the AXI Ethernet Standalone Driver page. Linux driver support information is available from the AXI Ethernet Linux Driver page.
3. For the supported versions of third-party tools, see the *Vivado Design Suite User Guide: Release Notes, Installation, and Licensing* (UG973).

Chapter 2

Overview

The AXI Ethernet Subsystem can be added to the canvas in the AMD Vivado™ IP integrator block design. The subsystem can also be used in a register transfer level (RTL) flow when selected from the IP catalog in the Vivado Integrated Design Environment (IDE). The catalog allows customization, instantiation within a design, output product generation, behavioral simulation, design elaboration, synthesis and implementation, and bitstream generation for the target device. The AXI Ethernet Subsystem represents a hierarchical design block containing multiple infrastructure cores that are configured and connected during the system design session. Each of the infrastructure cores can also be added directly to a block design (outside of the AXI Ethernet Subsystem).

This subsystem provides additional functionality and ease of use related to Ethernet. Based on the configuration, this subsystem creates interface ports, instantiates required infrastructure cores, and connects these cores. Infrastructure cores for this subsystem are the AMD Tri-Mode Ethernet MAC (TEMAC) and 1G/2.5G Ethernet PCS/PMA or Serial Gigabit Media Independent Interface (SGMII) cores. Additional functionality is provided using the AXI Ethernet Buffer core.

For detailed specifications, see Chapter 3: Product Specification. See the change log for this subsystem for the core versions used with this design. All core documents can be downloaded from Xilinx.com.

Navigating Content By Design Process

AMD Adaptive Computing documentation is organized around a set of standard design processes to help you find relevant content for your current development task. All AMD Versal™ adaptive SoC design process Design Hubs and the Design Flow Assistant materials can be found on the Xilinx.com website. This document covers the following design processes:

- **Hardware, IP, and Platform Development:**

  Creating the PL IP blocks for the hardware platform, creating PL kernels, functional simulation, and evaluating the AMD Vivado™ timing, resource use, and power closure. Also involves developing the hardware platform for system integration. Topics in this document that apply to this design process include:

  - Port Descriptions

- Register Space
- Clocking
- Resets
- Customizing and Generating the Subsystem
- Chapter 6: Example Design

How to Use This Document

Some of the information in this document is identical or very similar for all modes of the subsystem. 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 physical-side (PHY) interface selected.

Feature Summary

In addition to the features listed in Features, the subsystem includes the following features:

- Full-duplex support (Half-duplex is not supported)
- IEEE 1588 Support:
    - 1000BASE-X and SGMII PHY timestamping for 1G and 2.5G modes of operation
    - Supports optional 1588 hardware timestamping for one-step and two-step when enabled with 1000BASE-X/SGMII PHY targeting AMD Versal™ Adaptive SoC with GTY/GTYP transceivers, AMD UltraScale™/AMD UltraScale+™ FPGAs with GTH and GTY transceivers, and 7 series FPGAs with GTX and GTH transceivers
    - The system timer provided to the subsystem and the consequential timestamping taken from it are available in one of two formats which are selected during subsystem generation
        - Time-of-Day (ToD) format: IEEE 1588-2008 format consisting of a 48-bit second field and a 32-bit nanosecond field
        - Correction Field format: IEEE 1588-2008 numerical format consisting of a 64-bit field representing nanoseconds multiplied by 2^{16} (see IEEE 1588 clause 13.3.2.7)
- Option to omit 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 4 K, 8 K, 16 K, or 32 KB TX and RX frame buffer memory
- Optional extended filtering for multicast frames
- Optional TX and RX statistics gathering
- Auto PAD and frame check sequence (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 subsystem
- Optional support for board-based I/O constraints generation
- Support for 1000BASE-X and SGMII over Select Input/Output (I/O) Low Voltage Differential Signaling (LVDS) for UltraScale/UltraScale+. For Versal Adaptive SoC the Async 1000BASE-X and SGMII over LVDS support is with the Advanced I/O Wizard IP integration. Refer to the 1G Core Family Support table in the *1G/2.5G Ethernet PCS/PMA or SGMII LogiCORE IP Product Guide* (PG047) for the list of supported devices.

Licensing and Ordering

This AMD LogiCORE™ IP module is provided at no additional cost with the AMD Vivado™ Design Suite under the terms of the End User License.

***Note*:** To verify that you need a license, check the License column of the IP Catalog. Included means that a license is included with the AMD Vivado™ Design Suite; Purchase means that you have to purchase a license to use the subsystem.

For more information about this subsystem, visit the AXI Ethernet Subsystem product web page.

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

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

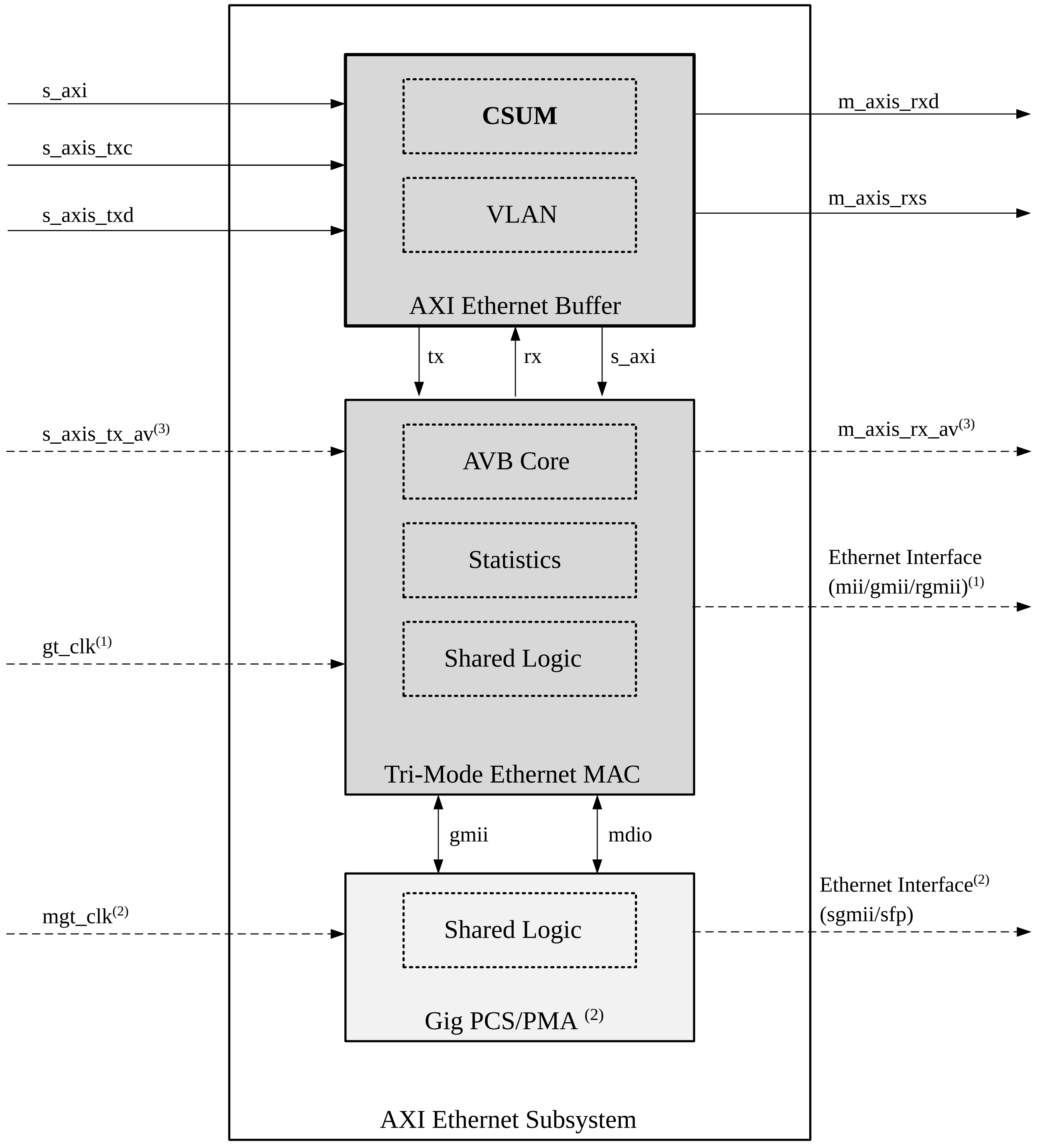
Product Specification

This chapter includes a description of the subsystem and details about the performance and resource utilization.

Functional Description

A high-level block diagram of the AXI Ethernet Subsystem is shown in the following figure.

Figure 1: **AXI Ethernet Subsystem Block Diagram**



(1) This interface is present only in MII, GMII or RGMII modes. It is not present in SGMII or 1000Base-X 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.

X14022

The subsystem provides an AXI4-Lite bus interface for a simple connection to the processor core to allow access to the registers. This AXI4-Lite slave interface supports single beat read and write data transfers (no burst transfers). 32-bit AXI4-Stream buses are provided for moving transmit and receive Ethernet data to and from the subsystem. These buses are designed to be used with an AXI Direct Memory Access (DMA) IP or AXI Multichannel Direct Memory Access (MCDMA) IP core, 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 required. The AXI4-Stream buses are described in Frame Transmission.

The PHY side of the subsystem 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 using the 1G/2.5G Ethernet PCS/PMA or SGMII module.

A 1000BASE-X PHY can be implemented directly in the FPGA using the 1G/2.5G Ethernet PCS/PMA or SGMII module configured in 1000BASE-X mode. This can be connected directly to an external optical module.

The AXI 1G/2.5G Ethernet subsystem can be configured in non-processor mode where the buffer module is not present. In this mode, all of the ports and interfaces are directly connected to the TEMAC and the PCS PMA. Non-processor mode can be enabled only for SGMII or 1000BASE-X.

The supported physical interface types are:

- **GMII:** The Gigabit Media Independent Interface (GMII) is defined by the IEEE 802.3 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 IEEE 802.3 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).
- **Both:** This feature is to support switching between the 1000BASE-X and SGMII standard. see the *1G/2.5G Ethernet PCS/PMA or SGMII LogiCORE IP Product Guide* (PG047).

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 infrastructure cores are used by the AXI Ethernet Subsystem:

- Tri-Mode Ethernet MAC core
- 1G/2.5G Ethernet PCS/PMA or SGMII core
- AXI Ethernet Buffer: Features such as 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 infrastructure 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 subsystem clock domain.

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 offloading in the subsystem using the C_TXCSUM and C_RXCSUM parameters. Including the checksum offload functions are a trade-off between using more FPGA resources and achieving 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 subsystem, using the AXI4-Stream Control and AXI4-Stream Status interfaces. These interfaces are described in AXI4-Stream Interface and the tables in this section show the checksum offload fields.

The use of the TCP/ UDP checksum offload function requires that the AXI DMA is connected to the AXI Ethernet Subsystem through the AXI4-Stream Control and AXI4-Stream data interfaces. SeeMapping 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 Subsystem.

There is a special case for checksums of UDP datagrams as specified in . 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 subsystem, 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 Subsystem 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 the following figures in this section. In these figures, the conditions shown in redare used by the hardware to determine if the TCP/UDP checksum and/or the IPv4 Header checksum is calculated.

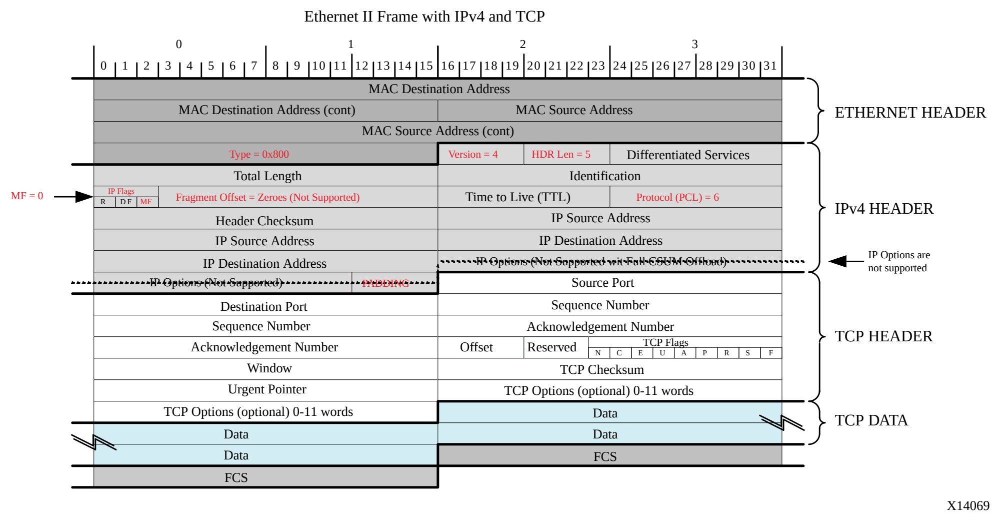
It is possible for the IPv4 Header checksum to be calculated even though the TCP/UDP checksum is not calculated. This can occur if the frame is Ethernet II or SNAP with an IPv4 Header that is five words in length, the IP flags are set to 0, and the fragment offset is set to 0 (the protocol field can 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.

The following figure 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: **Ethernet II Frame with IPv4 and TCP**

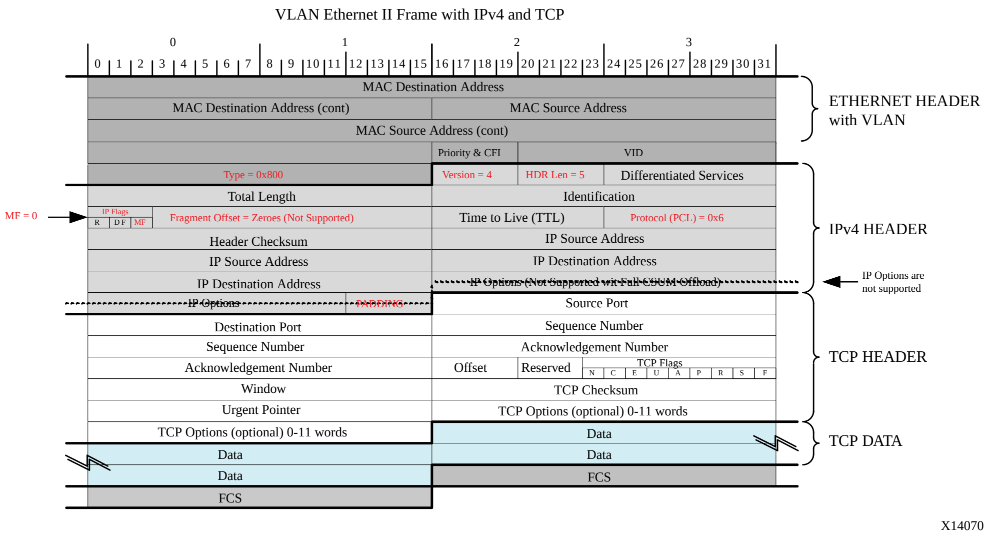


The following figure shows a VLAN Ethernet II frame with IPv4 and TCP. The following conditions must be met for the IPv4 Header checksum and TCP checksum to be calculated:

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

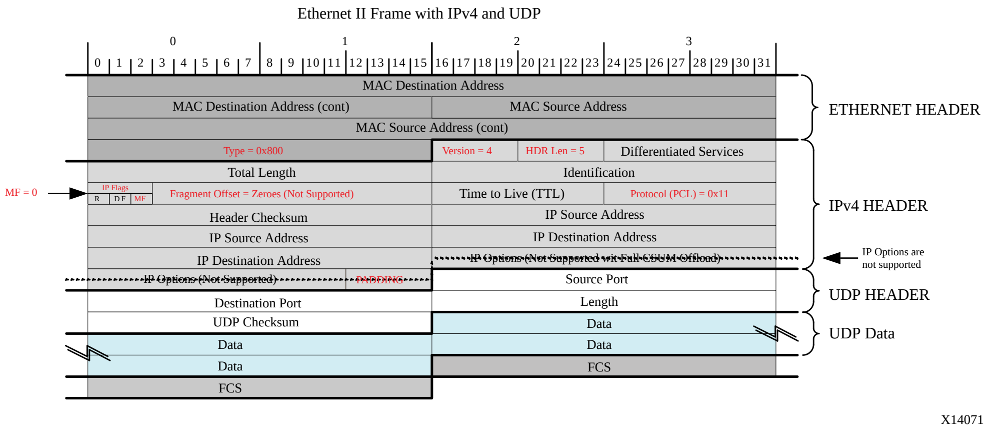


The following figure shows an Ethernet II frame with IPv4 and UDP. The following conditions must be met for the IPv4 Header checksum and UDP checksum to be calculated:

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

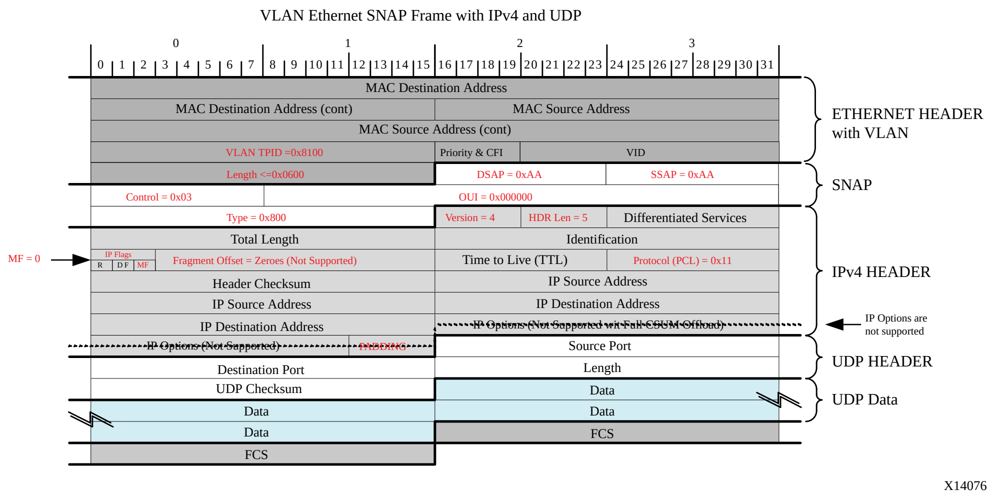


The following figure shows a VLAN Ethernet II frame with IPv4 and UDP. The following conditions must be met for the IPv4 Header checksum and UDP checksum to be calculated:

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



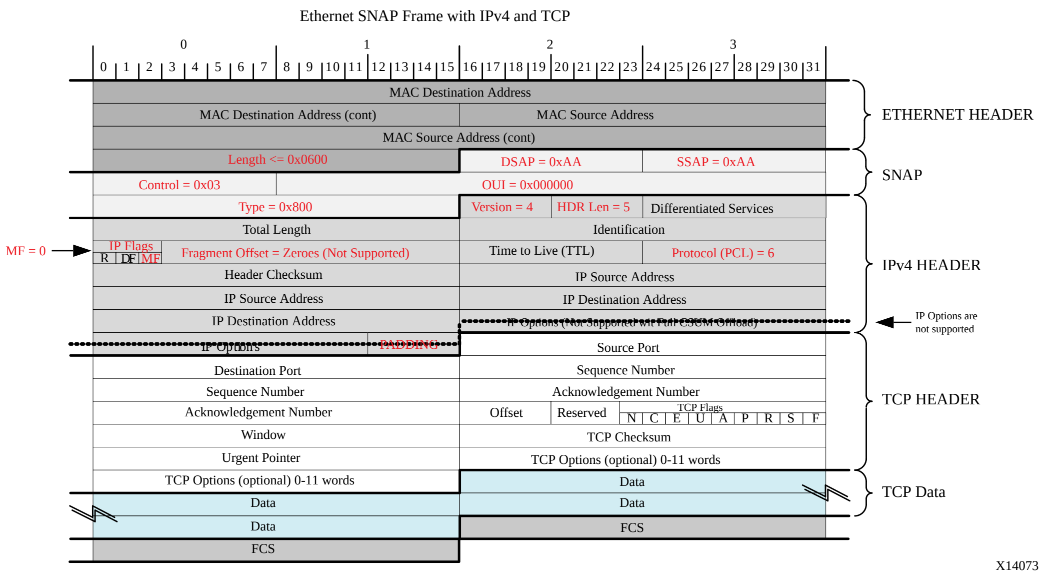
The following figure shows a SNAP frame with IPv4 and TCP. The following conditions must be met for the IPv4 Header checksum and TCP checksum to be calculated.

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

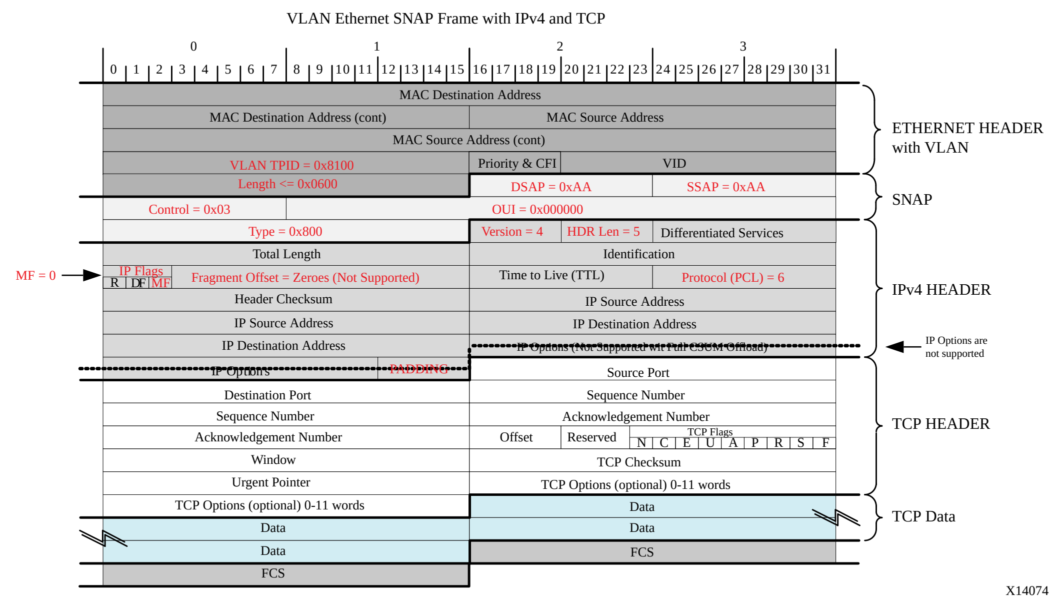


The following figure shows a VLAN SNAP frame with IPv4 and TCP. The following conditions must be met for the IPv4 Header checksum and TCP checksum to be calculated:

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

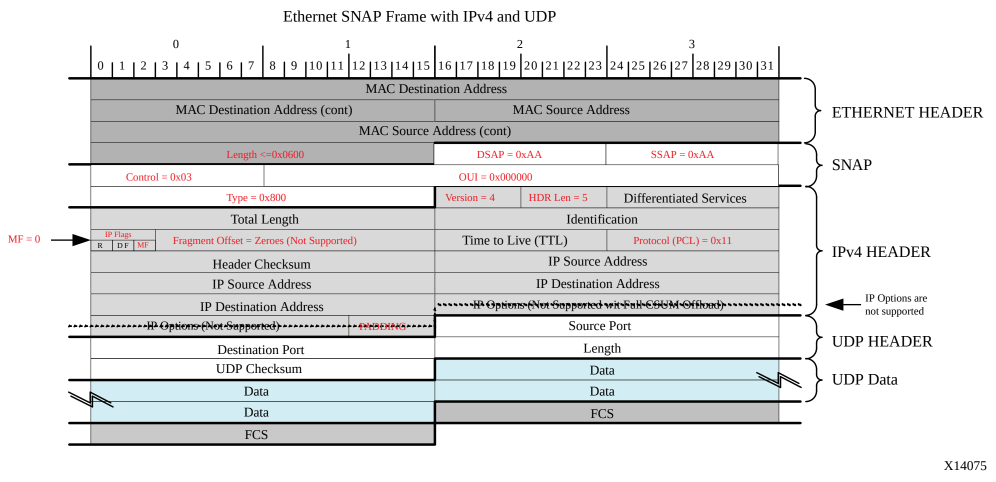


The following figure shows a SNAP frame with IPv4 and UDP. The following conditions must be met for the IPv4 Header checksum and UDP checksum to be calculated:

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

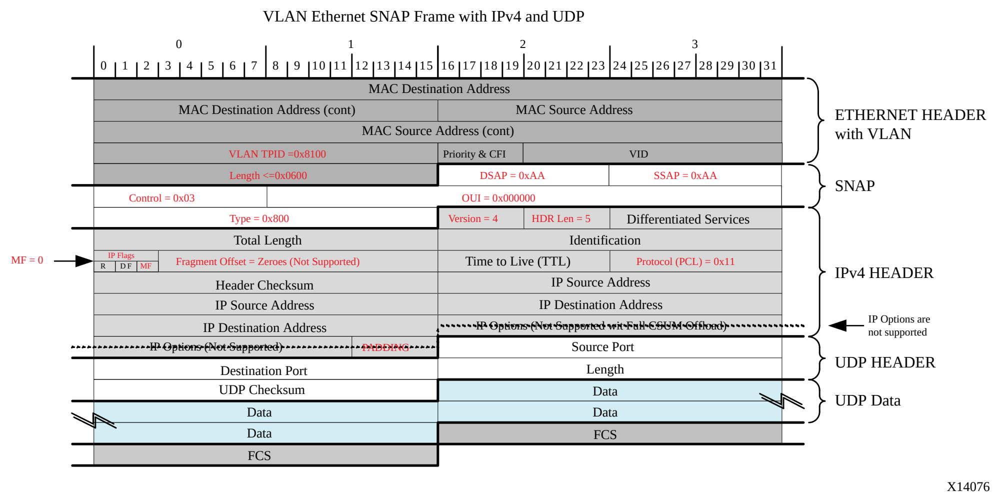


The following figure shows a VLAN SNAP frame with IPv4 and UDP. The following conditions must be met for the IPv4 Header checksum and UDP checksum to be calculated:

- 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 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 zeros 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 subsystem 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 subsystem, 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 subsystem 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 TEMAC Transmit Control (TC) register bit 29 (Table 46: TEMAC Transmit Configuration Register (0x408)).

Virtual LAN (VLAN) Frames

When transmitting VLAN frames (if enabled by the TC register bit 27, Table 46: TEMAC Transmit Configuration Register (0x408)) 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 46: TEMAC Transmit Configuration Register (0x408)) and you attempt to transmit a frame that exceeds the maximum legal length, the subsystem 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 Subsystem 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 Interrupt Status register bit 3 in Interrupt Status Register).

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 45: TEMAC Receive Configuration Word1 (RCW1) Register (0x404)) and Length/Type filed error checking is enabled (RCW1 register bit 25 Table 45: TEMAC Receive Configuration Word1 (RCW1) Register (0x404)), 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 45: TEMAC Receive Configuration Word1 (RCW1) Register (0x404)) 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 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 45: TEMAC Receive Configuration Word1 (RCW1) Register (0x404)). This is the basic native VLAN support provided by the TEMAC core. For more information about extended VLAN functions, see Extended VLAN Support.

Maximum Frame Length and Jumbo Frames

The maximum length of a frame specified in the IEEE Std 802.3-2008 specification is 1,518 bytes for non-VLAN tagged frames. VLAN tagged frames can be extended to 1,522 bytes. When jumbo frame handling is disabled (RCW1 register bit 30 Table 45: TEMAC Receive Configuration Word1 (RCW1) Register (0x404)) and a received frame exceeds the maximum legal length, the frame is dropped and a Receive Reject interrupt is activated (see bit 3 in Interrupt Status Register). 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 45: TEMAC Receive Configuration Word1 (RCW1) Register (0x404)) 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 Interrupt Status Register).

- 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). Using the Address Filters 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 Interrupt Status Register 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 Frame Filter Control Register. While the Filter Mask Index is set, the Address Filter register can be set accessed as shown in the following figure.

Figure 10: **Address Filter Access**

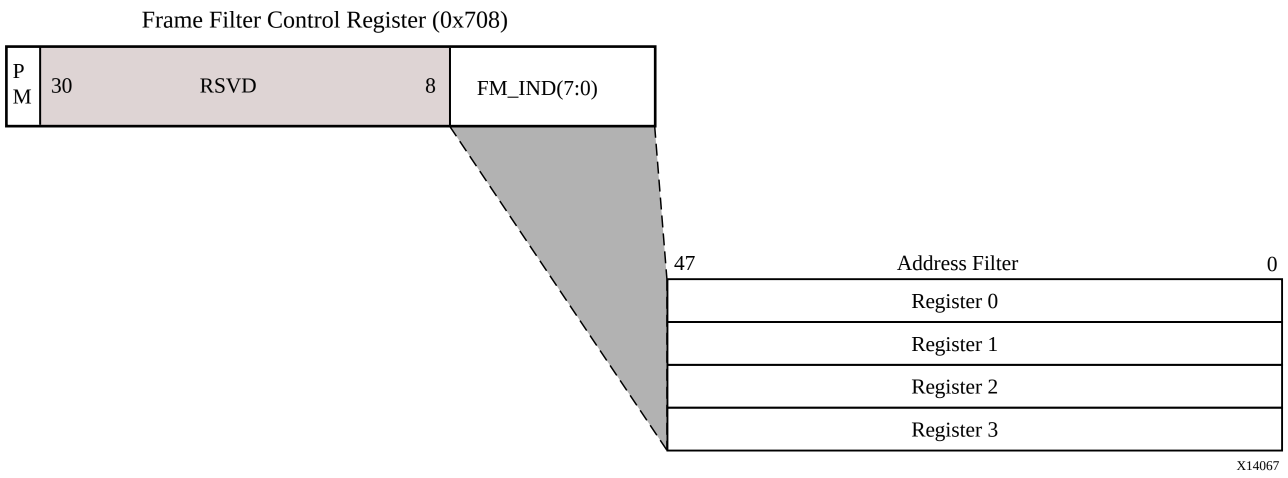
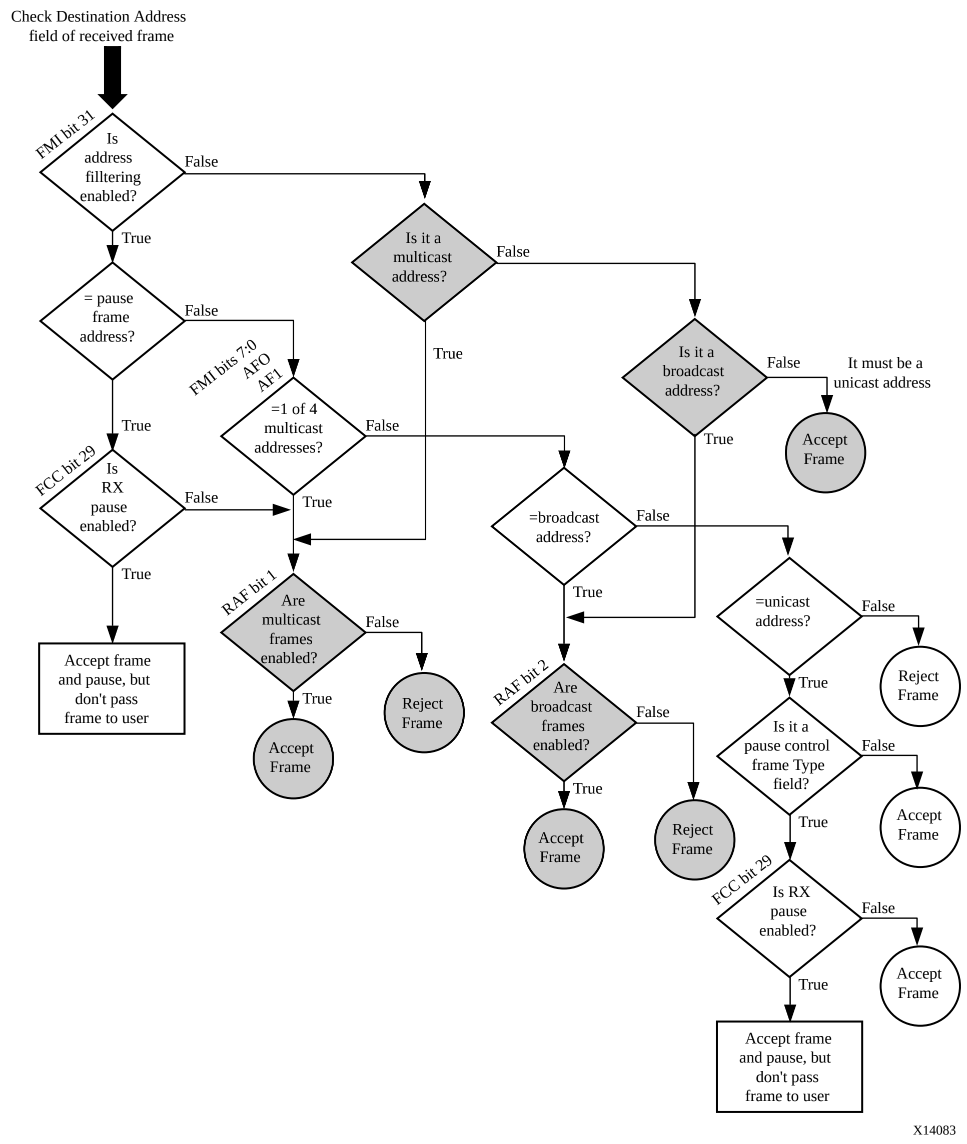


Figure 11: **Receive Address Basic Filtering**


Check Destination Address field of received frame
FMI bit 31
Is address filltering enabled?
False
True
Is it a multicast address?
False
True
= pause frame address?
False
True
FMI bits 7:0 AFO AF1
=1 of 4 multicast addresses?
False
True
Is it a broadcast address?
False
True
It must be a unicast address
Accept Frame
FCC bit 29
Is RX pause enabled?
False
True
=broadcast address?
False
True
=unicast address?
False
True
Reject Frame
RAF bit 1
Are multicast frames enabled?
False
True
Reject Frame
Accept Frame
Accept frame and pause, but don't pass frame to user
RAF bit 2
Are broadcast frames enabled?
False
True
Accept Frame
Reject Frame
Is it a pause control frame Type field?
False
True
Accept Frame
FCC bit 29
Is RX pause enabled?
False
True
Accept Frame
Accept frame and pause, but don't pass frame to user
X14083

Extended Multicast Address Filtering Mode

General

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). More multicast address values can be used by operating in promiscuous mode with software application filtering.

Extended multicast filtering is included at build time by setting the C_MCAST_EXTEND parameter to 1. This provides additional logic for address filtering. 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 mode to force it to pass all multicast frames through to the extended multicast address filtering logic.

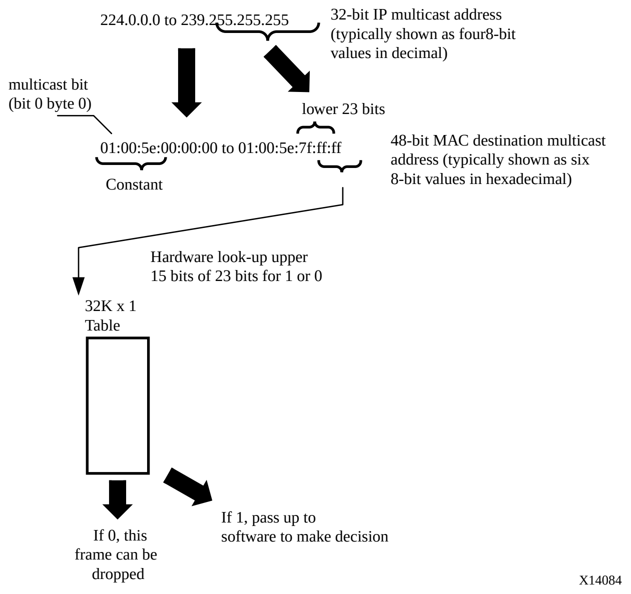
TEMAC in promiscuous mode will also pass through all unicast address frames. To avoid increasing the processor load for unicast address filtering, additional unicast address filtering has to be added to the extended multicast address filtering logic. Ensure that the TEMAC core is in promiscuous receive 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 taken 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 the following figure. 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 12: **Mapping IP Multicast Addresses to MAC Multicast Addresses**


224.0.0.0 to 239.255.255.255
32-bit IP multicast address (typically shown as four8-bit values in decimal)
multicast bit (bit 0 byte 0)
lower 23 bits
01:00:5e:00:00:00 to 01:00:5e:7f:ff:ff
48-bit MAC destination multicast address (typically shown as six 8-bit values in hexadecimal)
Constant
Hardware look-up upper 15 bits of 23 bits for 1 or 0
32K x 1 Table
If 0, this frame can be dropped
If 1, pass up to software to make decision
X14084

When a multicast address frame is received while extended multicast filtering is enabled, the subsystem 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, 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. Receive VLAN Tag Register shows the Receive VLAN Tag register bit definitions and Unicast Address Word Lower Register 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.

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.

Selecting **Extended Multicast address filtering** in AMD Vivado™ IDE allows to make decisions about the destination address without accessing the address from within the receive AXI4-Stream Data transfer. When using an AMD AXI DMA core, the information needed for filtering is in the buffer descriptor. Using this information, a decision can then be made regarding accepting or rejecting the frame without accessing the data buffer itself, which reduces memory access and buffer indexing overhead.

Flow Control

The flow control function is defined by IEEE Std 802.3-2008 Clause 31. The subsystem 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 in the Tri-Mode Ethernet MAC LogiCORE IP Product Guide (PG051).

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 overloaded interface can transmit a pause control frame to request that the link partner (the device connected on the other end of the Ethernet) stops transmitting for a defined period of time.

Transmitting a Pause Control Frame

For the AXI Ethernet Subsystem, a pause frame transmission can be initiated by writing a pause value to the Transmit Pause Frame Register while transmit pause processing is enabled (FCC register bit 30 in 1; see the Tri-Mode Ethernet MAC LogiCORE IP Product Guide (PG051).

Requesting a pause frame transmission does not interrupt a transmission in progress; 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 45: TEMAC Receive Configuration Word1 (RCW1) Register (0x404)).

Receiving a Pause Control Frame

When an error free frame is received by the subsystem, it examines the following information:

- The *destination address field* is compared to the pause control address and the configured unicast address.
- The *Length/Type field* is compared against the control type code (0x8808).
- 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, see the Tri-Mode Ethernet MAC LogiCORE IP Product Guide (PG051), the frame is ignored by the flow control logic and is passed to you. If the frames pass all three 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.

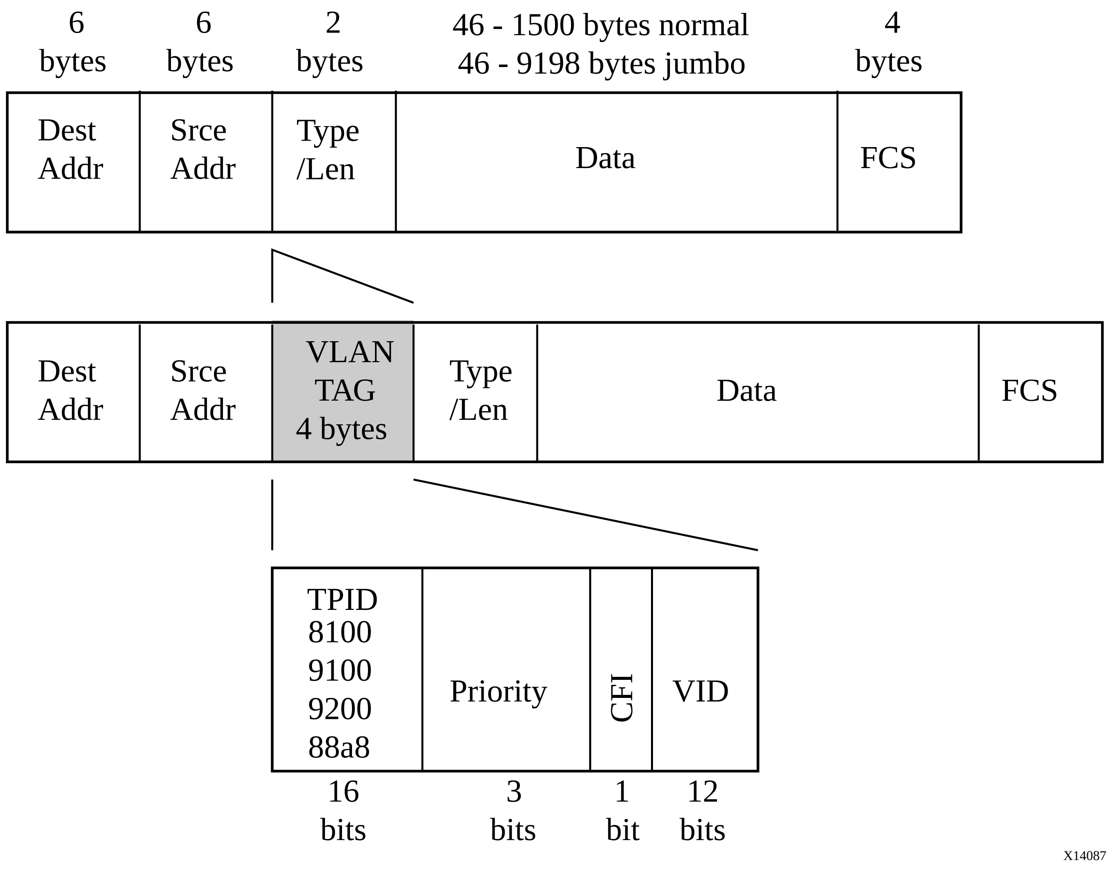
A Receive Reject interrupt is activated (see bit 28 in Interrupt Status Register), and the frame is not passed up to software. If the transmitter is paused and a second pause frame is received, the current pause value of the transmitter is replaced with the new pause value received, including a possible 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 which extends the overall frame by four bytes. The VLAN TAG field is further broken down into additional fields as shown in the following figure.

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



The TEMAC core provides basic VLAN support that can be enabled or disabled independent of Extended VLAN Support. 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

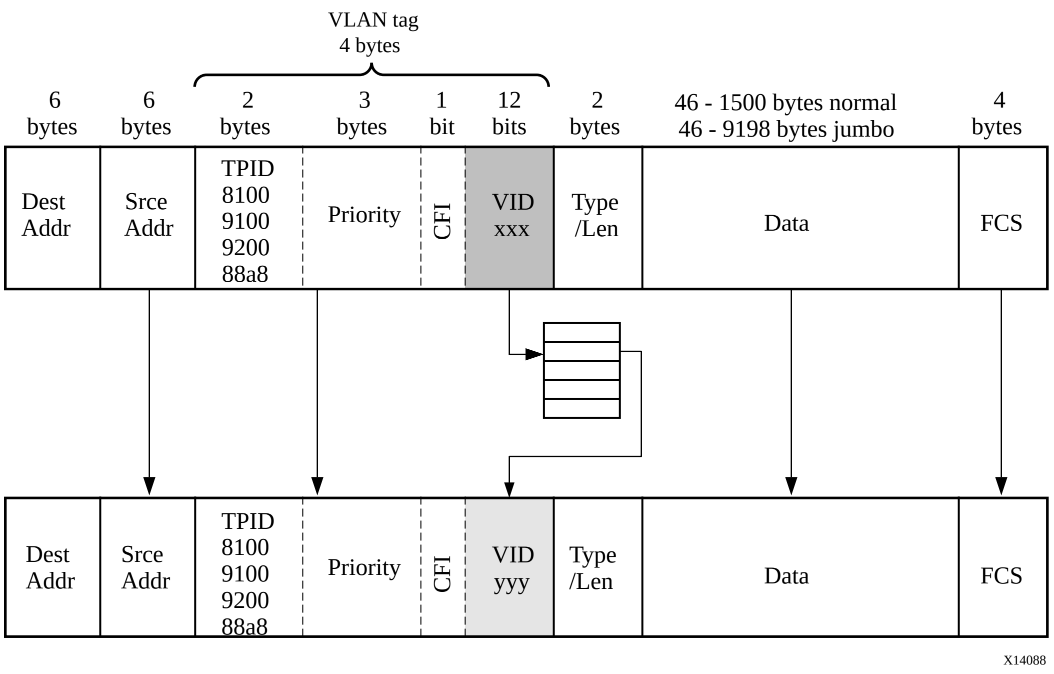
VLAN tags within one frame (referred to as double tagging, triple tagging). Additionally, some common operations are performed on VLAN frames that can be offloaded from software to hardware to reduce processor utilization. Some of these tasks, translation, stripping, and auto tagging, are available when the extended VLAN support is included in the TEMAC 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 the functions must be enabled at runtime by setting the New Functions enable bit (bit 11) of the Reset and Address Filter Register.

VLAN Translation

VLAN translation enables the AXI Ethernet Subsystem 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 subsystem in either the transmit or receive direction. See the following figure.

Figure 14: **VLAN VID Translation**



The TEMAC core recognizes neither transmission nor reception. 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 1,518 to 1,522 bytes. Additionally, multiple tagging is also not supported because of the even larger frame sizes.

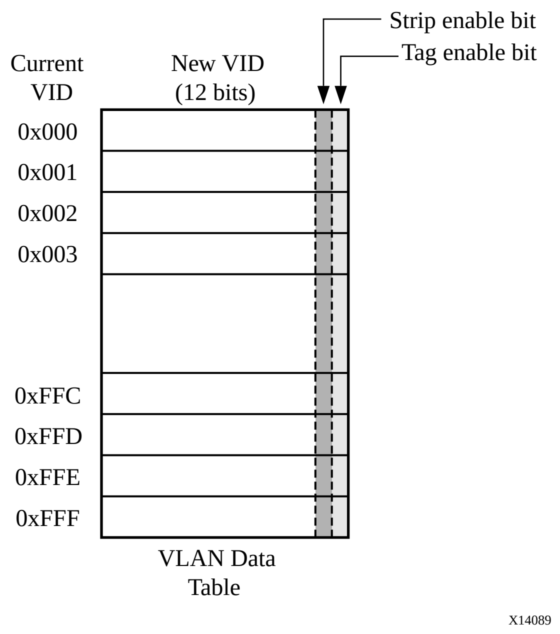
To support multiple VLAN tagging and the use of TPID values other than 0x8100 in the outer tag, jumbo frame mode must be used with basic VLAN mode disabled. Using this setting eliminates automatic invalidating (by the TEMAC core) of any frames that would be too large for *normal* frame sizes. To use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

Transmit Path

When transmitting frames, the outgoing frame is detected as a VLAN frame by recognizing a VLAN TPID in the Type/Length field and comparing it against user-defined values in the VLAN TPID Word 0 Register and VLAN TPID Word 1 Register. 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 to supply a replacement VID value which is substituted into the outgoing frame. See the following figure.

Figure 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.

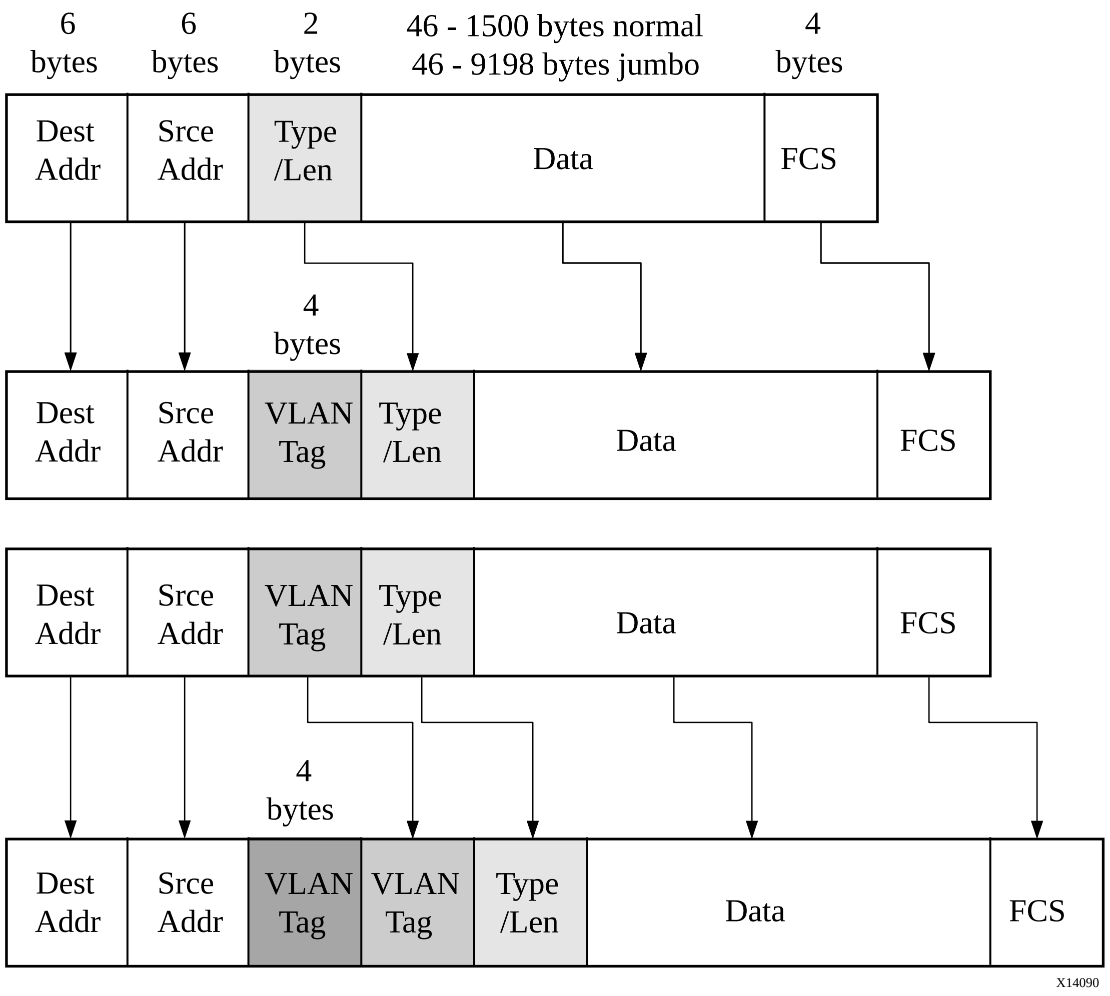
Receive Path

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 core to insert a predefined VLAN tag in select Ethernet frames as they pass through in either the transmit or receive direction.

Figure 16: **VLAN Tagging**



One VLAN tag is added depending on the mode of operation:

- Non-VLAN frames get one VLAN tag added to become single VLAN tagged frames.
- VLAN tagged frames receive an extra 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 1,522 bytes are 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 is used to select one of four VLAN tagging modes and the Transmit VLAN Tag Register and Receive VLAN Tag Register is used to hold the VLAN tag value that is inserted. The four VLAN tagging modes which are selectable at runtime are:

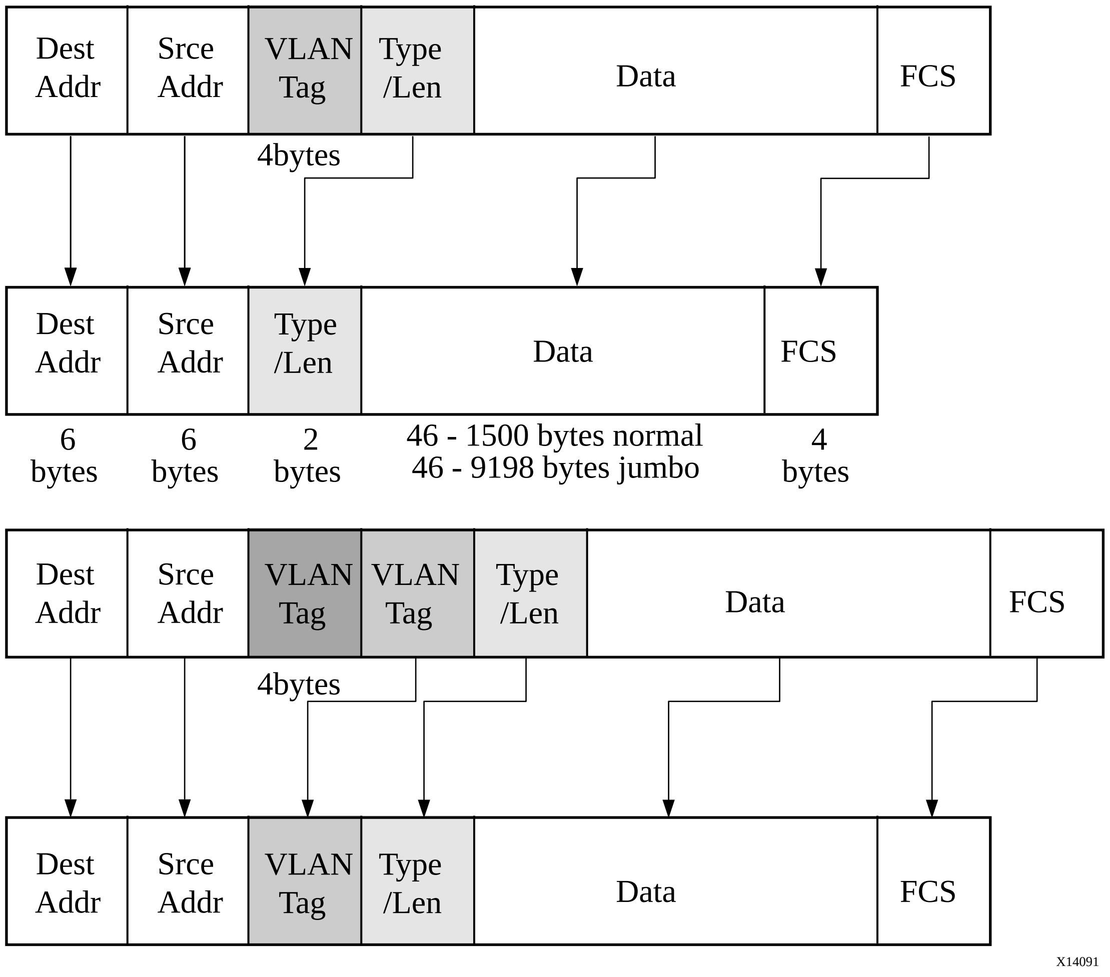
- Do not add tags to any frames.
- Add one tag to all frames.
- Add one tag only to frames that are already VLAN tagged.
- Add one tag only to select frames that are already VLAN tagged based on VID value.

The fourth mode requires a method for specifying which tagged frames should receive an additional VLAN tag. TheTEMAC Receive Configuration Word 0 Register and Receive VLAN Data Table are used for this purpose. A value of 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 the FCS field value is not 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 core to remove a VLAN tag in select Ethernet frames as they pass through the AXI Ethernet Subsystem in either the transmit or receive direction.

Figure 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 TEMAC core does not check to see how many VLAN tags it already has (if there are four tags, the 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 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 and Receive VLAN Data Table 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.

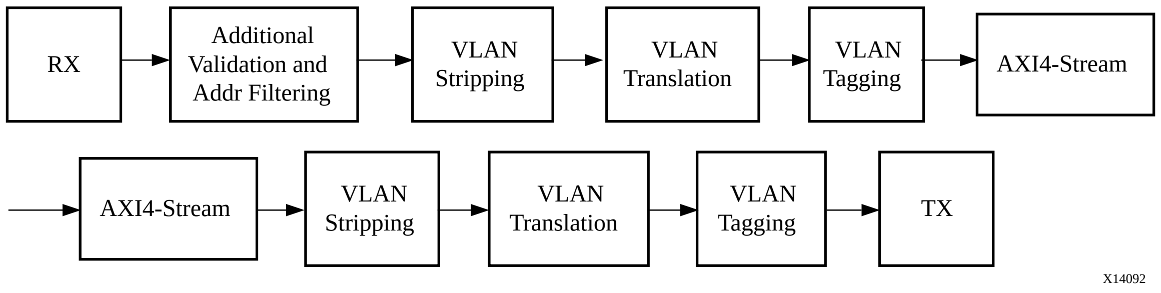
Again, transmit In-Band FCS mode is not allowed and receive FCS stripping is required when using VLAN stripping because the FCS field value would not be correct for the frame with the VLAN tag removed. Although receive stripping is enabled, any padding, if present, is not stripped due to the TYPE/LENGTH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

Order of Extended 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 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 internal or external to the FPGA. In SGMII and 1000BASE-X modes, the FPGA contains a single PHY. 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 *1G/2.5G Ethernet PCS/PMA or SGMII LogiCORE IP Product Guide* (PG047).

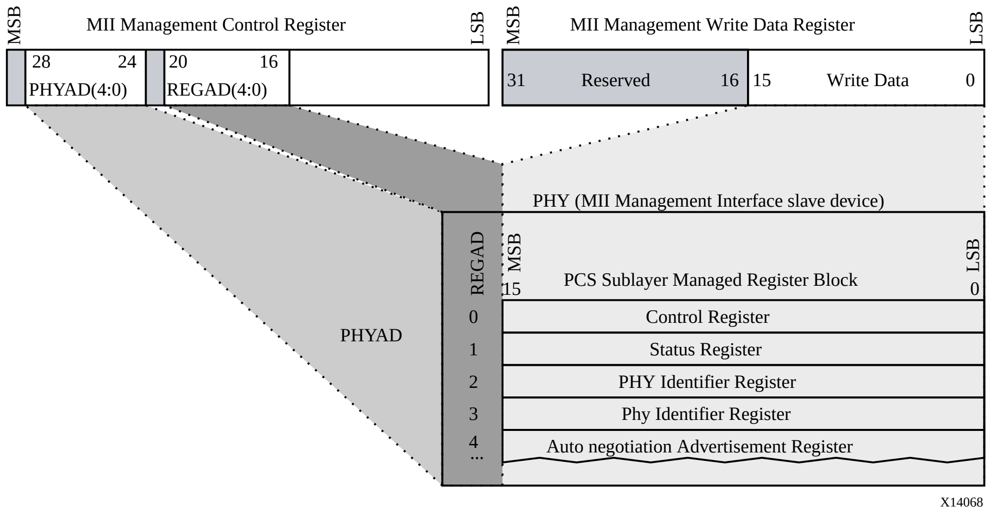
IMPORTANT! *Prior to any MII Management accesses, the MDIO Setup Register must be written with a valid CLOCK_DIVIDE value and the MDIOEN bit must be set.*

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 defined 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 the following figure.

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 1G/2.5G Ethernet PCS/PMA or SGMII registers, the PHYAD should match that value set by the parameter C_PHYADDR.

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


MSB
MII Management Control Register
LSB
28
24
PHYAD(4:0)
20
16
REGAD(4:0)
MSB
MII Management Write Data Register
LSB
31
Reserved
16
15
Write Data
0
PHY (MII Management Interface slave device)
REGAD
MSB
15
PCS Sublayer Managed Register Block
LSB
0
PHYAD
0
Control Register
1
Status Register
2
PHY Identifier Register
3
Phy Identifier Register
4
Auto negotiation Advertisement Register
X14068

The following table provides an example of a PHY register write through the MII Management Interface.

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

Register	Access	Value	Activity
MIIM Write Data register	WO	0x0000ABCD	Write the value that is written to the PHY register (0xABCD in this case).

Table 2: **Example of a PHY Register Write Through the MII Management Interface *(cont'd)***

Register	Access	Value	Activity
MII Management Control register	WO	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	RO	0x01024880	Poll the MII Management Control register bit 7. When set to 1, the data has been written.

Table 3: **Example of a PHY Register Read Through the MII Management Interface**

Register	Access	Value	Activity
MII Management Control register	WO	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	RO	0x01028880	Poll the MII Management Control register bit 7. When set to 1, the read data is available.
MII Management Read Data register	RO		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 the subsystem. 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 Serial GMII Specification v1.8.

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 Subsystem.

The results of the SGMII auto-negotiation can be read from the SGMII Management Auto-Negotiation Link Partner Ability Base register (Table 58: Management Auto-Negotiation Link Partner Ability Base Register (Register 5)). The speed of the subsystem should then be set to match.

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 54: Management Status Register (Register 1)).
2. Using the auto-negotiation complete interrupt (Interrupt Status Register and SGMII Management Auto-Negotiation Interrupt Control register, Table 63: Management Auto-Negotiation Interrupt Control Register (Register 16)).

When placed into loopback, data is routed from the transmitter to the receiver path at the last possible point in the PCS/PMA sublayer. 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.

SGMII over LVDS

The SGMII can also be implemented over LVDS. For more details on this mode, refer to the *1G/2.5G Ethernet PCS/PMA or SGMII LogiCORE IP Product Guide* (PG047). When the core is configured in the Include Shared Logic in Core mode, the input differential reference clock frequency can be selected from 125 MHz, 156.25 MHz and 625 MHz for AMD UltraScale™ devices.

In UltraScale devices, for the SGMII interface, an option is provided to choose between Synchronous SGMII over LVDS and Asynchronous SGMII over LVDS using the parameter, EnableAsyncSGMII.

The synchronous SGMII over LVDS solution uses component mode I/Os such as ISERDES, OSERDES, IDELAY, ODELAY components whereas the asynchronous implementation uses native mode HSSIO components such as BITSLICE and BITSLICE_CONTROL.

***Note*:** The Asynchronous 1000BASE-X/SGMII over LVDS implementation can fully support synchronous SGMII interfaces.

When using SGMII over LVDS in AMD UltraScale+™ or AMD Versal™ devices or using the Asynchronous SGMII over LVDS solution in AMD UltraScale™ devices, refer to the *1G/2.5G Ethernet PCS/PMA or SGMII LogiCORE IP Product Guide* (PG047) for port descriptions (especially for RIU interface, `dly_rdy` , `vtc_rdy` ports) and constraints relating to the design in the "Asynchronous 1000BASE-X/SGMII over LVDS" section in the *1G/2.5G Ethernet PCS/PMA or SGMII LogiCORE IP Product Guide* (PG047). For Versal devices, Asynchronous SGMII over LVDS is implemented using Advanced I/O Wizard.

1000BASE-X PCS/PMA

PCS/PMA

The Physical Coding Sublayer (PCS) for 1000BASE-X operation is defined in IEEE 802.3 clauses 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 deserialization) of code-groups for transmission (and reception) on the underlying serial Physical Medium Dependent (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 Gigabit Transceiver (GT).

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 gigabit interface converter (GBIC) or small form-factor pluggable (SFP) optical transceiver, which should be connected directly to the ports of the GT 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 58: Management Auto-Negotiation Link Partner Ability Base Register (Register 5)). The speed should be set to 1G and Duplex bit to Full duplex.

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 54: Management Status Register (Register 1)).
2. By using the auto-negotiation complete interrupt (and 1000BASE-X Management Auto-Negotiation Interrupt Control register, Table 63: Management Auto-Negotiation Interrupt Control Register (Register 16)).

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 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 53: Control Register (Register 0)).

1000BASE-X over LVDS

The 1000BASE-X can also be implemented over LVDS Mode. Refer to the *1G/2.5G Ethernet PCS/PMA or SGMII LogiCORE IP Product Guide* (PG047). In this mode, the locations tab is enabled. You have to select bitslice locations in this tab.

When using 1000BASE-X over LVDS, refer to the *1G/2.5G Ethernet PCS/PMA or SGMII LogiCORE IP Product Guide* (PG047) for port descriptions (especially for RIU interface, `dly_rdy`, `vtc_rdy` ports) and constraints relating to the design in the "Asynchronous 1000BASE-X/SGMII over LVDS" section in the *1G/2.5G Ethernet PCS/PMA or SGMII LogiCORE IP Product Guide* (PG047).

2.5G Data Rate Support

The 2.5G data rate support has been added as an option to the AXI 1G/2.5G Ethernet Subsystem. When you select this option, only SGMII and 2500BASE-X options are available for `phy_type`. The 2.5G data rate is an incremental speed upgrade mode of the 1 Gb/s data rate mode.

Generating Subsystem for 2.5G Operation

To configure the subsystem with 2.5G support, open the AXI 1G/2.5G Ethernet subsystem in Vivado IDE and perform the following steps:

1. In the Physical Interface tab, select the Ethernet speed as 2.5 Gb/s. 2.5G support is not available for AMD Artix™ 7 devices with speed grades other than -2, -2L, and -3.
2. Select the required `phy_type` from the physical interface selection.

For more details, see the Tri-Mode Ethernet MAC LogiCORE IP Product Guide (PG051). and *1G/2.5G Ethernet PCS/PMA or SGMII LogiCORE IP Product Guide* (PG047).

Standards

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

- The Gigabit Media Independent Interface (GMII) is defined by the IEEE 802.3 specification; it can provide support for Ethernet operation at 10 Mb/s, 100 Mb/s, and 1 Gb/s speeds.
- The Media Independent Interface (MII) is defined by the IEEE 802.3 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.
- IEEE 802.1AS Supports clock master functionality, clock slave functionality, and the Best Master Clock Algorithm (BMCA).
- IEEE 802.1Qav Supports arbitration between different priority traffic and implements bandwidth policing.

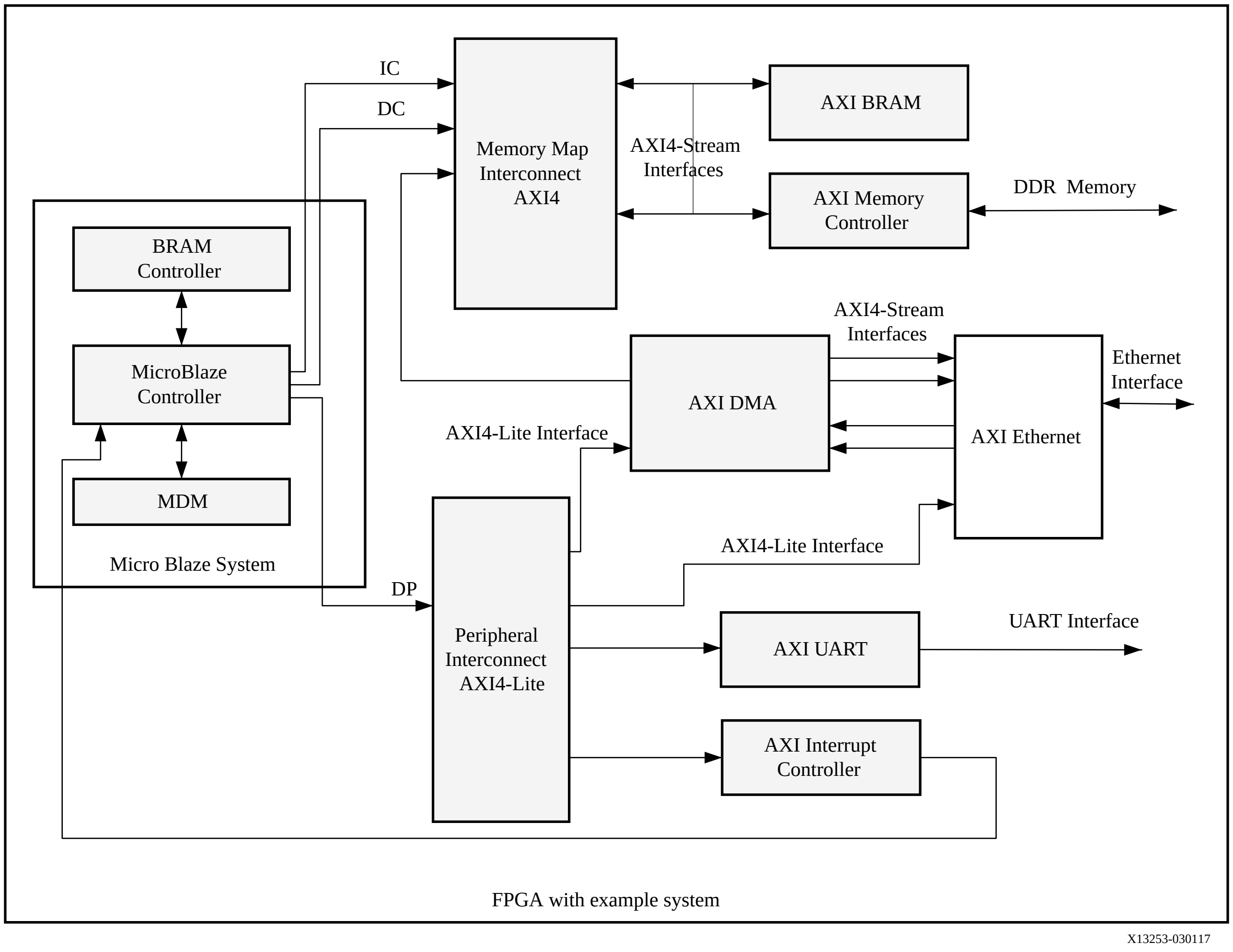
These standards are implemented by the respective cores used in the AXI Ethernet Subsystem. For more details related to these, see the Tri-Mode Ethernet MAC LogiCORE IP Product Guide (PG051). and *1G/2.5G Ethernet PCS/PMA or SGMII LogiCORE IP Product Guide* (PG047).

Performance

To measure the system performance of the AXI Ethernet Subsystem, a system was built in which the subsystem was added to each of the supported device families as the Device Under Test (DUT) as shown in the following figure in which the subsystem serves as the DUT block.

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

Figure 20: **System Configuration with the Subsystem 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 the following table.

Table 4: **System Performance F_{TYP}**

Target FPGA	AXI4-Lite	AXI4-Stream	MicroBlaze™
Supported families	100	100	100

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

Because the subsystem represents a hierarchical design block containing multiple IP instances, the latency, max frequency, throughput, and power values are provided by the IP instances that are present in a given configuration.

Resource Utilization

For details about resource utilization, visit Performance and Resource Utilization.

Port Descriptions

The subsystem ports depend on the mode of operation. These ports are grouped into interfaces based on their functionality. These interfaces have associated clock and reset ports. The details of the interfaces are given in the following table.

Table 5: **I/O Interfaces**

Interface Name	Description
s_axi	AXI4-Lite interface used to configure the AXI Ethernet Subsystem. In processor mode, this interface maps to the AXI Ethernet buffer which configures the TEMAC. In non-processor mode, this interface maps to and directly configures the TEMAC.
s_axis_txc	AXI4-Stream Transmit Control.
s_axis_txd	AXI4-Stream Transmit Data.
m_axis_rxd	AXI4-Stream Receive Data.
m_axis_rxs	AXI4-Stream Receive Status.
s_axis_tx_av	AXI4-Stream AVB Transmit Data. This interface is present only in AVB mode.
m_axis_rx_av	AXI4-Stream AVB Receive Data. This interface is present only in AVB mode.

Table 5: **I/O Interfaces** *(cont'd)*

Interface Name	Description
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 RGMII mode.
sgmii	Serial Gigabit Media Independent Interface. This interface is present only in SGMII mode.
sfp	In 1000BASE-X mode, this interface connects to the SFP cage. This interface is present only in 1000BASE-X mode.
mgt_clk	Differential clock input for the serial transceiver. This interface is present only in Shared Logic in Subsystem configuration in SGMII or 1000BASE-X modes.

Details of all the I/O signals that are included in this table and other individual signals are provided in the following sections.

AXI4-Lite Interface

The AXI4-Lite signal ports are described in the following table.

Table 6: **AXI4-Lite Signal Ports**

Signal Name	Direction	Description
s_axi_lite_clk	In	Clock.
s_axi_lite_resetn	In	Reset (active-Low). See Resets.
s_axi_awaddr[31:0]	In	Write address.
s_axi_awvalid	In	Write address valid: Indicates a valid write address and control information is available.
s_axi_awready	Out	Write address ready: Slave is ready to accept address and control information.
s_axi_wdata[31:0]	In	AXI write data bus.
s_axi_wstrb[3:0]	In	Write strobes: Indicates which byte lanes have valid data. s_axi_wstrb[n] corresponds to s_axi_wdata[(8xn)]+7:(8xn)]
s_axi_wvalid	In	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	Out	Write ready: Indicates the slave can accept the write data. • 1= Slave ready. • 0 = Slave not ready.

Table 6: **AXI4-Lite Signal Ports** *(cont'd)*

Signal Name	Direction	Description
s_axi_bresp[1:0]	Out	Write response: Indicates the status of the write transaction.
s_axi_bvalid	Out	Write response valid: Indicates a valid write response is available. • 1= Write response available. • 0 = Write response not available.
s_axi_bready	In	Response ready: Indicates the master can accept the response information. • 1 = Master ready. • 0 = Master not ready.
s_axi_araddr[31:0]	In	Read address
s_axi_arvalid	In	Read address valid: When High, this signal indicates the read address and control information is valid and remains valid until s_axi_arready is High. • 1 = Address and control information valid • 0= Address and control information not valid
s_axi_arready	Out	Address ready: Indicates the slave is ready to accept an address and associated control signals.
s_axi_rdata[31:0]	Out	Read data.
s_axi_rresp[1:0]	Out	Read response: Indicates the status of the read transaction.
s_axi_rvalid	Out	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	In	Read ready: Indicates the master can accept the read data and response information. • 1 = master ready • 0 = master not ready

AXI4-Stream Transmit Interface

The AXI4-Stream transmit signal ports are described in the following table.

Table 7: **AXI4-Stream Transmit Ports**

Signal Name	Direction	Description
axis_clk	In	AXI4-Stream clock for TXD RXD TXC and RXS interfaces.
axi_str_txd_aresetn	In	AXI4-Stream Transmit Data Reset. See Resets.
axi_str_txd_tvalid	In	AXI4-Stream Transmit Data Valid.
axi_str_txd_tready	Out	AXI4-Stream Transmit Data Ready.
axi_str_txd_tlast	In	AXI4-Stream Transmit Data Last Word.

Table 7: **AXI4-Stream Transmit Ports** *(cont'd)*

Signal Name	Direction	Description
axi_str_txd_tkeep[3:0]	In	AXI4-Stream Transmit Data Valid Strobes.
axi_str_txd_tdata[31:0]	In	AXI4-Stream Transmit Data bus.
axi_str_txc_aresetn	In	AXI4-Stream Transmit Control Reset. See Resets.
axi_str_txc_tvalid	In	AXI4-Stream Transmit Control Valid.
axi_str_txc_tready	Out	AXI4-Stream Transmit Control Ready.
axi_str_txc_tlast	In	AXI4-Stream Transmit Control Last Word.
axi_str_txc_tkeep[3:0]	In	AXI4-Stream Transmit Control Valid Strobes.
axi_str_txc_tdata[31:0]	In	AXI4-Stream Transmit Control bus.

AXI4-Stream Receive Interface

The AXI4-Stream receive signal ports are described in the following table.

Table 8: **AXI4-Stream Receive Ports**

Signal Name	Direction	Description
axi_str_rxd_aresetn	In	AXI4-Stream Receive Data Reset. See Resets.
axi_str_rxd_tvalid	Out	AXI4-Stream Receive Data Valid.
axi_str_rxd_tready	In	AXI4-Stream Receive Data Ready.
axi_str_rxd_tlast	Out	AXI4-Stream Receive Data Last Word.
axi_str_rxd_tkeep[3:0]	Out	AXI4-Stream Receive Data Valid Strobes.
axi_str_rxd_tdata[31:0]	Out	AXI4-Stream Receive Data bus.
axi_str_rxs_aresetn	In	AXI4-Stream Receive Control Reset. See .
axi_str_rxs_tvalid	Out	AXI4-Stream Receive Control Valid.
axi_str_rxs_tready	In	AXI4-Stream Receive Control Ready.
axi_str_rxs_tlast	Out	AXI4-Stream Receive Control Last Word.
axi_str_rxs_tkeep[3:0]	Out	AXI4-Stream Receive Control Valid Strobes.
axi_str_rxs_tdata[31:0]	Out	AXI4-Stream Receive Control bus.

Ethernet AVB Transmit Interface

The AVB transmit signal ports are described in the following table.

Table 9: **Ethernet AVB Transmit Ports**

Signal Name	Direction	Description
avb_tx_clk	Out	AXI4-Stream AVB Transmit Data Clock.
axi_str_avb_tx_aresetn	In	AXI4-Stream AVB Transmit Data Reset.
axi_str_avb_tx_tvalid	In	AXI4-Stream AVB Transmit Data Valid.
axi_str_avb_tx_tready	Out	AXI4-Stream AVB Transmit Data Ready.
axi_str_avb_tx_tlast	In	AXI4-Stream AVB Transmit Data Last Word.
axi_str_avb_tx_tdata[7:0]	In	AXI4-Stream AVB Transmit Data bus.
axi_str_avb_tx_tuser[0:0]	In	AXI4-Stream AVB Transmit User-defined signal.

Ethernet AVB Receive Interface

The AVB receive signal ports are described in the following table.

Table 10: **Ethernet AVB Receive Ports**

Signal Name	Direction	Description
avb_rx_clk	Out	AXI4-Stream AVB Receive Data Clock.
axi_str_avb_rx_aresetn	In	AXI4-Stream AVB Receive Data Reset.
axi_str_avb_rx_tvalid	Out	AXI4-Stream AVB Receive Data Valid.
axi_str_avb_rx_tlast	Out	AXI4-Stream AVB Receive Data Last Word.
axi_str_avb_rx_tdata[7:0]	Out	AXI4-Stream AVB Receive Data bus.
axi_str_avb_rx_tuser[0:0]	Out	Receive channel user information used to indicate if the received frame is good (active-Low) or bad (active-High).

Ethernet AVB Interrupt Interface

The AVB interrupt signal ports are described in the following table.

Table 11: **Ethernet AVB Interrupt Ports**

Signal Name	Direction	Description
interrupt_ptp_timer	Out	Asserted every 10 ms as a measure by the real time clock (RTC). This is used as a timer for the Precise Timing Protocol (PTP) software algorithms.
interrupt_ptp_tx	Out	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	Out	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 RTC Interface

The reference RTC signal ports are described in the following table. These ports can be used for 1722 logic.

Table 12: **Reference RTC Ports**

Signal Name	Direction	Description
rtc_nanosec_field[31:0]	Out	The synchronized nanosecond field from the RTC.
rtc_sec_field[47:0]	Out	The synchronized second field from the RTC
clk8k	Out	An 8 kHz clock which is derived from and is synchronized to the RTC. The period of this clock, 125 us, marks the isochronous cycle.

System Interface

The system ports are described in the following table.

Table 13: **System Ports**

Signal Name	Direction	Description
interrupt	Out	Interrupt indicator for subsystem.
mac_irq	Out	This is the interrupt output from the TEMAC interrupt controller related to the MDIO interface. For detailed information, see the Tri-Mode Ethernet MAC LogiCORE IP Product Guide (PG051).

Ethernet System Interface

The Ethernet system ports are described in the following table.

Table 14: **Ethernet System Ports**

Signal Name	Direction	Description
phy_rst_n	Out	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. Initial status 0.

Table 14: **Ethernet System Ports** *(cont'd)*

Signal Name	Direction	Description
ref_clk	In	This is a stable global clock used by signal delay primitives and transceivers in those respective modes. The clock frequency is 200 MHz for 7 series FPGAs. For UltraScale/UltraScale+/Versalarchitecture, the frequency range is 300-1333 MHz for GMII and RGMII modes. For detailed information, see the Tri-Mode Ethernet MAC LogiCORE IP Product Guide (PG051). When a PCS PMA core exists, it drives the independent clock of 1000BASE-X PCS/PMA. The clock frequency is 50 MHz for UltraScale/UltraScale+/Versal Adaptive SoC. For detailed information on the clock requirements, see the *1G/2.5G Ethernet PCS/PMA or SGMII LogiCORE IP Product Guide* (PG047).
gtx_clk	In	The 125 MHz clock used in all 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. This clock is also used when Ethernet Statistics are enabled with all supported device families. See Clocking.
mgt_clk_p	In	Positive polarity of differential clock used to drive GT serial transceivers. Must be connected to an external, high-quality differential reference clock of frequency of 125 MHz.
mgt_clk_n	In	Negative polarity of differential clock used to drive GT serial transceivers. Must be connected to an external, high-quality differential reference clock of frequency of 125 MHz.
signal_detect	In	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.
clk_en	In	This signal is present only in GMII mode and when the include_io option is deselected. In 10/100 MHz mode, this needs to be connected to the *1G/2.5G Ethernet PCS/PMA or SGMII LogiCORE IP* clock_en output port, else tie to 1.

Ethernet MII Interface

The Ethernet MII ports are described in the following table.

Table 15: **Ethernet MII Ports**

Signal Name	Direction	Description
mii_txd[3:0]	Out	TEMAC to PHY transmit data. Initial status 0.
mii_tx_en	Out	TEMAC to PHY transmit enable. Initial status 0.
mii_tx_er	Out	TEMAC to PHY transmit Error enable. Initial status 0.
mii_rxd[3:0]	In	PHY to TEMAC receive data.
mii_rx_dv	In	PHY to TEMAC receive data valid indicator.
mii_rx_er	In	PHY to TEMAC receive error indicator.
mii_rx_clk	In	PHY to TEMAC receive clock.
mii_tx_clk	In	PHY to TEMAC transmit clock (also used for GMII/MII mode). See Clocking.

Ethernet GMII Interface

The Ethernet GMII ports are described in the following table.

Table 16: **Ethernet GMII Ports**

Signal Name	Direction	Description
gmii_txd[7:0]	Out	TEMAC to PHY transmit data. Initial status 0.
gmii_tx_en	Out	TEMAC to PHY transmit enable. Initial status 0.
gmii_tx_er	Out	TEMAC to PHY transmit Error enable. Initial status 0.
gmii_gtx_clk	Out	TEMAC to PHY transmit clock. Initial status 0.
gmii_rxd[7:0]	In	PHY to TEMAC receive data.
gmii_rx_dv	In	PHY to TEMAC receive data valid indicator.
gmii_rx_er	In	PHY to TEMAC receive error indicator.
gmii_rx_clk	In	PHY to TEMAC receive clock.
gmii_tx_clk	In	PHY to TEMAC TX input clock in 10 and 100 Mb/s modes.

Ethernet SGMII and 1000BASE-X Interface

The Ethernet SGMII and 1000BASE-X signal ports are described in the following table.

Table 17: **Ethernet SGMII and 1000BASE-X Ports**

Signal Name	Direction	Description
txp	Out	TEMAC to PHY transmit data positive. Initial status 0.
txn	Out	TEMAC to PHY transmit data negative. Initial status 0.
rxp	In	PHY to TEMAC receive data positive.
rxn	In	PHY to TEMAC receive data negative.

Dynamic Switching Signal Port

The following table describes the signals present when the optional dynamic switching mode (between 1000BASE-X and SGMII standards) is selected. In this case, the device-specific transceiver ports are always present.

Table 18: **Optional Dynamic Switching**

Signal Name	Direction	Description
basex_or_sgmii	In	Used as the reset default to select the standard. Tie this signal to 0 or 1. 0: Core comes out of reset operating in 1000BASE-X. 1: Core comes out of reset operating in SGMII.

Notes:

1. Clock domain is userclk2.

Ethernet Transceiver Debug Interface

When you enable the Transceiver Control user parameter, the Ethernet Transceiver Debug Interface contains the DRP interface, DRP clock, and the Transceiver Debug interface. When IEEE 1588 mode is enabled, the Ethernet Transceiver Debug Interface contains only a transceiver interface. For detailed information on the signals and clock frequency information, see the *1G/2.5G Ethernet PCS/PMA or SGMII LogiCORE IP Product Guide* (PG047).

Table 19: **Ethernet Transceiver Debug Interface**

Name	Mode	Description
transceiver_debug	master	This interface contains transceiver debug signals.
gt_drp	slave	This interface contains DRP signals.
drp_clk	In	This is a DRP interface clock.

Ethernet RGMII Interface

The Ethernet RGMII are described in the following table.

Table 20: **Ethernet RGMII Ports**

Signal Name	Direction	Description
rgmii_txd[3:0]	Out	TEMAC to PHY transmit data. Initial status 0.
rgmii_tx_ctl	Out	TEMAC to PHY transmit control. Initial status 0.
rgmii_txc	Out	TEMAC to PHY transmit clock. Initial status 0.
rgmii_rxd[3:0]	In	PHY to TEMAC receive data.
rgmii_rx_ctl	In	PHY to TEMAC receive control.
rgmii_rxc	In	PHY to TEMAC receive clock.

IO Delay Calibration Ports for Ethernet RGMII Interface

The following are the ports that are used to calibrate the IO Delays used by the RGMII interface in Versal devices.

Table 21: **IO Delay Calibration Ports**

Signal Name	Direction	Description
idelay_cnt_val[4:0]	In	CNTVALUEIN input of all the IDELAYS that are driven by rgmii_rxd[3:0] and rgmii_rx_ctl
idelay_load	In	LOAD input of all the IDELAYS that are driven by rgmii_rxd[3:0] and rgmii_rx_ctl
odelay_cnt_val[4:0]	In	CNTVALUEIN input of the ODELAY that drives rgmii_tx_clk
odelay_load	In	LOAD input of the ODELAY that drives rgmii_tx_clk

Ethernet MII Management Interface

The Ethernet MII management interface signal ports are described in the following table.

Table 22: **Ethernet MII Management Interface (MIIM) Ports**

Signal Name	Direction	Initial Status	Description
mdc	Out	0	TEMAC to PHY MII management bus clock.
mdio	In/Out	1	3-stateable bidirectional MII Management data bus. If the MDIO interface of AXI Ethernet is not used, the MDIO_I signal must be tied High to allow MDIO communication to the internal Ethernet MAC. In all modes, individual mdio_i (input) mdio_o (output) and mdio_t (3-state enable) signals are provided.

IEEE 1588 Transmit Timestamp Ports

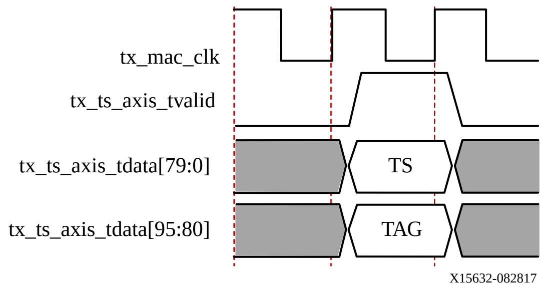
The captured timestamp is always presented out-of-band for 1-step and 2-step frame transmission, using a dedicated AXI4-Stream interface. The signal is defined in the following table.

The following figure shows a timing diagram showing the operation of this interface.

Table 23: **IEEE 1588 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 Ethernet MAC.

Figure 21: **Transmit 2-step Timestamp**



IEEE 1588 Received Timestamp Ports

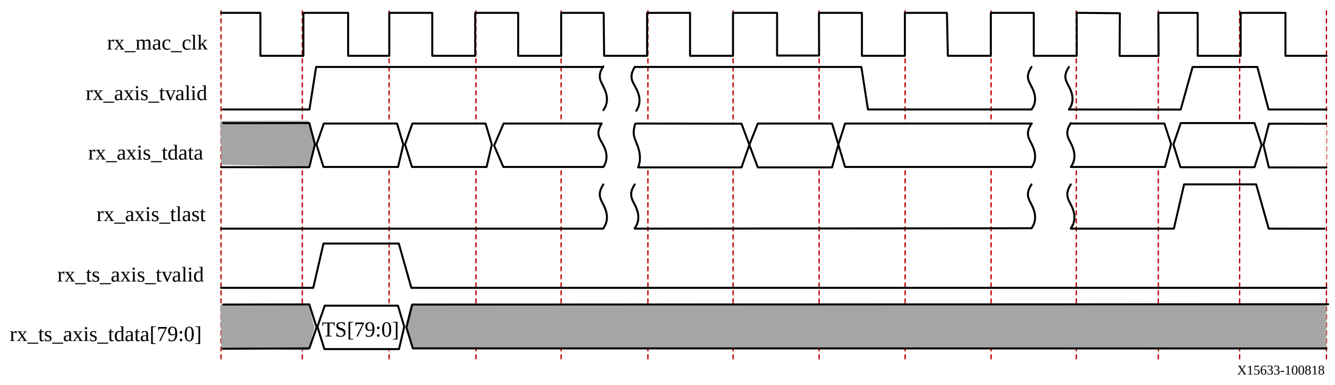
The captured timestamp is always presented out-of-band with TEMAC frame reception using a dedicated AXI4-Stream interface. The signal definition for this is defined in the following table.

A timing diagram showing the operation of this interface is shown in the following figure. 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 24: **IEEE 1588 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 Ethernet MAC.

Figure 22: **AXI4-Stream Interface Timing – Receive Timestamp**



IEEE 1588 System Timer Ports

The 1588 system-wide Time-of-Day (ToD) timer are provided to the subsystem using the ports defined in the following table.

Table 25: **IEEE 1588 System Timer Ports**

Name	Clock Domain	Description
systemtimer_clk[1]	–	System reference clock for the system timer provided to the subsystem.
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.

Notes:

1. If the system is using sync-E and the recovered clock is used to drive the systemtimer_clk, the frequency could be in multiples of userclk.

   For any other systems, AMD recommends that rising edges of the systemtimer_clk clock be close to rising edges of usrclk so that accurate value can be sampled on the userclk domain. This can be achieved by having the systemtimer_clk running at a much higher frequency as compared to usrclk.

Table 26: **1588 Correction Field Ports**

Name	Clock Domain	Description
systemtimer_clk[1]	–	Clock for the system timer provided to the subsystem.
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.

Notes:

1. If the system is using sync-E and the recovered clock is used to drive the systemtimer_clk, the frequency could be in multiples of userclk.

   For any other systems, AMD recommends that rising edges of the systemtimer_clk clock be close to rising edges of usrclk so that accurate value can be sampled on the userclk domain. This can be achieved by having the systemtimer_clk running at a much higher frequency as compared to usrclk.

Flow Control Interface

The flow control ports are described in the following table. These signals are used to request a flow-control action from the transmit engine. Valid flow control frames received by the MAC are automatically handled. The pause value in the received frame is used to inhibit the transmitter operation for the time defined in IEEE 802.3-2008 specification. The frame is then passed to the client with `rx_axis_mac_tuser` asserted to indicate to the client that it should be dropped.

Table 27: **Flow Control Interface Ports**

Signal Name	Direction	Description
s_axis_pause_tvalid	In	It provides the pause request to the TEMAC. Upon request the MAC transmits a pause frame upon completion of the current data packet.
s_axis_pause_tdata[15:0]	In	This is the pause value to the TEMAC. This value inserted into the parameter field of the transmitted pause frame.

Priority Flow Control Interface (802.1Qbb)

The priority flow control (PFC) interface is used to initiate the transmission of PFC frames from the core. The ports associated with this interface are shown in the following table. This interface is only present when priority-based flow control is enabled in the MAC features tab of the GUI. When the optional PFC is enabled, there are eight AXI4-Stream interfaces defined for each Class of Service.

Table 28: **Priority Flow Control Ports**

Signal Name	Direction	Description
s_axis_tx_pfc0_tvalid	In	Pause request from priority 0 FIFO in the TEMAC. This results in a PFC frame at the next available point.
s_axis_tx_pfc1_tvalid	In	Pause request from priority 1 FIFO in the TEMAC. This results in a PFC frame at the next available point.
s_axis_tx_pfc2_tvalid	In	Pause request from priority 2 FIFO in the TEMAC. This results in a PFC frame at the next available point.
s_axis_tx_pfc3_tvalid	In	Pause request from priority 3 FIFO in the TEMAC. This results in a PFC frame at the next available point.
s_axis_tx_pfc4_tvalid	In	Pause request from priority 4 FIFO in the TEMAC. This results in a PFC frame at the next available point.
s_axis_tx_pfc5_tvalid	In	Pause request from priority 5 FIFO in the TEMAC. This results in a PFC frame at the next available point.
s_axis_tx_pfc6_tvalid	In	Pause request from priority 6 FIFO in the TEMAC. This results in a PFC frame at the next available point.
s_axis_tx_pfc7_tvalid	In	Pause request from priority 7 FIFO in the TEMAC. This results in a PFC frame at the next available point.
m_axis_rx_pfc0_tready	In	Pause acknowledge from priority 0 RX FIFO in the TEMAC. The captured quanta only start to expire when this is asserted. If unused this should be tied High.
m_axis_rx_pfc0_tvalid	Out	Pause request to priority 0 RX FIFO in the TEMAC.
m_axis_rx_pfc1_tready	In	Pause acknowledge from priority 1 RX FIFO in the TEMAC. The captured quanta only start to expire when this is asserted. If unused this should be tied High.
m_axis_rx_pfc1_tvalid	Out	Pause request to priority 1 RX FIFO in the TEMAC.
m_axis_rx_pfc2_tready	In	Pause acknowledge from priority 2 RX FIFO in the TEMAC. The captured quanta only start to expire when this is asserted. If unused this should be tied High.
m_axis_rx_pfc2_tvalid	Out	Pause request to priority 2 RX FIFO in the TEMAC.

Table 28: **Priority Flow Control Ports** *(cont'd)*

Signal Name	Direction	Description
m_axis_rx_pfc3_tready	In	Pause acknowledge from priority 3 RX FIFO in the TEMAC. The captured quanta only start to expire when this is asserted. If unused this should be tied High.
m_axis_rx_pfc3_tvalid	Out	Pause request to priority 3 RX FIFO in the TEMAC.
m_axis_rx_pfc4_tready	In	Pause acknowledge from priority 4 RX FIFO in the TEMAC. The captured quanta only start to expire when this is asserted. If unused this should be tied High.
m_axis_rx_pfc4_tvalid	Out	Pause request to priority 4 RX FIFO in the TEMAC.
m_axis_rx_pfc5_tready	In	Pause acknowledge from priority 5 RX FIFO in the TEMAC. The captured quanta only start to expire when this is asserted. If unused this should be tied High.
m_axis_rx_pfc5_tvalid	Out	Pause request to priority 5 RX FIFO in the TEMAC.
m_axis_rx_pfc6_tready	In	Pause acknowledge from priority 6 RX FIFO in the TEMAC. The captured quanta only start to expire when this is asserted. If unused this should be tied High.
m_axis_rx_pfc6_tvalid	Out	Pause request to priority 6 RX FIFO in the TEMAC.
m_axis_rx_pfc7_tready	In	Pause acknowledge from priority 7 RX FIFO in the TEMAC. The captured quanta only start to expire when this is asserted. If unused this should be tied High.
m_axis_rx_pfc7_tvalid	Out	Pause request to priority 7 RX FIFO in the TEMAC.

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 the following table. The following table 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 the following table 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 23: **TEMAC Transmit Statistics Waveforms**

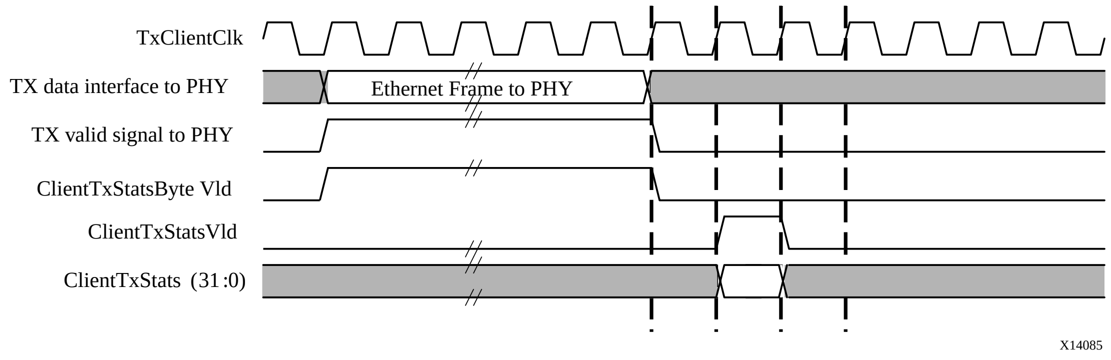


Table 29: **Transmit Statistics Bit Definitions**

Bits	Name	Description
31	PAUSE_FRAME_TRANSMITTED	Asserted if the previous frame was a pause frame initiated by writing to the Transmit Pause Frame (TPF) register.
30	BYTE_VALID	TEMAC: Asserted if an Ethernet 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.
24 [1]	Reserved (driven to zero)	Returns 0.
23 [1]	EXCESSIVE COLLISION	Full-Duplex: Returns 0s.
22 [1]	LATE_COLLISION	Full-Duplex: Returns 0s.
21 [1]	EXCESSIVE_DEFERRAL	Full-Duplex: Returns 0s.
20 [1]	TX_DEFERRED	Full-Duplex: Returns 0s.
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 16,838 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 the following figure. The waveform in the figure 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. The following table shows the bit definition of the receive statistics.

Figure 24: **Receive Statistics Waveforms**

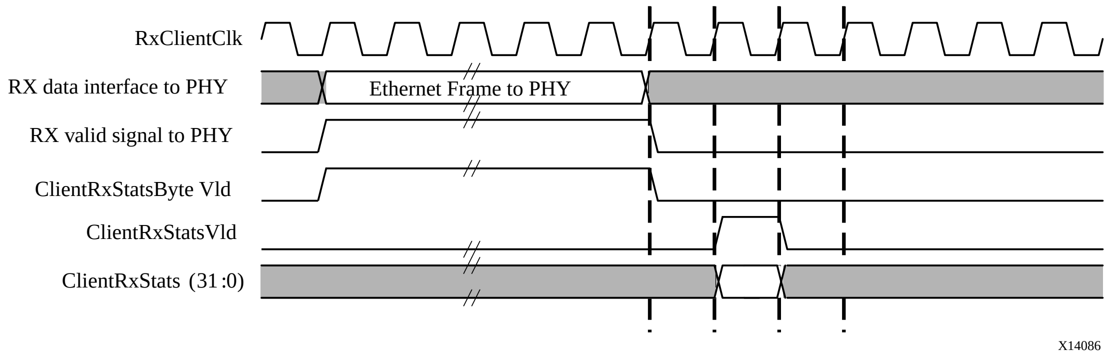


Table 30: **Receive Statistics Bit Definitions**

Bits	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.

Table 30: **Receive Statistics Bit Definitions** *(cont'd)*

Bits	Name	Description
22	BYTE_VALID	TEMAC: Asserted if an Ethernet 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 Ethernet 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.
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 16,383 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, a frame containing this type of error is marked as a GOOD_FRAME, providing no additional errors were detected.

Register Space

The subsystem contains memory and addressable registers for read and write operations as shown in the following table. All registers are directly accessible using a single AXI4-Lite interface. The base address is computed in the Vivado IP integrator system during the creation of the system. All the registers addresses mentioned here are 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 Tri-Mode Ethernet MAC LogiCORE IP Product Guide (PG051). for more information related to these registers. All reserved address spaces indicated in the following table return zeros when read.

In through , R/W stands for read/write and RO stands for read only.

Table 31: **AXI4-Lite Addressable Memory and Soft Registers**

Register Name	AXI4-Lite Address (offset from C_BASEADDR)	Access
Reset and Address Filter TEMAC (RAF)	0x00000000[1]	R/W
Transmit Pause Frame TEMAC (TPF)	0x00000004[1]	R/W
Transmit Inter Frame Gap Adjustment TEMAC (IFGP)	0x0000000[1]	R/W
Interrupt Status (IS)	0x0000000C[1]	R/W
Interrupt Pending (IP)	0x00000010[1]	RO
Interrupt Enable (IE)	0x00000014[1]	R/W
Transmit VLAN Tag TEMAC (TTAG)	0x00000018[1]	R/W
Receive VLAN Tag TEMAC (RTAG)	0x0000001C[1]	R/W
Unicast Address Word Lower TEMAC (UAWL)	0x00000020[1]	R/W
Unicast Address Word Upper TEMAC (UAWU)	0x00000024[1]	R/W
VLAN TPID TEMAC Word 0 (TPID0)	0x00000028[1]	R/W
VLAN TPID TEMAC Word 1 (TPID1)	0x0000002C[1]	R/W
PCS PMA TEMAC Status (PPST)	0x00000030[1]	RO
Reserved	0x00000034[1]–0x000001FC	Reserved
Statistics Counters	0x00000200–0x000003FC	RO
TEMAC Receive Configuration Word 0 (RCW)	0x00000400	R/W
TEMAC Receive Configuration Word 1 (RCW)	0x00000404	R/W
TEMAC Transmitter Configuration (TC)	0x00000408	R/W
TEMAC Flow Control Configuration (FCC)	0x0000040C	R/W
TEMAC Speed Configuration	0x00000410	R/W
RX Max Frame Configuration	0x00000414	R/W
TX Max Frame Configuration	0x00000418	R/W
TX Timestamp Adjust Control	0x0000041C	R/W
Reserved	0x00000424–0x000004F4	Reserved
Identification	0x000004F8	RO
Ability Register	0x000004FC	RO
MDIO Setup	0x00000500	R/W
MDIO Control	0x00000504	R/W
MDIO Write Data	0x00000508	R/W
MDIO Read Data	0x0000050C	RO
Reserved	0x00000510–0x000005FC	Reserved
Interrupt Status (TEMAC)	0x00000600	R/W
Reserved	0x00000604–0x0000060C	Reserved
Interrupt Pending (TEMAC)	0x00000610	RO

Table 31: **AXI4-Lite Addressable Memory and Soft Registers** *(cont'd)*

Register Name	AXI4-Lite Address (offset from C_BASEADDR)	Access
Reserved	0x00000614–0x0000061C	Reserved
Interrupt Enable (TEMAC)	0x00000620	R/W
Reserved	0x00000624–0x0000062C	Reserved
Interrupt Clear (TEMAC)	0x00000630	R/W
Reserved	0x00000634–0x000006FC	Reserved
Unicast Address Word 0 (UAW0)	0x00000700	R/W
Unicast Address Word 1 (UAW1)	0x00000704	R/W
Frame Filter Control	0x00000708	R/W
Frame Filter Enable	0x0000070C	Reserved
Frame Filter Value	0x00000710-0x0000074C	R/W
Frame Filter Mask Value	0x00000750–0x0000078C	R/W
Reserved	0x00000790–0x00000FFC	Reserved
Reserved	0x00001000–0x00003FFC	Reserved
Transmit VLAN Data Table TEMAC	0x00004000–0x00007FFC[1]	R/W
Receive VLAN Data Table TEMAC	0x00008000–0x0000BFFC	R/W
Reserved	0x0000C000–0x0000FFFC	Reserved
Ethernet AVB	0x00010000–0x00013FFC	R/W
Reserved	0x00014000–0x0001FFFC	Reserved
Multicast Address Table TEMAC	0x00020000–0x0003FFFC[1]	R/W

Notes:

1. Registers 0x00000000-0x00000034; 0x00004000-0x0000FFFF; 0x00020000–0x0003FFFF are available only when AXI Ethernet buffer is enabled.

.h Header File

AXI4 register information such as register address, register name with bit position, mask value, access type, and their default values are provided in header (.h) file format when the IP core is generated and the header file can be found under folder header_files of the project path.

The TEMAC subcore AXI4 register information is provided in the header (.h) file that is located within the TEMAC IP core folder.

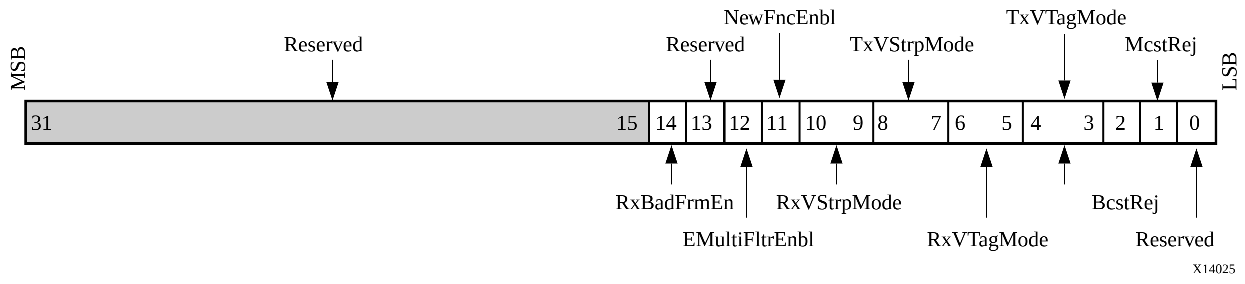
Reset and Address Filter Register

The Reset and Address Filter (RAF) register is shown in the following figure. This registerallows the software to block reception of multicast and broadcast Ethernet frames. 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 is set to promiscuous mode and the multicast reject bit of this register set to allow multicast frames. See for more information.

The 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 subsystem, bits in this register are used to control those new functions. To minimize the effect of these new bits on existing applications the default values of these bits disable this functionality. This ensures that when applications do not use the more recent functionality, the subsystem operates the way it did previously.

Figure 25: **Reset and Address Filter Register (0x0000_0000)**



The following table shows the Reset and Address Filter register bit definitions.

Table 32: **Reset and Address Filter Register (0x000_000)**

Bits	Name	Access	Reset Value	Description
31:15	Reserved	RO	0x0	**Reserved**: These bits are reserved for future use and always return zero.
14	RxBadFrmEn	R/W	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	RO	0	**Reserved**: These bits are reserved for future use and always return zero.

Table 32: **Reset and Address Filter Register (0x000_000)** *(cont'd)*

Bits	Name	Access	Reset Value	Description
12	EMultiFltrEnbl	R/W	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 subsystem includes the enhanced multicast address filtering function enabled at build time by the C_MCAST_EXTEND parameters. See for more details. • 0 – Disable enhanced multicast address filtering mode. • 1 – Enable enhanced multicast address filtering mode if present.
11	NewFncEnbl	R/W	0	**New Functions Enable**: This bit allows you to disable 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	R/W	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 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.

Table 32: **Reset and Address Filter Register (0x000_000)** *(cont'd)*

Bits	Name	Access	Reset Value	Description
8:7	TxVStrpMode	R/W	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 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	R/W	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 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.
4:3	TxVTagMode	R/W	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 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.

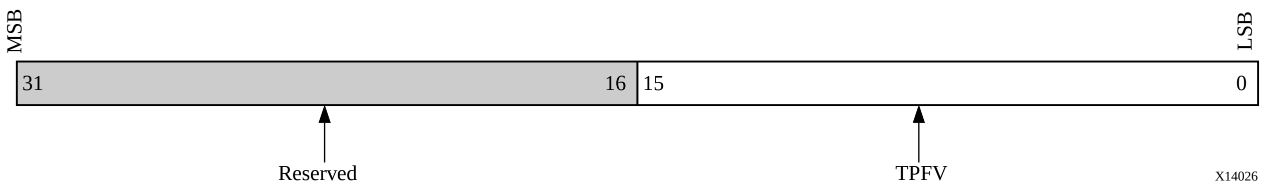
Table 32: **Reset and Address Filter Register (0x000_000)** *(cont'd)*

Bits	Name	Access	Reset Value	Description
2	BcstRej	R/W	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	R/W	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	RO	0	**Reserved**: These bits are reserved for future definition and always return zero.

Transmit Pause Frame Register

The Transmit Pause Frame (TPF) TEMAC register is shown in the following table. This register provides a value of pause when enabled by the FCC register (TEMAC Flow Control Configuration Register). 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 x 512-bit times.

Figure 26: **Transmit Pause Frame Register (0x0000_0004)**



The following table shows the Transmit Pause Frame register bit definitions.

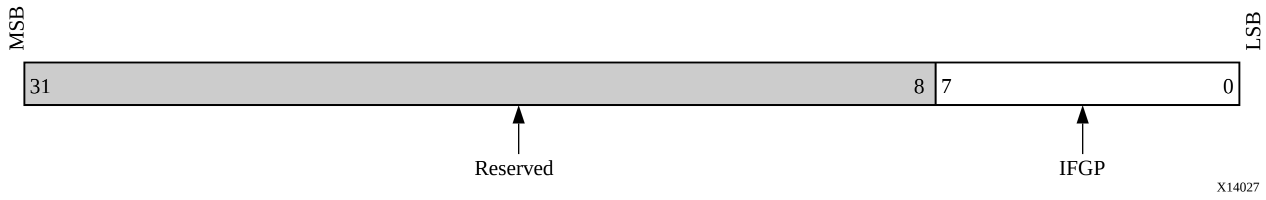
Table 33: **Transmit Pause Frame Register (0x0000_0004)**

Bits	Name	Access	Reset Value	Description
31:16	Reserved	RO	0x0	**Reserved**: These bits are reserved for future use and always return zero.
15:0	TPFV	R/W	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

The Transmit Inter Frame Gap Adjustment (IFGP) register is shown in the following figure. This register provides a duration value of Inter Frame Gap when enabled by the TC register (). 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 27: **Transmit Inter Frame Gap Adjustment Register (0x0000_0008)**



The following table shows the Transmit Inter Frame Gap Adjustment register bit definitions.

Table 34: **Transmit Inter Frame Gap Adjustment Register (0x0000_0008)**

Bits	Name	Access	Reset Value	Description
31:8	Reserved	RO	0x0	**Reserved**: These bits are reserved for future use and always return zero.
7:0	IFGP0	R/W	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 (25) 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

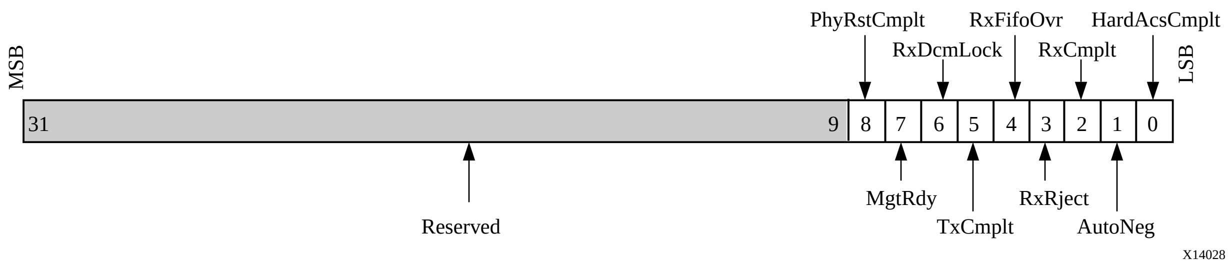
The Interrupt Status (IS) register is shown in Figure 43: Receive VLAN Table Entry with all Fields (0x0000_8000-0x0000_BFFF). This register combined with the IE, IP, MIS, and MIE registers define the interrupt interface of the subsystem. The Interrupt Status register uses one bit to represent each internal subsystem 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 Interrupt Status Register and Interrupt Enable Register.

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 Subsystem. The following figure shows the structure of the interrupt register.

Figure 28: **Interrupt Status Register (0x0000_000C)**



The following figure shows the Interrupt Status register bit definitions.

Table 35: **Interrupt Status Register (0x0000_000C)**

Bits	Name	Access	Reset Value	Description
31:9	Reserved	RO	0x0	**Reserved**: These bits are reserved for future use and always return zero.
8	PhyRstCmplt	R/W	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

Table 35: **Interrupt Status Register (0x0000_000C)** *(cont'd)*

Bits	Name	Access	Reset Value	Description
7	MgtRdy[1]	R/W	0 / 1	**Serial Transceiver Ready**: This bit indicates if the TEMAC core 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 core registers does not complete and the subsystem does not operate. In systems that do not use a serial transceiver, this signal goes to 1 immediately after reset. • 0 – serial transceiver / TEMAC not ready • 1 – serial transceiver / TEMAC ready
6	RxDcmLock	R/W	1	**Receive DCM Lock**: No longer used, but reserved for future use. This bit is always one. • 0 – RX digital clock manager (DCM) not locked • 1 – RX DCM Locked
5	TxCmplt	R/W	0	**Transmit Complete**: This bit indicates that a frame was successfully transmitted. • 0 – no frame transmitted • 1 – frame transmitted
4	RxMemOvr	R/W	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]	R/W	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	R/W	0	**Receive Complete**: This bit indicates that a packet was successfully received. • 0 – no frame received • 1 – frame received
1	AutoNeg	R/W	0	**Auto-Negotiation Complete**: This bit indicates that auto-negotiation of the SGMII or 1000BASE-X interface has completed. • 0 – auto-negotiation not complete. • 1 – auto-negotiation complete.

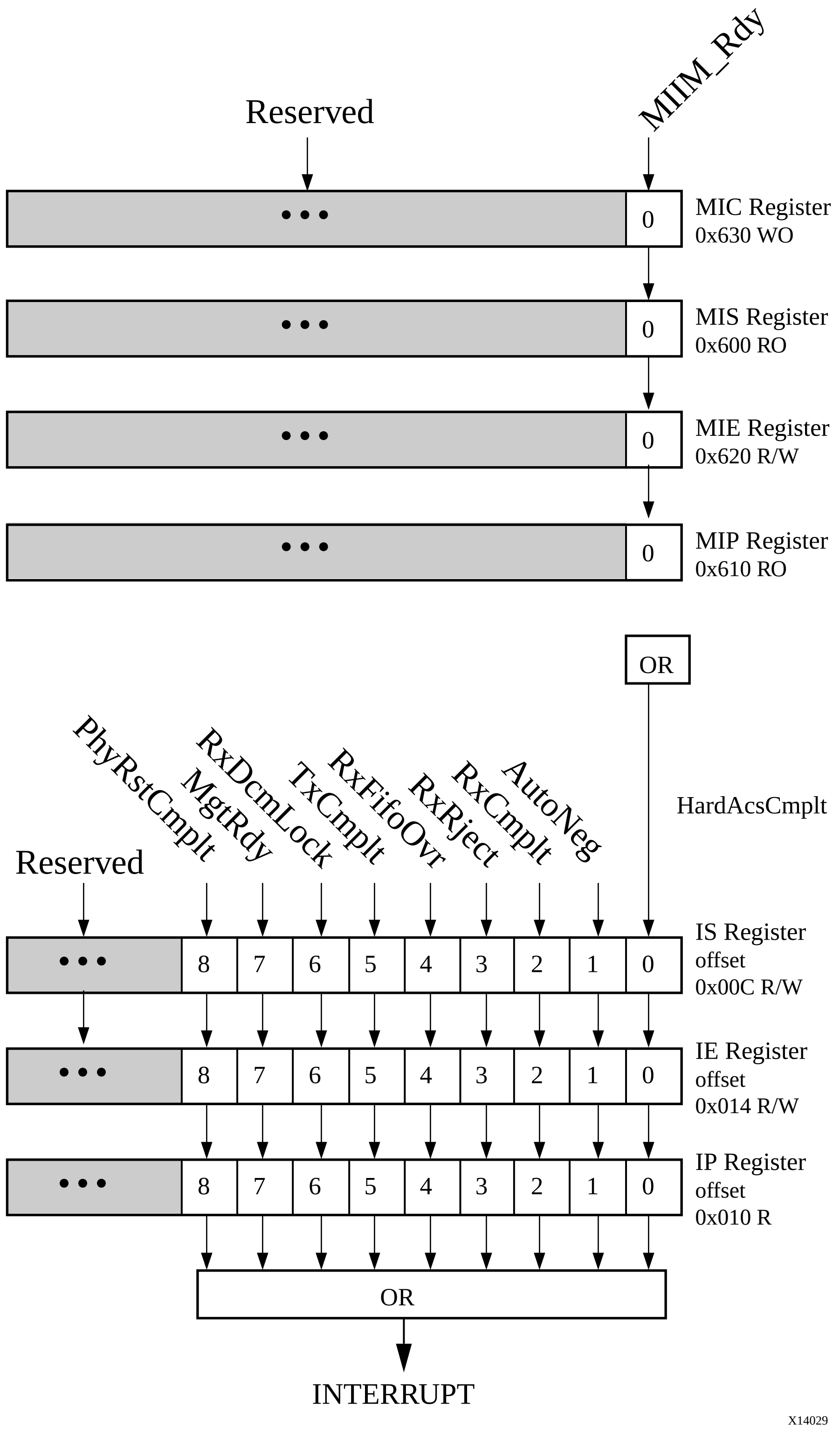
Table 35: **Interrupt Status Register (0x0000_000C)** *(cont'd)*

Bits	Name	Access	Reset Value	Description
0	HardAcsCmplt	R/W	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 the following figure for conditions that cause the receive frame reject interrupt to occur. The receive frame reject interrupt occurs for any of the following reasons:
    - The frame does not meet the Ethernet frame requirements (bad FCS, bad length, etc).
    - 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.
    - The subsystem was built to support extended multicast address filtering (C_MCAST_EXTEND=1).
    - 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.
    - 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 29: **Subsystem Interrupt Structure**



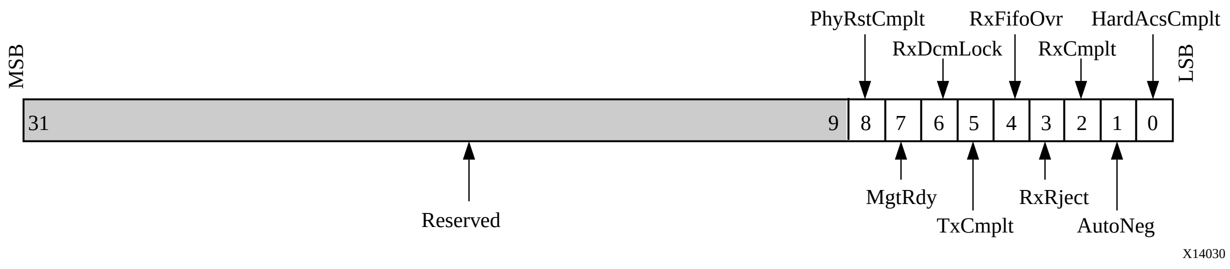
Interrupt Pending Register

The Interrupt Pending (IP) register is shown in the following figure. This register combined with the IS, IE, MIS, and MIE registers defines the interrupt interface of the subsystem. The Interrupt Pending register uses one bit to represent each subsystem 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 are set in the Interrupt Pending register, the `interrupt` signal is driven active-High out of the AXI Ethernet Subsystem.

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 30: **Interrupt Pending Register (0x0000_0010)**



The following table shows the Interrupt Pending register bit definitions.

Table 36: **Interrupt Pending Register (0x0000_0010)**

Bits	Name	Access	Reset Value	Description
31:9	Reserved	RO	0x0	**Reserved**: These bits are reserved for future use and always return zero.
8	PhyRstCmplt	R/W	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

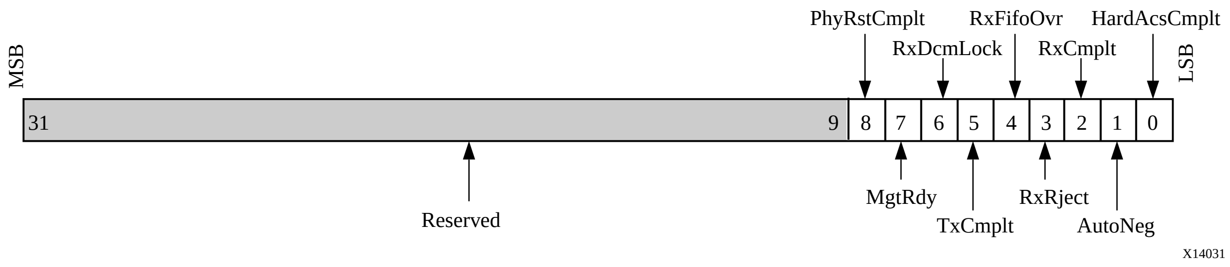
Table 36: **Interrupt Pending Register (0x0000_0010)** *(cont'd)*

Bits	Name	Access	Reset Value	Description
7	MgtRdy	R/W	0	**MGT Ready**: This bit indicates if the TEMAC core 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 to TEMAC core registers does not complete and the subsystem does not operate. In systems that do not use a serial transceiver, this signal goes to 1 immediately after reset. 0 – Serial Transceiver / TEMAC not ready 1 – Serial Transceiver / TEMAC ready
6	RxDcmLock	R/W	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	R/W	0	**Transmit Complete**: This bit indicates that a frame was successfully transmitted. 0 – no frame transmitted 1 – frame transmitted
4	RxMemOvr	R/W	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	R/W	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	R/W	0	**Receive Complete**: This bit indicates that a packet was successfully received. 0 – no frame received 1 – frame received
1	AutoNeg	R/W	0	**Auto-Negotiation Complete**: This bit indicates that auto-negotiation of the SGMII or 1000BASE-X interface has completed. • 0 – auto-negotiation not complete • 1 – auto-negotiation complete
0	HardAcsCmplt	R/W	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

The Interrupt Enable (IE) register is shown in the following figure. This register, combined with the IS, IP, MIS, and MIE registers, defines the interrupt interface of the subsystem. The Interrupt Enable register uses one bit to represent each subsystem 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 31: **Interrupt Enable Register (0x0000_0014)**



The following table shows the Interrupt Enable register bit definitions.

Table 37: **Interrupt Enable Register (0x0000_0014)**

Bits	Name	Access	Reset Value	Description
31:9	Reserved	RO	0x0	**Reserved**: These bits are reserved for future definition and always return zero.
8	PhyRstCmplt	R/W	0	**PHY Reset Complete**: Bit used to enable interrupt. • 0 – Interrupt Disabled • 1 – Interrupt Enabled
7	MgtRdy	R/W	0	MGT Ready: Bit used to enable interrupt. • 0 – Interrupt Disabled • 1 – Interrupt Enabled
6	RxDcmLock	R/W	0	**Receive DCM Lock**: Bit used to enable interrupt. • 0 – Interrupt Disabled • 1 – Interrupt Enabled
5	TxCmplt	R/W	0	**Transmit Complete**: Bit used to enable interrupt. • 0 – Interrupt Disabled • 1 – Interrupt Enabled
4	RxMemOvr	R/W	0	**Receive Memory Overrun:** Bit used to enable interrupt. • 0 – Interrupt Disabled • 1 – Interrupt Enabled

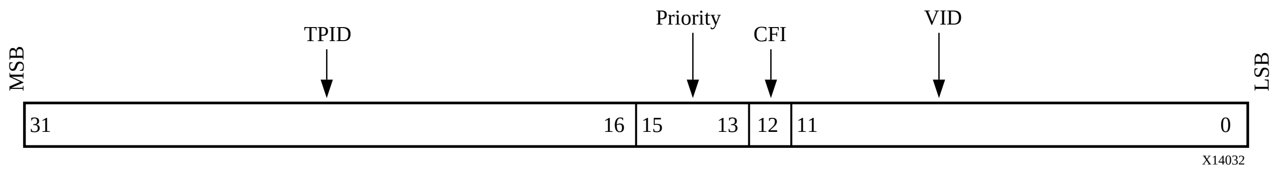
Table 37: **Interrupt Enable Register (0x0000_0014)** *(cont'd)*

Bits	Name	Access	Reset Value	Description
3	RxRject	R/W	0	**Receive Frame Rejected**: Bit used to enable interrupt. • 0 – Interrupt Disabled • 1 – Interrupt Enabled
2	RxCmplt	R/W	0	**Receive Complete:**Bit used to enable interrupt. • 0 – Interrupt Disabled • 1 – Interrupt Enabled
1	AutoNeg	R/W	0	**Auto-Negotiation Complete**: Bit used to enable interrupt. • 0 – Interrupt Disabled • 1 – Interrupt Enabled
0	HardAcsCmplt	R/W	0	**Hard Register Access Complete**: Bit used to enable interrupt. • 0 – Interrupt Disabled • 1 – Interrupt Enabled

Transmit VLAN Tag Register

The Transmit VLAN Tag (TTAG) register is shown in the following figure. This register is only used when the VLAN tagging is included in the subsystem 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 Support for more information about how VLAN tagging is performed.

Figure 32: **Transmit VLAN Tag Register (0x0000_0018)**



The following table shows the Transmit VLAN Tag register bit definitions.

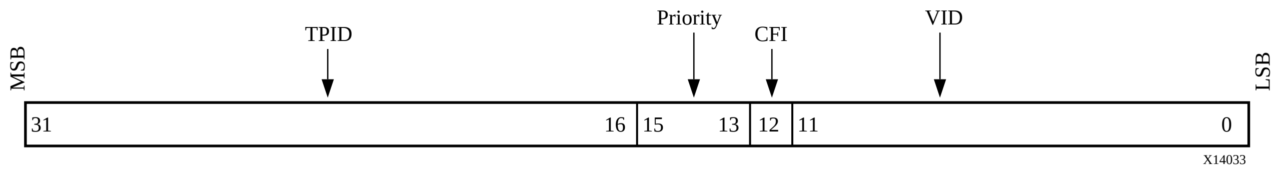
Table 38: **Transmit VLAN Tag Register (0x0000_0018)**

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

Receive VLAN Tag Register

The Receive VLAN Tag (RTAG) register is shown in the following figure. This register is only used when the VLAN tagging is included in the subsystem 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 33: **Receive VLAN Tag Register (0x0000_001C)**



The following table shows the Receive VLAN Tag register bit definitions.

Table 39: **Receive VLAN Tag Register (0x0000_001C)**

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

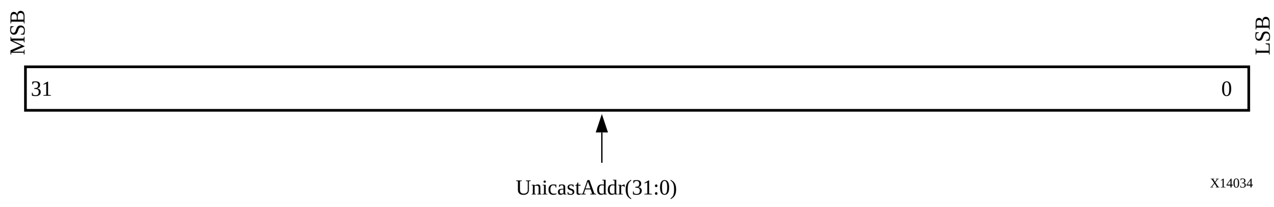
Unicast Address Word Lower Register

The Unicast Address Word Lower (UAWL) register is shown in the following figure. This register and the Unicast Address Word Upper (UAWU) register are *only used when extended multicast filtering is included* in the subsystem 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 subsystem 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 34: **Unicast Address Word Lower Register (0x020)**



The following table shows the Unicast Address Word Lower register bit definitions.

Table 40: **Unicast Address Word Lower Register (0x020)**

Bits	Name	Access	Reset Value	Description
31:0	UnicastAddr	R/W	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, an Ethernet MAC address of AA-BB-CC-DD-EE-FF would be stored in UnicastAddr(47:0) as 0xFFEEDDCCBBAA.

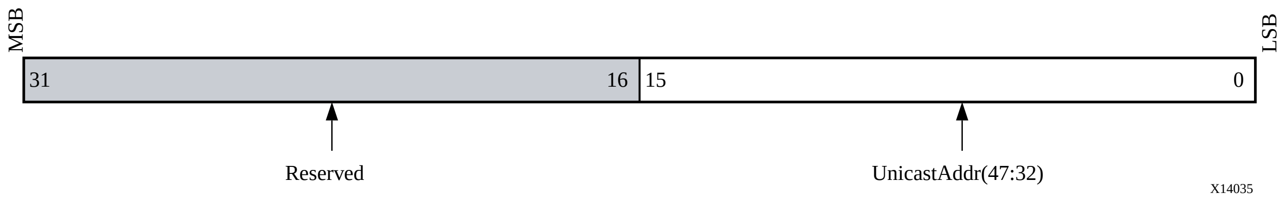
Unicast Address Word Upper Register

The Unicast Address Word Upper (UAWU) register is shown in the following register. This register and the register are only used when extended multicast filtering is included in the subsystem 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 35: **Unicast Address Word Upper Register (0x024)**



The following table shows the Unicast Address Word Upper register bit definitions.

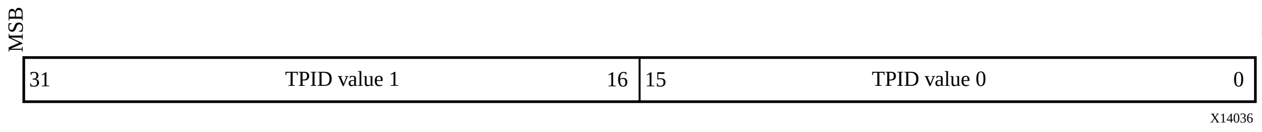
Table 41: **Unicast Address Word Upper Register (0x024)**

Bits	Name	Access	Reset Value	Description
31:16	Reserved	RO	0x0	**Reserved**: These bits are reserved for future use and always return zero.
15:0	UnicastAddr	R/W	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, an Ethernet MAC address of AA-BB-CC-DD-EE-FF would be stored in UnicastAddr(47:0) as 0xFFEEDDCCBBAA.

VLAN TPID Word 0 Register

The VLAN TPID Word 0 (TPID0) register is shown in the following figure. This register is only used when transmit and/or receive VLAN functions are included in the subsystem 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 four TPID values to be specified for recognizing VLAN frames for both the transmit and receive paths. The most common values for VLAN TPID are 0x8100, 0x9100, 0x9200, and 0x88A8. See Extended VLAN Support for more information about extended VLAN functions.

Figure 36: **VLAN TPID Word 0 Register (0x0000_0028)**



The following table shows the VLAN TPID Word 0 register bit definitions.

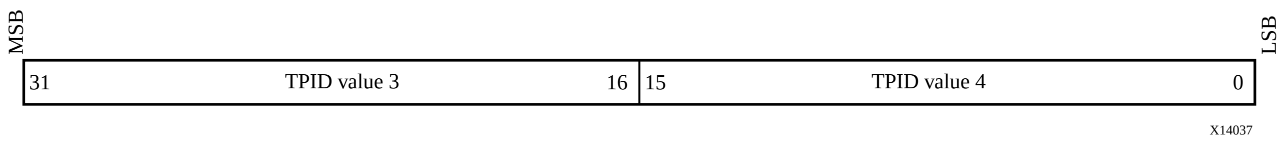
Table 42: **VLAN TPID Word 0 Register (0x0000_0028)**

Bits	Name	Access	Reset Value	Description
31:16	TPID value 1	R/W	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	R/W	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

The VLAN TPID Word 1 (TPID1) register is shown in the following figure. This register is only used when transmit and/or receive VLAN functions are included in the subsystem 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 four TPID values to 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 37: **VLAN TPID Word 1 Register (0x0000_002C)**



The following table shows the VLAN TPID Word 1 register bit definitions.

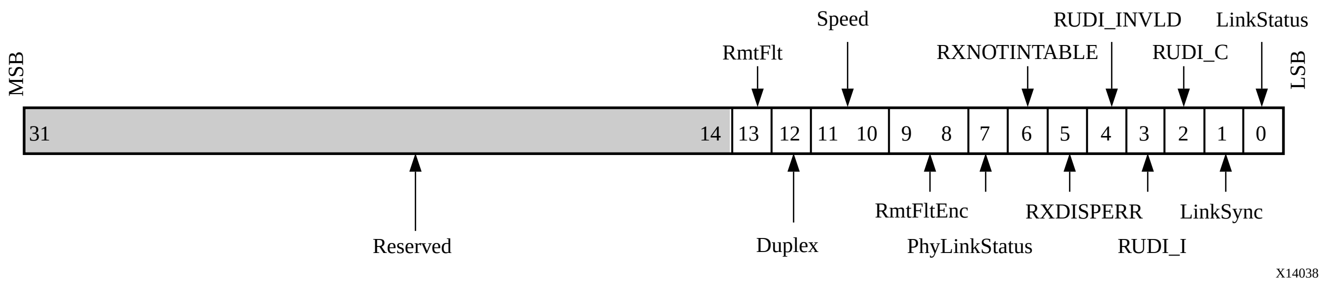
Table 43: **VLAN TPID Word 1 Register (0x0000_002C)**

Bits	Name	Access	Reset Value	Description
31:16	TPID value 3	R/W	0x0	**TPID Value 3**: These bits represent one TPID value that is used to recognize VLAN frames for both the transmit and receive paths.
15:0	TPID value 2	R/W	0x0	**TPID Value 2**: These bits represent one TPID value that is used to recognize VLAN frames for both the transmit and receive paths.

PCS PMA TEMAC Status Register

The PCS PMA TEMAC Status (PPST) register is shown in the following figure. This register reports valid information when the subsystem 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 zeros.

Figure 38: **PCS PMA TEMAC Status Register (0x0000_0030)**



The following table shows the PCS PMA TEMAC Status register bit definitions.

Table 44: **PCS PMA TEMAC Status Register (0x0000_0030)**

Bits	Name	Access	Reset Value	Description
31:14	Reserved	RO	0x000000	Reserved
13	RmtFlt	RO	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 de-asserted when an MDIO read is made to status register (register 1 in *See Management Status Register (Register 1)*). This signal has no significance in SGMII PHY mode.
12	Duplex	RO	0	**Duplex**: Duplex Mode. This bit indicates the Duplex mode negotiated with the link partner • 1 = Full-Duplex • 0 = Half-Duplex • Half-Duplex is not supported
11:10	Speed	RO	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
9:8	RmtFltEnc	RO	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.

Table 44: **PCS PMA TEMAC Status Register (0x0000_0030)** *(cont'd)*

Bits	Name	Access	Reset Value	Description
7	PhyLinkStatus	RO	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	RO	0	**RXNOTINTABLE**: Receive Not In Table. The subsystem has received a code group which is not recognized from the 8B/10B coding tables.
5	RXDISPERR	RO	0	**RXDISPERR**: Receive Disparity Error.The subsystem has received a running disparity error during the 8B/10B decoding function.
4	RUDI_INVLD	RO	0	**RUDI_INVLD**: RUDI(/INVALID/). The subsystem has received invalid data while receiving/C/ or/I/ ordered set.
3	RUDI_I	RO	0	**RUDI_I**: RUDI(/I/). The subsystem is receiving /I/ ordered sets (Idles)
2	RUDI_C	RO	0	**RUDI_C**: RUDI(/C/). The subsystem is receiving /C/ ordered sets (Auto-Negotiation Configuration sequences).
1	LinkSync	RO	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	RO	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

The set of 64-bit counters is only present when selected at build-time. The counters keep track of statistics for the transmit and receive Ethernet traffic and are defined in the *Tri-Mode Ethernet MAC LogiCORE IP Product Guide* (PG051). The Half-Duplex counters are omitted because this subsystem does not support Half-Duplex.

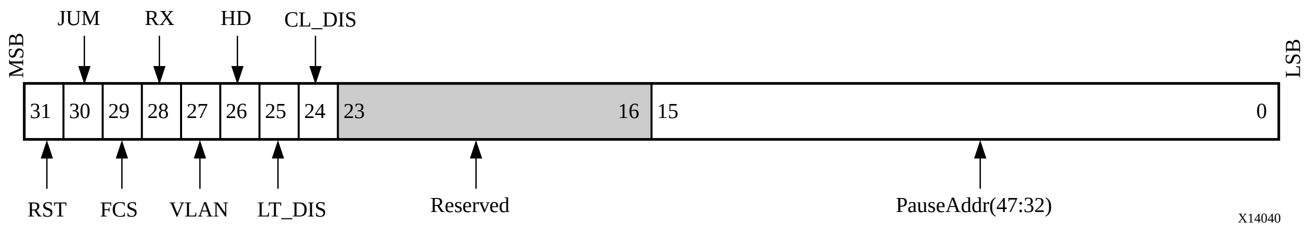
TEMAC Receive Configuration Word 0 Register

Refer to the Tri-Mode Ethernet MAC LogiCORE IP Product Guide (PG051) for details.

TEMAC Receive Configuration Word 1 Register

The TEMAC Receive Configuration Word 1 (RCW1) register is shown in the following figure. 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 39: **TEMAC Receive Configuration Word 1 (RCW1) Register (0x404)**



The following table shows the TEMAC Receive Configuration Word1 register bit definitions.

TEMAC Transmit Configuration Register

Table 45: **TEMAC Receive Configuration Word1 (RCW1) Register (0x404)**

Bits	Name	Access	Reset Value	Description
31	RST	R/W	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 the subsystem can place the subsystem in an unknown state. • 0 – no reset • 1 – initiate a receiver reset

Table 45: **TEMAC Receive Configuration Word1 (RCW1) Register (0x404)** *(cont'd)*

Bits	Name	Access	Reset Value	Description
30	JUM [1]	R/W	0	**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	R/W	0	**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[2]	R/W	1	**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	R/W	0	**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	R/W	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	R/W	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	R/W	0x0	**Control Frame Length Check Disable**: When this bit is 1, control frames larger than the minimum frame length can be accepted
23	Reserved			**Reserved** .

Table 45: **TEMAC Receive Configuration Word1 (RCW1) Register (0x404)** *(cont'd)*

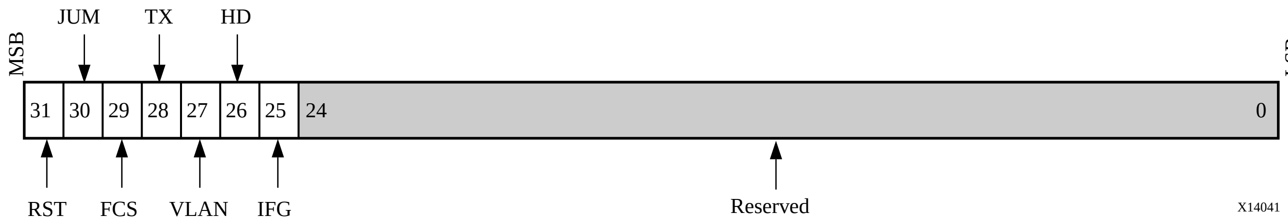
Bits	Name	Access	Reset Value	Description
22		R/W	0	Inband 1588 Timestamp Enable. When 0, Timestamp is only provided out-of-band. When 1, the Timestamp is provided in-Line in addition to out-of-band. When the TEMAC does not include 1588 functionality, this bit is ignored because no timestamp is present.
21:16	Reserved	RO	0x0	**Reserved**: These bits are reserved for future use and always return zero.
15:0	PauseAddr	R/W	0xFFFF	**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, an Ethernet 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.

The TEMAC Transmit Configuration (TC) register is shown in the following figure. 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 40: **TEMAC Transmit Configuration Register (0x408)**



The following table shows the TEMAC Transmit Configuration register bit definitions.

Table 46: **TEMAC Transmit Configuration Register (0x408)**

Bits	Name	Access	Reset Value	Description
31	RST	R/W	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 the subsystem can place the subsystem in an unknown state. • 0 – no reset • 1 – initiate a transmitter reset
30	JUM[1]	R/W	0	**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	R/W	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[2]	R/W	1	**Transmit Enable**: When this bit is 1, the transmit logic is enabled to operate. • 0 – transmit disabled • 1 – transmit enabled
27	VLAN	R/W	0	**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	R/W	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	R/W	0	**Inter Frame Gap Adjustment Enable**: When this bit is 1, the transmitter uses the value of the IFGP register (Transmit Inter Frame Gap Adjustment Register) 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

Table 46: **TEMAC Transmit Configuration Register (0x408)** *(cont'd)*

Bits	Name	Access	Reset Value	Description
24:23		RO		Reserved
22		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.
21:0		RO		**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

Refer to the Tri-Mode Ethernet MAC LogiCORE IP Product Guide (PG051) for details.

TEMAC Ethernet MAC Speed Configuration Register

Refer to the Tri-Mode Ethernet MAC LogiCORE IP Product Guide (PG051) for details.

Receive Max Frame Configuration Register

Refer to the Tri-Mode Ethernet MAC LogiCORE IP Product Guide (PG051) for details.

Transmit Max Frame Configuration Register

Refer to the Tri-Mode Ethernet MAC LogiCORE IP Product Guide (PG051) for details.

Transmit Timestamp Adjust Control Register

This register is present only when the TEMAC includes 1588 functionality. See the following table.

Table 47: **Transmitter Timestamp Adjust Control Register (0x41C)**

Bits	Default Value	Access	Description
31:17	N/A	RO	Reserved

Table 47: **Transmitter Timestamp Adjust Control Register (0x41C)** *(cont'd)*

Bits	Default Value	Access	Description
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"
15:0	0xD3 (211 ns)	RW	TX latency adjust value. In ToD mode: This value is in units of nanoseconds and is initialized to reflect the delay following the timestamping position through the MAC, 1000BASE-X FPGA logic, and GTX transceiver components. In Correction Field Format: The default value is 387 ns decimal value.

ID Register

Refer to the Tri-Mode Ethernet MAC LogiCORE IP Product Guide (PG051) for details.

Ability Register

Refer to the Tri-Mode Ethernet MAC LogiCORE IP Product Guide (PG051) for details.

MDIO Setup Register

Refer to the Tri-Mode Ethernet MAC LogiCORE IP Product Guide (PG051) for details.

MDIO Control Register

Refer to the Tri-Mode Ethernet MAC LogiCORE IP Product Guide (PG051) for details.

MDIO Write Data Register

Refer to the Tri-Mode Ethernet MAC LogiCORE IP Product Guide (PG051) for details.

MDIO Read Data Register

Refer to the Tri-Mode Ethernet MAC LogiCORE IP Product Guide (PG051) for details.

Interrupt Status Register

Refer to the Tri-Mode Ethernet MAC LogiCORE IP Product Guide (PG051) for details.

Interrupt Pending Register

Refer to the Tri-Mode Ethernet MAC LogiCORE IP Product Guide (PG051) for details.

Interrupt Enable Register

Refer to the Tri-Mode Ethernet MAC LogiCORE IP Product Guide (PG051) for details.

Interrupt Clear Register

Refer to the Tri-Mode Ethernet MAC LogiCORE IP Product Guide (PG051) for details.

Unicast Address Word 0 Register

Refer to the Tri-Mode Ethernet MAC LogiCORE IP Product Guide (PG051) for details.

Unicast Address Word 1 Register

Refer to the Tri-Mode Ethernet MAC LogiCORE IP Product Guide (PG051) for details.

Frame Filter Control Register

Refer to the Tri-Mode Ethernet MAC LogiCORE IP Product Guide (PG051) for details.

Frame Filter Enable Register

Refer to the Tri-Mode Ethernet MAC LogiCORE IP Product Guide (PG051) for details.

Frame Filter Value Register

Refer to the Tri-Mode Ethernet MAC LogiCORE IP Product Guide (PG051) for details.

Transmit VLAN Data Table

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 the first following figure while the second figure 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 41: **Transmit VLAN Table Entry with all Fields (0x0000_4000-0x0000_7FFF)**

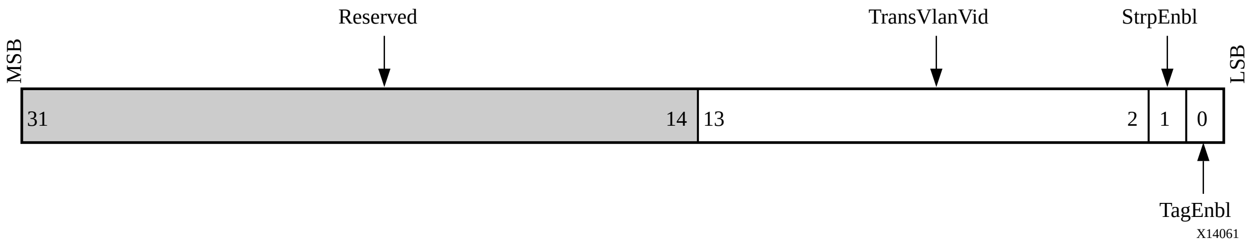
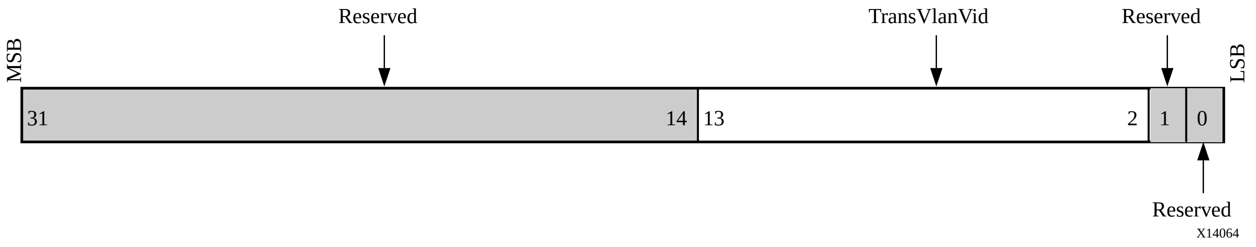


Figure 42: **Transmit VLAN Table Entry with One Field (offset 0x0000_4000-0x0000_7FFF)**



Receive VLAN Data Table

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 the first following figure while the second figure 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 43: **Receive VLAN Table Entry with all Fields (0x0000_8000-0x0000_BFFF)**

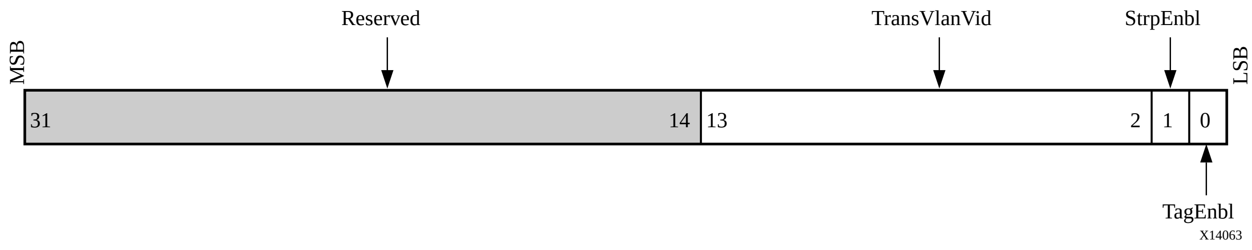
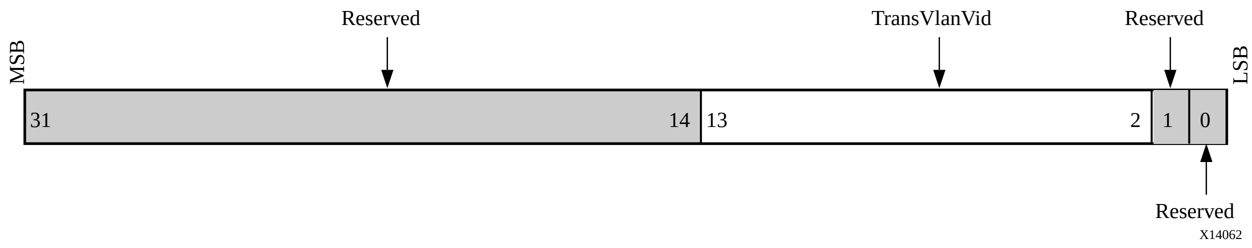


Figure 44: **Receive VLAN Table Entry with One Field (offset 0x0000_8000-0x0000_BFFF)**



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

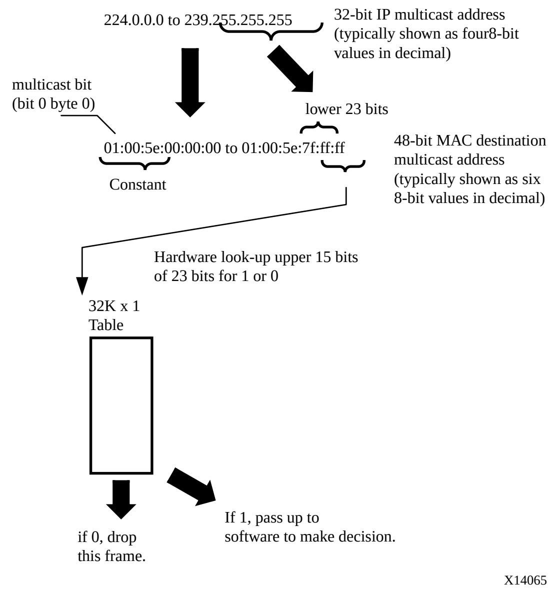
See the version of the Tri-Mode Ethernet MAC core listed in the change log for more information.

Multicast Address Table — Offset 0x0002_0000-0x0003_FFFF

The Multicast Address Table entry is shown in Figure 46: Multicast Address Table Entry (0x0002_0000-0x0003_FFFF). 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 subsystem 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 an Ethernet MAC multicast address is defined as any 48-bit Ethernet MAC address that has bit 0 (LSB) set to 1 (for example 01:00:00:00:00:00), in most cases the Ethernet MAC multicast address is created from a IP multicast address, as shown in the following figure.

Figure 45: **Mapping IP Multicast Addresses to MAC Multicast Addresses**


224.0.0.0 to 239.255.255.255
32-bit IP multicast address (typically shown as four8-bit values in decimal)
multicast bit (bit 0 byte 0)
lower 23 bits
01:00:5e:00:00:00 to 01:00:5e:7f:ff:ff
48-bit MAC destination multicast address (typically shown as six 8-bit values in decimal)
Constant
Hardware look-up upper 15 bits of 23 bits for 1 or 0
32K x 1 Table
if 0, drop this frame.
If 1, pass up to software to make decision.
X14065

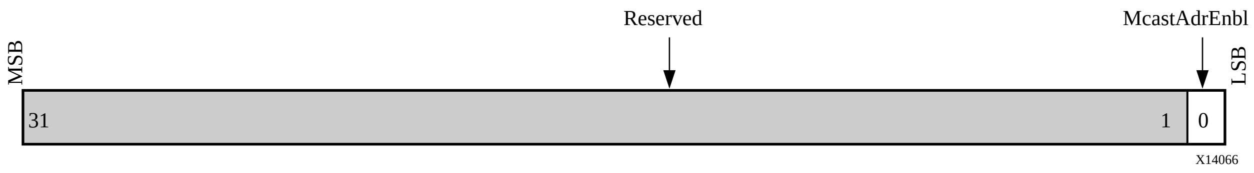
When a multicast address frame is received while this extended multicast filtering is enabled, the subsystem first verifies that the initial 24 bits are 01:00:5E and then uses the upper 15 bits of the unique 23 bit Ethernet MAC multicast address to index this memory. If the associated memory location 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 46: **Multicast Address Table Entry (0x0002_0000-0x0003_FFFF)**



The following table shows the Multicast Address Table bit definitions.

Table 48: **Multicast Address Table (0x0002_0000-0x0003_FFFF)**

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

Enhancements to the 1G/2.5G Ethernet PCS/PMA or Subsystem 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 *1G/2.5G Ethernet PCS/PMA or SGMII LogiCORE IP Product Guide* (PG047).

Table 49: **1588 Control: Vendor Specific Register 19**

Bits	Default Value	Access	Description
15:4	N/A	RO	Reserved

Table 49: **1588 Control: Vendor Specific Register 19** *(cont'd)*

Bits	Default Value	Access	Description
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 the following table. 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 subsystem is initialized following a power-on, reset, or recovery from loss of synchronization; it then remains constant for normal operation. When 0, no adjustment is made to compensate for measurable variable known latencies.

Table 50: **RX PHY Fixed Latency: Vendor Specific Register 20**

Bits	Default Value	Access	Description
15:0	0xC8	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 prior to the timestamping position.

Table 51: **RX PHY Variable Latency: Vendor Specific Register 21**

Bits	Default Value	Access	Description
15:0	N/A	RO	RX 1000BASE-X variable RX Delay in UI. This value is measured within the subsystem 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 subsystem is initialized following a power-on, reset, or recovery from loss of synchronization; it then remains constant for normal operation.

Gigabit Ethernet PCS/PMA Management Registers

The Gigabit Ethernet PCS PMA core 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 provided here for quick reference. For more information about these registers see the *1G/2.5G Ethernet PCS/PMA or SGMII LogiCORE IP Product Guide* (PG047).

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 the subsystem 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 52: **Gigabit Ethernet PCS PMA Internal Management Registers**

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

Table 53: **Control Register (Register 0)**

Bits	Name	Access	Reset Value	Description
15	Reset	R/W Self clearing	0	• 1 = Subsystem Reset • 0 = Normal Operation
14	Loopback	R/W	0	• 1 = Enable Loopback Mode • 0 = Disable Loopback Mode When used with a device-specific transceiver, the subsystem 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	R/W	1	• 1 = Enable Auto-Negotiation Process • 0 = Disable Auto-Negotiation Process

Table 53: **Control Register (Register 0)** *(cont'd)*

Bits	Name	Access	Reset Value	Description
11	Power Down	R/W	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.
10	Isolate[1]	R/W	1	• 1 = Electrically Isolate PHY from GMII • 0 = Normal operation
9	Restart Auto-Negotiation	R/W 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	R/W	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.

Notes:

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 subsystem is not operational until this is completed.

The following table shows the Gigabit Ethernet PCS PMA Management Status register bit definitions.

Table 54: **Management Status Register (Register 1)**

Bits	Name	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.

Table 54: **Management Status Register (Register 1)** *(cont'd)*

Bits	Name	Access	Reset Value	Description
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	RO	0	• 0 – auto-negotiation process not completed • 1 – auto-negotiation process complete
4	Remote Fault	RO 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	RO 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.

The following table shows the first Management PHY Identifier register bit definitions.

Table 55: **Management PHY Identifier (Register 2)**

Bits	Name	Access	Reset Value	Description
15:0	OUI	RO	0x0000	Organizationally Unique Identifier (OUI).

The following table shows the second Management PHY Identifier register bit definitions.

Table 56: **Management PHY Identifier (Register 3)**

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

The following table shows the Management Auto-Negotiation Advertisement register bit definitions.

Table 57: **Management Auto-Negotiation Advertisement Register (Register 4)**

Bits	Name	Access	Reset Value	Description
15	Next Page	R/W	0	• 0 – next page functionality is not advertised • 1 – next page functionality is advertised
14	Reserved	Returns 0s	0	Always return zeros.
13:12	Remote Fault	R/W self clearing after auto-negotiation	0x0	• 00 – no error • 01 – off line • 10 – link failure • 11 – auto-negotiation error
11:9	Reserved	Returns 0s	0x0	Always return zeros.
8:7	Pause	R/W	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	R/W	1	• 0 – full-duplex mode is not advertised • 1 – full-duplex mode is advertised
4:0	Reserved	Returns 0s	0x0	Always return zeros.

The following table shows the TEMAC Internal 1000BASE-X PCS/PMA Management Auto-Negotiation Link Partner Ability Base register bit definitions.

Table 58: **Management Auto-Negotiation Link Partner Ability Base Register (Register 5)**

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

The following table shows the Management Auto-Negotiation Expansion register bit defiitions.

Table 59: **Management Auto-Negotiation Expansion Register (Register 6)**

Bits	Name	Access	Reset Value	Description
15:3	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	RO 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.

The following table shows the Management Auto-Negotiation Next Page Transmit register bit definitions.

Table 60: **Management Auto-Negotiation Next Page Transmit Register (Register 7)**

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

The following table shows the Management Auto-Negotiation Next Page Receive register bit definitions.

Table 61: **Management Auto-Negotiation Next Page Receive Register (Register 8)**

Bits	Name	Access	Reset Value	Description
15	Next Page	RO	0	• 0 – last page • 1 – additional next pages to follow
14	Acknowledge	RO	0	Used by auto-negotiation function to indicate reception of a link partner base or next page.

Table 61: **Management Auto-Negotiation Next Page Receive Register (Register 8)** *(cont'd)*

Bits	Name	Access	Reset Value	Description
13	Message Page	RO	0	• 0 – unformatted page • 1 – message page
12	Acknowledge 2	RO	0	• 0 – cannot comply with message • 1 – complies with message
11	Toggle	RO	0	Value toggles between subsequent pages.
10:0	Message or unformatted Code Field	RO	0x0 (null message code)	Message code field or unformatted page encoding as dictated by bit 13.

The following table shows the Management Extended Status register bit definitions.

Table 62: **Management Extended Status Register (Register 15)**

Bits	Name	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.
11:0	Reserved	Returns 0s	0x0	Always return zeros.

The following table shows the Management Auto-Negotiation Interrupt Control register bit definitions.

Table 63: **Management Auto-Negotiation Interrupt Control Register (Register 16)**

Bits	Name	Access	Reset Value	Description
15:2	Reserved	Returns 0s	0	Always return zeros.

Table 63: **Management Auto-Negotiation Interrupt Control Register (Register 16)** *(cont'd)*

Bits	Name	Access	Reset Value	Description
1	Interrupt Status	R/W	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	R/W	1	• 0 – interrupt is disabled • 1 – interrupt is enabled

Table 64: **Management Loopback Control Register (Register 17)**

Bits	Name	Access	Reset Value	Description
15:1	Reserved	Returns 0s	0	Always return zeros.
0	Loopback Position	R/W	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

Chapter 4

Designing with the Subsystem

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

Design Guidelines

This chapter provides design guidelines that need to be considered when using the subsystem. It is important to understand the features and interfaces provided by the subsystem and as described in the following sections.

Customize and Generate the Subsystem

Generate the subsystem with your desired options using the IP catalog as described in Customizing and Generating the Subsystem.

Examine the Generated Design for This Hierarchical Block

- HDL is generated for this hierarchical block.
- 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 the AXI Ethernet Subsystem in the IP integrator Address editor tab. The AXI Ethernet Subsystem requires a 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 an AMD 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 subsystem. 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 AMD 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 Constraining the Subsystem. Also, see the infrastructure core documentation for more information.

Make Only Allowed Modifications

The AXI Ethernet Subsystem should not be modified. Modifications can have adverse effects on system timing and protocol compliance. Supported user configurations of this subsystem can only be made by selecting the options from within the AMD Vivado™ design tools when the subsystem is generated. See Customizing and Generating the Subsystem. 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) for the targeted FPGA family for more information.

In the Include Shared Logic in IP Example Design configuration, the `s_axi_lite_clk`, `ref_clk`, and `axis_clk` signals for the core are generated from the `*_clocks_resets` module (within the example design) through an MMCME3_ADV followed by BUFGs.

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

While using the core with GT in Example Design on Versal and UltraScale/UltraScale+ devices, the clocks are connected to the core from GT as follows.

Table 65: **User Clocks from GT in Versal and Ultrascale+ devices**

Clock	Description
userclk	Clock for the 16-bit TX data path from the core to the GT. Frequency: 1G: 62.5 Mhz. 2.5G: 156.25 Mhz.
userclk2	Clock for the 8-bit internal data path within the PCS. Frequency: 1G: 125 Mhz. 2.5G: 312.5 Mhz.
rxuserclk	Not used
rxuserclk2	Clock for the 16-bit RX data path from the GT to the core. Unused in 1000/2500 BaseX modes where 1588 timestamping is disabled. The userclk is used when rxuserclk2 is not used. Frequency: 1G: 62.5 Mhz. 2.5G: 156.25 Mhz.

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 Subsystem 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 subsystem. Failure to comply causes unpredictable behavior.

In a system in which Ethernet operates at 10 Mb/s, the Ethernet MAC interface operates at 2.5 MHz. If the AXI4-Lite interface operates at 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 x 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 Subsystem streaming reset inputs.

RECOMMENDED: *As a general design guideline, AMD 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 Subsystem that is uniquely tailored to your design, These parameters can be configured through the Vivado IDE. In addition to the parameters listed in the following table, embedded development kit (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 the following table.

Table 66: **Design Parameters**

Parameter	Default Value	Description
USE_BOARD_FLOW	FALSE	If TRUE, this enables generation of Board based I/O Constraints. If you select a board in the Project settings, the default value will be TRUE.
PHY_TYPE	GMII	Select the PHY Interface type
Enable_1588	FALSE	Enables 1588 timestamping support. This parameter is enabled only when PHY_TYPE is set to 1000BASE-X or SGMII and not in LVDS mode.
Enable_1588_firstep	TRUE	TRUE: 1-step support FALSE: 2-step support
TIMER_CLK_PERIOD	4000	1588 System Timer reference clock period in picoseconds
SupportLevel	1	This verifies if the Shared Logic is in the subsystem or not. Default value is to include shared logic in the subsystem. If Support Level is 1, it specifies *Include Shared Logic in Core. If* SupportLevel is 0, it specifies *Include Shared Logic in IP Example Design*.
ENABLE_LVDS	FALSE	Enable standard I/O (LVDS) for SGMII or 1000BASEX
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 [1]	MDIO PHY Address, range 1 to 31
Statistics_Width	64 bit	Statistics Counter Width

Table 66: **Design Parameters** *(cont'd)*

Parameter	Default Value	Description
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 subsystem. This is available only in GMII mode. When this is FALSE, TEMAC is configured in internal mode.
processor_mode	TRUE	When TRUE, buffer module is included and drivers are available. When FALSE, the buffer module is not available and the drivers are not present. This parameter can only be updated when PHY_TYPE is SGMII or 1000BASE-X.
TransceiverControl	FALSE	This enables the debug interface when a transceiver is instantiated in the AXI 1G/2.5G Ethernet subsystem. When the transceiver debug interface is enabled, the drp clock input port is available at the top level and its frequency is programmable. When the transceiver debug port is disabled, the drp clock port of the transceiver is not available at the top level. It is internally connected to reference clock port and its frequency is fixed.
Enable_Pfc	FALSE	This enables the support for the Priority Flow Control.
speed_1_2p5	1G	This sets the data rate to 1G or 2.5G. • 2.5G: phy_type can only be of 2500BASE-X or 2.5G SGMII mode. The processor_mode is disabled. It supports single data rate of 2.5G. • 1G: The core is backward compatible and supports 10/100/1G speeds for GMII, RGMII and SGMII modes, 1G for 1000BASE-X mode, and 10/100M for MII mode.
drpclkrate	50.0	This is the frequency (in MHz) connected to the drp_clk. This is enabled only when TransceiverControl parameter is enabled. In AMD UltraScale™ devices when TransceiverControl is disabled, this value is fixed at 50.0 MHz.
gtrefclkrate	125	This is the frequency (in MHz) of the GT reference clock. Valid values are selected through Vivado IDE. This is applicable only for UltraScale devices.
lvdsclkrate	125	This is the frequency (in MHz) of the LVDS reference clock. Valid values are selected through Vivado IDE. This is applicable only for UltraScale devices.

Table 66: **Design Parameters** *(cont'd)*

Parameter	Default Value	Description
gt_type	GTH	This selects the GT type when the project part supports multiple GT types. Valid Values are selected through Vivado IDE.

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 Table 70: Transmit AXI4-Stream Control Word 2 – APP1 supported does not vary based on the Ethernet MAC type selected. See the following table through for the supported PHY interface with TEMAC.

Table 67: **Supported PHY Speeds Based on PHY Modes**

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

Notes:

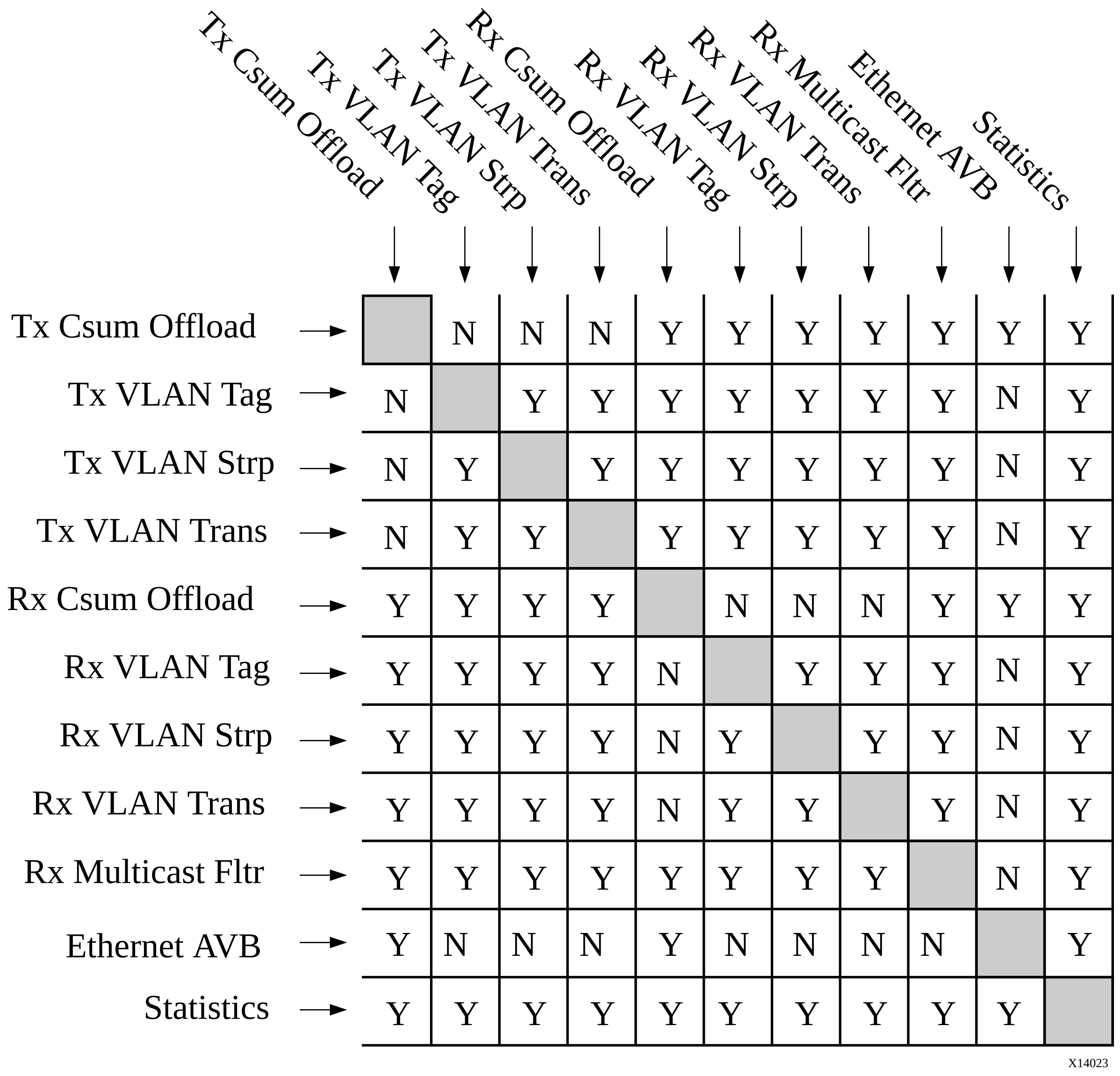
1. For AMD UltraScale+™ devices, the RGMII interface cannot be placed on HD I/O, because there are no IDELAY/ODELAYs in HD I/O.

For supported IO voltages in the following devices, refer to the documents mentioned below:

- *UltraScale Architecture SelectIO Resources User Guide* (UG571)
- *Versal Adaptive SoC SelectIO Resources Architecture Manual* (AM010)

Some of the optional functions provided by the AXI Ethernet Subsystem are not compatible with other optional functions. The following figure shows which of these functions are compatible with each other. In the following figure , TX/RX CSUM Offload refers to both Partial Checksum Offloading and Full Checksum Offloading.

Figure 47: **Option Function Compatibility**



	Tx Csum Offload	Tx VLAN Tag	Tx VLAN Strp	Tx VLAN Trans	Rx Csum Offload	Rx VLAN Tag	Rx VLAN Strp	Rx VLAN Trans	Rx Multicast Fltr	Ethernet AVB	Statistics
Tx Csum Offload		N	N	N	Y	Y	Y	Y	Y	Y	Y
Tx VLAN Tag	N		Y	Y	Y	Y	Y	Y	Y	N	Y
Tx VLAN Strp	N	Y		Y	Y	Y	Y	Y	Y	N	Y
Tx VLAN Trans	N	Y	Y		Y	Y	Y	Y	Y	N	Y
Rx Csum Offload	Y	Y	Y	Y		N	N	N	Y	Y	Y
Rx VLAN Tag	Y	Y	Y	Y	N		Y	Y	Y	N	Y
Rx VLAN Strp	Y	Y	Y	Y	N	Y		Y	Y	N	Y
Rx VLAN Trans	Y	Y	Y	Y	N	Y	Y		Y	N	Y
Rx Multicast Fltr	Y	Y	Y	Y	Y	Y	Y	Y		N	Y
Ethernet AVB	Y	N	N	N	Y	N	N	N	N		Y
Statistics	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	

The AXI Ethernet Subsystem 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 Subsystem 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 subsystem 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 subsystem. 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 Subsystem provides memory buffering of transmit and receive Ethernet frames, thereby allowing more optimal transfer to and from the subsystem 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 Subsystem 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 Subsystem 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.

AXI4-Stream Interface

The Ethernet frame data to be transmitted and the frame data that is received passes between the AXI Ethernet Subsystem and the rest of the integrated 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 Subsystem AXI4-Stream interface.

The AXI4-Stream interface is a high-performance, synchronous, point-to-point connection which is described in Arm AMBA 4 AXI4-Stream Protocol v1.0 Specification (ARM IHI 0051A). This section describes the specific 32-bit implementation used by the AXI Ethernet Subsystem to transfer transmit and receive Ethernet frame data with the rest of the integrated system. The AXI4-Stream model used is called Packetized and Aligned Strobe, which is defined in the following 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 in the middle of a transfer, or at the end of a transfer. This means that the 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 indicate 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 integrated system, typically a DMA controller, and the *Destination* is the AXI Ethernet Subsystem. For the receive datapath the *Source* is the AXI Ethernet and the *Destination* is the integrated 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 (Refer to the following figure and Receive Status Transmit AXI4-Stream Transfer – Flag=0x5). The write strobe signals for the control and status buses are always in the active state (0xF).

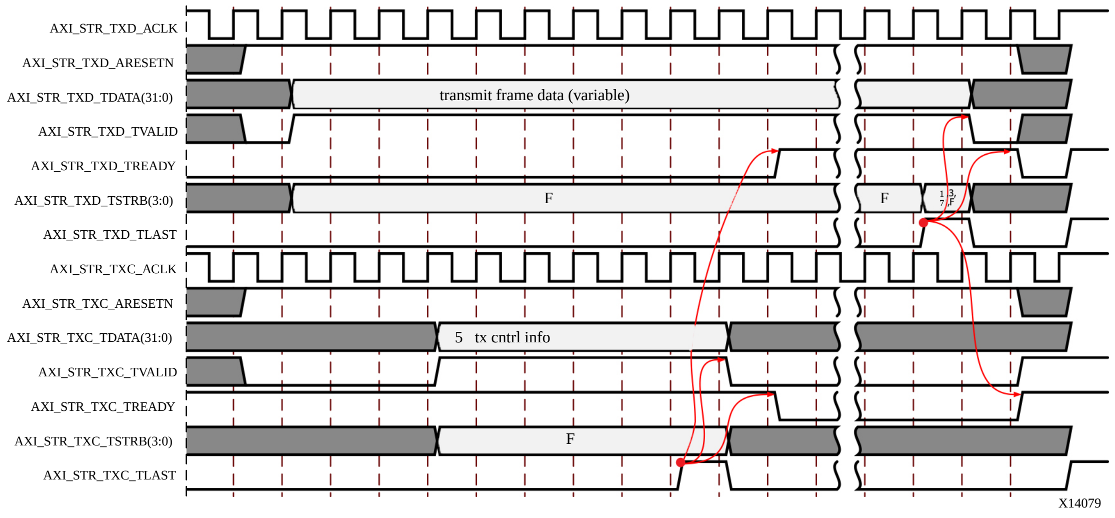
The right-most write strobe signal for the data bus is always in the active state (0x1, 0x3, 0x7, or 0xF). These signals can be tied off rather than routing signals from the AXI4-Stream source to the destination. The AXI Ethernet Subsystem provides these ports to be compliant with the standard; however, 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 Subsystem to any external core receiving the TX data.

The following figure 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 48: **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 Subsystem. The following figure and tables 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. 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 the following figure for the Transmit AXI4-Stream Control Word definition.

Figure 49: **Transmit AXI4-Stream Control Words**

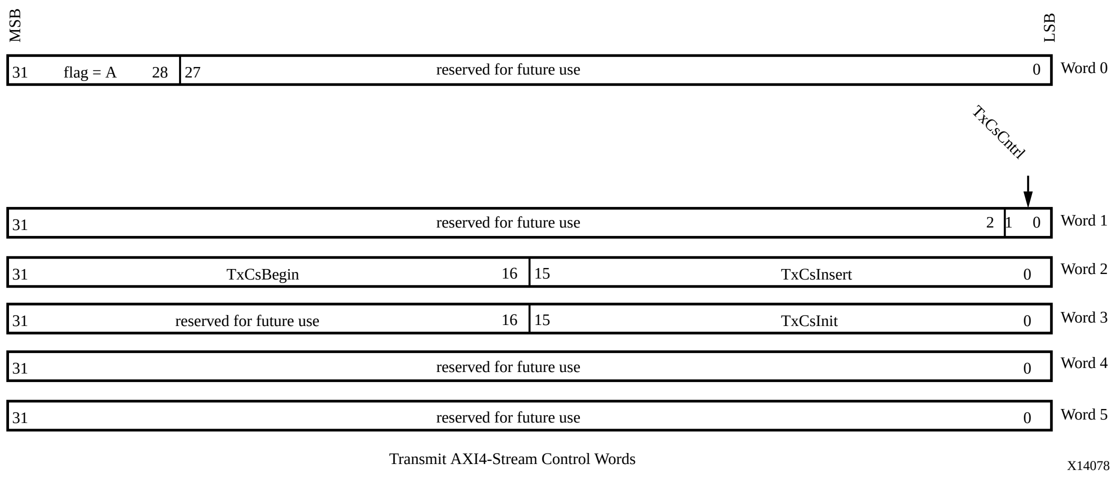

MSB
LSB
31 flag = A 28
27 reserved for future use 0
Word 0
TxCsCntrl
31 reserved for future use 2 1 0
Word 1
31 TxCsBegin 16
15 TxCsInsert 0
Word 2
31 reserved for future use 16
15 TxCsInit 0
Word 3
31 reserved for future use 0
Word 4
31 reserved for future use 0
Word 5
Transmit AXI4-Stream Control Words
X14078

Table 68: **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 69: **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 70: **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.

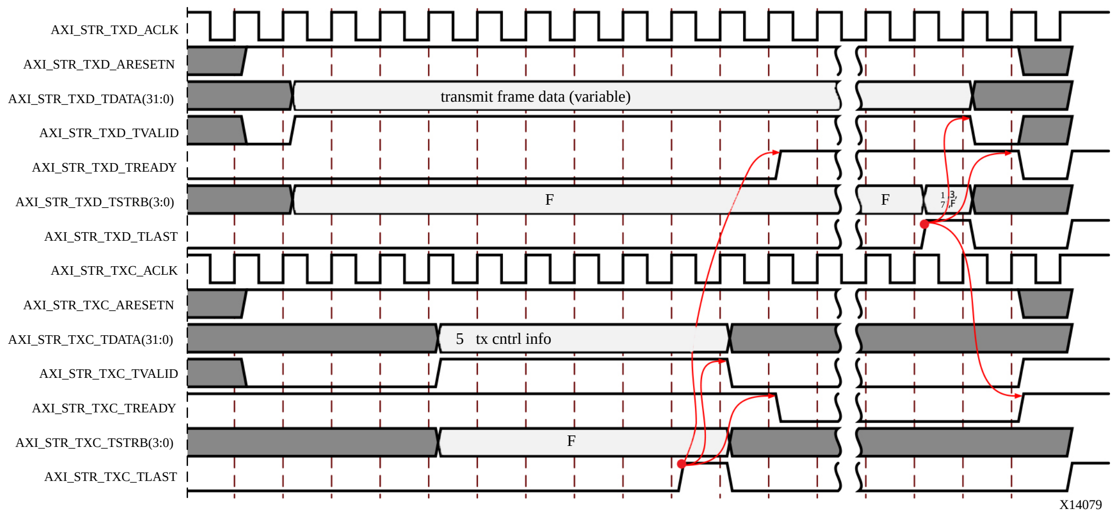
Table 71: **Transmit AXI4-Stream Control Word 3 – APP2**

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 used 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 subsystem at build time. Note the different identification flag value in the following figure.

Figure 50: **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 the Ethernet data, the receive channel initiates transfer on the AXI4-Stream Status Interface before starting the transfer on AXI4-Stream Data Interface. The completion of the transfers on these interfaces depends on their throttling. See Figure 53: Receive AXI DMA Buffer Descriptor AXI4-Stream Field Mapping 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 contains six 32-bit status words (words 0 to 5). The following figure and tables 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 51: **Receive AXI4-Stream Status Words**

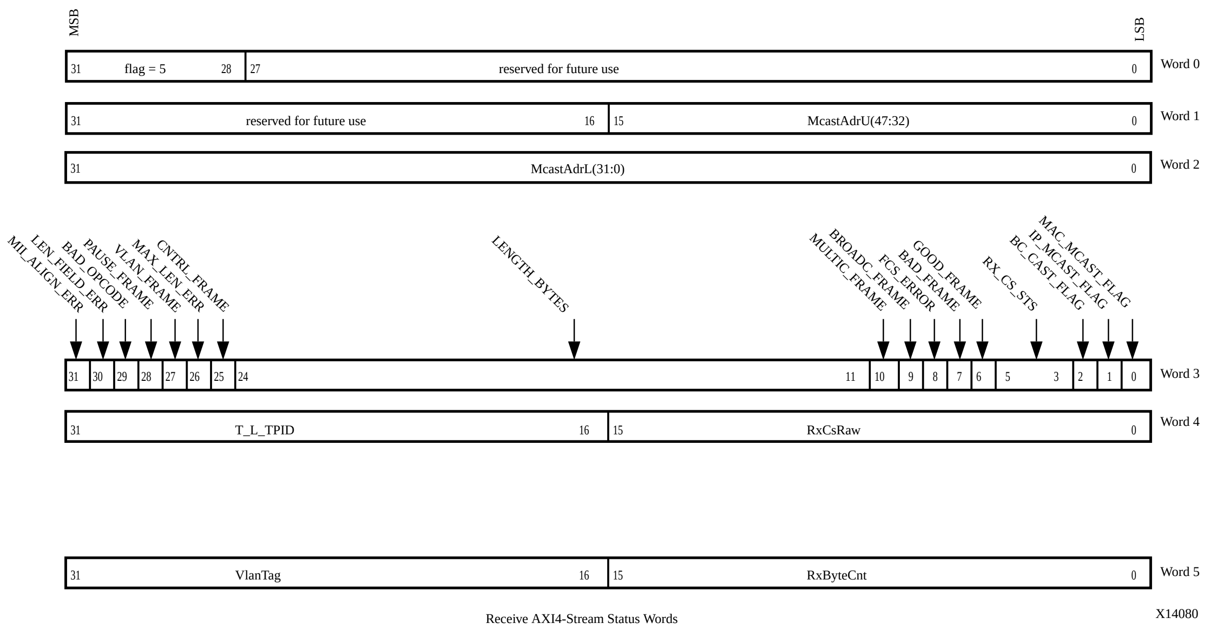


Table 72: **Receive AXI4-Stream Status Word 0 – TAG**

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

Table 73: **Receive AXI4-Stream Status Word 1 – APP0**

Bits	Name	Description
31–16	Reserved	Undefined value.

Table 73: **Receive AXI4-Stream Status Word 1 – APP0** *(cont'd)*

Bits	Name	Description
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, an Ethernet MAC address of AA-BB-CC-DD-EE-FF would be stored in UnicastAddr(47:0) as 0xFFEEDDCCBBAA. This word would be 0xFFEE.

Table 74: **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, an Ethernet MAC address of AA-BB-CC-DD-EE-FF would be stored in UnicastAddr(47:0) as 0xFFEEDDCCBBAA. This word would be 0xDDCCBBAA.

Table 75: **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.

Table 75: **Receive AXI4-Stream Status Word 3 – APP2** *(cont'd)*

Bits	Name	Description
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 16,383 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.
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 an Ethernet MAC multicast frame that has passed the hardware multicast address filtering.

Table 76: **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 77: **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 1fifth and 1fourth 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 Subsystem requires that certain AXI4-Stream Control/Status words be used to support TCP/IP Checksum Offload. The subsystem 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,AMD 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 Subsystem. 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.

The following first figure shows the mapping between the AXI DMA Buffer Descriptor words in external memory and the fields in the transmit AXI4-Stream case and the second figure 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 Subsystem.

Figure 52: **Transmit AXI DMA Buffer Descriptor AXI4-Stream Field Mapping**

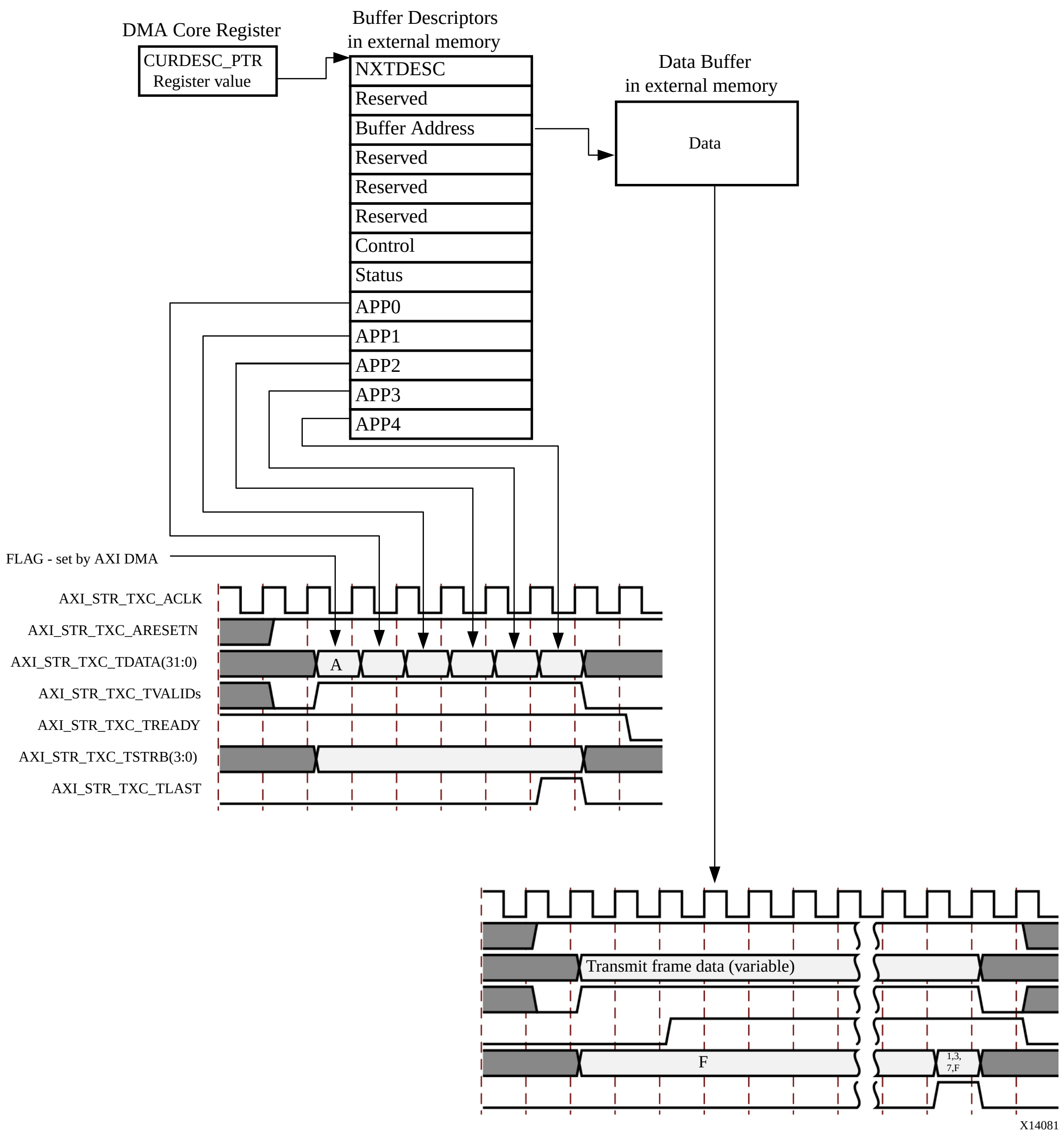
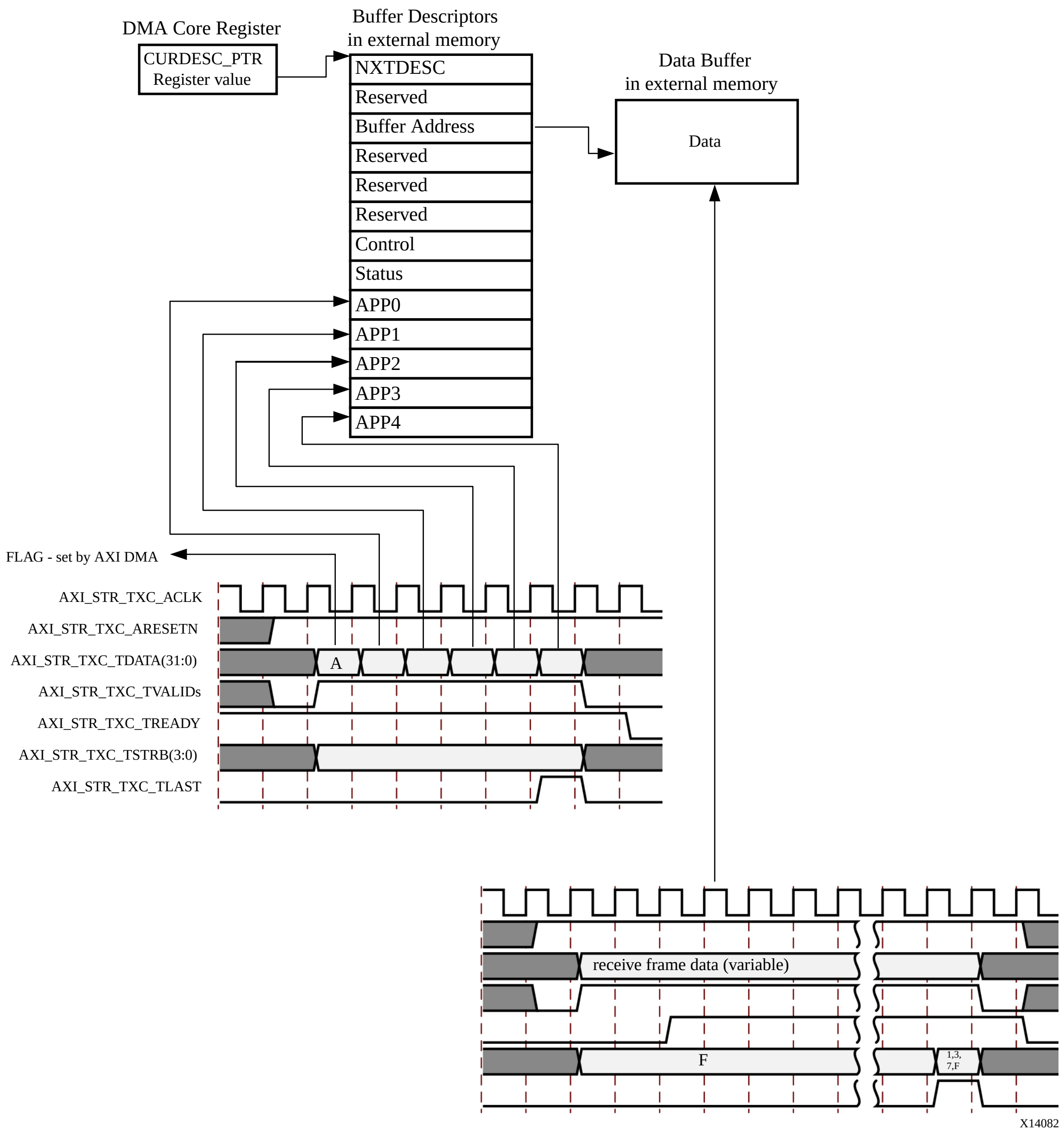


Figure 53: **Receive AXI DMA Buffer Descriptor AXI4-Stream Field Mapping**



Ethernet Audio Video Bridging

Ethernet Audio Video Bridging (AVB) functionality is supported by the subsystem. 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 Tri-Mode Ethernet MAC LogiCORE IP Product Guide (PG051).

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 Subsystem 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 AMD. Several third-party partners have created this logic, have tested it with the subsystem, 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 performed in FPGA logic.

Ethernet AVB frames are passed back and forth to the Ethernet through the AVB AXI4-Stream interfaces. Internal to AXI Ethernet Subsystem, 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 Subsystem 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 Subsystem 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 Subsystem 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 Subsystem 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 Subsystem 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. The following first figure 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. The second figure shows the TX client interface operating at 100 Mb/s.

Figure 54: **1000 Mb/s Transmit AVB AXI4-Stream (back-to back)**

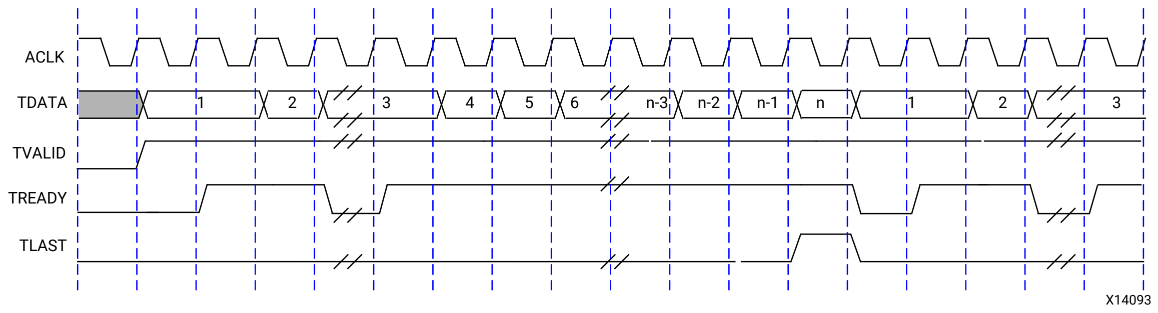
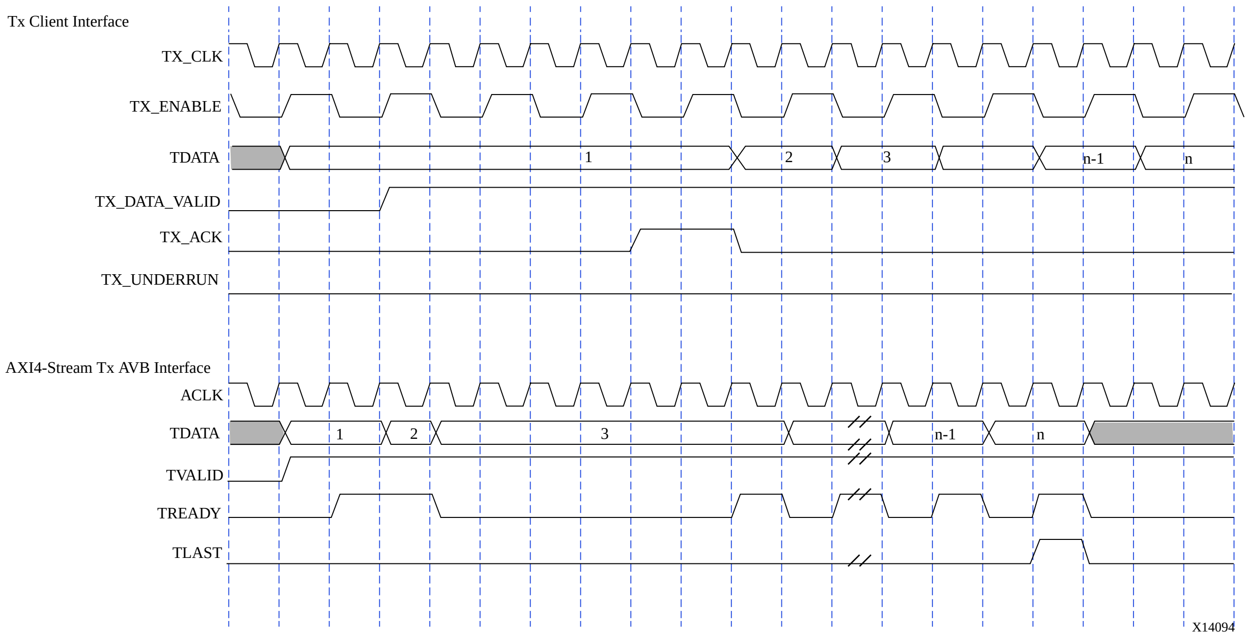


Figure 55: **100 Mb/s Transmit AVB AXI4-Stream**



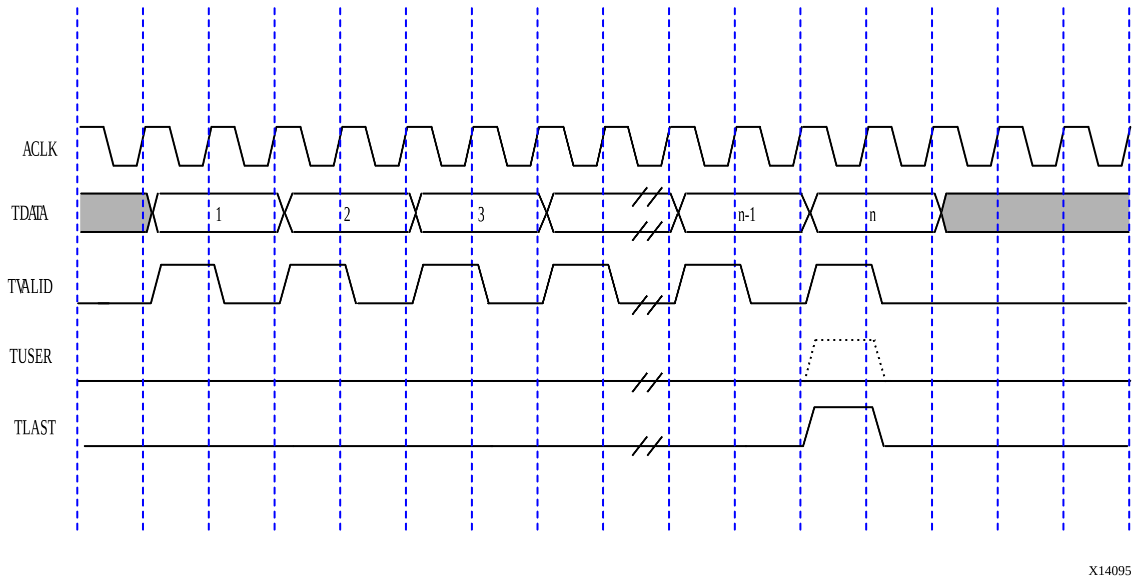
Receive Interface

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 the AXI Ethernet Subsystem 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 the AXI Ethernet Subsystem 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`. The following figure shows the receive waveforms for the AVB interface operating at 100 Mb/s.

Figure 56: **100 Mb/s Receive AVB AXI4-Stream**



Chapter 5

Design Flow Steps

This section describes customizing and generating the subsystem, constraining the subsystem, and the simulation, synthesis, and implementation steps that are specific to this IP subsystem. More detailed information about the standard AMD Vivado™ design flows and 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)
- *Vivado Design Suite User Guide: Designing with IP* (UG896)
- *Vivado Design Suite User Guide: Getting Started* (UG910)
- *Vivado Design Suite User Guide: Logic Simulation* (UG900)

Customizing and Generating the Subsystem

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

The AXI Ethernet Subsystem can be customized and generated 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). You can customize the AXI Ethernet Subsystem for use in your design by specifying values for the various parameters associated with the subsystem following these 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 Subsystem to the block design.
3. Double-click the AXI Ethernet Subsystem to open up a configuration dialog box in the 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 the AXI Ethernet Subsystem. If you want to edit advanced parameters of the sub-cores, you must build the system by directly instantiating that core. You cannot modify internal connections of the AXI Ethernet Subsystem. Configurations that are not allowed or not valid in a particular mode are grayed 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.

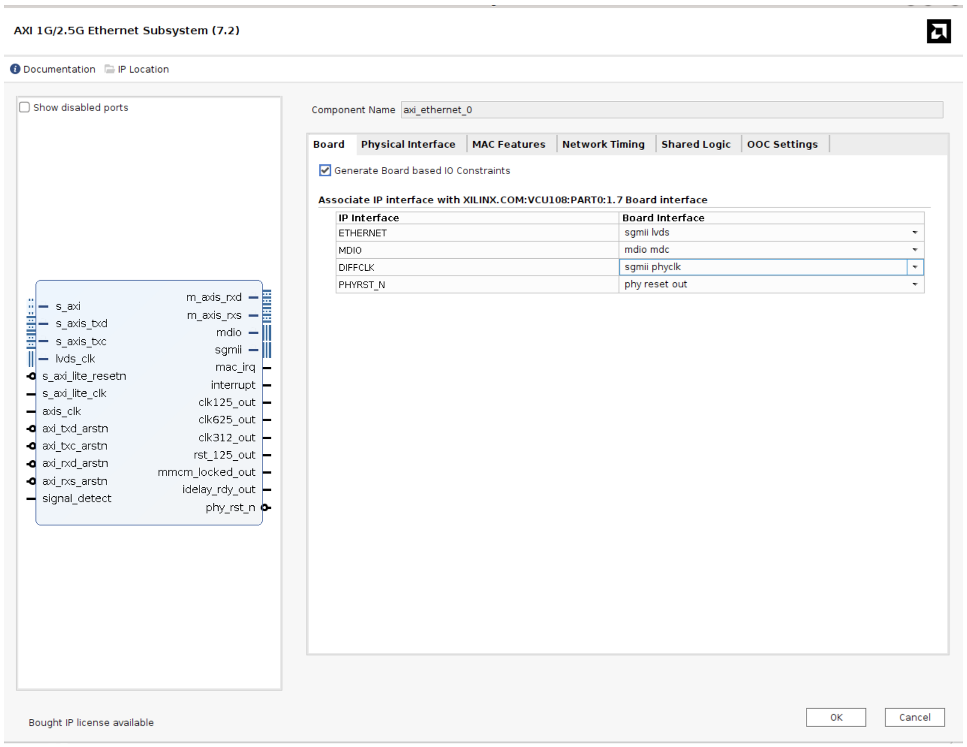
Customizing the Subsystem in the Vivado IDE

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

The Board tab (available for non-AMD Versal™ Adaptive SoC only): The Board tab (see the following figure) is visible only when you select a non-Versal Adaptive SoC board 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).

- **Generate Board based IO Constraints:** 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 the board.
    - **DIFF_CLK:** This option allows you to select the serial transceiver clock source if that source is available on the board.
    - **PHYRST_N:** This option allows you to generate reset for an external PHY device. The minimum duration of this reset is 10ms at the maximum AXI4-Stream clock.

Figure 57: **Board Tab of the Configuration Dialog Box**



Component Name axi_ethernet_0

[✓] Generate Board based IO Constraints

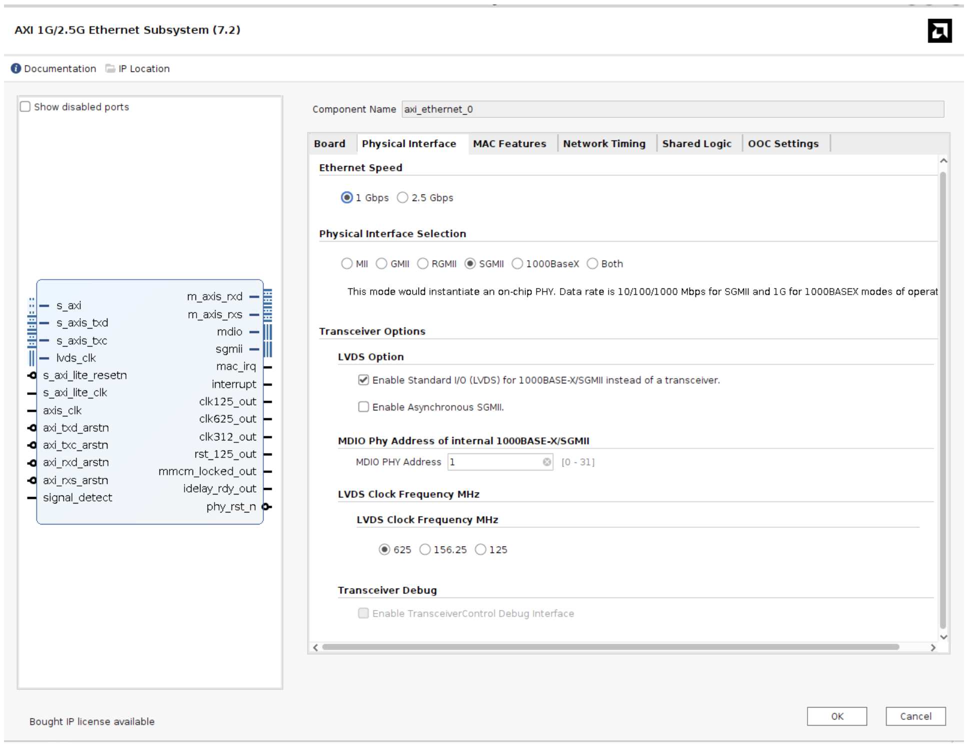
Associate IP interface with XILINX.COM:VCU108:PART0:1.7 Board interface

IP Interface	Board Interface
ETHERNET	sgmii lvds
MDIO	mdio mdc
DIFFCLK	sgmii phyclk
PHYRST_N	phy reset out

The Physical Interface tab: This tab is related to the physical interfaces that are available. The following options are available in this tab:

- **Ethernet Speed:** Supported types are 1 Gb/s and 2.5 Gb/s.
- **Physical Interface Selection:** The physical interface type is selected using this option. The PHY types supported are MII, GMII, RGMII, SGMII, 1000BASE-X, and both.
- **SGMII LVDS Option:** The LVDS mode is enabled only in the SGMII or 1000BASE-X mode of operation.
- **MDIO PHY Address:** PHY address is the address for the MDIO interface for the Gigabit Ethernet PCS PMA PHY address. The PHY address can be set using this option.
- **GT Reference Clock Frequency MHz:** This is the reference clock input to GT. The valid values are 125 MHz, 156.25 MHz, 250 MHz, and 312.5 MHz. For Versal devices, 500MHz and 625 MHz reference clock input is supported in addition to the above list.
- **Enable Transceiver Control Debug Interface:** If selected, enables addition transceiver control ports for TX Driver, RX Equalization and other features such as PRBS. This is available only in non LVDS mode in SGMII or 1000BASE-X modes.

Figure 58: **Physical Interface Tab of the Configuration Dialog Box**



Component Name axi_ethernet_0

Ethernet Speed

[✓ 1 Gbps] 2.5 Gbps

Physical Interface Selection

MII GMII RGMII [✓ SGMII] 1000BaseX Both

This mode would instantiate an on-chip PHY. Data rate is 10/100/1000 Mbps for SGMII and 1G for 1000BASEX modes of operat

Transceiver Options

LVDS Option

[✓] Enable Standard I/O (LVDS) for 1000BASE-X/SGMII instead of a transceiver.

Enable Asynchronous SGMII.

MDIO Phy Address of internal 1000BASE-X/SGMII

MDIO PHY Address 1 [0 - 31]

LVDS Clock Frequency MHz

LVDS Clock Frequency MHz

[✓ 625] 156.25 125

Transceiver Debug

Enable TransceiverControl Debug Interface

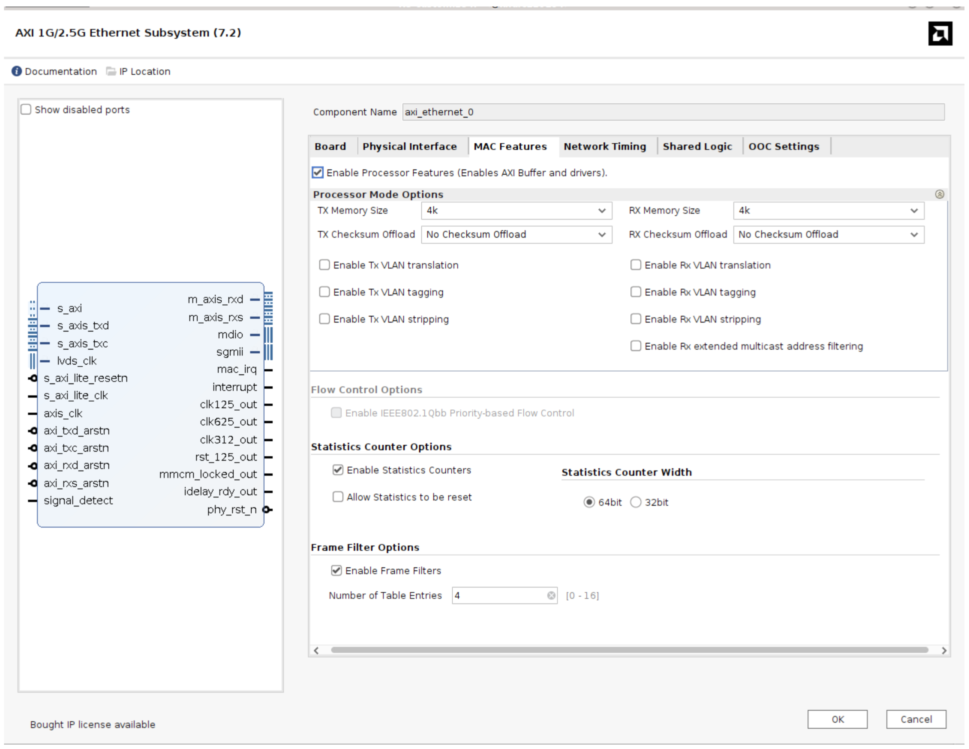
Bought IP license available

The MAC Features tab: The MAC Features tab has options related to the Ethernet MAC functionality. The following options are available in this tab:

- **Enable Processor Features**: In processor mode, the axi_ethernet_buffer is present and driver support is available. Processor mode can be disabled only when PHY_TYPE is 1000BASE-X or SGMII. For 7 series devices, there is no processor mode support for the 2.5G data rate.
- Processor Mode Options
  - TX and RX memory sizes can be selected using the TX Memory Size and RX Memory Size options.
  - TX and RX checksum offload capability can be selected using the RX Checksum Offload and TX Checksum Offload options.
  - RX extended multicast filtering can be enabled with the 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 options.
- **Flow Control Options**: Enables priority-based flow control options. This can be enabled only when processor_mode is deselected.

- **Statistics Counter Options**: Use this option to enable the statistic counters. Enable Statistics Counters should be selected for this purpose. By selecting the **Allow Statistics to be reset** option, the statistics reset capability can be enabled. The width of the statistics counter can be selected using the Statistics Counter Width option.
- **Frame Filter Options**: Enables the TEMAC filters.

Figure 59: **MAC Features Tab of the Configuration Dialog Box**



AXI 1G/2.5G Ethernet Subsystem (7.2)

Documentation | IP Location

[ ] Show disabled ports

Component Name: axi_ethernet_0

Board | Physical Interface | MAC Features | Network Timing | Shared Logic | OOC Settings

[✓] Enable Processor Features (Enables AXI Buffer and drivers).

Processor Mode Options

TX Memory Size: 4k RX Memory Size: 4k

TX Checksum Offload: No Checksum Offload RX Checksum Offload: No Checksum Offload

[ ] Enable Tx VLAN translation [ ] Enable Rx VLAN translation

[ ] Enable Tx VLAN tagging [ ] Enable Rx VLAN tagging

[ ] Enable Tx VLAN stripping [ ] Enable Rx VLAN stripping

[ ] Enable Rx extended multicast address filtering

Flow Control Options

[ ] Enable IEEE802.1Qbb Priority-based Flow Control

Statistics Counter Options

[✓] Enable Statistics Counters

[ ] Allow Statistics to be reset

Statistics Counter Width

● 64bit ○ 32bit

Frame Filter Options

[✓] Enable Frame Filters

Number of Table Entries: 4 [0 - 16]

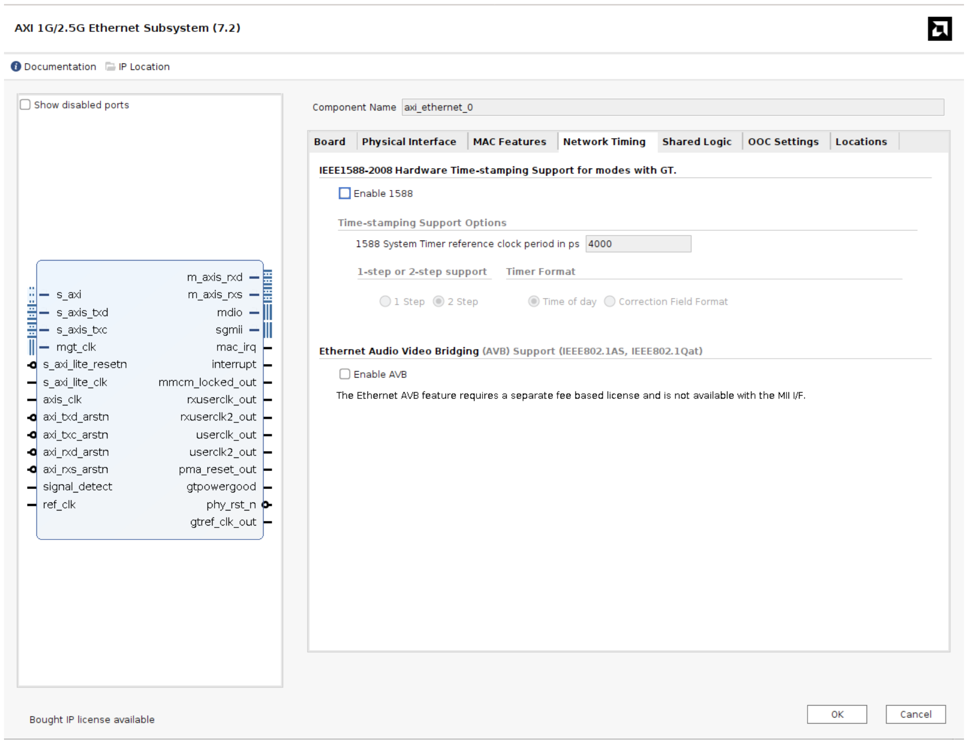
Bought IP license available

OK | Cancel

The Network Timing tab: The 1588 options are enabled only when PHY_TYPE is 1000BASE-X or SGMII in 1G and 2.5G modes of operation and not in LVDS mode. This tab is used to configure the 1588 and AVB modes. 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. TCP/UDP hardware checksum offload is *not* supported/calculated when the 1588 feature 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 Format timer and timestamp format.
  - **1588 System Timer reference clock period in ps** : This is the 1588 system reference clock period in picoseconds.
- **Enable AVB**: This enables Audio Video Bridging functions. This can be enabled only when 1588 mode is disabled.

Figure 60: **Network Timing Tab of the Configuration Dialog Box**



AXI 1G/2.5G Ethernet Subsystem (7.2)

Component Name axi_ethernet_0

IEEE1588-2008 Hardware Time-stamping Support for modes with GT.

☐ Enable 1588

Time-stamping Support Options

1588 System Timer reference clock period in ps 4000

1-step or 2-step support: 1 Step [✓ 2 Step]

Timer Format: [✓ Time of day] Correction Field Format

Ethernet Audio Video Bridging (AVB) Support (IEEE802.1AS, IEEE802.1Qat)

☐ Enable AVB

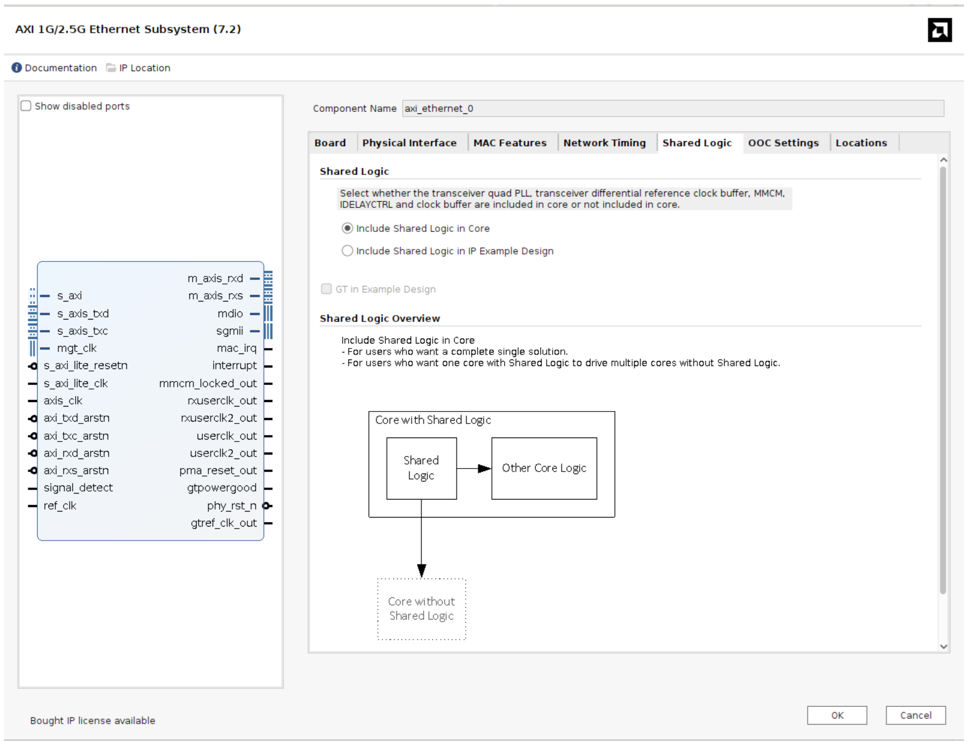
The Ethernet AVB feature requires a separate fee based license and is not available with the MII I/F.

Bought IP license available

The Shared Logic tab (available for non-Versal Adaptive SoC only): The Shared Logic tab (See the following figure) selects whether shared logic is included in the subsystem. 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 AMD Artix™ 7 or AMD Kintex™ 7 devices are shared logic. There is no shareable logic in MII mode. In SGMII using transceiver mode or 1000BASE-X Mode, the transceiver differential reference clock buffer, mixed mode clock manager (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 subsystem.

GT in Example Design: This option is available only when you select Include Shared Logic in IP Example Design option (available for non-Versal Adaptive SoC only). This moves the transceiver from the core to the Support level instance. When this option is selected, generation of additional transceiver control and status ports is disabled. This option is available only for AMD UltraScale™ and AMD UltraScale+™ architecture designs.

Figure 61: **Shared Logic Tab of the Configuration Dialog Box**



AXI 1G/2.5G Ethernet Subsystem (7.2)

Documentation   IP Location

☐ Show disabled ports

Component Name axi_ethernet_0

Board | Physical Interface | MAC Features | Network Timing | Shared Logic | OOC Settings | Locations

Shared Logic

Select whether the transceiver quad PLL, transceiver differential reference clock buffer, MMCM, IDELAYCTRL and clock buffer are included in core or not included in core.

[✓ Include Shared Logic in Core]

Include Shared Logic in IP Example Design

☐ GT in Example Design

Shared Logic Overview

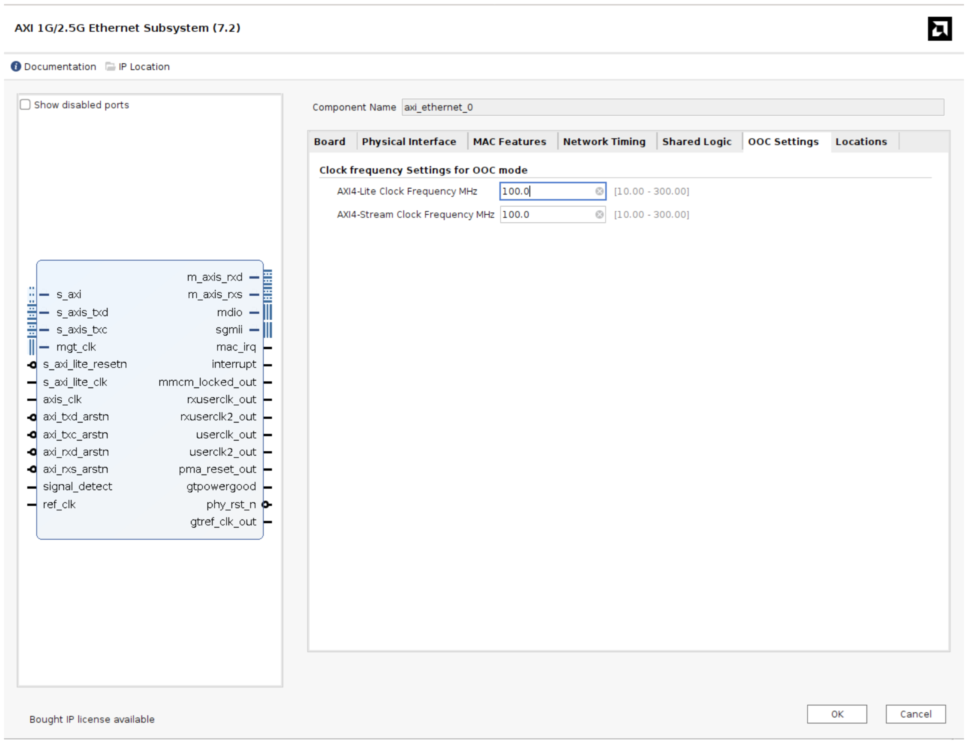
Include Shared Logic in Core
- For users who want a complete single solution.
- For users who want one core with Shared Logic to drive multiple cores without Shared Logic.

Bought IP license available

OK   Cancel

The OOC Settings tab: In OOC mode, these clock frequency values are used by the synthesis tool.

Figure 62: **OOC Settings Tab of the Configuration Dialog Box**



AXI 1G/2.5G Ethernet Subsystem (7.2)

Documentation IP Location

Show disabled ports

Component Name axi_ethernet_0

Board | Physical Interface | MAC Features | Network Timing | Shared Logic | OOC Settings | Locations

Clock frequency Settings for OOC mode

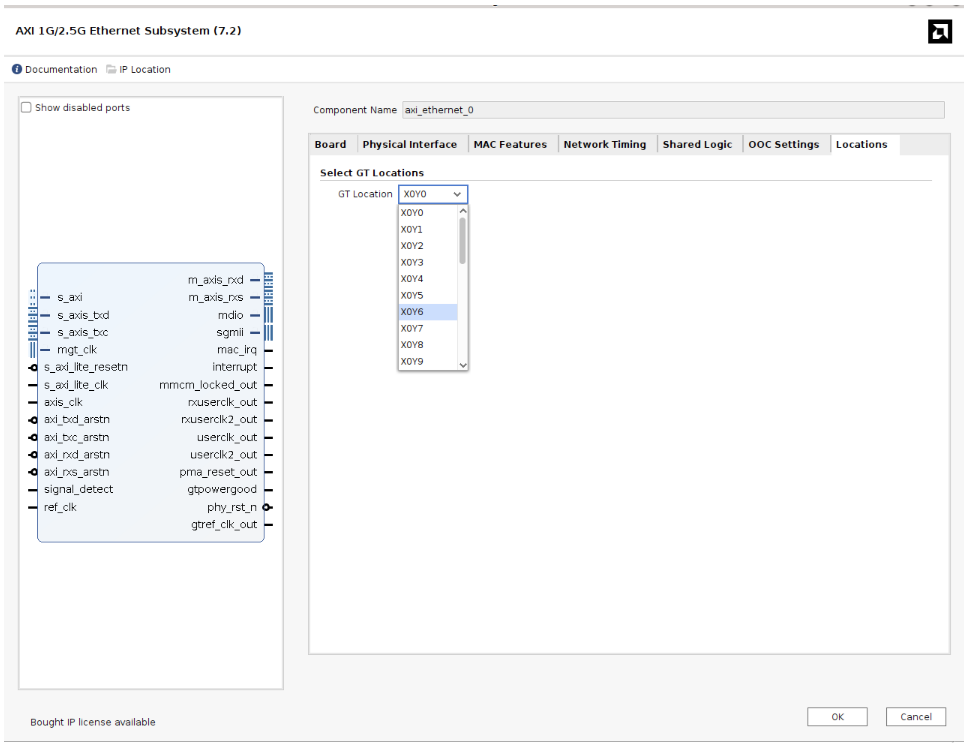
AXI4-Lite Clock Frequency MHz 100.0 [10.00 - 300.00]

AXI4-Stream Clock Frequency MHz 100.0 [10.00 - 300.00]

Bought IP license available

The Locations tab: This tab is available for UltraScale and UltraScale+ devices in SGMII and 1000BASE-X modes. In non-LVDS mode the gt xy coordinates can be selected. In Asynchronous 1000BASE-X/SGMII over LVDS mode the bit slice locations need to be selected. Refer to the “Layout and Placement” section of the *1G/2.5G Ethernet PCS/PMA or SGMII LogiCORE IP Product Guide* (PG047).

Figure 63: **Locations Tab**


AXI 1G/2.5G Ethernet Subsystem (7.2)

Documentation IP Location

Show disabled ports

Component Name axi_ethernet_0

Board | Physical Interface | MAC Features | Network Timing | Shared Logic | OOC Settings | Locations

Select GT Locations

GT Location X0Y0

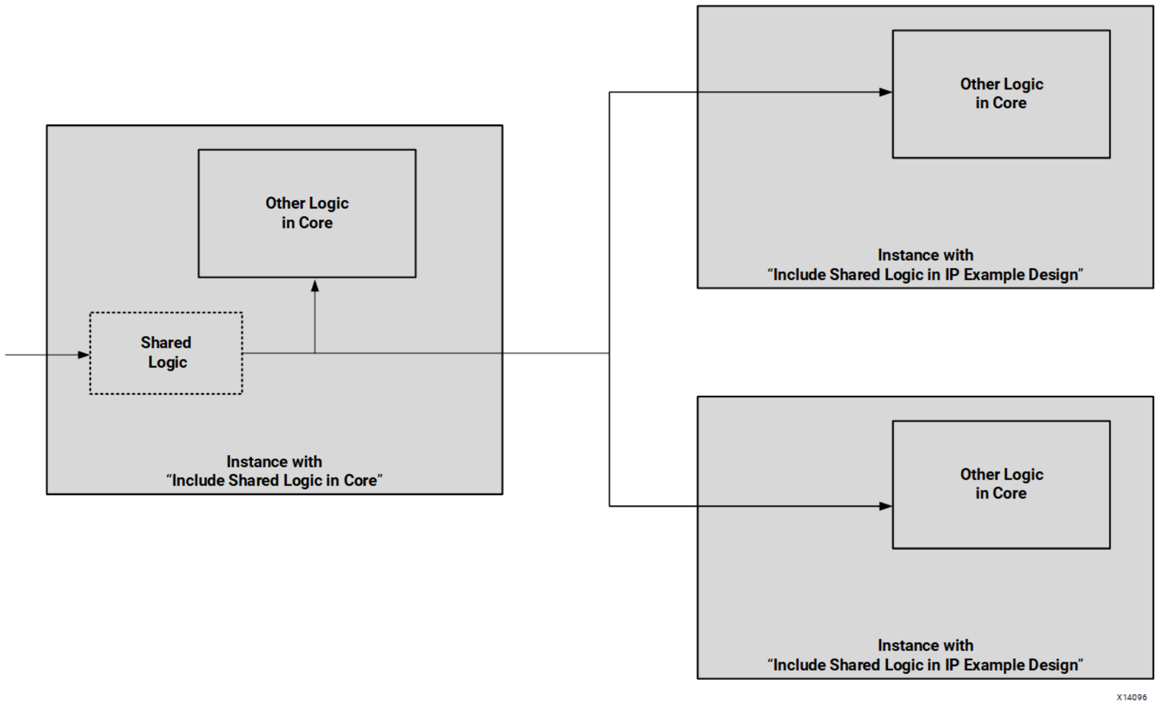
Bought IP license available

Customizing the Subsystem to Share Resources across Multiple Instances

When using multiple instances of the subsystem, based on the mode, certain types of resources can be shared across these instances. Particularly with devices that are not Versal Adaptive SoC, one IP instance should be configured to Include Shared Logic in Core and the remaining instances configured to Include Shared Logic in IP Example Design. The following figure shows that the outputs from a subsystem configured to Include Shared Logic in Core are connected to the inputs of the subsystem configured to Include Shared Logic in IP Example Design .

If only one instance of AXI Ethernet is present, it is recommended to configure it in Include Shared Logic in Core mode.

Figure 64: **Subsystem Sharing Resources among Multiple Instances Using Shared Logic**

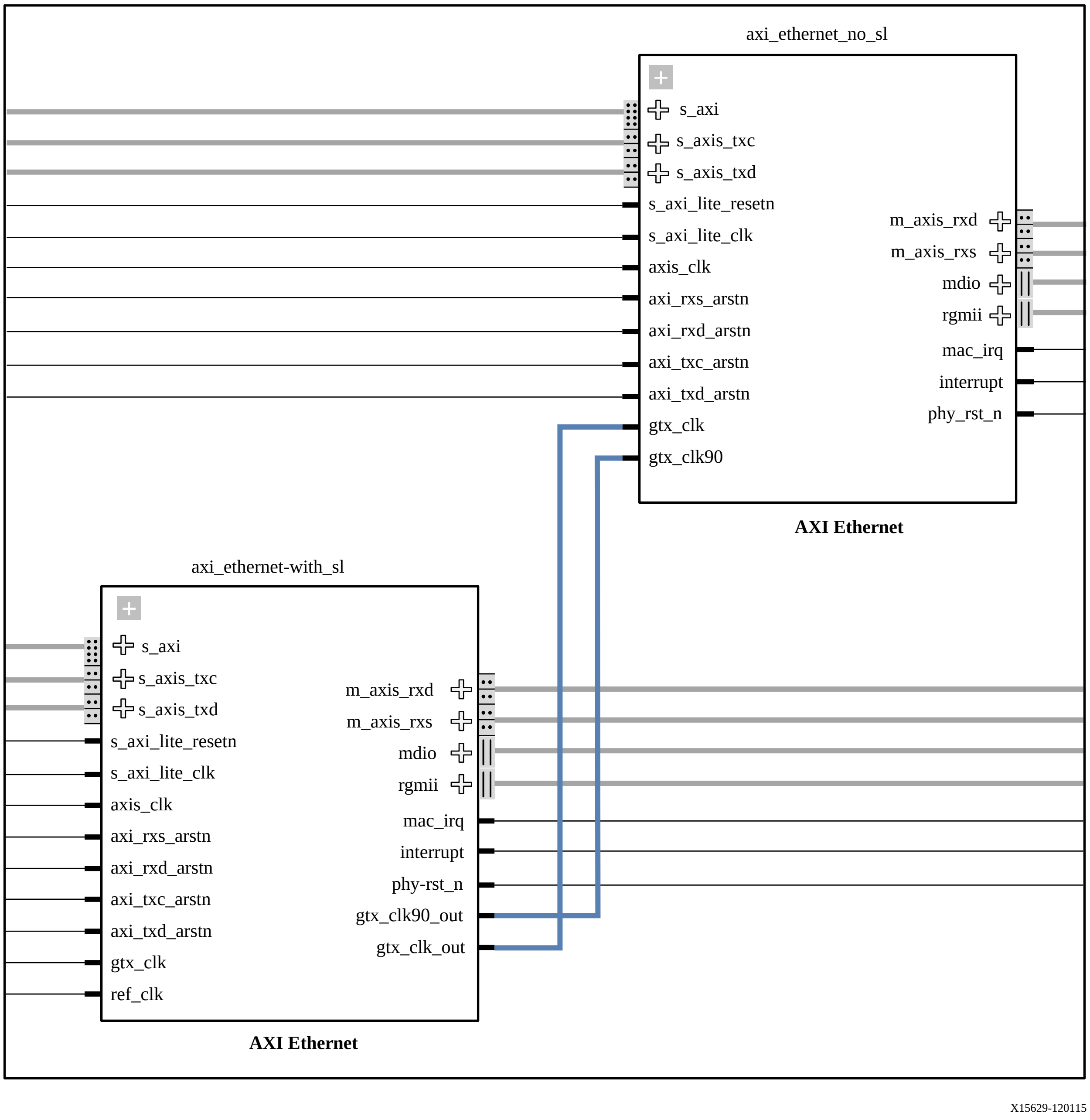

Other Logic in Core
Instance with "Include Shared Logic in IP Example Design"
Other Logic in Core
Shared Logic
Instance with "Include Shared Logic in Core"
Other Logic in Core
Instance with "Include Shared Logic in IP Example Design"
X14096

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

Example 1: Sharing IDELAYCTRL in RGMII Mode among Multiple Subsystem Instances(Applicable for Non-Versal Devices)

When multiple subsystem 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 to Include Shared Logic in Core need to be connected to inputs of the instance configured to Include Shared Logic in IP Example Design . This is shown in the following figure. The connections shown in blue are specific to connections for sharing logic. The rest are regular connections.

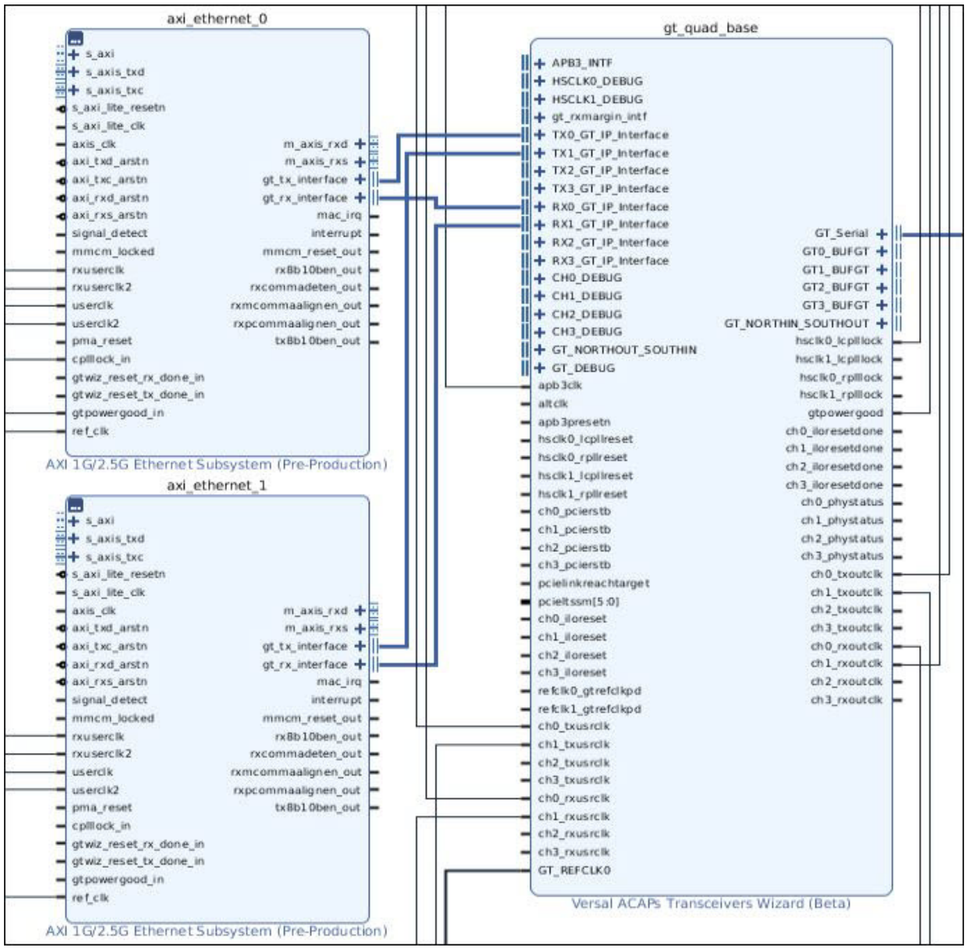
Figure 65: **Connections to Enable Sharing Logic among Two Instances**



Example 2: Sharing GT Quad Base IP in 1000BASE-X Mode among Multiple AXI Ethernet Subsystem Instances (Applicable for Versal Adaptive SoC)

For Versal Adaptive SoC, the GT is always present in the example design so that you can map multiple instances of the AXI Ethernet IP to different channels of the same GT Quad Base IP, or with a new GT Quad Base.

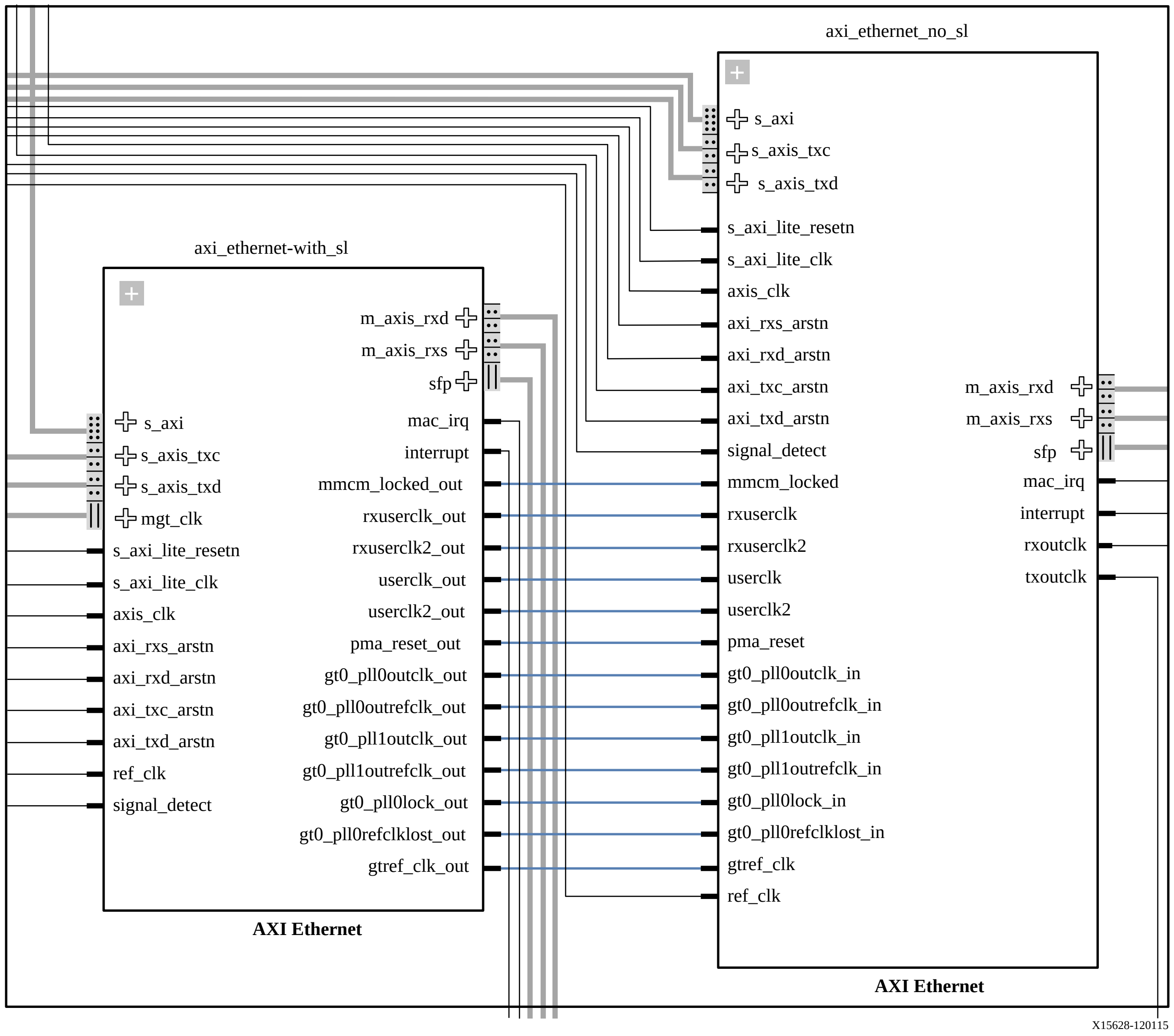
Figure 66: **Connections to Enable Sharing of the GT Quad Base IP among Two Instances**



Example 3: Sharing GT COMMON in 1000BASE-X Mode among Multiple Subsystem Instances (Applicable for Non-Versal Devices)

When multiple subsystem instances are targeted for transceivers in the same quad, the GTCOMMON and IBUFDS need to be shared. The transceiver outputs of the instance configured to Include Shared Logic in Core need to be connected to inputs of an instance configured to Include Shared Logic in IP Example Design . An example where the RX clock is synchronous between both the instances is shown in the following figure. The connections shown in blue are specific to connections for sharing logic. The rest are regular connections.

Figure 67: **Connections to Enable Sharing of GTCOMMON among Two Instances**



For shared logic connectivity guidelines, see Table 3-1 in the *1G/2.5G Ethernet PCS/PMA or SGMII LogiCORE IP Product Guide* (PG047).

For Clock Management with Multiple Core Instances for SGMII, see the “Clock Sharing Across Multiple Cores with Transceivers and FPGA Logic Elastic Buffer” section in *1G/2.5G Ethernet PCS/PMA or SGMII LogiCORE IP Product Guide* (PG047).

CAUTION! *The RXOUTCLKs cannot be shared between multiple cores unless the transmitting devices are synchronous when using SGMII with the fabric buffer.*

Example 4: Sharing Clock and Reset Sequence for Asynchronous 1000BASE-X/SGMII over LVDS Mode among Multiple Subsystem Instances

When multiple subsystem instances are targeted for LVDS ports using Asynchronous 1000BaseX/SGMII over LVDS in the same bank, the clocking resources and reset sequence can be shared. The signals to/from instance configured to Include Shared Logic in Core needs to be connected to inputs of an instance configured to Include Shared Logic in IP Example Design. For Versal devices, XPLL is always included in the core and RIU reset state machine is included in example design. This is described in detail in the "Asynchronous 1000BASE-X/SGMII over LVDS" section in the *1G/2.5G Ethernet PCS/PMA or SGMII LogiCORE IP Product Guide* (PG047).

Using Designer Assistance for the Subsystem

After the subsystem is added to the block design canvas, design assistance is enabled and Run Block Automation and Run Connection Automation commands are enabled. See For Kintex UltraScale KCU105 Board.

The Run Block Automation command connects the subsystem AXI4-Stream interface to a DMA or a FIFO as selected in the Vivado IDE. The `gtx_clk`, `ref_clk`, and `lvds clocks` are connected to respective clock sources when the clock source is already present, else it would instantiate a clock wizard. It also connects the mii/gmii/rgmii/sgmii/sfp along with the `mdio` and `phy_rst_n` ports based on the interface available for the selected board or part.

***Note*:** You cannot configure the subsystem using the block automation Vivado IDE; therefore, it is recommended first to configure the subsystem and then run block automation.

The `axis_clk` signal should be connected to the same clock source as the AXI4-Stream interface. When using a FIFO, `axis_clk` should be connected to the `AXI_lite` clock of the FIFO; for the DMA this needs to be connected to the same clock source as `m_axi_mm2s_aclk` and `m_axi_s2mm_aclk` of DMA.

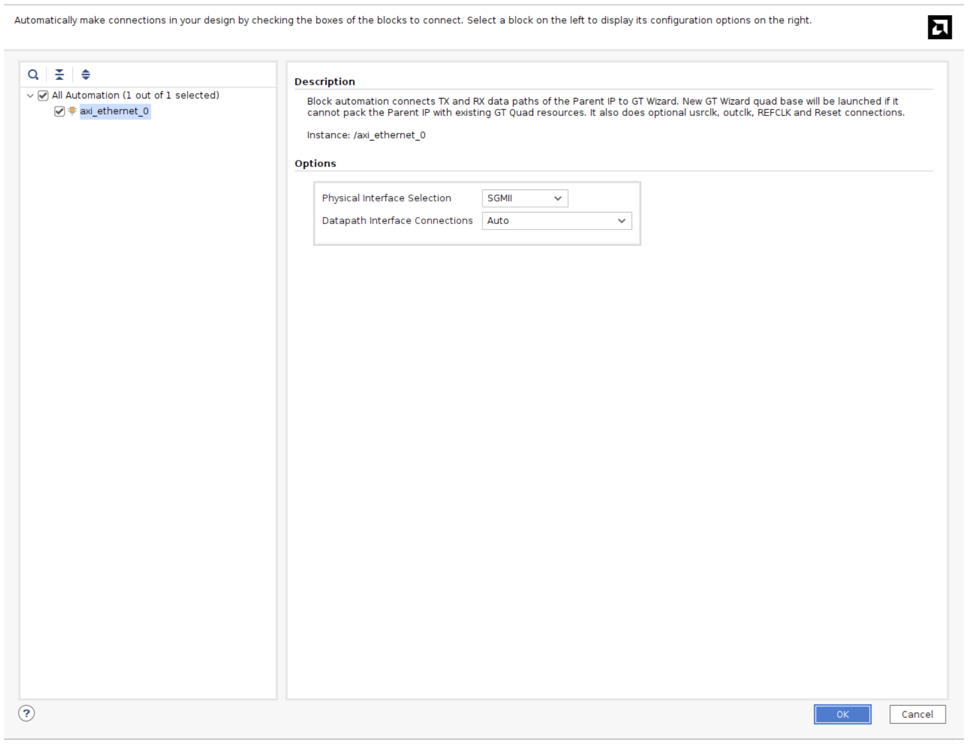
The Run Connection Automation command connects the AXI4-Lite interface of the subsystem to the peripheral interface of the processor. This also connects the AXI4-Lite clock and the AXI4-Lite reset to the respective sources. When the project is set for a board that has the related interfaces, the Run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

For Versal Adaptive SoC Designs

To run block automation for Versal Adaptive SoC designs, perform the following steps.

1. Create a new block design.
2. Place the AXI Ethernet Subsystem IP in the block design.
3. Run block automation. A pop-up window appears, as shown in the following figure.

Figure 68: **Versal Adaptive SoC Block Automation Popup**



Automatically make connections in your design by checking the boxes of the blocks to connect. Select a block on the left to display its configuration options on the right.

All Automation (1 out of 1 selected)
axi_ethernet_0

Description

Block automation connects TX and RX data paths of the Parent IP to GT Wizard. New GT Wizard quad base will be launched if it cannot pack the Parent IP with existing GT Quad resources. It also does optional usrclk, outclk, REFCLK and Reset connections.

Instance: /axi_ethernet_0

Options

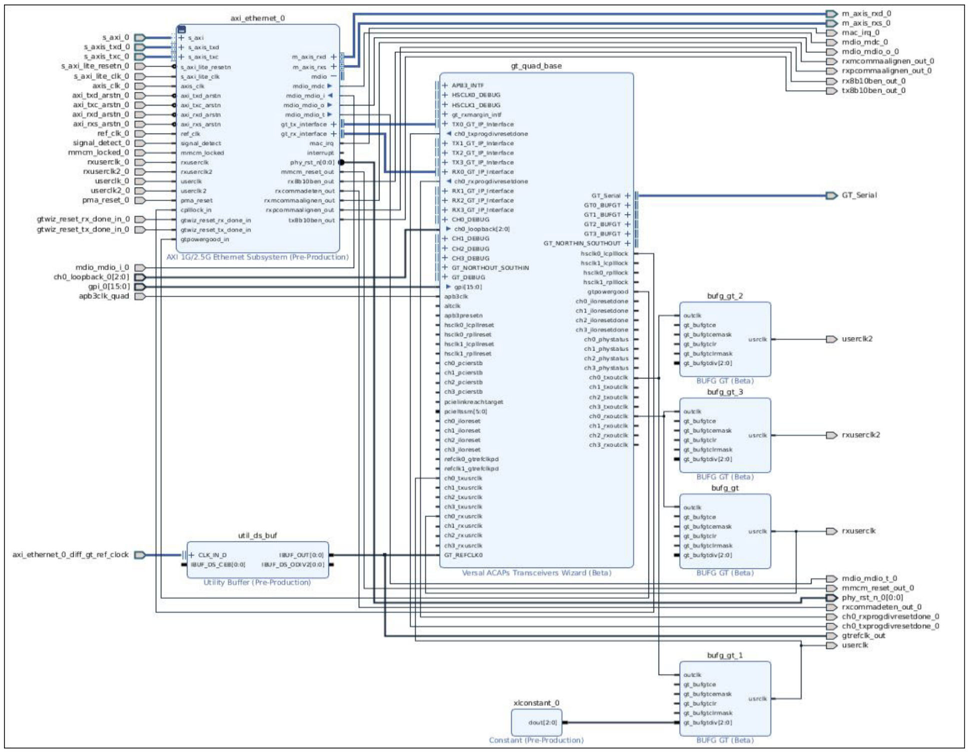
Physical Interface Selection: SGMII
Datapath Interface Connections: Auto

The Physical Interface Selection contains the GT-based configurations: that is, the SGMII, 1000BaseX, and Both options. The Datapath Interface Connections default to “Auto” but the dropdown has the option to select the GT Quad Base connection. In order to connect to a specific lane of an existing GT Quad, first add the AXI Ethernet IP and GT Quad Base IP in the block design, then run the Block Automation with "Customized_Connections" option from the Datapath Interface Connection drop-down as shown in the previous figure. The availability of GT Quads and Lanes is subject to GT resource sharing rules.

***Note*:** The Physical Interface selection must be set to the same value as that of the IP instance on which block automation is being applied.

For example, in the following figure, the selected physical interface is SGMII, and the Datapath interface connections is Auto:

Figure 69: **Versal Adaptive SoC Block Automation Example**



For Kintex UltraScale KCU105 Board

Make sure to verify the following aspects for the Kintex UltraScale KCU105 board:

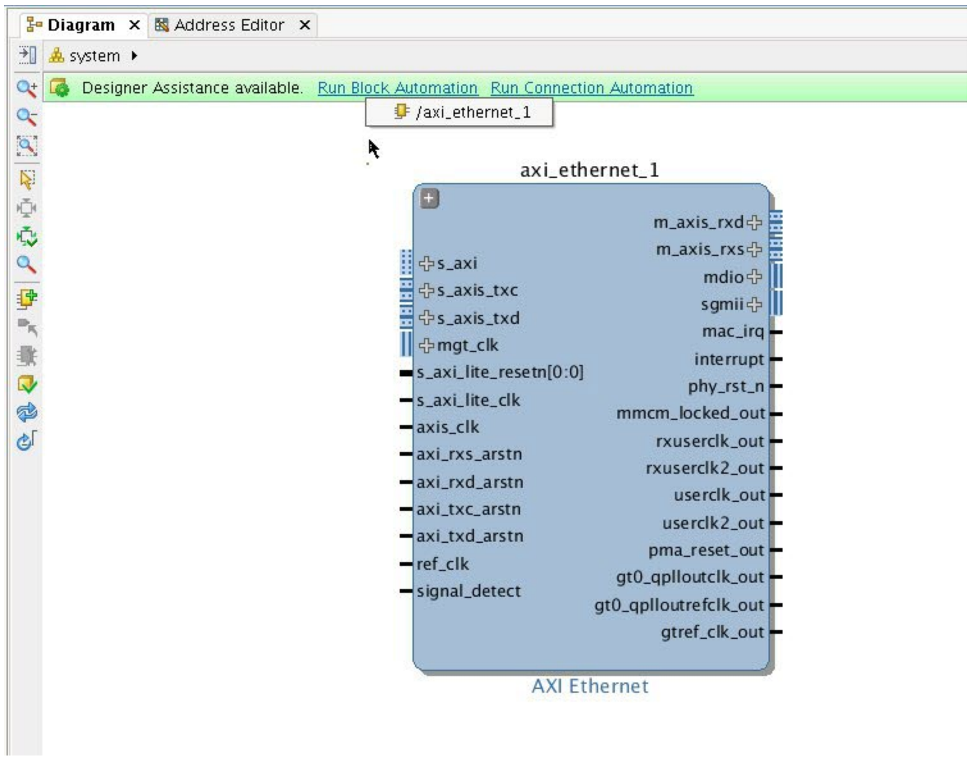
1. Do not use the `phy_rst_n` automation in SGMII over LVDS mode.

   Using this `phy_rst_n` causes a deadlock condition during reset. The on-board PHY of this board does not generate the reference clock during reset that is required by the AXI 1G/2.5G Ethernet subsystem to de-assert the `phy_rst_n`, thus causing a deadlock.

2. When using this board, select **Include Shared Logic in IP Example Design** for the AXI 1G/2.5G Ethernet subsystem.

The on-board PHY generates a 625 MHz clock. Using a clock wizard, the required clocks for this mode can be generated. Block automation also helps in connecting to a clock wizard. This mode also requires an IDELAY control module to be instantiated outside the AXI 1G/2.5G Ethernet subsystem.

Figure 70: **Designer Assistance for the Subsystem**



User Parameters

The following table shows the relationship between the Vivado IDE and the User Parameters.

Table 78: **Vivado IDE Parameter to User Parameter Relationship**

Vivado IDE Parameter/User Parameter	Vivado IDE Parameter Value: User Parameter Value	Default Value
SupportLevel	Include Shared Logic in core: 1 Include Shared Logic in example design: 0	1
speed_1_2p5	1 Gb/s: 1G 2.5 Gb/s: 2p5G	1G
TXCSUM and RXCSUM	No Checksum Offload: None Partial (legacy) Checksum Offload: Partial Full Checksum Offload: Full	None
Timer_Format	Time_of_day: 0 Correction_Field_Format: 1	0
Enable_1588_1step	1_Step: true 2_Step: false	true

Notes:

1. Parameter values are listed in the table where the Vivado IDE parameter value differs from the user parameter value. Such values are shown in this table as indented below the associated parameter

Output Generation

For details, see *Vivado Design Suite User Guide: Designing with IP* (UG896).

Constraining the Subsystem

This section contains information about constraining the subsystem in the AMD Vivado™ Design Suite.

Required Constraints

Because the subsystem is hierarchical, it enables the use of timing constraints from the infrastructure cores. The subsystem 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 core. The timing constraints related to VLAN and others are provided by the AXI Ethernet Buffer. The timing constraints related to the transceiver and elastic buffers are provided by the 1G/2.5G Ethernet PCS/PMA or SGMII core.

In addition to the 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. On selection of the board interface, the constraints for that interface are generated automatically. The subsystem propagates the required inputs to the sub cores and these generate the constraints in their file list. These constraints are usually available in a file that has file name ending with `_board.xdc`.

GMII or RGMII interface

When the core is configured to use GMII or RGMII interface on UltraScale or UltraScale+ devices, you need to set the `REFCLK_FREQUENCY` attribute of all the IDELAY and ODELAY primitives used by the core to the frequency of the clock driving the `ref_clk` port. This can be done through XDC constraints. For more information, refer to the example design in Vivado.

```
set_property REFCLK_FREQUENCY <ref_clk> [get_cells -hier -filter {NAME
=~ *<cell_name>}]
```

Note: The AXI Ethernet Subsystem is a hierarchical IP, and consequently does not have a core XDC. The subcore XDC constraints are automatically applied. It is recommended to check the AXI Ethernet IP example design XDC and apply the constraints (if any) to the top XDC for board and part based subsystem designs.

Device, Package, and Speed Grade Selections

The selections need to be done in accordance with the requirements of the infrastructure cores. See the *1G/2.5G Ethernet PCS/PMA or SGMII LogiCORE IP Product Guide* (PG047) and Tri-Mode Ethernet MAC LogiCORE IP Product Guide (PG051) for more details.

Clock Frequencies

This section is not applicable for this subsystem.

Clock Management

This section is not applicable for this subsystem.

Clock Placement

The clock buffer needs to be placed in accordance with the requirements of the infrastructure cores. See the *1G/2.5G Ethernet PCS/PMA or SGMII LogiCORE IP Product Guide* (PG047) and Tri-Mode Ethernet MAC LogiCORE IP Product Guide (PG051) for more details.

Banking

There is no information provided for this subsystem.

Transceiver Placement

The transceiver placements should be done in accordance with the requirement of Gigabit Ethernet PCS PMA. See the *1G/2.5G Ethernet PCS/PMA or SGMII LogiCORE IP Product Guide* (PG047).

I/O Standard and Placement

The I/O standards and placements needs to be placed in accordance with the requirements of the infrastructure cores. See the *1G/2.5G Ethernet PCS/PMA or SGMII LogiCORE IP Product Guide* (PG047) and Tri-Mode Ethernet MAC LogiCORE IP Product Guide (PG051) for more details.

Simulation

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).

IMPORTANT! *For cores targeting 7 series or AMD Zynq™ 7000 devices, UNIFAST libraries are not supported. AMD IP is tested and qualified with UNISIM libraries only.*

Includes all simulation sources required by the core. Simulation of the core is not supported without the addition of a test bench (not supplied). Simulation of the example design is supported.

When SIMULATION_MODE is set, the link timer for auto-negotiation is pre-loaded with a smaller value. When SIMULATION_MODE is set, the core also pre-loads the 3 ms counter for the 7 series transceiver in startup FSM with a smaller value to decrease the simulation time.

For SGMII over LVDS, setting SIMULATION_MODE pre-loads the `eye_mon_wait_time` counter to a lower value to decrease the simulation time.

***Note*:** The SIMULATION_MODE generic is provided in all modes to reduce simulation time. In simulation, the value of SIMULATION_MODE should be 1. In implementation, the value of SIMULATION_MODE should be 0. To change the SIMULATION_MODE attribute you need to use the following command before the generation of the output products:

```
set_property CONFIG.SIMULATION_MODE {1} [get_ips <component_name> ]
```

IMPORTANT! *SIMULATION_MODE generic should be set to 1 only for simulation.*

***Note*:** In case of Asynchronous SGMII/1000BASE-X over LVDS solution, SIMULATION_MODE should be 1 to simulate the design in post-synthesis and post-implementation simulations. This is because the RIU register used in the calibration sequence in the `clock_reset` module is not supported in simulation and always returns 0 value when read. However, SIMULATION_MODE should be set to 0 when the design is implemented in FPGA. It should be reset to 0.

When SGMII or 1000Base-X is selected as the PHY type, a mixed hardware description language (HDL) license is required to successfully run the simulation.

Synthesis and Implementation

For details about synthesis and implementation, see the *Vivado Design Suite User Guide: Designing with IP* (UG896).

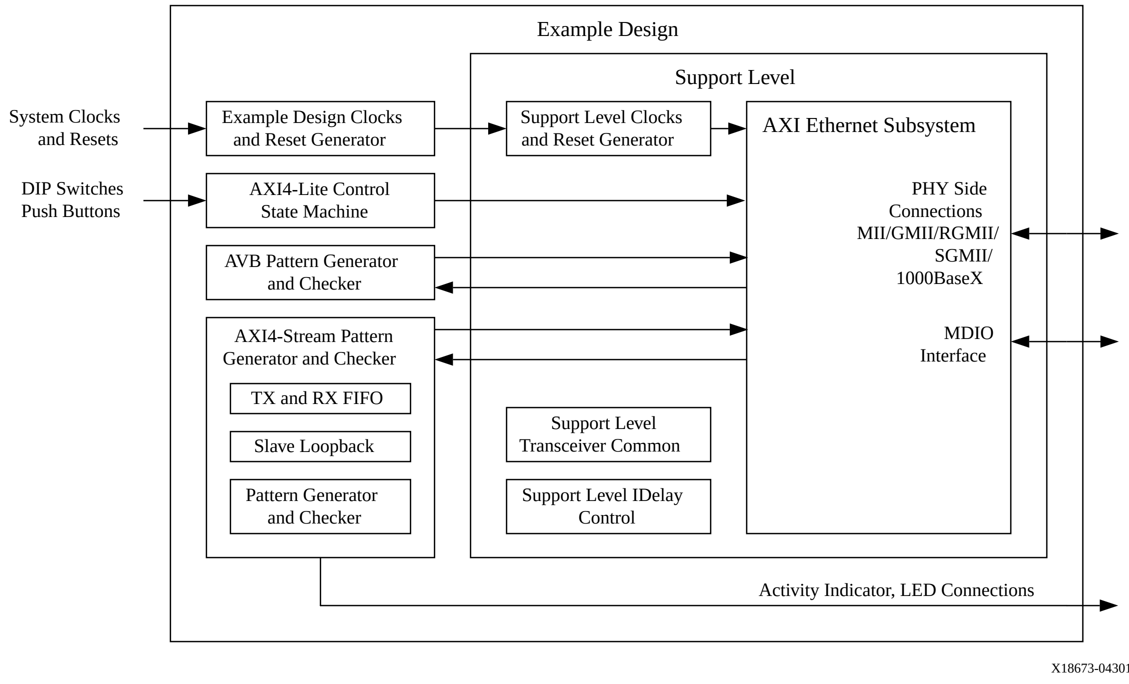
Example Design

This chapter contains information about the example design provided in the AMD Vivado™ Design Suite. This example design is intended to directly target the key AMD demonstration families under certain core configurations. The current example design targets the AMD Kintex™ 7 FPGA KC705 Evaluation Kit board. Information about targeting the example design to the Kintex AMD UltraScale™ KCU105 board is also provided in this chapter.

The example design includes a basic state machine that uses the AXI4-Lite interface to bring up the external PHY and Ethernet MAC allowing basic frame transfers. A simple frame generator and frame checker are also included to provide a packet generator with optional checking of any received data. If the AXI Ethernet Subsystem is generated with the optional AVB endpoint, another frame generator and frame checker are included to exercise the additional AV datapath.

Loopback functionality is provided as either MAC RX to TX (loopback logic replaces the packet generator as the packet source), or PHY TX to RX (loopback logic replaces the demonstration test bench stimulus and checker). Push buttons and DIP switches on the board provide basic control of the state machine allowing Ethernet MAC bit rate change. See the board-specific sections in Targeting the Example Design to a Board. The following figure illustrates the top-level of the AXI Ethernet Subsystem HDL example design.

Figure 71: **HDL Example Design**


Example Design
Support Level
System Clocks and Resets
DIP Switches Push Buttons
Example Design Clocks and Reset Generator
AXI4-Lite Control State Machine
AVB Pattern Generator and Checker
AXI4-Stream Pattern Generator and Checker
TX and RX FIFO
Slave Loopback
Pattern Generator and Checker
Support Level Clocks and Reset Generator
Support Level Transceiver Common
Support Level IDelay Control
AXI Ethernet Subsystem
PHY Side Connections MII/GMII/RGMII/ SGMII/ 1000BaseX
MDIO Interface
Activity Indicator, LED Connections
X18673-043019

Components of Example Design

The example design of the AXI Ethernet Subsystem can be divided in to different components and hierarchies. The Support Level hierarchy contains elements that belong to the shared logic. The example design hierarchy is the top level for this HDL example design. HDL example design contains the following components:

- An instance of the AXI Ethernet Subsystem
- Clock management logic, including MMCM and Global Clock Buffer instances, where required
- MII, GMII, RGMII, SGMII, or 1000BASE-X interface logic, including IOB and DDR registers instances, where required
- User Transmit and Receive FIFOs with AXI4-Stream interfaces
- User basic pattern generator module containing a frame generator and a frame checker with loopback logic
- User AVB pattern generator module providing a second frame generator and checker for designs including the AVB endpoint
- A simple state machine to bring up the PHY (if any) and the Ethernet MAC to ready the design for frame transfer

The HDL example design provides basic loopback functionality on the user side of the AXI Ethernet Subsystem and connects the GMII/RGMII interface to external IOBs. The design also operates as a pattern generator with optional PHY-side external data loopback with automatic checking.

This configuration allows the functionality of the subsystem to be demonstrated either by using a simulation package as discussed in this guide, or directly in hardware when placed on a suitable board. The simple state machine assumes standard AMD demonstration board PHY address and register content.

10/100/1000 Mb/s Ethernet FIFO

The 10/100/1000 Mb/s Ethernet FIFO contains an instance of `tx_client_fifo` to connect to the Ethernet MAC TX AXI4-Stream interface, and an instance of the `rx_client_fifo` to connect to the Ethernet MAC RX AXI4-Stream interface. Both transmit and receive FIFO components implement an AXI4-Stream user interface through which the frame data can be read and written. This module is present only when the AXI Ethernet Subsystem is not configured in processor mode. In processor mode, the pattern generator and checker read and write directly to the streaming interfaces.

rx_client_fifo

The `rx_client_fifo` is built around a dual-port inferred RAM giving a total memory capacity of 4,096 bytes. The receive FIFO writes data received through the TEMAC core. If not errored, the frame is presented on the AXI4-Stream FIFO interface to be read by the `basic_pat_gen` module. If the frame is errored, it is dropped by the receive FIFO. If the receive FIFO memory overflows, the frame currently being received is dropped.

The FIFO can overflow if the receiver clock is running at a faster rate than the transmitter clock or if the inter-packet gap between the received frames is smaller than the interpacket gap between the transmitted frames. In this case, the TX FIFO is unable to read data from the RX FIFO as fast as it is being received.

The FIFO size of 4,096 bytes limits the size of the frames that it can store without error. If a frame is larger than 4,000 bytes, the FIFO can overflow causing lost data. It is therefore recommended that the example design should not be used with the TEMAC solution in jumbo frame mode for frames of larger than 4,000 bytes.

tx_client_fifo

The `tx_client_fifo` is built around a dual-port inferred RAM, giving a total memory capacity of 4,096 bytes.

When a full frame has been written into the transmit FIFO, the FIFO presents data to the Ethernet MAC transmitter. The Ethernet MAC uses `tx_axis_mac_tready` to throttle the data until it has control of the medium.

If the FIFO memory fills up, the `tx_axis_fifo_tready` signal halts the AXI4-Stream interface from writing in data until space becomes available in the FIFO. If the FIFO memory fills up but no full frames are available for transmission, such as when a frame larger than 4,000 bytes is written into the FIFO, the FIFO asserts the `tx_overflow` signal and continues to accept the rest of the frame. The overflow frame is dropped by the FIFO ensuring that the AXI4-Stream FIFO interface does not lock up.

Basic Pattern Generator Module

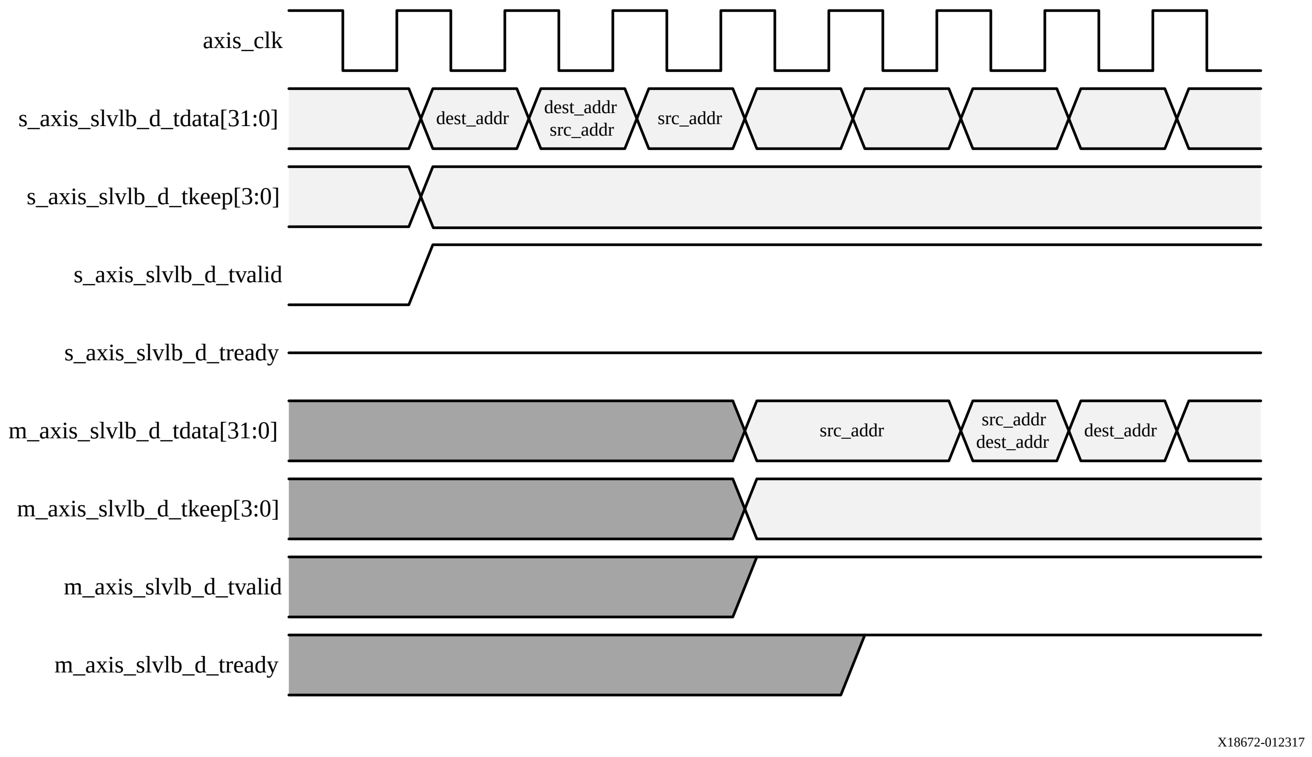
The basic pattern generator has two main functional modes: loopback and generator. In loopback, the data from the RX FIFO is passed to the address swap module which passes the data to the TX FIFO. In Generator mode, the TX data is provided by the pattern generator while RX data is checked by the pattern checker.

Address Swap

The address swap module is configured for use on the loopback path. This permits the example design, when targeted to a suitable board, to be connected to an Ethernet protocol tester. The address swap module waits until both the DA and SA are received before starting to send data on to the TX FIFO.

If enabled, the address swap module swaps the destination and source addresses of each frame as shown in the following figure to ensure that the outgoing frame destination address matches the source address of the link partner. If not enabled, the DA and SA are left untouched. The address swap module transmits the frame control signals with an equal latency to the frame data.

Figure 72: **Modification of Frame Data by Address Swap Module**



Pattern Generator

The pattern generator can be enabled/disabled using a DIP switch. When enabled, the data from the RX FIFO is flushed and the `axi_pat_gen` module drives the `address_swap` module inputs.

Using parameters, the pattern generator allows user modification of these items:

- Destination address
- Source address
- Minimum frame size
- Maximum frame size

When enabled using a dedicated input mapped to a DIP switch on the board, the pattern generator begins with the minimum frame size and, after each frame is sent, increments the frame size until the maximum value is reached. The cycle then repeats beginning with the minimum frame size.

The destination and source addresses are as provided by HDL parameters. The Type/Length field is dependent upon the frame size and the frame data consists of the count value.

Pattern Checker

The pattern checker module provides simple verification that data is being received correctly. The pattern checker module uses the same parameters as the pattern generator module to quickly verify that it is receiving the same frame contents and frame size increments as those being generated. When enabled, the output from the AXI Ethernet streaming interface is monitored. The value in the type/length field is captured to identify the data location within the frame sequence. If an error is detected, an error condition is raised on the mismatched byte or bytes. The error condition is latched to a dedicated output and displayed using a board LED. The error condition can be cleared using a DIP switch.

The pattern checker module also provides a simple activity monitor which toggles a dedicated output to flash a board LED indicating that RX Data is being received correctly. This ensures that the lack of a detected error is not the result of all frames being dropped.

AXI4-Lite Control State Machine

The AXI4-Lite state machine is generated with the AXI4-Lite interface providing basic access to initialize the PHY and the Ethernet MAC for basic frame transfers.

Targeting the Example Design to a Board

For each supported board, there are certain AXI Ethernet Subsystem configurations that target the design directly to the board. The XDC file included with the example design provides the required pin placements for the specific board. In each case, the board DIP switches, push buttons, and LEDs are used to provide basic control over the Ethernet MAC functionality. The operation of the switches and indicators is described in more detail in this section.

Board-Specific Configurations

There are two basic requirements that must be met for the example design to function properly when targeted to a specific board:

- Select the Ethernet IP interface as `gmii`, `sfp_sgmii`, or `sfp` in the Board tab of the Vivado IDE.
- Apply the appropriate board interface settings in the remainder of the Board tab.

Bring-Up Sequence

To ensure that the example design works properly on the targeted board, verify that these conditions are met:

- Ensure that jumpers J27, J28, J29, J30, and J64 are in the correct positions.
- If SGMII or 1000BASE-X is configured, SFP with PHY or loopback SFP must also be configured.

KC705 Board

When the KC705 evaluation board is selected in the Vivado Design Suite project options, an XDC file is generated to target the example design to the KC705 board. The LED functions for the KC705 evaluation board are described in the following table.

Table 79: **KC705 LED Indicators**

LED Location	Signal Name	Indication
GPIO_LED1	mtrlb_activity_flash	Activity. Blinking LED indicates that pattern checker is receiving frames.
GPIO_LED2	mtrlb_pktchk_error	Error. Illuminated LED indicates that pattern checker has received an error packet.

The push button functions for the KC705 board are described in the following table.

Table 80: **KC705 Push Buttons**

Switch	Signal Name	Function
SW6	start_config	Update configuration. When pressed, the push button configuration is applied to the example design.
SW7(CPU_RESET)	sys_rst	System reset.
SW14(FPGA_PROG_B)		FPGA configuration reset.

DIP switch SW11 functions for the KC705 board are described in the following table.

Table 81: **KC705 DIP Switch SW11**

Switch Position	Function
4’h1	Set bit rate to 1 Gb/s.
4’h2	Set bit rate to 100 Mb/s.
4’h3	Set bit rate to 10 Mb/s.
4’h4	Reset packet check error.
4’h5	Set BIST mode, loopback at PHY interface.

Table 81: **KC705 DIP Switch SW11** *(cont'd)*

Switch Position	Function
4'h9	Set demonstration mode, loopback at AXI4-Stream interface.

KCU105 Board

The KCU105 board supports only SGMII over LVDS to connect to the on-board PHY. In this mode, the MGT CLOCK is provided by the on-board PHY at 625 MHz only. Also, the on-board PHY receives the reset from the FPGA. Because the AXI Ethernet Subsystem generates `phy_reset_n` on the GTX clock, there might be a deadlock in the reset state. To avoid this, the example design needs to be modified to disconnect the `phy_reset_n` generated by the AXI Ethernet Subsystem and connect another active-Low reset. This can be done in the example design wrapper. The required number of LEDs, DIP switches, and push button switches are same as that are required for the KC705 board. The required location constraints need to be added.

VCU118/KCU116 Board

The VCU118 and KCU116 boards with on-board TI PHY support are added for SGMII over LVDS configuration.

In the case of the VCU118 board, the SGMII over LVDS mode has the MGT CLOCK which is provided by the on-board TI PHY at 625 MHz. Also, the on-board PHY receives the reset from the FPGA. The other I/O ports are assigned with the LEDs, DIP switches, and push button switches available in the VCU118 board. DIP switch for `control_data` functions are the same as described in the previous table.

Chapter 7

Test Bench

This chapter contains information about the test bench provided in the AMD Vivado™ Design Suite.

Test Bench Functionality

The demonstration test bench is defined in this file:

```
demo_tb.v
```

The demonstration test bench is a simple Verilog program that exercises the example design and the subsystem itself. The test bench has two modes of operation:

- DEMO
- Built-in Self Test (BIST)

BIST mode is the default.

The test bench consists of these component blocks:

- Clock generators
- DEMO (stimulus) – A stimulus block that connects to the PHY-side receiver interface of the example design (MII, GMII, RGMII, SFP or SGMII interfaces)
- DEMO (monitor) – A monitor block to check data returned through the PHY side transmitter interface
- DEMO – Basic frame filter that looks at the DA/SA fields of frame inserted into the PHY side receiver interface
- BIST – A simple loopback from the PHY side transmit interface to the receiver
- BIST (AVB only) – A basic AV data bandwidth monitor
- A management block to control the speed selection
- An MDIO monitor/stimulus block to check and respond to MDIO accesses

DEMO Mode

The demonstration test bench performs these steps:

1. Input clock signals are generated.
2. A reset is applied to the example design.
3. The required bit rate is selected using the control interface.
4. The MDIO stimulus/response block responds to a read with all 1s indicating no PHY is present.
5. Four frames are pushed into the PHY-side receiver interface at the highest Ethernet MAC bit rate supported:
    a. The first frame is a minimum-length frame.
    b. The second frame is a type frame.
    c. The third frame is an errored frame.
    d. The fourth frame is a padded frame.
6. The frames received at the PHY-side transmitter interface are checked against the stimulus frames to ensure that data is the same. The monitor process takes into account the source/destination address field and FCS modifications resulting from the address swap module.
7. The test bench updates the example design for next maximum speed available.
8. The MDIO stimulus/response block responds to a read with all 1s indicating no PHY is present.
9. Four frames similar to those in step 5 are sent to the PHY side interface and checked against the stimulus frames.
10. The preceding steps are repeated until the lowest supported Ethernet MAC bit rate is achieved. Then the packets are checked again at the highest supported Ethernet MAC bit rate.

DEMO Mode with Frame Filter

The test bench performs the additional task of checking the destination address and source address of the frame using these steps:

1. A fifth frame is pushed into the PHY side receiver interface.
2. The address filter looks at the DA/SA fields of each frame.
3. If there is a mismatch between the DA/SA field of a frame and the 12-byte value the address filter has been pre-programmed with (in the AXI4-Lite state machine), then the frame is dropped.
4. By default, only the fifth frame is dropped because of this mismatch.

BIST Mode

In BIST mode, the demonstration test bench performs these steps:

1. Input clock signals are generated.
2. A reset is applied to the example design.
3. The required Ethernet MAC bit rate is selected.
4. The MDIO stimulus/response block responds to a read with all 1s indicating no PHY is present.
5. The pattern generator(s) and checker(s) are enabled.
6. The simulation runs for a fixed duration, allowing a large number of frames to pass.
7. Any detected errors or lack of RX activity are reported as errors.
8. If the AVB endpoint is included, the bandwidth of the two data streams is reported.

Changing the Test Bench

The test bench can be in configured in DEMO mode or BIST mode by changing the parameter `TB_MODE` in the `demo_tb.v` source file. To change the mode, set the parameter value to *DEMO* or *BIST*.

DEMO Mode

Changing Frame Data

The contents of the frame data passed into the TEMAC receiver can be changed by editing the DATA fields for each frame defined in the test bench. The test bench automatically calculates the new FCS field to pass into the TEMAC and also calculates the new expected FCS value. Further frames can be added by defining a new frame of data.

If frame filter is enabled, the first four frames have the same DA and SA data matching the value of the frame filter setting. The fifth frame is dropped because, by default, it has different DA/SA values than the first four frames.

If none of the frames have this DA/SA setup, all frames are dropped by the address filter.

Changing Frame Error Status

Errors can be inserted into any of the predefined frames by changing the error field to 1 in any column of that frame. When an error is introduced into a frame, the `bad_frame` field for that frame must be set to disable the monitor checking for that frame. The error currently written into the third frame can be removed by setting all error fields for the frame to 0 and clearing the `bad_frame` field.

BIST Mode

In BIST mode, the data is provided by the basic pattern generator module. This arrangement allows a degree of control over the frames generated using these module parameters:

- DEST_ADDR
- SRC_ADDR
- MAX_SIZE
- MIN_SIZE
- VLAN_ID
- VLAN_PRIORITY

The pattern generator does not have an error injection capability.

IEEE 1588 Timestamping

This appendix contains information about configuring the core for IEEE 1588 hardware timestamping support.

Generating the Subsystem for 1588 Operation

To configure the subsystem with IEEE 1588 hardware timestamping support, open the AXI Ethernet AMD Vivado™ IDE and perform the following steps:

1. From the Physical Interface tab, select the **1000BASE-X** or **SGMII** 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 Format 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 25: IEEE 1588 System Timer Ports.
6. Click **OK** to generate the subsystem.

Functional Description

IEEE 1588 hardware timestamping support has been added as an option to the AXI Ethernet Subsystem. This feature is built on the existing functionality provided by the following two cores.

- *Tri-Mode Ethernet MAC LogiCORE IP Product Guide* (PG051)
- *1G/2.5G Ethernet PCS/PMA or SGMII LogiCORE IP Product Guide* (PG047)

Refer, in particular, to the Tri-Mode Ethernet MAC LogiCORE IP Product Guide (PG051) 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 or SGMII PHY targeting 7 series (GTX and GTH), AMD UltraScale™, AMD UltraScale+™ (GTH and GTY), and AMD Versal™ device (GTY and GTYP) transceivers.

IEEE 1588 Supported Features

- Hardware timestamping at full Ethernet line rate on both transmit and receive paths. Timestamp accuracy is 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 the ports defined in IEEE 1588 Received Timestamp Ports. 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 ports defined in Table 26: 1588 Correction Field Ports. 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 IEEE 1588 Transmit Timestamp Ports.
    - 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.

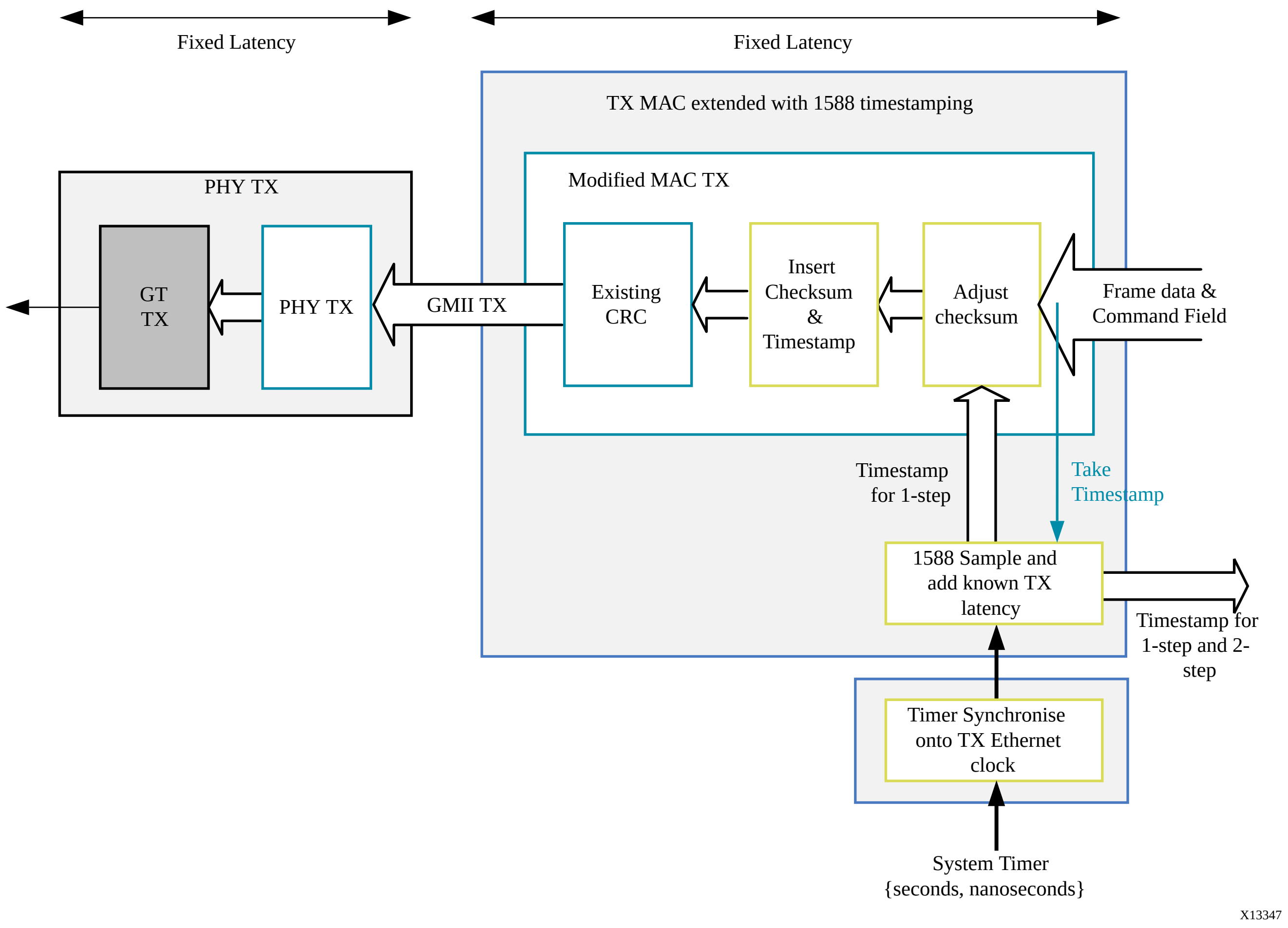
- Supported 1588 PTP frame types are:
    - For transmit 1-step:
        - Raw Ethernet frames
        - UDP IPv4 frames
        - UDP IPv6 frames
    - For transmit 2-step, all Precise Timing Protocol (PTP) frame formats can be supported.
    - For receive, all PTP frame formats can be supported.

Architecture Overview

Transmitter

The following figure shows the Transmitter portions of the Ethernet MAC and 1000BASE-X PHY enhanced with 1588 support.

Figure 73: **Transmitter Block Level Diagram**


Fixed Latency
Fixed Latency
TX MAC extended with 1588 timestamping
PHY TX
Modified MAC TX
GT TX
PHY TX
GMII TX
Existing CRC
Insert Checksum & Timestamp
Adjust checksum
Frame data & Command Field
Timestamp for 1-step
Take Timestamp
1588 Sample and add known TX latency
Timestamp for 1-step and 2-step
Timer Synchronise onto TX Ethernet clock
System Timer {seconds, nanoseconds}
X13347

1588 Frame-by-Frame Timestamp Operation

The frame sent to the Ethernet 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 Ethernet MAC transmitter ports (defined in See IEEE 1588 AXI4-Stream Interface Ports - Transmit 2-Step Timestamp) 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, the CRC value of the frame should be updated/recalculated when the timestamp value is inserted into the frame. 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 Ethernet MAC because both the checksum and the correction field must both be fully contained in the Ethernet 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.

The following table provides a definition of the command field.

Table 82: **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 subsystem is required to recalculate.
[15:9]	Reserved	Reserved for future use. Values are ignored.

Table 82: **Command Field Definition** *(cont'd)*

Bits	Name	Description
[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.
[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

The previous figure illustrates the *1588 Sample and add known TX latency* block. The timestamp is sampled when the Ethernet Start of Frame Delimiter (SFD) is observed at the 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.

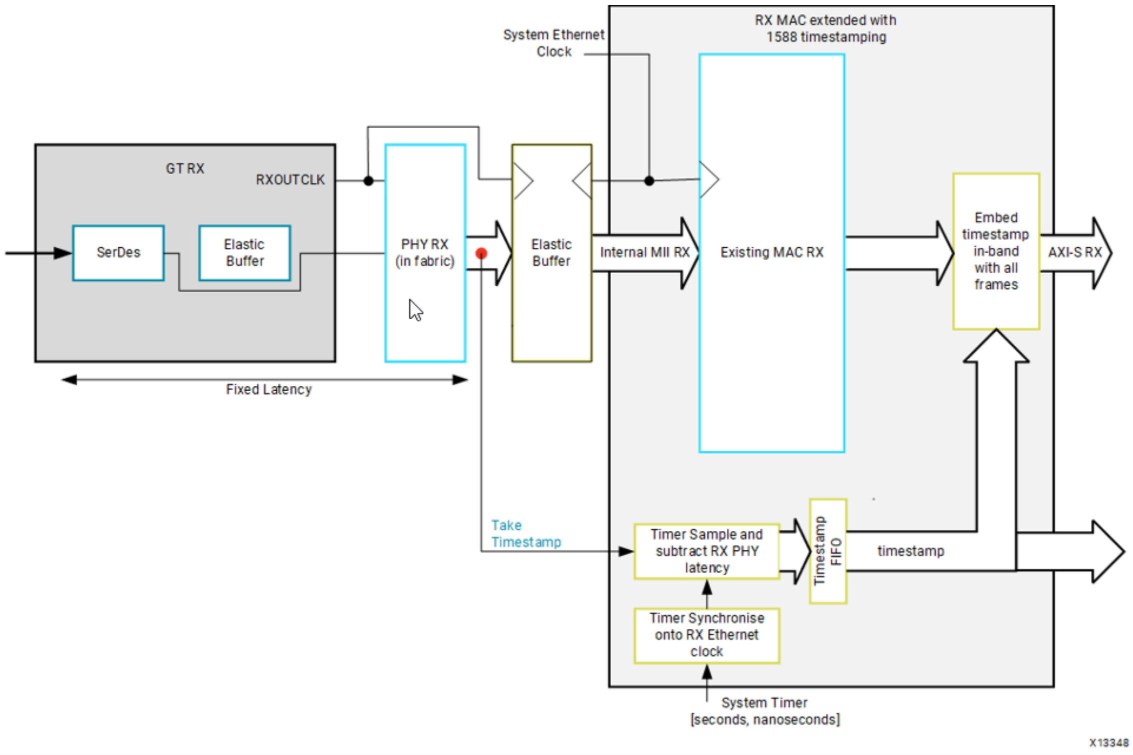
The logic is also then capable of adjusting the ToD timestamp value taken by adding a configurable duration (see bit 22 of Table 46: TEMAC Transmit Configuration Register (0x408)). This value is user adjustable, but its default is initialized with the entire transmitter path latency (through TEMAC, 1000BASE-X, 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

The following figure shows the Receiver portions of the Ethernet MAC and 1000BASE-X PHY enhanced with 1588 support.

Figure 74: **Receiver Block Level Diagram**


System Ethernet Clock
RX MAC extended with 1588 timestamping
GT RX
RXOUTCLK
SerDes
Elastic Buffer
PHY RX (in fabric)
Elastic Buffer
Internal MII RX
Existing MAC RX
Embed timestamp in-band with all frames
AXI-S RX
Fixed Latency
Take Timestamp
Timer Sample and subtract RX PHY latency
Timestamp FIFO
timestamp
Timer Synchronise onto RX Ethernet clock
System Timer [seconds, nanoseconds]
X13348

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 ports defined in IEEE 1588 Received Timestamp Ports. 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 the ports defined in Table 26: 1588 Correction Field Ports. 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

The previous figure 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 1000BASE-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 bit 22 of Table 45: TEMAC Receive Configuration Word1 (RCW1) Register (0x404)). 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 subsystem in "1588 Correction Field Mode" using a 64-bits port. This Correction Field value is in a numerical format as defined in IEEE 1588 clause 13.3.2.7. These details are provided in Table 26: 1588 Correction Field Ports.

Latency Values for the 1588 Enabled Configuration

Latency Values – Simulation

1. Transmitter path
    a. MAC latency: 48 ns
    b. PCS latency: 64 ns
    c. GT Latency: 77 ns

    Total TX latency: 1.a + 1.b + 1.c = 48 + 64 + 77 = 189 ns

    Value in hex for TX latency: 0xBD

2. Receiver path
    a. MAC latency: 8 ns
    b. PCS latency: 56 ns
    c. GT latency
        i. GT latency from RXP to parallel data: 163 ns
        ii. GT latency barrel shifter: 18 ns

        Calculated GT latency: 2.c.i – 2.c.ii = 163 -18 = 145 ns.

        Total RX latency: 2.a + 2.b + 2.c = 8 + 56 +145 = 209 ns

        Value in hex for RX latency: 0xD1

Latency Values – Theoretical

1. Transmitter path
    a. MAC latency: 48 ns
    b. PCS latency: 64 ns
    c. GT Latency: 84 ns

    Total TX latency: 1.a + 1.b + 1.c = 48 + 64 + 84 = 196 ns

    Value in hex for TX latency: 0xC4
2. Receiver path
    a. MAC latency: 8 ns
    b. PCS latency: 56 ns
    c. GT latency: 113 ns

    Total RX latency: 2.a + 2.b + 2.c = 8 + 56 +113 = 177 ns.

    Value in hex for RX latency: 0xB1.

Received Timestamp Ports (Out-of-Band)

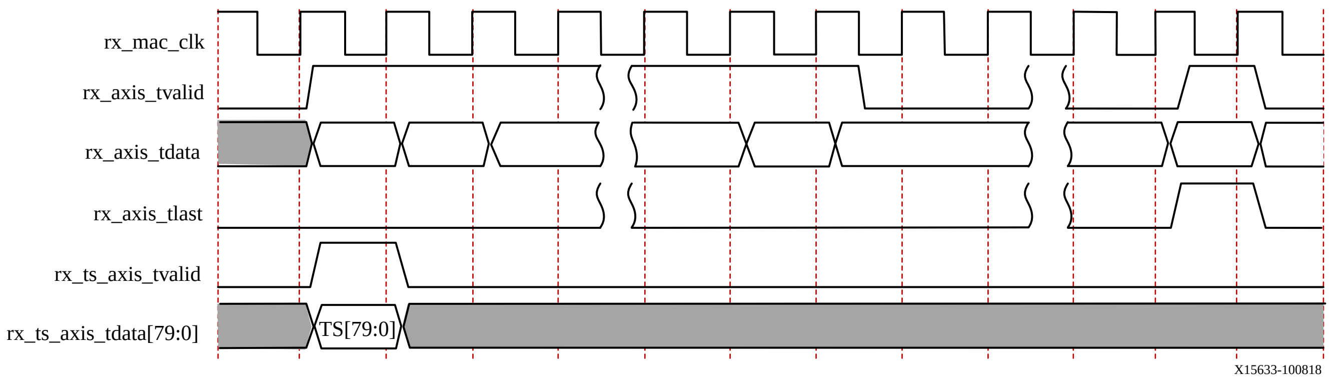
The captured timestamp will always be presented out-of-band with TEMAC frame reception using a dedicated AXI4-Stream interface. The signal definition for this is defined in the following table.

A timing diagram showing the operation of this interface follows the table. To summarize, the timestamp will be 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 83: **AXI4-Steam Interface Ports – Receive Timestamp**

Name	Direction	Description
rx_ts_axis_tdata[127:0]	Out	AXI4-Stream Receive Timestamp from the TEMAC. Bits[127:80] - Reserved Bits [79:32] - Captured Timestamp Seconds field Bits [31:0] - Captured Timestamp Nano-seconds field
rx_ts_axis_tvalid	Out	AXI4-Stream Receive Timestamp Data Valid from the MAC

Figure 75: **AXI4-Stream Interface Timing – Receive Timestamp**



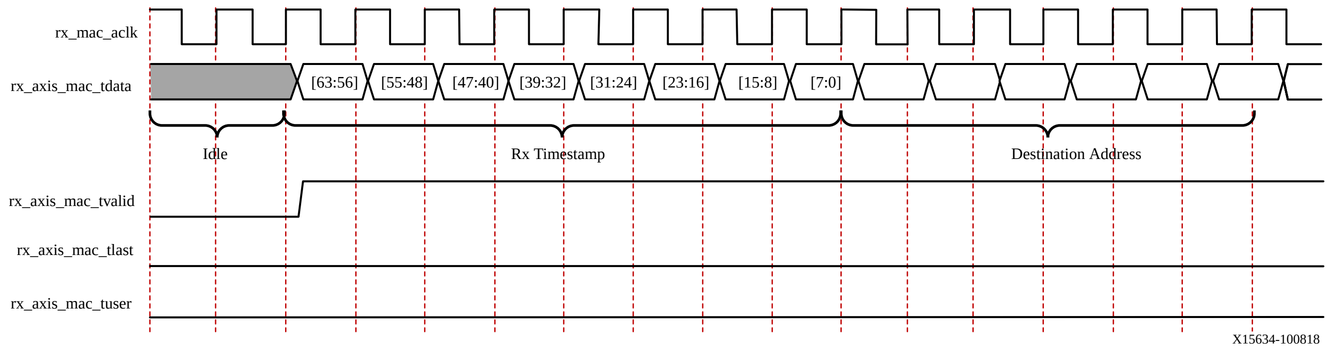
Received Frame Timestamp In-line with Frame Reception

The captured timestamp can optionally be provided in-line with the received frame using the TEMAC existing AXI4-Stream interface – Receive Tri-Mode Ethernet MAC LogiCORE IP Product Guide (PG051). Enable this mode of operation through an AXI4-Lite configuration bit (see Table 45: TEMAC Receive Configuration Word1 (RCW1) Register (0x404), 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 the previous figure.

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

Figure 76: **Timing Diagram of the In-Line Timestamp and Received Frame**



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 signal from that already defined in the current AXI4-Stream Interface – Transmit (see the Tri-Mode Ethernet MAC LogiCORE IP Product Guide (PG051).

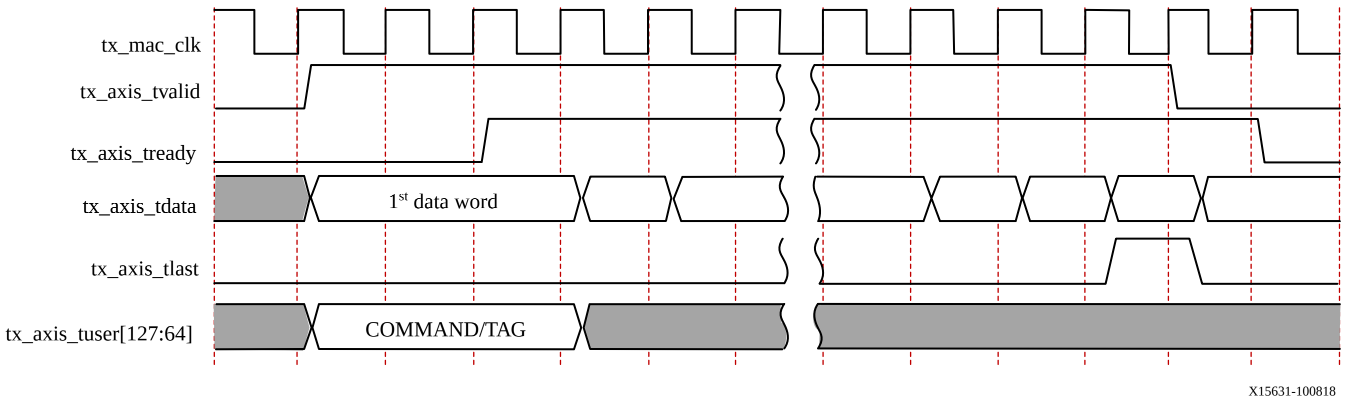
Selecting this mode is through an AXI4-Lite addressable configuration bit (see the following table).

The signal definition for this expanded `tx_axis_tuser` is defined in the following table. A timing diagram showing the operation of this signal for normal frame transmission follows the following table. 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 84: **tx_axis_tuser Bit Field Definition**

Bits	Name	Description
tx_axis_tuser [0]	Underrun	AXI4-Stream user signal used to signal explicit underrun. This is defined in the *1G/2.5G Ethernet PCS/PMA or SGMII LogiCORE IP Product Guide* (PG047) and *Tri-Mode Ethernet MAC LogiCORE IP Product Guide* (PG051).
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 82: Command Field Definition. 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 77: **AXI4-Stream Interface Timing – Out-of-Band Command Field**



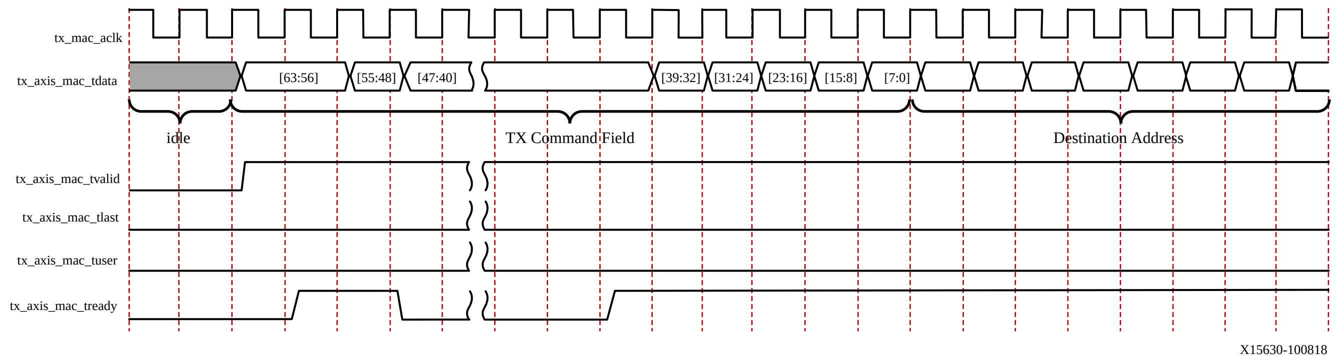
Providing the Command Field In-Line

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 Tri-Mode Ethernet MAC LogiCORE IP Product Guide (PG051).

Enabling this mode is through an AXI4-Lite addressable configuration bit (see Table 46: TEMAC Transmit Configuration Register (0x408), bit 22).

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

Figure 78: **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 the following table.

Table 85: **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	AXI4-Stream interface for RX Data
s_axis_tx	tx_axis_mac	Slave	AXI4-Stream interface for TX Data
m_axis_rx_ts	rx_axis_ts	Master	AXI4-Stream interface for timestamping streaming packet
m_axis_tx_ts	tx_axis_ts	Master	AXI4-Stream interface for timestamping streaming packet

Logic Utilization

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

Table 86: **Logic Utilization for the IEEE 1588 Hardware Timestamping Solutions**

Option	LUT as logic	LUT as Distributed RAM	LUT as Shift Register	Register as Flip-flop	Register as Latch	RAMB36	RAMB 18	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

Chapter 9

Upgrading

This appendix contains information about migrating a design from ISE® to the AMD Vivado™ Design Suite, and for upgrading to a more recent version of the subsystem. For customers upgrading in the Vivado Design Suite, important details (where applicable) about any port changes and other impact on user logic are included utilization.

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).

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 subsystem in the Vivado Design Suite.

The AXI Ethernet Subsystem can be upgraded from an older version to the latest version. When the AXI Ethernet Subsystem is upgraded, there are a few ports renamed and a few parameters added The following sections summarize updated ports and any action to be taken. Parameters are automatically taken care of.

Changes from v7.1 to v7.2

- Added AMD Versal™ devices support.
- Added support for dynamic switchable between 1000BaseX and SGMII.

Changes from v7.0 to v7.1

Added `gt_powergood`. The port is useful when using `gtrefclk` for other applications. Refer to the gtwizard user guide for usage guidelines.

Changes from v5.0 to v6.0

The following ports were renamed.

- `axi_str_txc` to `s_axis_txc` AXI4-Stream Transmit Control. Connect to DMA/FIFO Transmit Control interface.
- `axi_str_txd` to `s_axis_txd` AXI4-Stream Transmit Data. Connect to DMA/FIFO Transmit Data interface.
- `axi_str_rxd` to `m_axis_rxd`

  AXI4-Stream Receive Data. Connect to DMA/FIFO Receive Data Interface.
- `axi_str_rxs` to `m_axis_rxs`

  AXI4-Stream Receive Status. Connect to DMA Receive Status Interface
- `axi_str_avb_tx` to `s_axis_tx_av`

  AXI4-Stream AVB Transmit Data. This interface is present only in AVB mode. Connect to AVB streaming TX master interface.
- `axi_str_avb_rx` to `m_axis_rx_av`

  AXI4-Stream AVB Receive Data. This interface is present only in AVB mode. Connect to AVB streaming RX slave interface.
- `txp`, `txn`, `rxp`, `rxn` ports to `sgmii`

  Serial Gigabit Media Independent Interface. Make this as an external interface.
- `txp`, `txn`, `rxp`, `rxn` ports to `sfp`

  In 1000BASE-X mode, this interface connects to the SFP cage. Make this an external interface.
- `mgt_clk_p` , `mgt_clk_n` ports to `mgt_clk`

  Differential clock input for the serial transceiver. Make this an external interface.

Changes from v4.0 to v5.0

- `refclk` to `ref_clk`

  Reference clock to the delayctrl for RGMII mode and the independent clock source for 1000BASE-X and SGMII. Reconnect this to reference clock source.

Chapter 10

Debugging

This appendix includes details about resources available on the AMD Support website and debugging tools.

Finding Help with AMD Adaptive Computing Solutions

To help in the design and debug process when using the subsystem, the Support web page contains key resources such as product documentation, release notes, answer records, information about known issues, and links for obtaining further product support. The Community Forums are also available where members can learn, participate, share, and ask questions about AMD Adaptive Computing solutions.

Documentation

This product guide is the main document associated with the subsystem. This guide, along with documentation related to all products that aid in the design process, can be found on the Support web page or by using the AMD Adaptive Computing Documentation Navigator. Download the Documentation Navigator from the Downloads page. For more information about this tool and the features available, open the online help after installation.

Answer Records

Answer Records include information about commonly encountered problems, helpful information on how to resolve these problems, and any known issues with an AMD Adaptive Computing product. Answer Records are created and maintained daily to ensure that users have access to the most accurate information available.

Answer Records for this subsystem can be located by using the Search Support box on the main Support web page. To maximize your search results, use 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 Subsystem

AR: 54668

Technical Support

AMD Adaptive Computing provides technical support on the Community Forums for this AMD LogiCORE™ IP product when used as described in the product documentation. AMD Adaptive Computing cannot guarantee timing, functionality, or support if you do any of the following:

- Implement the solution in devices that are not defined in the documentation.
- Customize the solution beyond that allowed in the product documentation.
- Change any section of the design labeled DO NOT MODIFY.

To ask questions, navigate to the Community Forums.

Debug Tools

There are many tools available to address AXI Ethernet Subsystem design issues. It is important to know which tools are useful for debugging various situations.

Vivado Design Suite Debug Feature

The AMD Vivado™ Design Suite debug feature inserts logic analyzer and virtual I/O cores directly into your design. The debug feature also allows 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 AMD devices.

The Vivado logic analyzer is used to interact with the logic debug LogiCORE 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).

Reference Boards

Various AMD development boards support the AXI Ethernet Subsystem core. These boards 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 Design Suite debug feature is a valuable resource to use in hardware debug. The signal names mentioned in the following individual sections can be probed using the Vivado Design Suite debug feature for debugging the specific problems.

Ensure that all the timing constraints for the subsystem 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 subsystem clocks are toggling and that the enables are also asserted.
- If the simulation has been run, verify in simulation and/or the Vivado Design Suite debug feature 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 subsystem cannot send data.
- If the receive `<interface_name>_tvalid` is stuck Low, the subsystem is not receiving data.
- Check that the aclk inputs are connected and toggling.
- Check that the AXI4-Stream waveforms are being followed. See Normal Transmit AXI4-Stream Transfer – Flag=0xA, Receive Status Transmit AXI4-Stream Transfer – Flag=0x5, and Mapping AXI DMA IP Buffer Descriptor Fields to AXI4-Stream Fields.
- Check subsystem configuration.
- Add appropriate core specific checks.

Additional Resources and Legal Notices

Finding Additional Documentation

Technical Information Portal

The AMD Technical Information Portal is an online tool that provides robust search and navigation for documentation using your web browser. To access the Technical Information Portal, go to https://docs.amd.com.

Documentation Navigator

Documentation Navigator (DocNav) is an installed tool that provides access to AMD Adaptive Computing documents, videos, and support resources, which you can filter and search to find information. To open DocNav:

- From the AMD Vivado™ IDE, select **Help → Documentation and Tutorials**.
- On Windows, click the **Start** button and select **Xilinx Design Tools → DocNav**.
- At the Linux command prompt, enter `docnav`.

***Note*:** For more information on DocNav, refer to the *Documentation Navigator User Guide* (UG968).

Design Hubs

AMD Design Hubs provide links to documentation organized by design tasks and other topics, which you can use to learn key concepts and address frequently asked questions. To access the Design Hubs:

- In DocNav, click the **Design Hubs View** tab.
- Go to the Design Hubs web page.

Support Resources

For support resources such as Answers, Documentation, Downloads, and Forums, see Support.

References

These documents provide supplemental material useful with this guide:

1. Tri-Mode Ethernet MAC LogiCORE IP Product Guide (PG051)
2. *1G/2.5G Ethernet PCS/PMA or SGMII LogiCORE IP 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 Data Sheet: Overview* (DS180)
9. *ISE to Vivado Design Suite Migration Guide* (UG911)
10. *Vivado Design Suite User Guide: Programming and Debugging* (UG908)
11. *AXI Interconnect LogiCORE IP Product Guide* (PG059)
12. *Vivado Design Suite User Guide: Designing IP Subsystems using IP Integrator* (UG994)
13. *UltraScale Architecture SelectIO Resources User Guide* (UG571)
14. *Versal Adaptive SoC SelectIO Resources Architecture Manual* (AM010)

Revision History

The following table shows the revision history for this document.

Section	Revision Summary
06/05/2023 Version 7.2	
Clocking	Added a table.
11/15/2023 Version 7.2	
Editorial updates	Rebranding updates.

Section	Revision Summary
05/17/2023 Version 7.2	
GMII or RGMII interface	Added the topic.
Ethernet System Interface	Updated the table.
11/16/2022 Version 7.2	
N/A	Added Versal devices support.
05/11/2022 Version 7.2	
IO Delay Calibration Ports for Ethernet RGMII Interface	Added the topic.
20/03/2021 Version 7.2	
N/A	Added support for Versal Adaptive SoC.
01/14/2021 Version 7.2	
Dynamic Switching Signal Port	Added the topic.
Chapter 5: Design Flow Steps	Added support for Versal Adaptive SoC.
06/24/2020 Version 7.2	
Latency Values for the 1588 Enabled Configuration	Added the topic.
Chapter 5: Design Flow Steps	Updated GUI details.
Functional Description	Updated the topic.
05/22/2019 Version 7.1	
N/A	• Updated Figure 2-10. • Added note to Table 2-23 and Table 2-24. • Updated Table 2-29. • Updated Figure 2-25.
12/05/2018 Version 7.1	
N/A	• Updated the Standalone driver link in Features. • Updated Artix 7 speed grade in Features. • Updated speed grade in Generating Subsystem for 2.5G Operation. • Added Flow Control Interface and Priority Flow Control Interface (802.1Qbb). • Added note in Allowable Parameter Combinations. • Added shared logic description in Example 3: Sharing GT COMMON in 1000BASE-X Mode among Multiple Subsystem Instances (Applicable for Non-Versal Devices). • Added VCU118/KCU116 Board section. • Updated description in Generating the Subsystem for 1588 Operation.

Section	Revision Summary
04/04/2018 Version 7.1	
N/A	• Updated the SGMII Auto-Negotiation section in Chapter 2, Product Specification. • Added the .h Header File section in Chapter 2, Product Specification. • Added a new paragraph to the Clocking section of Chapter 3, Designing with the Subsystem. • Removed a paragraph about the Include Shared Logic in IP Example Design configuration before Figure 4-5. • Added a paragraph about ports to the Using Designer Assistance for the Subsystem section in Chapter 4, Design Flow Steps.
10/04/2017 Version 7.1	
N/A	• Added clarification of register availability depending on AXI Ethernet Buffer enabling. See the Note for Table 2-27. • Added information about signals generated from *_clocks_resets module in the Include Shared Logic in IP Example Design configuration in Chapter 4, Design Flow Steps. • Added a caution after Figure 4-10 about RXOUTCLKs.
06/07/2017 Version 7.1	
N/A	• Updated screen displays in Chapter 4. • Updated description of phy_rst_n.in Table 2-14. • Added gt_powergood. See Changes from v7.0 to v7.1 in Appendix B. • Replaced Table B-1 with sections describing port changes for versions.

Section	Revision Summary
04/05/2017 Version 7.0	
N/A	• Added Verilog and VHDL source HDL Model to Simulation Model row in IP Facts table. • Added text for timestamping support of 1G and 2.5G mode of operation and support for over Select Input/ Output (I/O) Low Voltage Differential Signaling (LVDS) in the Feature Summary in Chapter 1, Overview. • Added text about AMD UltraScale™ and AMD UltraScale+™ to the SGMII over LVDS section in Chapter 2. • Added text about ports and SGMII in asynchronous and synchronous mode in the 1000BASE-X over LVDS section in Chapter 2. • Changed "promiscuous address mode" to "promiscuous mode" throughout. • Updated some text in Figure 2-20. Updated AXI4-Lite registers from 0x500-0x78c in Table 2-27. • Removed all register descriptions from the Identification register through Address Filter Mask register. Replaced with references to PG051. Also many of the register names were changed. • Updated Reset Values in Table 2-43 and 2-44. • Added Example 3 section to Chapter 4. • Added several paragraphs to the Simulation section in Chapter 4.
10/05/2016 Version 7.0	
N/A	• Added information about the Location tabs in Chapter 4, Design Flow Steps. • Added SGMII and 1000BASEX over LVDS support for UltraScale and UltraScale+ devices. • Added support for 1588 in SGMII mode. • Added Spartan-7 device support.
06/08/2016 Version 7.0	
N/A	Added support for GT in example design for UltraScale and UltraScale+ devices.
04/06/2016 Version 7.0	
N/A	Added clock frequency rates for UltraScale devices in User Parameters.
11/18/2015 Version 7.0	
N/A	Added support for UltraScale+ families.
09/30/2015 Version 7.0	
N/A	Added SGMII over LVDS support on UltraScale devices.
06/24/2015 Version 7.0	
N/A	Updated the Received Timestamp Ports (Out of Band) and Received Frame Timestamp In-line with Frame Reception sections in Appendix-A: IEEE 1588 Timestamping.
04/01/2015 Version 7.0	
N/A	Added support for 2.5G Ethernet on 7 series devices with GTH and GTX transceivers and UltraScale transceivers.

Section	Revision Summary
10/01/2014 Version 6.2	
N/A	• Numerous changes to support subsystem nomenclature • Added Example Design and Test Bench chapters
04/02/2014 Version 6.1	
N/A	• Added Correction Field update in 1588 functionality. • Updated Figure 4-5.
10/02/2013 Version 6.0	
N/A	• 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 Version 5.0	
N/A	• 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 Version 1.0	
N/A	Initial release.

Please Read: Important Legal Notices