

40G/50G High Speed Ethernet Subsystem v1.1

Product Guide

Vivado Design Suite

PG211 June 8, 2016

Table of Contents

IP Facts

Chapter 1: Overview

| | |
|--|---|
| Feature Summary | 5 |
| Applications | 6 |
| Licensing and Ordering Information | 6 |

Chapter 2: Product Specification

| | |
|--|----|
| Typical Operation | 8 |
| Statistics Gathering | 9 |
| Testability Functions | 9 |
| Pause Operation | 9 |
| Standards | 10 |
| Performance and Resource Utilization | 10 |
| Port Descriptions | 10 |
| Auto-Negotiation (AN) and Link Training (LT) | 40 |
| PCS Variant | 56 |
| AXI4-Lite Register Space | 66 |

Chapter 3: Designing with the Core

| | |
|---|-----|
| General Design Guidelines | 116 |
| Clocking | 117 |
| LogiCORE Example Design Clocking and Resets | 124 |
| IEEE PTP 1588v2 for 40G/50G Subsystem | 129 |
| Status/Control Interface | 143 |

Chapter 4: Design Flow Steps

| | |
|--|-----|
| Customizing and Designing the Core | 147 |
| Constraining the Core | 159 |
| Simulation | 160 |
| Synthesis and Implementation | 160 |

Chapter 5: Example Design

| | |
|--|-----|
| Example Design Hierarchy | 161 |
| User Interface | 164 |
| Core xci Top Level Port List | 165 |
| Duplex Mode of Operation | 219 |
| Shared Logic Implementation | 219 |
| AXI4-Lite Interface Implementation | 222 |

Chapter 6: Test Bench

Appendix A: Migrating and Upgrading

| | |
|--|-----|
| Upgrading in the Vivado Design Suite | 224 |
|--|-----|

Appendix B: Debugging

| | |
|--------------------------------------|-----|
| Finding Help on Xilinx.com | 227 |
| Debug Tools | 228 |
| Simulation Debug | 229 |
| Hardware Debug | 231 |
| Interface Debug | 233 |
| Protocol Debug | 235 |

Appendix C: Pause Processing Interface

| | |
|--------------------------------|-----|
| TX Pause Generation | 236 |
| RX Pause Termination | 238 |

Appendix D: Additional Resources and Legal Notices

| | |
|--|-----|
| Xilinx Resources | 241 |
| References | 241 |
| Revision History | 242 |
| Please Read: Important Legal Notices | 242 |

Introduction

The Xilinx® LogiCORE™ IP High Speed Ethernet IP Subsystem implements the 40G or 50G Ethernet Media Access Controller (MAC) with a Physical Coding Sublayer (PCS) or standalone PCS.

Features

- Designed to the Ethernet requirements for 50 Gb/s operation as defined in Schedule 3 of the 25G Ethernet Consortium See the Note at the bottom of the page.
- Designed to the requirements for 40 Gb/s operation as defined in IEEE 802.3 Clause 82 [Ref 1].
- Includes complete Ethernet MAC and PCS functions or standalone PCS.
- Simple packet-oriented user interface.
- Low latency mode
- Comprehensive statistics gathering.
- Status signals for all major functional indicators.
- Delivered with a top-level wrapper including functional transceiver wrapper, IP netlist, sample test scripts, and Vivado® design tools compile scripts.
- Optional fee based AN/LT/KR Forward Error Correction (FEC) features

Note: Xilinx recommends that you join the 25 Gigabit Ethernet Consortium to gain access to the 25G/50G specification. For more information on membership, visit the [25G 50G Ethernet Consortium](#).

| LogiCORE IP Facts Table | |
|---|--|
| Core Specifics | |
| Supported Device Family | Virtex® UltraScale+™ Kintex® UltraScale+ Zynq® UltraScale+ Virtex UltraScale Kintex UltraScale |
| Supported User Interfaces | AXI4-Stream |
| Resources | Performance and Resource Utilization web page |
| Provided with Core | |
| Design Files | Encrypted register transfer level (RTL) |
| Example Design | Verilog |
| Test Bench | Verilog |
| Constraints File | Xilinx Design Constraints (XDC) |
| Simulation Model | Verilog |
| Supported S/W Driver | Not Applicable |
| Tested Design Flows⁽¹⁾ | |
| Design Entry ⁽²⁾ | Vivado Design Suite |
| Simulation | For supported simulators, see the Xilinx Design Tools: Release Notes Guide |
| Synthesis | Synopsys or Vivado synthesis |
| Support | |
| Provided by Xilinx at the Xilinx Support web page | |

Notes:

1. For the supported versions of the tools, see the [Xilinx Design Tools: Release Notes Guide](#).
2. Contact Xilinx Technical Support for your design requirements.

Overview

The Xilinx® 40G/50G High Speed Ethernet Subsystem implements a 40G/50G Ethernet Media Access Controller (MAC) module with 40G/50G PCS or standalone 40G/50G PCS.

The 40G/50G High Speed Ethernet Subsystem is designed to schedule 3 of the 25G and 50G Ethernet Consortium specification r1.6 for the 50 Gb/s operation and IEEE 802.3 for 40 Gb/s operation; it is hardware proven, and offers system designers with a risk-free and quick path for systems that implement 40G/50G Ethernet protocols.

This guide also describes the 40G/50G High Speed Ethernet Subsystem in detail and provides the information required to integrate the 40G/50G High Speed Ethernet Subsystem into user designs. The document assumes you are familiar with the IEEE Std 802.3-2012 protocol and FPGA design and methodology. See [Chapter 2, Product Specification](#) for detailed information. For Xilinx device platform-specific information, see [Xilinx Support](#). See also the IEEE Std 802.3-2012 [\[Ref 1\]](#).

Feature Summary

- Supports custom preambles
- Programmable Inter Packet Gap (IPG)
- Low latency design
- Optional Clause 74 Forward Error Correction (FEC)
- Optional 1588v2 PTP 2-step timestamping
- Optional Auto-Negotiation and Link Training

Applications

The Xilinx 40G/50G High Speed Ethernet Subsystem is designed to function as the network interface for applications that require a very high bit rate, such as:

- Ethernet switches
- IP routers
- Data center switches
- Communications equipment

The capability to interconnect devices at 50 Gb/s Ethernet rates becomes especially relevant for next-generation data center networks where:

- To keep up with increasing CPU and storage bandwidth, rack or blade servers must support aggregate throughputs faster than 10 Gb/s (single lane) or 20 Gb/s (dual lane) from their Network Interface Card (NIC) or LAN-on-Motherboard (LOM) networking ports.
- Given the increased bandwidth to endpoints, uplinks from Top-of-Rack (TOR) or Blade switches need to transition from 40 Gb/s (four lanes) to 100 Gb/s (four lanes) while ideally maintaining the same per-lane breakout capability.
- Due to the expected adoption of 100GBASE-CR4/KR4/SR4/LR4, SerDes and cabling technologies are already being developed and deployed to support 25 Gb/s per physical lane, twin-ax cable, or fiber.

Licensing and Ordering Information

The 40G/50G Ethernet IP core is provided under the terms of the [Xilinx Core License Agreement](#). The module is shipped as part of the Vivado Design Suite. For full access to all core functionalities in simulation and in hardware, you must purchase one or more licenses for the core. Contact your [local Xilinx sales representative](#) for information about pricing and availability.

Note: The 40G/50G Ethernet MAC + BASE-R, XLAUI/LAUI, and 40GBASE-KR4/50GBASE-KR2 IP options require separate part numbers.

Contact your [local Xilinx sales representative](#) for more information on the 40G/50G Ethernet core pricing and availability. For more information, see:

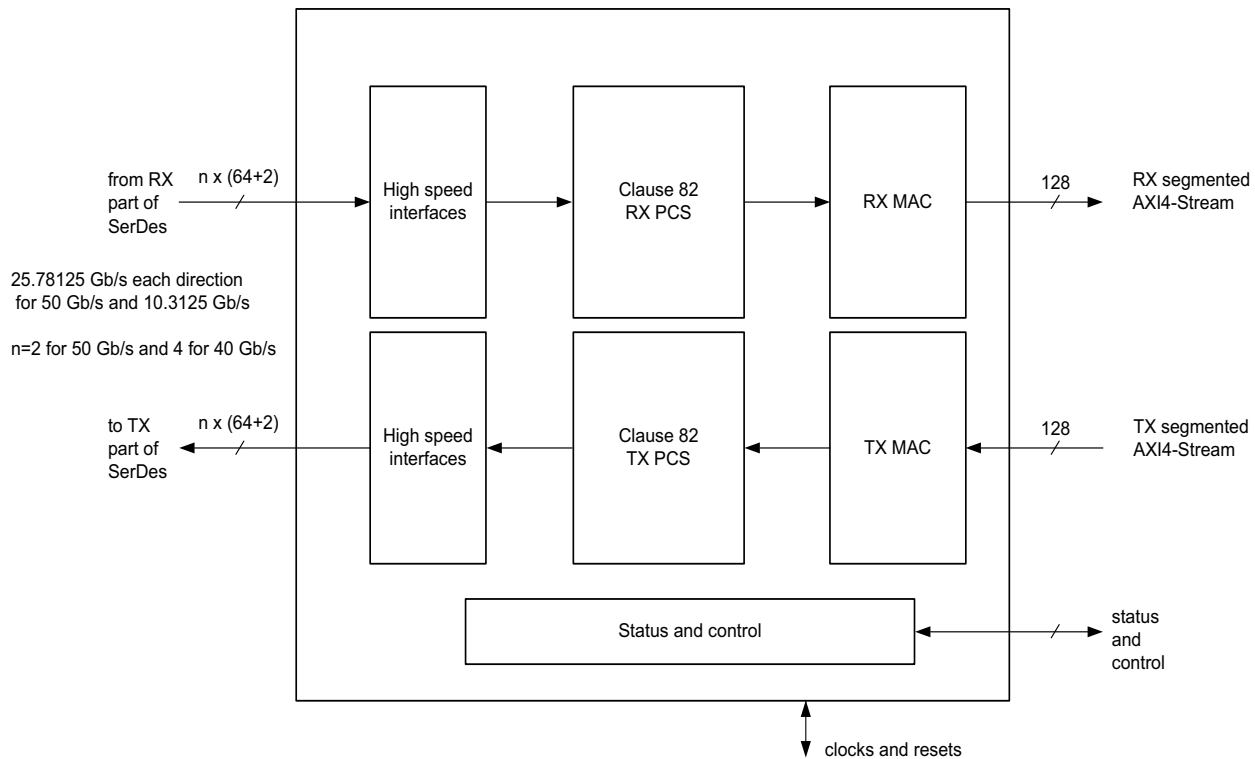
For more information, visit the [10G/25G Ethernet Subsystem page](#).

Information about additional Xilinx LogiCORE™ modules are available on the [Xilinx Intellectual Property](#) page.

Product Specification

The 40G/50G High Speed Ethernet IP subsystem provides ease of use connecting to the High Speed Ethernet core (40G/50G High Speed Ethernet Subsystem). Based on the configuration, this subsystem creates interface ports, instantiates the 40G/50G High Speed Ethernet Subsystem and high speed serialize and deserializer (SerDes) blocks, provides the appropriate clock signals, and connects to the AXI4-Stream user-side interface.

The block diagram for the 40G/50G High Speed Ethernet Subsystem is shown in [Figure 2-1](#). The right-hand side is the user interface and the left-hand side is the external device interface.



X16599-052516

Figure 2-1: 40G/50G High Speed Ethernet Subsystem Block Diagram

The PCS architecture is based on distributing (or striping) parts of a packet over several (relatively) lower speed physical interfaces by the transmitting device. The receiving device PCS layer is then responsible for stripping the different parts and rebuilding the packet before handing it off to the Ethernet MAC block. The receiver PCS layer must also deskew the data from the different physical interfaces as these might encounter different delays as they are transported throughout the network. Additionally, the core handles PCS Lane swapping across all received PCS Lanes, allowing the 40G/50G High Speed Ethernet Subsystem to be used with all optical transport systems.

The PCS and Ethernet MAC layers of the core operate at the maximum line-rate of the interface, and have been optimized to operate in Xilinx FPGAs. The PCS layer includes scrambling/descrambling and 64B/66B encoders/decoders operating at full 40G/50G line rate. The Ethernet MAC block includes a high-speed and optimized Frame Check Sequence (FCS) generation and checking module. In addition to checking the FCS integrity of the packet, the FCS module is capable of optionally inserting and deleting the FCS bytes of the packet at full 40G/50G line rate.

The Control and Status block provides several statistics counters for monitoring data traffic. Additionally, the status interface of the 40G/50G High Speed Ethernet Subsystem provides detailed information about the health of the overall interface, each individual physical interface, and every single PCS lane. The status information includes sync header alignment, PCS alignment and PCS deskew status.

Typical Operation

The core handles all protocol-related functions to communicate with the PCS and Ethernet MAC interface of another device. This includes handshaking, synchronizing and error checking. You provide packet data through the AXI4-Stream TX interface, and receive packet data from the AXI4-Stream RX interface. A detailed description is given in [AXI4-Stream Interface Ports in Chapter 2](#).

The 40G/50G High Speed Ethernet Subsystem is designed to be as flexible as possible and can be used in many different applications. The RX path does not perform any buffering other than the pipelining required to perform the required operations. Received data is passed directly to the user interface in a cut-through manner, allowing you the flexibility to implement any required buffering scheme. Also, the core TX path consists of a single pipeline with minimal buffering to provide reliable cut-through operation.

Statistics Gathering

The 40G/50G High Speed Ethernet Subsystem provides a flexible and user-friendly mechanism for gathering statistics. For every one of the supported statistics, the core has an output signal (or bus) that indicates an increment value for the statistic in a given clock cycle. This mechanism allows you to select which statistics are required in the system without having the cost overhead of a full set of counters. Additionally, and more importantly, you can implement any counter and statistics gathering mechanism required by the system. For example, you can build 32-bit or 64-bit counters as needed, or implement clear-on-read or saturated counters, as required. An optional AXI4-Lite register implementation is available which includes statistics counters. A detailed description of the optional AXI4-Lite implementation is given in [AXI4-Lite Interface Implementation in Chapter 5](#).

For the purposes of TX statistics, good packets are defined as packets without FCS or other errors; bad packets are defined as packets with FCS or any other error.

For the purposes of RX statistics, good packets are defined as packets without FCS or other errors including length error; bad packets are defined as packets with FCS or any other error. The length field error includes length field error, oversize and undersized packets.

Testability Functions

The core implements the test pattern generation and checking as defined in Clause 82.2.10 (test-pattern generators) and 82.2.17 (test-pattern checkers). For details, see the *IEEE Standard for Ethernet* (IEEE Std 802.3-2012) [\[Ref 1\]](#).

Pause Operation

The 40G/50G High Speed Ethernet Subsystem is capable of handling 802.3x and priority-based pause operation. The RX path parses pause packets and presents the extracted quanta on the status interface; the TX path can accept pause packet requests from the control interface and injects the requested packets into the data stream. Both global pause packets and priority-based pause packets are handled. Details are described in [Appendix C, Pause Processing Interface](#).

Standards

The core is designed to the IEEE Std 802.3-2012; it is hardware proven, and offers system designers with a risk-free and quick path for systems that implement 40G/50G Ethernet protocols.

Performance and Resource Utilization

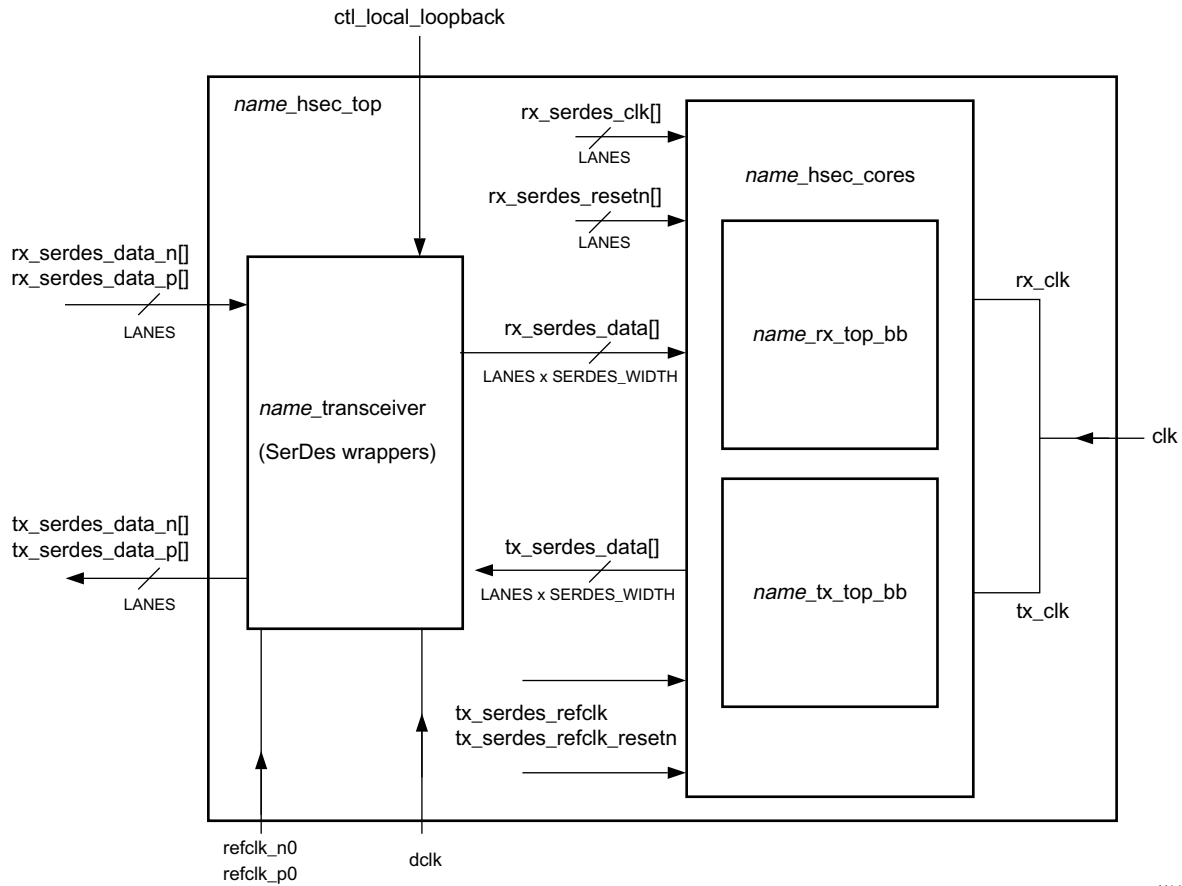
For full details about performance and resource utilization, visit the [Performance and Resource Utilization web page](#).

Port Descriptions

The following tables list all ports applicable to the 40G/50G subsystem including the ports used for optional features at the name_hsec_cores hierarchy level. At the xci top level hierarchy the ports change some to include the instantiated transceiver core and other shared logic. The port list at the XCI level of the hierarchy is described in [Core xci Top Level Port List in Chapter 5](#). Other Ports are described in [PCS Variant](#) and [Auto-Negotiation \(AN\) and Link Training \(LT\)](#).

[Figure 2-2](#) shows the relationship between the hierarchical blocks and illustrates the differences among the blocks in relation to the signals.

Note: When you generate the optional AXI4-Lite registers, some of these ports can be accessed by the corresponding register instead of a broadside bus.



X14116

Figure 2-2: Port List Hierarchy

The following sections describe the ports. The VL_LANES parameter is 4 and the LANES parameter is 2 for 50 Gb/s operation and 4 for 40 Gb/s operation.

Transceiver Ports

Table 2-1 describes the transceiver I/O ports.

Table 2-1: Transceiver I/O Port List

| Name | Direction | Clock Domain | Description |
|-----------------------------------|-----------|-------------------------------------|--|
| rx_serdes_data[LANES-1:0][64-1:0] | Input | Refer to Clocking . | Data bus from the SerDes macros. There are LANES rx_serdes_data buses; one bus for each SerDes lane and each bus has 64 bits. By definition, bit [64-1] is the first bit received by the 40G/50G High Speed Ethernet Subsystem. Bit [0] is the last bit received. A typical width is 64. |
| tx_serdes_data[LANES-1:0][64-1:0] | Output | Refer to Clocking . | Data bus to the SerDes macros. There are LANES tx_serdes_data buses; one bus for each SerDes lane and each bus has 64 bits. By definition, bit [64-1] is the first bit transmitted by the 40G/50G High Speed Ethernet Subsystem. Bit [0] is the last bit transmitted. A typical width is 64. |
| rx_serdes_clk[LANES-1:0] | Input | Refer to Clocking . | Recovered clock of each SerDes lane. The rx_serdes_data bus for each lane is synchronized to the positive edge of the corresponding bit of this bus. |
| rx_serdes_reset[LANES-1:0] | Input | Refer to Clocking . | Reset for each RX SerDes lane. The recovered clock for each SerDes lane has associated with it an active-High reset. This signal should be asserted whenever the associated recovered clock is not operating at the correct frequency. Generally this signal is connected to a phase-locked loop (PLL) lock signal. This is a synchronous reset. |
| tx_serdes_refclk | Input | Refer to Clocking . | Reference clock for the TX datapath. This clock must be frequency locked to the tx_serdes_clk inputs. Typically, the same reference clock that is used to drive the TX SerDes is connected to this input. |
| tx_serdes_refclk_reset | Input | Refer to Clocking . | Reset for TX Reference clock. This signal should be asserted whenever the tx_serdes_refclk input is not operating at the correct frequency. This is a synchronous reset. |

AXI4-Stream Interface Ports

This section describes how to connect the AXI4-Stream data interfaces of the 40G/50G subsystem. The AXI4-Stream interface follows the standard AXI4-Stream specification but makes extensive use of "user" signals. Refer to the Xilinx AXI4-Stream specification for a detailed description of the interface.

In the AXI4-Stream interface, [Table 2-2](#) describes the clock/reset signals, [Table 2-3](#) describes the Receive interface signals, [Table 2-4](#) describes the Transmit interface signals, [Table 2-5](#) describes the TX path control/status signals, and [Table 2-6](#) describes the RX path control/status signals.

Table 2-2: AXI4-Stream Interface – Clock/Reset Signals

| Name | Direction | Clock Domain | Description |
|-----------|-----------|--------------|--|
| clk | Input | | AXI4-Stream clock. All signals between the 40G/50G High Speed Ethernet Subsystem and the user-side logic are synchronized to the positive edge of this signal. |
| dclk | Input | | This must be a convenient stable clock, for example 75 MHz. Refer to the current transceiver guide for up to date information. |
| rx_reset | Input | | Reset for the RX circuits. This signal is active-High (1 = reset) and must be held High until clk is stable. The 40G/50G High Speed Ethernet Subsystem handles synchronizing the rx_reset input to the appropriate clock domains within the core. This is a synchronous reset. |
| refclk_n0 | Input | | Differential reference clock for the transceiver (N). |
| refclk_p0 | Input | | Differential reference clock for the transceiver (P). |
| tx_reset | Input | | Reset for the TX circuits. This signal is active-High (1 = reset) and must be held High until clk is stable. The 40G/50G High Speed Ethernet Subsystem handles synchronizing the tx_reset input to the appropriate clock domains within the core. This is a synchronous reset. |

Receive AXI4-Stream Interface

The receive AXI4-Stream interface is similar to the transmit side, with the RX data corresponding to the received Ethernet frame. The other signals on the RX AXI bus have a meaning analogous to the signals on the TX bus

The following table shows the AXI4-Stream receive interface signals.

Table 2-3: AXI4-Stream Receive Interface Signals

| Signal | Direction | Clock Domain | Description |
|--|-----------|--------------|---|
| rx_clk_out | Out | | All RX AXI signals are referenced to this clock. |
| rx_axis_tdata[127:0] | Out | rx_clk_out | AXI4-Stream Data to user logic. |
| rx_axis_tuser_tvalid | Out | rx_clk_out | AXI4-Stream Data Valid. When this signal is 1, there is valid data on the RX AXI data bus. |
| rx_axis_tuser_sop0 rx_axis_tuser_sop1 | Out | rx_clk_out | This signal, when asserted, indicates the start of a received Ethernet frame. |
| rx_axis_tuser_eop0 rx_axis_tuser_eop1 | Out | rx_clk_out | This signal, when asserted, indicates the end of a received Ethernet frame. There are two bits—one for each segment. |
| rx_axis_tuser_err0 rx_axis_tuser_err1 | Out | rx_clk_out | RX AXI error indication signal. <ul style="list-style-type: none"> • 1 indicates a bad packet has been received. • 0 indicates a good packet has been received. There are two bits—one for each segment. |
| rx_axis_tuser_mty0[2:0] rx_axis_tuser_mty1[2:0] | Out | rx_clk_out | This bus indicates how many bytes of the rx_axis_tdata bus are empty or invalid for the last transfer of the current packet. This bus is only valid during cycles when both rx_axis_tuser_ena and rx_axis_tuser_eop are 1. There are two bits—one for each segment. |
| rx_axis_tuser_ena0 rx_axis_tuser_ena1 | Out | rx_clk_out | Receive AXI4-Stream Enable for each Segment. When asserted, this signal indicates that data for the associated segment is valid. |

Normal Frame Reception

The timing of a normal inbound frame transfer is represented below. The client must be prepared to accept data at any time as there is insufficient buffering within the core to allow for latency in the receive client. When frame reception begins, data is transferred on consecutive clock cycles to the receive client. During frame reception, rx_axis_tuser_tvalid is asserted to indicate that valid frame data is being transferred to the client on rx_axis_tdata. All bytes are always valid throughout the frame, as indicated by all rx_axis_tuser_mty bits being set to 0, except during the final transfer of the frame when rx_axis_tuser_eop is asserted. During this final transfer of data for a frame, rx_axis_tuser_mty bits indicate the final valid bytes of the frame.

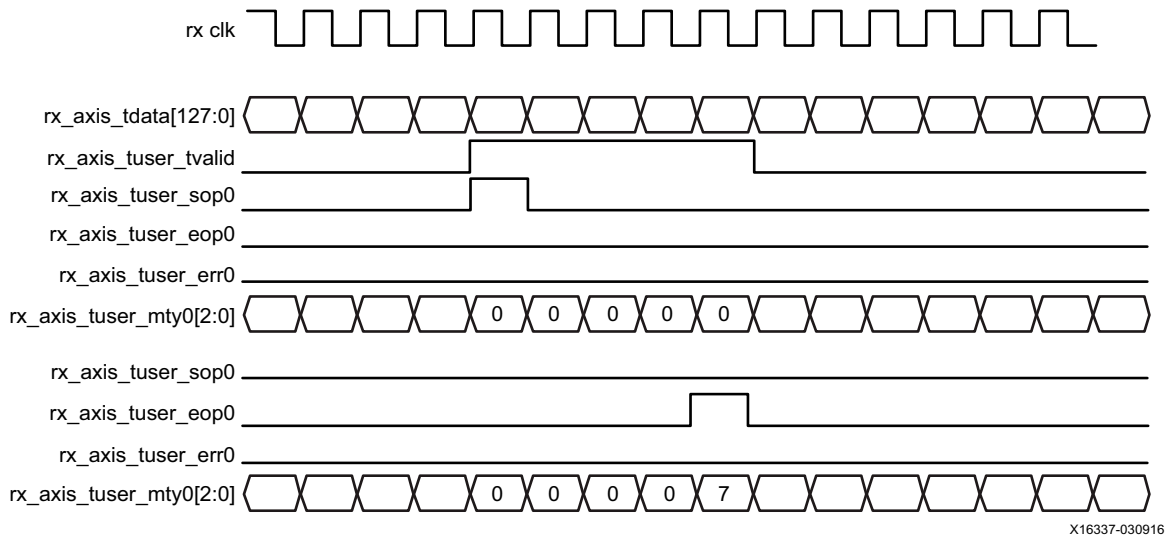


Figure 2-3: RX Waveform

The signals shown in the Figure 2-3 waveform are described in the following subsections.

rx clk

The rx_clk_out signal should be used as the clock reference for all RX AXI4-Stream signals. All RX AXI signals are aligned to the rising edge of this clock.

rx_axis_tdata[127:0]

This bus provides the packet-oriented data corresponding to the received Ethernet frame. Data is clocked by the rx_clk_out signal.

Figure 2-4 illustrates how the end of an Ethernet frame is mapped onto the bit positions of the RX AXI4-Stream interface. Note the positions of the ENA0 and ENA1 signals relative to the bits positions of the RX AXI4-Stream bus.

This mapping is for a 128-bit AXI4-Stream bus.

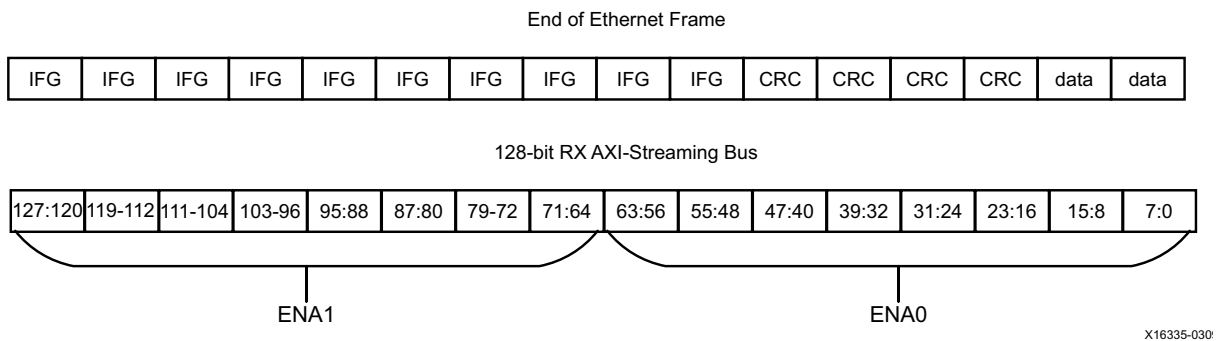


Figure 2-4: RX Mapping

rx_axis_tuser_tvalid

When asserted, this signal indicates that data on the RX AXI bus is valid.

rx_axis_tuser_ena0, rx_axi_tuser_ena1

When asserted, this signal indicates that data for the associated segment is valid.

rx_axis_tuser_sop0, rx_axis_tuser_sop1

This signal indicates the start of an Ethernet frame on the RX AXI bus. There is one SOP signal for each segment.

rx_axis_tuser_eop0, rx_axis_tuser_eop1

This signal indicates the end of an Ethernet frame on the RX AXI bus. There is one EOP signal for each segment.

rx_axis_tuser_err0, rx_axis_tuser_err1

When this signal is asserted, it indicates that there is an error in the received frame. It is valid during the EOP cycle when `ena` and `tvalid` are asserted. There is one error signal for each segment.

The types of errors that may exist include:

- There was an FCS error
- The length was out of the valid range
- A bad 64B/66B code was received during receipt of the packet

rx_axis_tuser_mty0[2:0], rx_axis_tuser_mty1[2:0]

As in the TX AXI interface, the `mty` signal indicates how many bytes of the current cycle are unused (empty) during the last cycle of a received packet (the EOP cycle). There is one `mty` signal for each RX segment.

Synchronous RX AXI4-Stream Interface

The synchronous RX AXI4-Stream interface provides packet-oriented data much like the TX AXI4-Stream interface accepts. All signals are synchronous with the rising-edge of the AXI4-Stream clock. [Figure 2-5](#) shows a sample waveform for data transaction for a 65-byte packet using a 128-bit data bus.

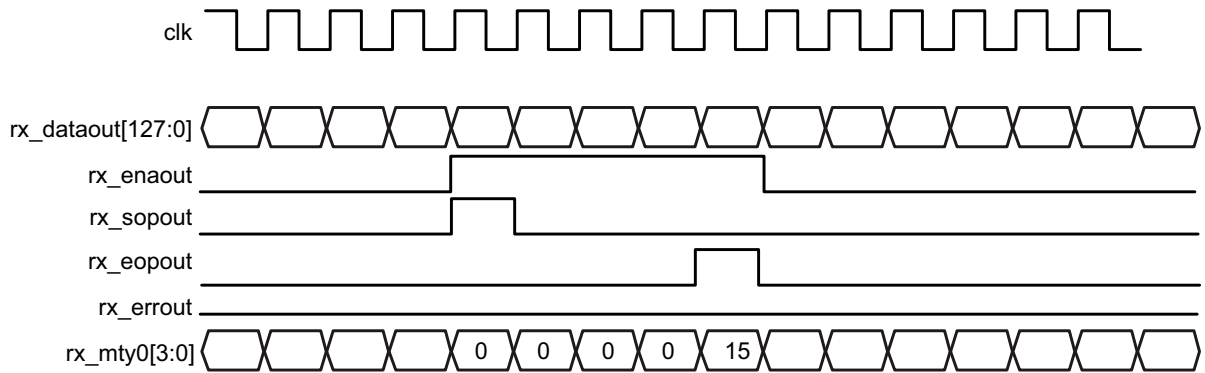


Figure 2-5: Sample RX Waveform With a 256-bit Data Bus

Data is supplied by the 40G/50G High Speed Ethernet Subsystem on every `clk` clock cycle when `rx_enaout` is asserted. This signal qualifies the other outputs of the RX AXI4-Stream interface.

Similar to the TX AXI4-Stream interface, `rx_sopout` identifies the start of a packet and `rx_eopout` identifies the end of a packet. Both `rx_sopout` and `rx_eopout` are asserted during the same cycle for packets that are less than or equal to the bus width.

Similar to the TX AXI4-Stream interface, the first byte of a packet is supplied on the most significant bits of `rx_dataout`. For a 128-bit wide bus, the first byte of the packet is written on bits [7:0], the second byte on bits [15:8], and so forth.

Similar to the TX AXI4-Stream interface, portions of packets are written on the bus in the full width of the bus unless `rx_eopout` is asserted. When `rx_eopout` is asserted, the `rx_mtyout` bus indicates how many byte lanes in the data bus are invalid. The encoding is the same as for `tx_mtyin`.

During the last cycle of a packet, when `rx_eopout` is asserted with `rx_enaout`, `rx_errout` might also be asserted. This indicates the packet received either had an FCS error, the length was out of the valid range (valid range is least 64 bytes and no more than 9,216 bytes), or had a bad 64B/66B code that was received during the receipt of the packet.

There is no mechanism to back pressure the RX AXI4-Stream interface. The user logic must be capable of receiving data when `rx_enaout` is asserted.

Transmit AXI4-Stream Interface

Table 2-4 shows the AXI4-Stream transmit interface signals.

Table 2-4: AXI4-Stream Transmit Interface Signals

| Signal | Direction | Clock Domain | Description |
|--|-----------|--------------|---|
| tx_clk_out | Out | | Transmit AXI clock. All TX signals are referenced to this clock. |
| tx_axis_tready | Out | tx_clk_out | When High, this signal indicates that the TX AXI interface is ready to accept data. You must respond immediately when tx_axis_tready goes Low by stopping data transfers. |
| tx_axis_tdata[127:0] | In | tx_clk_out | Transmit AXI4-Stream data (128-bit interface). The TX AXI data bus receives user-supplied packet data. |
| tx_axis_tvalid | In | tx_clk_out | AXI4-Stream Data Valid input. Data transfers are only completed when this signal is 1. |
| tx_axis_tuser_ena0 tx_axis_tuser_ena1 | In | tx_clk_out | Enable signal for the TX AXI bus transfers. A High on this signal enables transfer of data to the TX. |
| tx_axis_tuser_sop0 tx_axis_tuser_sop1 | In | tx_clk_out | AXI4-Stream signal indicating the Start of Ethernet Packet. There is one Start of Packet (SOP) signal per segment. |
| tx_axis_tuser_eop0 tx_axis_tuser_eop1 | In | tx_clk_out | AXI4-Stream signal indicating End of Ethernet Packet. There is one End of Packet (EOP) signal per segment. |
| tx_axis_tuser_err0 tx_axis_tuser_err0 | In | tx_clk_out | This signal is used to indicate that a packet contains an error when it is sampled as a 1 and is 0 for all other transfers of the packet. This signal is sampled only in cycles when tx_enain and tx_eopin are sampled as 1. When this signal is sampled as a 1, the last data word is replaced with the 802.3 Error Code control word that guarantees the partner device receives the packet in error. If a packet is input with this signal set to a 1, the FCS checking and reporting is disabled (only for that packet). There is one signal per segment. |
| tx_axis_tuser_mty0[2:0] tx_axis_tuser_mty1[2:0] | In | tx_clk_out | Transmit Empty. This bus is used to indicate how many bytes of the tx_datain bus are empty or invalid for the last transfer of the current packet. This bus is sampled only in cycles when tx_axis_valid and tx_axis_user_eopin are sampled as 1. |

The synchronous TX AXI bus interface accepts packet-oriented data. All signals are synchronous relative to the rising edge of the tx_clk_out port.

The AXI4-Stream transmit interface consists of two segments, where each is 64 bits (8 bytes) wide. This segmented approach allows for greater efficiency such that a packet can start and end in any given segment or cycle.

A normal transmit cycle is shown in the following waveform. These waveforms illustrate the transfer of a 73-byte packet over a 128-bit wide AXI4-Stream interface. There are two segments shown, numbered 0 and 1.

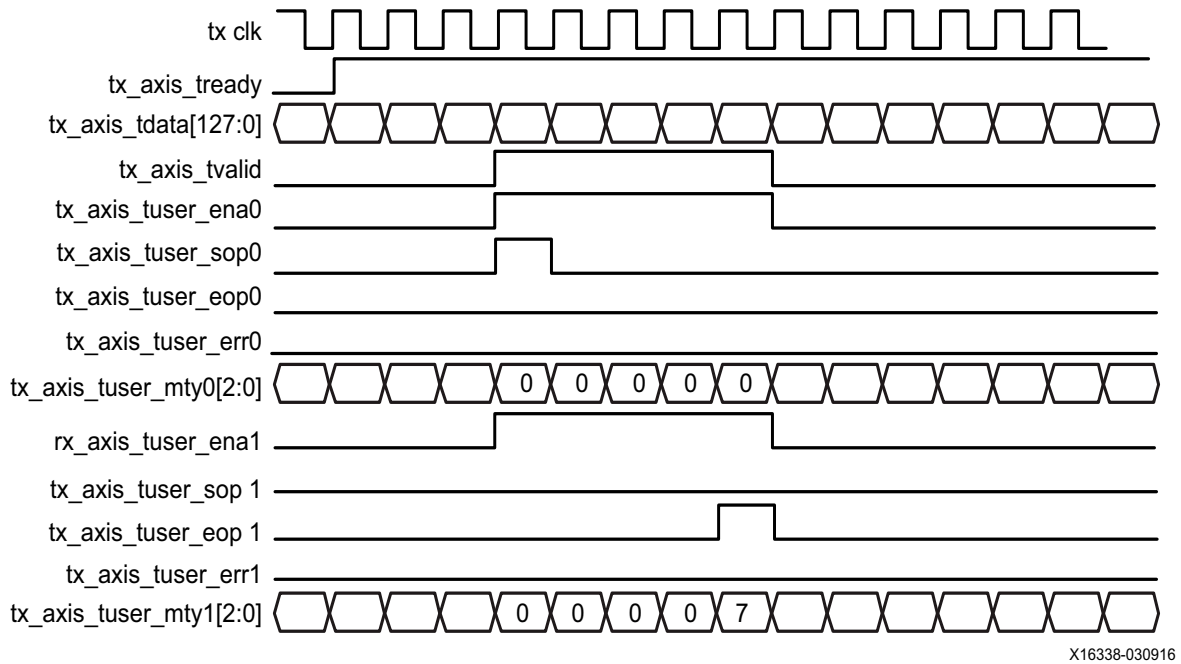


Figure 2-6: TX Waveform

The signals shown in the Figure 2-6 waveform are described in the following subsections.

tx_clk

This is the signal `tx_clk_out` output of the subsystem. All TX AXI signals are referenced to this clock. The frequency of this clock is normally 390.625 MHz for 50G operation and 312.5 MHz for 40G operation.

tx_axis_tready

When asserted, this signal indicates that the TX AXI4-Stream interface is able to accept data. When `tx_axi_tready` goes Low, you must stop sending data immediately, or it will not be accepted by the TX AXI4-Stream interface.

tx_axis_tdata[127:0]

This is the bus for the frame to be transmitted.

Figure 2-7 illustrates how the start of an Ethernet frame is mapped onto the bit positions of the TX AXI4-Stream interface. Note the positions of the ENA0 and ENA1 signals relative to the bits positions of the TX AXI4-Stream bus.

This mapping is for a 128-bit AXI4-Stream bus.

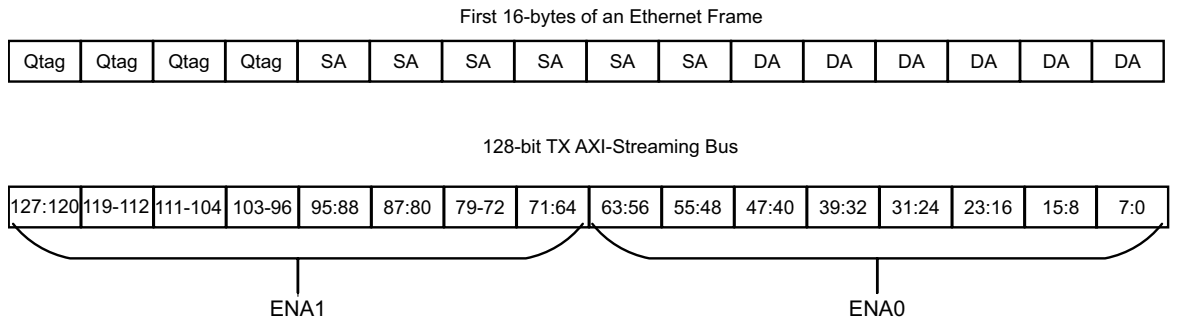


Figure 2-7: TX Mapping

tx_axis_tvalid

When High, this signal indicates that there is valid data on the TX AXI bus.

tx_axis_tuser_ena0, tx_axis_tuser_ena1

These signals enable the transfer of data over the TX bus when asserted. Data transfer has to be validated with the tvalid signal in order for a transfer to take place.

There is an enable signal for each AXI segment.

tx_axis_tuser_sop0, tx_axis_tuser_sop1

These signals indicate the start of an Ethernet frame in that cycle. Only one SOP is permitted in a bus cycle. There is a separate SOP signal for each AXI segment.

tx_axis_tuser_eop0, tx_axis_tuser_eop1

These signals indicate the end of an Ethernet frame in that cycle. Only one EOP is permitted in a bus cycle. There is a separate EOP signal for each AXI segment.

tx_axis_tuser_mty0[2:0], tx_axis_tuser_mty1[2:0]

These signals indicate which bytes of the corresponding segment are not used ("empty"). If tx_mtyin has a value of 0x0, there are no empty byte lanes, or in other words, all bits of the data bus are valid.

For example, if tx_axis_tuser_mty0[2:0] = 2, the last 2 bytes of the segment do not contain data and will be ignored. The value of mty can only be non-zero during the last cycle of a packet transfer (the EOP cycle).

tx_unfout

Not shown on the waveforms in [Figure 2-6](#) is the `tx_unfout` output indicator. When this signal is High, it indicates that there has not been a sufficient data transfer and the Ethernet interface will underflow. This must not be allowed to occur. You must ensure that you transfer data whenever `tx_axis_tready` is High until you reach the end of the Ethernet frame.

tx_axis_user_err0, tx_axis_user_err1

This signal is used to indicate that a packet contains an error when it is sampled as a 1 and is 0 for all other transfers of the packet. This signal is sampled only in cycles when `tx_axis_tuser_ena` and `tx_axis_tuser_eop` are sampled as 1. When this signal is sampled as a 1, the last data word is replaced with the 802.3 Error Code control word that guarantees the partner device receives the packet in error. If a packet is input with this signal set to a 1, the FCS checking and reporting is disabled (only for that packet).

There is one `tx_axis_user_err` signal per segment.

TX Path Control/Status Signals

Table 2-5 describes the other status/control signals.

Table 2-5: TX Path Control/Status Signals

| Name | Direction | Clock Domain | Description |
|-----------------------|-----------|--------------|---|
| ctl_rate_mode | Input | static | This signal causes the IP core to switch between 50G operation (0) and 40G operation (1). Note that the clock frequencies must be corrected for the mode chosen. |
| ctl_tx_enable | Input | clk | TX Enable. This signal is used to enable the transmission of data when it is sampled as a 1. When sampled as a 0, only idles are transmitted by the 40G/50G High Speed Ethernet Subsystem. This input should not be set to 1 until the receiver it is sending data to (that is, the receiver in the other device) is fully aligned and ready to receive data (that is, the other device is not sending a remote fault condition). Otherwise, loss of data can occur. If this signal is set to 0 while a packet is being transmitted, the current packet transmission is completed and the 40G/50G High Speed Ethernet Subsystem stops transmitting anymore packets. |
| ctl_tx_send_rfi | Input | clk | Transmit Remote Fault Indication (RFI) code word. If this input is sampled as a 1, the TX path only transmits Remote Fault code words. This input should be set to 1 until the RX path is fully aligned and is ready to accept data from the link partner. |
| ctl_tx_send_lfi | Input | clk | Transmit Local Fault Indication (LFI) code word. Takes precedence over RFI. |
| ctl_tx_send_idle | Input | clk | Transmit Idle code words. If this input is sampled as a 1, the TX path only transmits Idle code words. This input should be set to 1 when the partner device is sending Remote Fault Indication (RFI) code words. |
| ctl_tx_fcs_ins_enable | Input | clk | Enable FCS insertion by the TX core. If this bit is set to 0, the 40G/50G High Speed Ethernet Subsystem does not add FCS to the packet. If this bit is set to 1, the 40G/50G High Speed Ethernet Subsystem calculates and adds the FCS to the packet. This input cannot be changed dynamically between packets. |

Table 2-5: TX Path Control/Status Signals (Cont'd)

| Name | Direction | Clock Domain | Description |
|---|-----------|--------------|---|
| ctl_tx_ignore_fcs | Input | clk | <p>Enable FCS error checking at the AXI4-Stream interface by the TX core. This input only has effect when <code>ctl_tx_fcs_ins_enable</code> is Low. If this input is Low and a packet with bad FCS is being transmitted, it is not binned as good. If this input is High, a packet with bad FCS is binned as good.</p> <p>The error is flagged on the signals <code>stat_tx_bad_fcs</code> and <code>stomped_fcs</code>, and the packet is transmitted as it was received.</p> <p>Note: Statistics are reported as if there was no FCS error.</p> |
| ctl_tx_vl_length_minus1[15:0] | Input | static | <p>Number of words in between PCS Lane markers minus one. Default value, as defined in IEEE Std 802.3-2012, should be set to 16,383. This input should only be changed while the corresponding reset input is asserted.</p> |
| ctl_tx_vl_marker_id[VL_LANES-1:0][63:0] | Input | static | <p>These inputs set the PCS Lane markers for each PCS lane. For 802.3 default values, see the IEEE Std 802.3-2012 [Ref 1]. This input should only be changed while the corresponding reset input is asserted.</p> |
| stat_tx_local_fault | Output | clk | <p>A value of 1 indicates the transmit encoder state machine is in the TX_INIT state. This output is level sensitive.</p> |
| ctl_tx_custom_preamble_enable | Input | tx_clk | <p>When asserted, this signal treats the first 64 bits of a packet on the <code>rx_serdes_clk</code> as a custom preamble instead of inserting a standard preamble.</p> <p>When asserted, this signals enables the use of <code>tx_preamblein</code> as a custom preamble instead of inserting a standard preamble.</p> |
| tx_preamblein[55:0] | Input | tx_clk | <p>This bus represents the custom preamble when the signal <code>ctl_tx_custom_preamble_enable</code> is asserted. It should be asserted on the first cycle of the packet (start of packet).</p> |

Table 2-6: RX Path Control/Status Signals

| Name | Direction | Clock Domain | Description |
|-----------------------|-----------|---------------|--|
| ctl_rate_mode | Input | static | This signal causes the IP core to switch between 50G operation (0) and 40G operation (1). Note that the clock frequencies will need to be correct for the mode chose. |
| ctl_rx_enable | Input | rx_serdes_clk | RX Enable. For normal operation, this input must be set to 1. When this input is set to 0, after the RX completes the reception of the current packet (if any), it stops receiving packets by keeping the PCS from decoding incoming data. In this mode, there are no statistics reported and the AXI4-Stream interface is idle. |
| ctl_rx_check_preamble | Input | rx_serdes_clk | When asserted, this input causes the Ethernet MAC to check the preamble of the received frame. |
| ctl_rx_check_sfd | Input | rx_serdes_clk | When asserted, this input causes the Ethernet MAC to check the start of frame Delimiter of the received frame. |
| ctl_rx_force_resync | Input | rx_serdes_clk | RX force resynchronization input. This signal is used to force the RX path to reset, re-synchronize, and realign. A value of 1 forces the reset operation. A value of 0 allows normal operation. Note: This input should normally be Low and should only be pulsed (1 cycle minimum pulse) to force realignment. |
| ctl_rx_delete_fcs | Input | rx_serdes_clk | Enable FCS removal by the RX core. If this bit is set to 0, the 40G/50G High Speed Ethernet Subsystem does not remove the FCS of the incoming packet. If this bit is set to 1, the 40G/50G High Speed Ethernet Subsystem deletes the FCS to the received packet. FCS is not deleted for packets that are ≤ 8 bytes long. This input should only be changed while the corresponding reset input is asserted. |

Table 2-6: RX Path Control/Status Signals (Cont'd)

| Name | Direction | Clock Domain | Description |
|---|-----------|---------------|---|
| ctl_rx_ignore_fcs | Input | rx_serdes_clk | <p>Enable FCS error checking at the AXI4-Stream interface by the RX core. If this bit is set to 0, a packet received with an FCS error is sent with the rx_errout pin asserted during the last transfer (rx_eopout and rx_enaout sampled 1). If this bit is set to 1, the 40G/50G High Speed Ethernet Subsystem does not flag an FCS error at the AXI4-Stream interface.</p> <p>Note: The statistics are reported as if the packet is good. The <i>stat_rx_bad_fcs</i> signal, however, reports the error.</p> |
| ctl_rx_max_packet_len[14:0] | Input | rx_serdes_clk | <p>Any packet longer than this value is considered to be oversized. If a packet has a size greater than this value, the packet is truncated to this value and the rx_errout signal is asserted along with the rx_eopout signal. Packets less than 64 bytes are dropped. The allowed value for this bus can range from 64 to 16,383. ctl_rx_max_packet_len[14] is reserved and must be set to 0.</p> |
| ctl_rx_min_packet_len[7:0] | Input | rx_serdes_clk | <p>Any packet shorter than this value is considered to be undersized. If a packet has a size less than this value, the rx_errout signal is asserted during the rx_eopout asserted cycle. Packets that are less than 64 bytes are dropped.</p> |
| ctl_rx_vl_length_minus1[15:0] | Input | rx_serdes_clk | <p>Number of words in between PCS Lane markers minus one. Default value, as defined in IEEE Std 802.3-2012, should be set to 16,383. This input should only be changed while the corresponding reset input is asserted.</p> |
| ctl_rx_vl_marker_id[VL_LANES-1:0][63:0] | Input | rx_serdes_clk | <p>These inputs set the PCS Lane markers for each PCS lane. These inputs should be set to the values as defined in the IEEE Std 802.3-2012. For IEEE 802.3 default values, see Section 5.3 [Ref 1]. This input should only be changed while the corresponding reset input is asserted.</p> |

Table 2-6: RX Path Control/Status Signals (Cont'd)

| Name | Direction | Clock Domain | Description |
|--|-----------|--------------|---|
| stat_rx_framing_err_[VL_LANES-1:0][3:0] | Output | rx_clk_out | RX sync header bits framing error. Each PCS Lane has a four-bit bus that indicates how many sync header errors were received for that PCS Lane. The value of the bus is only valid when the corresponding stat_rx_framing_err_valid_[VL_LANES-1:0] is a 1. The values on these buses can be updated at any time and are intended to be used as increment values for sync header error counters. |
| stat_rx_framing_err_valid_[VL_LANES-1:0] | Output | rx_clk_out | Valid indicator for stat_rx_framing_err_[VL_LANES-1:0]. When one of these outputs is sampled as a 1, the value on the corresponding stat_rx_framing_err_[VL_LANES-1:0] is valid. |
| stat_rx_local_fault | Output | rx_clk_out | This output is High when stat_rx_internal_local_fault or stat_rx_received_local_fault is asserted. This output is level sensitive. |
| stat_rx_synced[VL_LANES-1:0] | Output | rx_clk_out | Word Boundary Synchronized. These signals indicate whether a PCS lane is word boundary synchronized. A value of 1 indicates the corresponding PCS lane has achieved word boundary synchronization and it has received a PCS lane marker. Corresponds to management data input/output (MDIO) register bit 3.52.7:0 and 3.53.11:0 as defined in Clause 82.3. This output is level sensitive. |
| stat_rx_synced_err[VL_LANES-1:0] | Output | rx_clk_out | Word Boundary Synchronization Error. These signals indicate whether an error occurred during word boundary synchronization in the respective PCS lane. A value of 1 indicates that the corresponding PCS lane lost word boundary synchronization due to sync header framing bits errors or that a PCS lane marker was never received. This output is level sensitive. |

Table 2-6: RX Path Control/Status Signals (Cont'd)

| Name | Direction | Clock Domain | Description |
|-------------------------------------|-----------|--------------|---|
| stat_rx_mf_len_err[VL_LANES-1:0] | Output | rx_clk_out | PCS Lane Marker Length Error. These signals indicate whether a PCS Lane Marker length mismatch occurred in the respective lane (that is, PCS Lane Markers were received not every <code>ctl_rx_vl_length_minus1</code> words apart). A value of 1 indicates that the corresponding lane is receiving PCS Lane Markers at wrong intervals. This remains High until the error condition is removed. |
| stat_rx_mf_repeat_err[VL_LANES-1:0] | Output | rx_clk_out | PCS Lane Marker Consecutive Error. These signals indicate whether four consecutive PCS Lane Marker errors occurred in the respective lane. A value of 1 indicates an error in the corresponding lane. This output remains High until the error condition is removed. |
| stat_rx_mf_err[VL_LANES-1:0] | Output | rx_clk_out | PCS Lane Marker Word Error. These signals indicate that an incorrectly formed PCS Lane Marker Word was detected in the respective lane. A value of 1 indicates an error occurred. This output is pulsed for one clock cycle to indicate the error condition. Pulses can occur in back-to-back cycles. |
| stat_rx_aligned | Output | rx_clk_out | All PCS Lanes Aligned/Deskewed. This signal indicates whether or not all PCS lanes are aligned and deskewed. A value of 1 indicates all PCS lanes are aligned and deskewed. When this signal is a 1, the RX path is aligned and can receive packet data. When this signal is 0, a local fault condition exists. This also corresponds to MDIO register bit 3.50.12 as defined in Clause 82.3. This output is level sensitive. |
| stat_rx_status | Output | rx_clk_out | PCS status. A value of 1 indicates that the PCS is aligned and not in hi_ber state. Corresponds to MDIO register bit 3.32.12 as defined in Clause 82.3. This output is level sensitive. |
| stat_rx_block_lock[VL_LANES-1:0] | Output | rx_clk_out | Block lock status for each PCS lane. A value of 1 indicates that the corresponding lane has achieved block lock as defined in Clause 82. Corresponds to MDIO register bit 3.50.7:0 and 3.51.11:0 as defined in Clause 82.3. This output is level sensitive. |

Table 2-6: RX Path Control/Status Signals (Cont'd)

| Name | Direction | Clock Domain | Description |
|----------------------------------|-----------|--------------|---|
| stat_rx_aligned_err | Output | rx_clk_out | Loss of Lane Alignment/Deskew. This signal indicates that an error occurred during PCS lane alignment or PCS lane alignment was lost. A value of 1 indicates an error occurred. This output is level sensitive. |
| stat_rx_misaligned | Output | rx_clk_out | Alignment Error. This signal indicates that the lane aligner did not receive the expected PCS lane marker across all lanes. This signal is not asserted until the PCS lane marker has been received at least once across all lanes and at least one incorrect lane marker has been received. This occurs one metaframe after the error. This signal is not asserted if the lane markers have never been received correctly. Lane marker errors are indicated by the corresponding stat_rx_mf_err signal. This output is pulsed for one clock cycle to indicate an error condition. Pulses can occur in back-to-back cycles. |
| stat_rx_remote_fault | Output | rx_clk_out | Remote fault indication status. If this bit is sampled as a 1, it indicates a remote fault condition was detected. If this bit is sampled as a 0, a remote fault condition does not exist. This output is level sensitive. |
| stat_rx_vl_number_[3:0][1:0] | Output | rx_clk_out | There are a total of VL_LANES separate stat_rx_vl_number[4]1:0 buses. stat_rx_vl_number_# indicates which PCS lane is being received on the corresponding physical lane. This bus is only valid when the corresponding bit of stat_rx_synced[VL_LANES-1:0] is a 1. These outputs are level sensitive. |
| stat_rx_vl_demuxed[VL_LANES-1:0] | Output | rx_clk_out | PCS Lane Marker found. If a signal of this bus is sampled as 1, it indicates that the receiver has properly de-muxed that PCS lane. These outputs are level sensitive. |

Table 2-6: RX Path Control/Status Signals (Cont'd)

| Name | Direction | Clock Domain | Description |
|------------------------------|-----------|--------------|---|
| stat_rx_bad_fcs[n:0] | Output | rx_clk_out | Bad FCS indicator. The value on this bus indicates packets received with a bad FCS, but not a stomped FCS. A stomped FCS is defined as the bitwise inverse of the expected good FCS. This output is pulsed for one clock cycle to indicate an error condition. Pulses can occur in back-to-back cycles. |
| stat_rx_stomped_fcs[n:0] | Output | rx_clk_out | Stomped FCS indicator. The value on this bus indicates packets were received with a stomped FCS. A stomped FCS is defined as the bitwise inverse of the expected good FCS. This output is pulsed for one clock cycle to indicate the stomped condition. Pulses can occur in back-to-back cycles. |
| stat_rx_truncated | Output | rx_clk_out | Packet truncation indicator. A value of 1 indicates that the current packet in flight is truncated due to its length exceeding <code>ctl_rx_max_packet_len[14:0]</code> . This output is pulsed for one clock cycle to indicate the truncated condition. Pulses can occur in back-to-back cycles. |
| stat_rx_internal_local_fault | Output | rx_clk_out | This signal goes High when an internal local fault is generated due to any one of the following: test pattern generation, bad lane alignment, or high bit error rate. This signal remains High as long as the fault condition persists. |
| stat_rx_received_local_fault | Output | rx_clk_out | This signal goes High when enough local fault words are received from the link partner to trigger a fault condition as specified by the IEEE fault state machine. This signal remains High as long as the fault condition persists. |
| stat_rx_bip_err[VLANES-1:0] | Output | rx_clk_out | BIP8 error indicator. A non-zero value indicates the BIP8 signature byte was in error for the corresponding PCS lane. A non-zero value is pulsed for one clock cycle. This output is pulsed for one clock cycle to indicate an error condition. Pulses can occur in back-to-back cycles. |
| stat_rx_hi_ber | Output | rx_clk_out | High Bit Error Rate (BER) indicator. When set to 1, the BER is too high as defined by IEEE Std 802.3-2012. Corresponds to MDIO register bit 3.32.1 as defined in Clause 82.3. This output is level sensitive. |

Table 2-6: RX Path Control/Status Signals (Cont'd)

| Name | Direction | Clock Domain | Description |
|-------------------------------|-----------|--------------|---|
| ctl_rx_custom_preamble_enable | Input | rx_clk_out | When asserted, this signal causes the preamble to be presented on rx_preambleout. |
| rx_preambleout[55:0] | Output | rx_clk | This bus represents the preamble bytes when the ctl_rx_custom_preamble_enable signal is asserted. It is valid on the first cycle of the packet. |

Miscellaneous Status/Control Ports

Table 2-7 describes the other status/control signals.

Table 2-7: Miscellaneous Status/Control Signals

| Name | Direction | Clock Domain | Description |
|-----------------------|-----------|--------------|---|
| ctl_tx_ipg_value[3:0] | Input | | This signal can be optionally present. The ctl_tx_ipg_value defines the target average minimum Inter Packet Gap (IPG, in bytes) inserted between rx_serdes_clk packets. Typical value is 12. The ctl_tx_ipg_value can also be programmed to a value in the 0 to 7 range, but in that case, it is interpreted as meaning "minimal IPG", so only Terminate code word IPG is inserted; no Idles are ever added in that case and that produces an average IPG of around 4 bytes when random-size packets are transmitted. |
| stat_rx_got_signal_os | Output | rx_clk_out | Signal OS indication. If this bit is sampled as a 1, it indicates that a Signal OS word was received. Note: Signal OS should not be received in an Ethernet network. |
| ctl_rx_process_lfi | Input | rx_clk_out | When this input is set to 1, the RX core expects and processes Local Fault (LF) control codes coming in from the SerDes. When set to 0, the RX core ignores LF control codes coming in from the SerDes. |
| ctl_rx_test_pattern | Input | rx_clk_out | Test pattern checking enable for the RX core. A value of 1 enables test mode as defined in Clause 82.2.17. Corresponds to MDIO register bit 3.42.2 as defined in Clause 82.3. Checks for scrambled idle pattern. |

Table 2-7: Miscellaneous Status/Control Signals (Cont'd)

| Name | Direction | Clock Domain | Description |
|--|-----------|--------------|---|
| ctl_tx_test_pattern | Input | clk | Test pattern generation enable for the TX core. A value of 1 enables test mode as defined in Clause 82.2.10. Corresponds to MDIO register bit 3.42.3 as defined in Clause 82.3. Generates a scrambled idle pattern. |
| stat_rx_test_pattern_mismatch[3 2 1 0:0] | Output | rx_clk_out | Test pattern mismatch increment. A non zero value in any cycle indicates how many mismatches occurred for the test pattern in the RX core. This output is only active when ctl_rx_test_pattern is set to a 1. This output can be used to generate MDIO register 3.43.15:0 as defined in Clause 82.3. This output is pulsed for one clock cycle. |

Statistics Interface Ports

In the Statistics Interface, [Table 2-8](#) describes the RX path signals, and [Table 2-9](#) describes the TX path signals.

Table 2-8: Statistics Interface – RX Path Signals

| Name | Direction | Clock Domain | Description |
|--------------------------------|-----------|--------------|---|
| stat_rx_total_bytes[6 5 3 2:0] | Output | clk | Increment for the total number of bytes received. |
| stat_rx_total_packets[n:0] | Output | clk | Increment for the total number of packets received. |
| stat_rx_total_good_bytes[13:0] | Output | clk | Increment for the total number of good bytes received. This value is only non-zero when a packet is received completely and contains no errors. |
| stat_rx_total_good_packets | Output | clk | Increment for the total number of good packets received. This value is only non-zero when a packet is received completely and contains no errors. |
| stat_rx_packet_bad_fcs | Output | clk | Increment for packets between 64 and ctl_rx_max_packet_len bytes that have FCS errors. |
| stat_rx_packet_64_bytes | Output | clk | Increment for good and bad packets received that contain 64 bytes. |
| stat_rx_packet_65_127_bytes | Output | clk | Increment for good and bad packets received that contain 65 to 127 bytes. |
| stat_rx_packet_128_255_bytes | Output | clk | Increment for good and bad packets received that contain 128 to 255 bytes. |
| stat_rx_packet_256_511_bytes | Output | clk | Increment for good and bad packets received that contain 256 to 511 bytes. |
| stat_rx_packet_512_1023_bytes | Output | clk | Increment for good and bad packets received that contain 512 to 1,023 bytes. |

Table 2-8: Statistics Interface – RX Path Signals (Cont'd)

| Name | Direction | Clock Domain | Description |
|--------------------------------|-----------|--------------|--|
| stat_rx_packet_1024_1518_bytes | Output | clk | Increment for good and bad packets received that contain 1,024 to 1,518 bytes. |
| stat_rx_packet_1519_1522_bytes | Output | clk | Increment for good and bad packets received that contain 1,519 to 1,522 bytes. |
| stat_rx_packet_1523_1548_bytes | Output | clk | Increment for good and bad packets received that contain 1,523 to 1,548 bytes. |
| stat_rx_packet_1549_2047_bytes | Output | clk | Increment for good and bad packets received that contain 1,549 to 2,047 bytes. |
| stat_rx_packet_2048_4095_bytes | Output | clk | Increment for good and bad packets received that contain 2,048 to 4,095 bytes. |
| stat_rx_packet_4096_8191_bytes | Output | clk | Increment for good and bad packets received that contain 4,096 to 8,191 bytes. |
| stat_rx_packet_8192_9215_bytes | Output | clk | Increment for good and bad packets received that contain 8,192 to 9,215 bytes. |
| stat_rx_packet_small[n:0] | Output | clk | Increment for all packets that are less than 64 bytes long. Packets that are less than 64 bytes are dropped. |
| stat_rx_packet_large | Output | clk | Increment for all packets that are more than 9,215 bytes long. |
| stat_rx_unicast | Output | clk | Increment for good unicast packets. |
| stat_rx_multicast | Output | clk | Increment for good multicast packets. |
| stat_rx_broadcast | Output | clk | Increment for good broadcast packets. |
| stat_rx_oversize | Output | clk | Increment for packets longer than <code>ctl_rx_max_packet_len</code> with good FCS. |
| stat_rx_toolong | Output | clk | Increment for packets longer than <code>ctl_rx_max_packet_len</code> with good and bad FCS. |
| stat_rx_undersize[n:0] | Output | clk | Increment for packets shorter than <code>stat_rx_min_packet_len</code> with good FCS. |
| stat_rx_fragment[n:0] | Output | clk | Increment for packets shorter than <code>ctl_rx_min_packet_len</code> with bad FCS. |
| stat_rx_vlan | Output | clk | Increment for good 802.1Q tagged VLAN packets. |
| stat_rx_inrangeerr | Output | clk | Increment for packets with Length field error but with good FCS. |
| stat_rx_jabber | Output | clk | Increment for packets longer than <code>ctl_rx_max_packet_len</code> with bad FCS. |
| stat_rx_pause | Output | clk | Increment for 802.3x Ethernet MAC Pause packet with good FCS. |
| stat_rx_user_pause | Output | clk | Increment for priority-based pause packets with good FCS. |

Table 2-8: Statistics Interface – RX Path Signals (Cont'd)

| Name | Direction | Clock Domain | Description |
|-----------------------------|-----------|--------------|--|
| stat_rx_bad_code[3 2 1 0:0] | Output | clk | Increment for 64B/66B code violations. This signal indicates that the RX PCS receive state machine is in the RX_E state as specified by the IEEE Std 802.3-2012. This output can be used to generate MDIO register 3.33:7:0 as defined in Clause 82.3. |
| stat_rx_bad_sfd | Output | clk | Increment bad start of frame delimiter (SFD). This signal indicates if the Ethernet packet received was preceded by a valid SFD. A value of 1 indicates that an invalid SFD was received. |
| stat_rx_bad_preamble | Output | clk | Increment bad preamble. This signal indicates if the Ethernet packet received was preceded by a valid preamble. A value of 1 indicates that an invalid preamble was received. |

Table 2-9: Statistics Interface – TX Path Signals

| Name | Direction | Clock Domain | Description |
|--------------------------------|-----------|--------------|---|
| stat_tx_total_bytes[6 5 3 2:0] | Output | clk | Increment for the total number of bytes transmitted. |
| stat_tx_total_packets | Output | clk | Increment for the total number of packets transmitted. |
| stat_tx_total_good_bytes[13:0] | Output | clk | Increment for the total number of good bytes transmitted. This value is only non-zero when a packet is transmitted completely and contains no errors. |
| stat_tx_total_good_packets | Output | clk | Increment for the total number of good packets transmitted. |
| stat_tx_bad_fcs | Output | clk | Increment for packets greater than 64 bytes that have FCS errors. |
| stat_tx_packet_64_bytes | Output | clk | Increment for good and bad packets transmitted that contain 64 bytes. |
| stat_tx_packet_65_127_bytes | Output | clk | Increment for good and bad packets transmitted that contain 65 to 127 bytes. |
| stat_tx_packet_128_255_bytes | Output | clk | Increment for good and bad packets transmitted that contain 128 to 255 bytes. |
| stat_tx_packet_256_511_bytes | Output | clk | Increment for good and bad packets transmitted that contain 256 to 511 bytes. |
| stat_tx_packet_512_1023_bytes | Output | clk | Increment for good and bad packets transmitted that contain 512 to 1,023 bytes. |
| stat_tx_packet_1024_1518_bytes | Output | clk | Increment for good and bad packets transmitted that contain 1,024 to 1,518 bytes. |

Table 2-9: Statistics Interface – TX Path Signals (Cont'd)

| Name | Direction | Clock Domain | Description |
|--------------------------------|-----------|--------------|--|
| stat_tx_packet_1519_1522_bytes | Output | clk | Increment for good and bad packets transmitted that contain 1,519 to 1,522 bytes. |
| stat_tx_packet_1523_1548_bytes | Output | clk | Increment for good and bad packets transmitted that contain 1,523 to 1,548 bytes. |
| stat_tx_packet_1549_2047_bytes | Output | clk | Increment for good and bad packets transmitted that contain 1,549 to 2,047 bytes. |
| stat_tx_packet_2048_4095_bytes | Output | clk | Increment for good and bad packets transmitted that contain 2,048 to 4,095 bytes. |
| stat_tx_packet_4096_8191_bytes | Output | clk | Increment for good and bad packets transmitted that contain 4,096 to 8,191 bytes. |
| stat_tx_packet_8192_9215_bytes | Output | clk | Increment for good and bad packets transmitted that contain 8,192 to 9,215 bytes. |
| stat_tx_packet_small | Output | clk | Increment for all packets that are less than 64 bytes long. Packets that are less than 64 bytes are not transmitted. |
| stat_tx_packet_large | Output | clk | Increment for all packets that are more than 9,215 bytes long. |
| stat_tx_unicast | Output | clk | Increment for good unicast packets. |
| stat_tx_multicast | Output | clk | Increment for good multicast packets. |
| stat_tx_broadcast | Output | clk | Increment for good broadcast packets. |
| stat_tx_vlan | Output | clk | Increment for good 802.1Q tagged VLAN packets. |
| stat_tx_pause | Output | clk | Increment for 802.3x Ethernet MAC Pause packet with good FCS. |
| stat_tx_user_pause | Output | clk | Increment for priority-based pause packets with good FCS. |
| stat_tx_frame_error | Output | clk | Increment for packets with tx_errin set to indicate an EOP abort. |

Pause Interface Ports

Table 2-10 describes the control signals, Table 2-11 describes the RX path signals, and Table 2-12 describes the TX path signals.

Table 2-10: Pause Interface – Control Signals

| Name | Direction | Clock Domain | Description |
|--------------------------|-----------|---------------|---|
| ctl_rx_pause_enable[8:0] | Input | rx_serdes_clk | RX pause enable signal. This input is used to enable the processing of the pause quanta for the corresponding priority. Note: This signal only affects the RX user interface, not the pause processing logic. |
| ctl_tx_pause_enable[8:0] | Input | clk | TX pause enable signal. This input is used to enable the processing of the pause quanta for the corresponding priority. This signal gates transmission of pause packets. |

Table 2-11: Pause Interface – RX Path Signals

| Name | Direction | Clock Domain | Description |
|-----------------------------|-----------|---------------|---|
| ctl_rx_enable_gcp | Input | rx_serdes_clk | A value of 1 enables global control packet processing. |
| ctl_rx_check_mcast_gcp | Input | rx_serdes_clk | A value of 1 enables global control multicast destination address processing. |
| ctl_rx_check_ucast_gcp | Input | rx_serdes_clk | A value of 1 enables global control unicast destination address processing. |
| ctl_rx_pause_da_ucast[47:0] | Input | rx_serdes_clk | Unicast destination address for pause processing. |
| ctl_rx_check_sa_gcp | Input | rx_serdes_clk | A value of 1 enables global control source address processing. |
| ctl_rx_pause_sa[47:0] | Input | rx_serdes_clk | Source address for pause processing. |
| ctl_rx_check_etype_gcp | Input | rx_serdes_clk | A value of 1 enables global control ethertype processing. |
| ctl_rx_check_opcode_gcp | Input | rx_serdes_clk | A value of 1 enables global control opcode processing. |
| ctl_rx_opcode_min_gcp[15:0] | Input | rx_serdes_clk | Minimum global control opcode value. |
| ctl_rx_opcode_max_gcp[15:0] | Input | rx_serdes_clk | Maximum global control opcode value. |
| ctl_rx_etype_gcp[15:0] | Input | rx_serdes_clk | Ethertype field for global control processing. |
| ctl_rx_enable_pcp | Input | rx_serdes_clk | A value of 1 enables priority control packet processing. |
| ctl_rx_check_mcast_pcp | Input | rx_serdes_clk | A value of 1 enables priority control multicast destination address processing. |
| ctl_rx_check_ucast_pcp | Input | rx_serdes_clk | A value of 1 enables priority control unicast destination address processing. |
| ctl_rx_pause_da_mcast[47:0] | Input | rx_serdes_clk | Multicast destination address for pause processing. |

Table 2-11: Pause Interface – RX Path Signals (Cont'd)

| Name | Direction | Clock Domain | Description |
|-----------------------------|-----------|---------------|---|
| ctl_rx_check_sa_pcp | Input | rx_serdes_clk | A value of 1 enables priority control source address processing. |
| ctl_rx_check_etype_pcp | Input | rx_serdes_clk | A value of 1 enables priority control ethertype processing. |
| ctl_rx_etype_pcp[15:0] | Input | rx_serdes_clk | Ethertype field for priority control processing. |
| ctl_rx_check_opcode_pcp | Input | rx_serdes_clk | A value of 1 enables priority control opcode processing. |
| ctl_rx_opcode_min_pcp[15:0] | Input | rx_serdes_clk | Minimum priority control opcode value. |
| ctl_rx_opcode_max_pcp[15:0] | Input | rx_serdes_clk | Maximum priority control opcode value. |
| ctl_rx_enable_gpp | Input | rx_serdes_clk | A value of 1 enables global pause packet processing. |
| ctl_rx_check_mcast_gpp | Input | rx_serdes_clk | A value of 1 enables global pause multicast destination address processing. |
| ctl_rx_check_ucast_gpp | Input | rx_serdes_clk | A value of 1 enables global pause unicast destination address processing. |
| ctl_rx_check_sa_gpp | Input | rx_serdes_clk | A value of 1 enables global pause source address processing. |
| ctl_rx_check_etype_gpp | Input | rx_serdes_clk | A value of 1 enables global pause ethertype processing. |
| ctl_rx_etype_gpp[15:0] | Input | rx_serdes_clk | Ethertype field for global pause processing. |
| ctl_rx_check_opcode_gpp | Input | rx_serdes_clk | A value of 1 enables global pause opcode processing. |
| ctl_rx_opcode_gpp[15:0] | Input | rx_serdes_clk | Global pause opcode value. |
| ctl_rx_enable_ppp | Input | rx_serdes_clk | A value of 1 enables priority pause packet processing. |
| ctl_rx_check_mcast_ppp | Input | rx_serdes_clk | A value of 1 enables priority pause multicast destination address processing. |
| ctl_rx_check_ucast_ppp | Input | rx_serdes_clk | A value of 1 enables priority pause unicast destination address processing. |
| ctl_rx_check_sa_ppp | Input | rx_serdes_clk | A value of 1 enables priority pause source address processing. |
| ctl_rx_check_etype_ppp | Input | rx_serdes_clk | A value of 1 enables priority pause ethertype processing. |
| ctl_rx_etype_ppp[15:0] | Input | rx_serdes_clk | Ethertype field for priority pause processing. |
| ctl_rx_check_opcode_ppp | Input | rx_serdes_clk | A value of 1 enables priority pause opcode processing. |
| ctl_rx_opcode_ppp[15:0] | Input | rx_serdes_clk | Priority pause opcode value. |

Table 2-11: Pause Interface – RX Path Signals (Cont'd)

| Name | Direction | Clock Domain | Description |
|---------------------------------|-----------|---------------|---|
| stat_rx_pause_req[8:0] | Output | rx_serdes_clk | Pause request signal. When the RX receives a valid pause frame, it sets the corresponding bit of this bus to a 1 and keeps it at 1 until the pause packet has been processed. See Appendix C, Pause Processing Interface for pause interface details. |
| ctl_rx_pause_ack[8:0] | Input | rx_serdes_clk | Pause acknowledge signal. This bus is used to acknowledge the receipt of the pause frame from the user logic. See Appendix C, Pause Processing Interface for pause interface details. |
| ctl_rx_check_ack | Input | rx_serdes_clk | Wait for acknowledge. If this input is set to 1, the 40G/50G High Speed Ethernet Subsystem uses the ctl_rx_pause_ack[8:0] bus for pause processing. If this input is set to 0, ctl_rx_pause_ack[8:0] is not used. |
| ctl_rx_forward_control | Input | rx_serdes_clk | A value of 1 indicates that the 40G/50G High Speed Ethernet Subsystem forwards control packets to you. A value of 0 causes the 40G/50G High Speed Ethernet Subsystem to drop control packets. See Appendix C, Pause Processing Interface for control/pause packet processing. |
| stat_rx_pause_valid[8:0] | Output | rx_serdes_clk | This bus indicates that a pause packet was received and the associated quanta on the stat_rx_pause_quanta[8:0][15:0] bus is valid and must be used for pause processing. If an 802.3x Ethernet MAC Pause packet is received, bit[8] is set to 1. |
| stat_rx_pause_quanta[8:0][15:0] | Output | rx_serdes_clk | These nine buses indicate the quanta received for each of the eight priorities in priority-based pause operation and global pause operation. If an 802.3x Ethernet MAC Pause packet is received, the quanta are placed in value [8]. |

Table 2-12: Pause Interface – TX Path Signals

| Name | Direction | Clock Domain | Description |
|---------------------------------------|-----------|--------------|--|
| ctl_tx_pause_req[8:0] | Input | clk | If a bit of this bus is set to 1, the 40G/50G High Speed Ethernet Subsystem transmits a pause packet using the associated quanta value on the ctl_tx_pause_quanta[8:0][15:0] bus. If bit[8] is set to 1, a global pause packet is transmitted. All other bits cause a priority pause packet to be transmitted. |
| ctl_tx_pause_quanta[8:0][15:0] | Input | clk | These nine buses indicate the quanta to be transmitted for each of the eight priorities in priority-based pause operation and the global pause operation. The value for ctl_tx_pause_quanta[8] is used for global pause operation. All other values are used for priority pause operation. |
| ctl_tx_pause_refresh_timer[8:0][15:0] | Input | clk | These nine buses set the retransmission time of pause packets for each of the eight priorities in priority-based pause operation and the global pause operation. The value for ctl_tx_pause_refresh_timer[8] is used for global pause operation. All other values are used for priority pause operation. |
| ctl_tx_da_gpp[47:0] | Input | clk | Destination address for transmitting global pause packets. |
| ctl_tx_sa_gpp[47:0] | Input | clk | Source address for transmitting global pause packets. |
| ctl_tx_ethertype_gpp[15:0] | Input | clk | Ethertype for transmitting global pause packets. |
| ctl_tx_opcode_gpp[15:0] | Input | clk | Opcode for transmitting global pause packets. |
| ctl_tx_da_ppp[47:0] | Input | clk | Destination address for transmitting priority pause packets. |
| ctl_tx_sa_ppp[47:0] | Input | clk | Source address for transmitting priority pause packets. |
| ctl_tx_ethertype_ppp[15:0] | Input | clk | Ethertype for transmitting priority pause packets. |
| ctl_tx_opcode_ppp[15:0] | Input | clk | Opcode for transmitting priority pause packets. |

Table 2-12: Pause Interface – TX Path Signals (Cont'd)

| Name | Direction | Clock Domain | Description |
|--------------------------|-----------|--------------|---|
| ctl_tx_resend_pause | Input | clk | Re-transmit pending pause packets. When this input is sampled as 1, all pending pause packets are retransmitted as soon as possible (that is, after the current packet in flight is completed) and the retransmit counters are reset. This input should be pulsed to 1 for one cycle at a time. |
| stat_tx_pause_valid[8:0] | Output | clk | If a bit of this bus is set to 1, the 40G/50G High Speed Ethernet Subsystem has transmitted a pause packet. If bit[8] is set to 1, a global pause packet is transmitted. All other bits cause a priority pause packet to be transmitted. |

Auto-Negotiation (AN) and Link Training (LT)

The 40G/50G IP core supports Auto-Negotiation and Link Training.

A block diagram of the 40G/50G IP core with AN and LT is illustrated in [Figure 2-8](#).

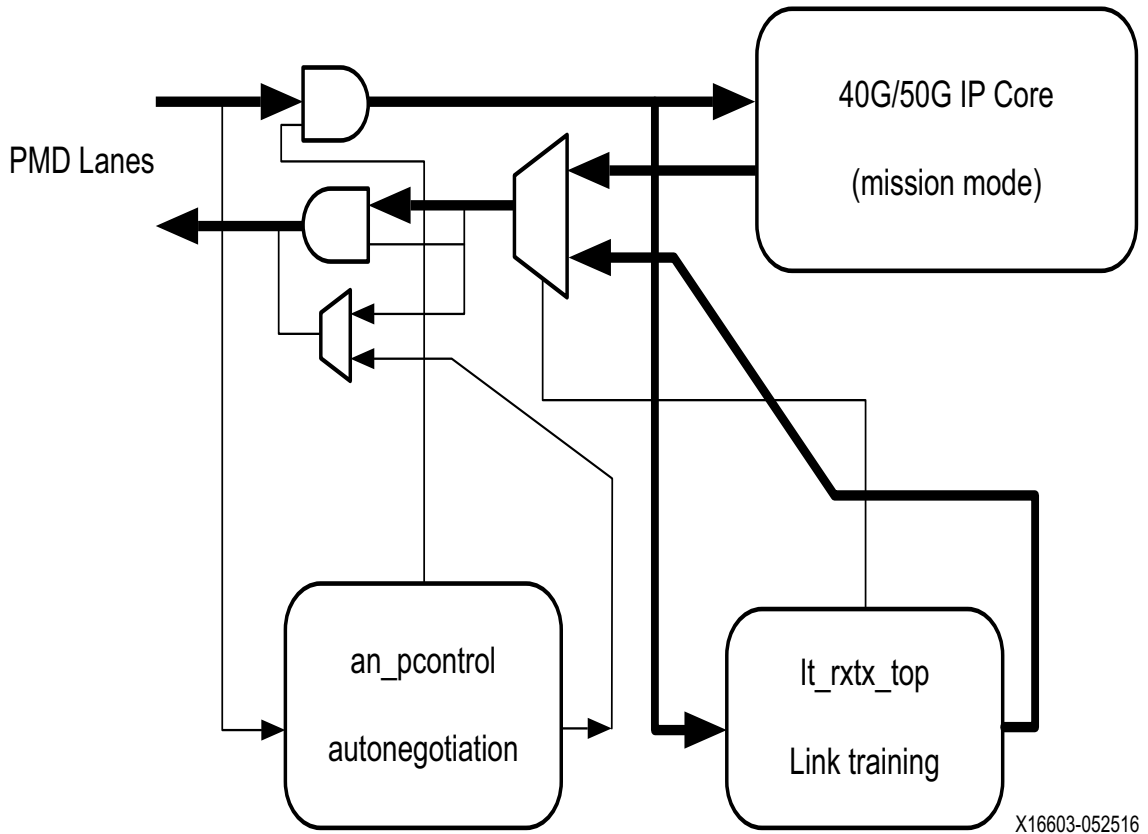


Figure 2-8: 40G/50G IP core with Auto-Negotiation and Link Training

The Auto-Negotiation function allows an Ethernet device to advertise the modes of operation it possesses to another device at the remote end of a Backplane Ethernet link and to detect corresponding operational modes the other device might be advertising. The objective of this Auto-Negotiation function is to provide the means to exchange information between two devices and to automatically configure them to take maximum advantage of their abilities. It has the additional objective of supporting a digital signal detect to ensure that the device is attached to a link partner rather than detecting a signal due to crosstalk. When Auto-Negotiation is complete, the ability is reported according to the available modes of operation.

Link Training (LT) is performed after AN if the LT function is supported by both ends of the link. Link training is typically required due to frequency-dependent losses that can occur as digital signals traverse the backplane. The primary function of the LT block included with this IP core is to provide register information and a training sequence over the backplane link which is then analyzed by a receiving circuit (part of the SerDes).

The other function of the LT block is to communicate training feedback from the receiver to the corresponding transmitter so that its pre-emphasis circuit (part of the SerDes) can be adjusted as required. The decision-making algorithm is not part of this IP core.

When AN and LT are complete, the datapath is switched to mission mode (see [Figure 2-8](#)).

Port List

The following additional signals are used for the auto-negotiation function. These signals are found at the `*wrapper.v` hierarchy.

Table 2-13: Auto-Negotiation Ports

| Port Name | Direction | Clock Domain | Description and Notes |
|------------------------|-----------|--------------|---|
| an_clk | Input | | Input Clock for the Auto-Negotiation circuit. The required frequency is indicated in the readme file for the release. |
| an_reset | Input | an_clk | Synchronous active-High reset corresponding to an_clk domain. |
| ctl_autoneg_enable | Input | an_clk | Enable signal for auto-negotiation. |
| ctl_autoneg_bypass | Input | an_clk | Input to disable auto-negotiation and bypass the auto-negotiation function. If this input is asserted, auto-negotiation is turned off, but the PCS is connected to the output to allow operation. |
| ctl_an_nonce_seed[7:0] | Input | an_clk | 8-bit seed to initialize the nonce field polynomial generator. Non-zero. The auto-negotiation will not function if this is zero. |

Table 2-13: Auto-Negotiation Ports (Cont'd)

| Port Name | Direction | Clock Domain | Description and Notes |
|---|-----------|--------------|--|
| ctl_an_pseudo_sel | Input | an_clk | Selects the polynomial generator for the bit 49 random bit generator. If this input is 1, then the polynomial is x^7+x^6+1 . If this input is zero, the polynomial is x^7+x^3+1 . |
| ctl_restart_negotiation | Input | an_clk | This input is used to trigger a restart of the auto-negotiation, regardless of what state the circuit is currently in. |
| ctl_an_local_fault | Input | an_clk | This input signal is used to set the local_fault bit of the transmit link codeword. |
| Signals Used for PAUSE Ability Advertising | | | |
| ctl_an_pause | Input | an_clk | This input signal is used to set the PAUSE bit, (C0), of the transmit link codeword. This signal might not be present if the core does not support pause. |
| ctl_an_asmdir | Input | an_clk | This input signal is used to set the ASMDIR bit, (C1), of the transmit link codeword. This signal might not be present if the core does not support pause. |
| Ability Signal Inputs | | | |
| ctl_an_ability_1000base_kx | Input | an_clk | These inputs identify the Ethernet protocol abilities that are advertised in the transmit link codeword to the link partner. A value of 1 indicates that the interface advertises that it supports the protocol. |
| ctl_an_ability_100gbase_cr10 | Input | an_clk | |
| ctl_an_ability_100gbase_cr4 | Input | an_clk | |
| ctl_an_ability_100gbase_kp4 | Input | an_clk | |
| ctl_an_ability_100gbase_kr4 | Input | an_clk | |
| ctl_an_ability_10gbase_kr | Input | an_clk | |
| ctl_an_ability_10gbase_kx4 | Input | an_clk | |
| ctl_an_ability_25gbase_krcr | Input | an_clk | |
| ctl_an_ability_25gbase_cr1 | Input | an_clk | |
| ctl_an_ability_25gbase_krcr_s | Input | an_clk | |
| ctl_an_ability_25gbase_kr1 | Input | an_clk | |
| ctl_an_ability_40gbase_cr4 | Input | an_clk | |
| ctl_an_ability_40gbase_kr4 | Input | an_clk | |
| ctl_an_ability_50gbase_cr2 | Input | an_clk | |
| ctl_an_ability_50gbase_kr2 | Input | an_clk | |
| ctl_an_fec_10g_request | Input | an_clk | Used to set the clause 74 FEC request bit in the transmit link codeword. This signal applies only to PMDs running at 10 Gb/s. It is ignored in PMDs running any other rate. |

Table 2-13: Auto-Negotiation Ports (Cont'd)

| Port Name | Direction | Clock Domain | Description and Notes |
|---------------------------------------|-----------|--------------|---|
| ctl_an_fec_ability_override | Input | an_clk | Used to set the clause 74 FEC ability bit in the transmit link codeword. If this input is set, the FEC ability bit in the transmit link codeword is cleared. This signal might not be present if the IP core does not support clause 74 FEC. |
| ctl_an_cl91_fec_ability | Input | an_clk | This bit is used to set clause 91 FEC ability. |
| ctl_an_cl91_fec_request | Input | an_clk | This bit is used to request clause 91 FEC. |
| ctl_an_fec_25g_rs_request | Input | an_clk | Used to set the RS-FEC request bit in the transmit link codeword. This signal applies only to PMDs running at 25Gb/s. It is ignored in PMDs running any other rate. |
| ctl_an_fec_25g_baser_request | Input | an_clk | Used to set the clause 74 FEC request bit in the transmit link codeword. This signal applies only to PMDs running at 25Gb/s. It is ignored in PMDs running any other rate. |
| stat_an_rxcdrhold | Output | an_clk | Used to set the rxcdrhold_in of the GT during auto-negotiation. |
| stat_an_link_cntl_1000base_kx[1:0] | Output | an_clk | Link Control outputs from the auto-negotiation controller for the various Ethernet protocols. Settings are as follows: <ul style="list-style-type: none"> • 00: DISABLE; PCS is disconnected; • 01: SCAN_FOR_CARRIER; RX is connected to PCS; • 11: ENABLE; PCS is connected for mission mode operation. • 10: not used |
| stat_an_link_cntl_100gbase_cr10[1:0] | Output | an_clk | |
| stat_an_link_cntl_100gbase_cr4[1:0] | Output | an_clk | |
| stat_an_link_cntl_100gbase_kp4[1:0] | Output | an_clk | |
| stat_an_link_cntl_100gbase_kr4[1:0] | Output | an_clk | |
| stat_an_link_cntl_10gbase_kr[1:0] | Output | an_clk | |
| stat_an_link_cntl_10gbase_kx4[1:0] | Output | an_clk | |
| stat_an_link_cntl_25gbase_krcr[1:0] | Output | an_clk | |
| stat_an_link_cntl_25gbase_cr1[1:0] | Output | an_clk | |
| stat_an_link_cntl_25gbase_krcr_s[1:0] | Output | an_clk | |
| stat_an_link_cntl_25gbase_kr1[1:0] | Output | an_clk | |
| stat_an_link_cntl_40gbase_cr4[1:0] | Output | an_clk | |
| stat_an_link_cntl_40gbase_kr4[1:0] | Output | an_clk | |
| stat_an_link_cntl_50gbase_cr2[1:0] | Output | an_clk | |
| stat_an_link_cntl_50gbase_kr2[1:0] | Output | an_clk | |
| stat_an_fec_enable | Output | an_clk | Used to enable the use of clause 74 FEC on the link. |
| stat_an_rs_fec_enable | Output | an_clk | Used to enable the use of clause 91 FEC on the link. |
| stat_an_tx_pause_enable | Output | an_clk | Used to enable station-to-station (global) pause packet generation in the transmit path to control data flow in the receive path. |

Table 2-13: Auto-Negotiation Ports (Cont'd)

| Port Name | Direction | Clock Domain | Description and Notes |
|-----------------------------------|-----------|--------------|---|
| stat_an_rx_pause_enable | Output | an_clk | Used to enable station-to-station (global) pause packet interpretation in the receive path, in order to control data flow from the transmitter. |
| stat_an_autoneg_complete | Output | an_clk | Indicates the auto-negotiation is complete and rx link status from the PCS has been received. |
| stat_an_parallel_detection_fault | Output | an_clk | Indicated a parallel detection fault during auto-negotiation. |
| stat_an_lp_ability_1000base_kx | Output | an_clk | These signals indicate the advertised protocol from the link partner. They all become valid when the output signal stat_an_lp_ability_valid is asserted. A value of 1 indicates that the protocol is advertised as supported by the link partner. |
| stat_an_lp_ability_100gbase_cr10 | Output | an_clk | |
| stat_an_lp_ability_100gbase_cr4 | Output | an_clk | |
| stat_an_lp_ability_100gbase_kp4 | Output | an_clk | |
| stat_an_lp_ability_100gbase_kr4 | Output | an_clk | |
| stat_an_lp_ability_10gbase_kr | Output | an_clk | |
| stat_an_lp_ability_10gbase_kx4 | Output | an_clk | |
| stat_an_lp_ability_25gbase_krcr | Output | an_clk | |
| stat_an_lp_ability_25gbase_krcr_s | Output | an_clk | |
| stat_an_lp_ability_40gbase_cr4 | Output | an_clk | |
| stat_an_lp_ability_40gbase_kr4 | Output | an_clk | |
| stat_an_lp_ability_25gbase_cr1 | Output | an_clk | |
| stat_an_lp_ability_25gbase_kr1 | Output | an_clk | Indicates the advertised protocol from the link partner. Becomes valid when the output signal stat_an_lp_extended_ability_valid is asserted. A value of 1 indicates that the protocol is advertised as supported by the link partner. |
| stat_an_lp_ability_50gbase_cr2 | Output | an_clk | Indicates the advertised protocol from the link partner. Becomes valid when the output signal stat_an_lp_extended_ability_valid is asserted. A value of 1 indicates that the protocol is advertised as supported by the link partner. |
| stat_an_lp_ability_50gbase_kr2 | Output | an_clk | Indicates the advertised protocol from the link partner. Becomes valid when the output signal stat_an_lp_extended_ability_valid is asserted. A value of 1 indicates that the protocol is advertised as supported by the link partner. |

Table 2-13: Auto-Negotiation Ports (Cont'd)

| Port Name | Direction | Clock Domain | Description and Notes |
|----------------------------------|-----------|--------------|--|
| stat_an_lp_pause | Output | an_clk | This signal indicates the advertised value of the PAUSE bit, (C0), in the receive link codeword from the link partner. It becomes valid when the output signal stat_an_lp_ability_valid is asserted. |
| stat_an_lp_asm_dir | Output | an_clk | This signal indicates the advertised value of the ASMDIR bit, (C1), in the receive link codeword from the link partner. It becomes valid when the output signal stat_an_lp_ability_valid is asserted. |
| stat_an_lp_fec_10g_ability | Output | an_clk | This signal indicates the advertised value of the clause 74 FEC ability bit in the receive link codeword on the corresponding 10 Gb/s PMD interface from the link partner. It becomes valid when the output signal stat_an_lp_ability_valid is asserted. |
| stat_an_lp_fec_10g_request | Output | an_clk | This signal indicates the advertised value of the clause 74 FEC Request bit in the receive link codeword on the corresponding 10 Gb/s PMD interface from the link partner. It becomes valid when the output signal stat_an_lp_ability_valid is asserted. |
| stat_an_lp_fec_25g_rs_request | Output | an_clk | This signal indicates the advertised value of the RS-FEC request bit in the receive link codeword on the corresponding 25Gb/s PMD interface from the link partner. It becomes valid when the output signal. |
| stat_an_lp_fec_25g_baser_request | Output | an_clk | This signal indicates the advertised value of the clause 74 FEC request bit in the receive link codeword on the corresponding 25Gb/s PMD interface from the link partner. It becomes valid when the output signal stat_an_lp_ability_valid is asserted. |
| stat_an_lp_autoneg_able | Output | an_clk | This output signal indicates that the link partner is able to perform auto-negotiation. It becomes valid when the output signal stat_an_lp_ability_valid is asserted. |
| stat_an_lp_ability_valid | Output | an_clk | This signal indicates when all of the link partner advertisements become valid. |
| an_loc_np_data[47:0] | Input | an_clk | Local Next Page codeword. This is the 48 bit codeword used if the loc_np input is set. In this data field, the bits NP, ACK, & T, bit positions 15, 14, 12, and 11, are not transferred as part of the next page codeword. These bits are generated in the AN IP. However, the Message Protocol bit, MP, in bit position 13, is transferred. |

Table 2-13: Auto-Negotiation Ports (Cont'd)

| Port Name | Direction | Clock Domain | Description and Notes |
|--------------------------------------|-----------|---------------|--|
| an_lp_np_data[47:0] | Output | an_clk | Link Partner Next Page Data. This 48-bit word is driven by the AN IP with the 48 bit next page codeword from the remote link partner. |
| ctl_an_loc_np | Input | an_clk | Local Next Page indicator. If this bit is 1, the AN IP transfers the next page word at input loc_np_data to the remote link partner. If this bit is 0, the AN IP does not initiate the next page protocol. If the link partner has next pages to send and the loc_np bit is clear, the AN IP transfers null message pages. |
| stat_fec_inc_cant_correct_count[3:0] | Output | rx_serdes_clk | Logical indication of uncorrectable errors. If an uncorrectable packet is encountered, this output signal cycles once. The signal is High for a minimum of 16 clocks and goes Low for a minimum of 16 clocks. There is one per lane. |
| stat_fec_inc_correct_count[3:0] | Output | rx_serdes_clk | Logical indication of correctable errors. If a correctable packet is encountered, this output signal cycles once. The signal is High for a minimum of 16 clocks and goes Low for a minimum of 16 clocks. There is one per lane. |
| stat_fec_lock_error[3:0] | Output | rx_serdes_clk | Logical indication of a failure to achieve a frame lock. The receiver scans the incoming data stream for about 10,000,000 bits, attempting all possible bit alignments for frame synchronization. After this time, this signal is asserted High and remains High until the receiver achieves a frame lock. There is one per lane. |
| stat_fec_rx_lock[3:0] | Output | rx_serdes_clk | Logical indication of a frame lock. The receiver asserts this signal High when it achieves a frame lock to the incoming bitstream. There is one per lane. |
| ctl_an_lp_np_ack | Input | an_clk | Link Partner Next Page Acknowledge. This is used to signal the AN IP that the next page data from the remote link partner at output pin lp_np_data has been read by the local host. When this signal goes High, the AN IP acknowledges reception of the next page codeword to the remote link partner and initiate transfer of the next codeword. During this time, the AN IP removes the lp_np signal until the new next page information is available. |

Table 2-13: Auto-Negotiation Ports (Cont'd)

| Port Name | Direction | Clock Domain | Description and Notes |
|--------------------------------------|-----------|--------------|--|
| stat_an_loc_np_ack | Output | an_clk | This signal is used to indicate to the local host that the local next page data, presented at input pin loc_np_data, has been taken. This signal pulses High for 1 clock period when the AN IP samples the next page data on input pin loc_np_data. When the local host detects this signal High, it must replace the 48 bit next page codeword at input pin loc_np_data with the next 48 bit codeword to be sent. If the local host has no more next pages to send, it must clear the loc_np input. |
| stat_an_lp_np | Output | an_clk | Link Partner Next Page. This signal is used to indicate that there is a valid 48 bit next page codeword from the remote link partner at output pin lp_np_data. This signal is driven Low when the lp_np_ack input signal is driven High, indicating that the local host has read the next page data. It remains Low until the next codeword becomes available on the lp_np_data output pin; then the lp_np output is driven High again. |
| stat_an_lp_ability_extended_fec[1:0] | Output | an_clk | This output indicates the extended FEC abilities as defined in Schedule 3. |
| stat_an_lp_extended_ability_valid | Output | an_clk | When this bit is 1, it indicates that the detected extended abilities are valid. |
| stat_an_lp_rf | Output | an_clk | This bit indicates link partner remote fault. |

Table 2-13: Auto-Negotiation Ports (Cont'd)

| Port Name | Direction | Clock Domain | Description and Notes |
|-----------------------------|-----------|--------------|---|
| stat_an_start_tx_disable | Output | an_clk | When ctl_autoneg_enable is High and ctl_autoneg_bypass is Low, this signal, stat_an_start_tx_disable, cycles High for 1 clock cycle at the very start of the TX_DISABLE phase of auto-negotiation. That is, when auto-negotiation enters the state TX_DISABLE, this output will cycle High for 1 clock period. It effectively signals the start of auto-negotiation. |
| stat_an_start_an_good_check | Output | an_clk | When ctl_autoneg_enable is High and ctl_autoneg_bypass is Low, this signal, stat_an_start_an_good_check, cycles High for 1 clock cycle at the very start of the AN_GOOD_CHECK phase of auto-negotiation. That is, when auto-negotiation enters the state AN_GOOD_CHECK, this output will cycle High for 1 clock period. It effectively signals the start of link training. However, if link training is not enabled, that is, if the input ctl_lt_training_enable is Low, the stat_an_start_an_good_check output effectively signals the start of mission-mode operation. |

The following additional signals are used for the link-training function. These signals are found at the `*wrapper.v` hierarchy.

Table 2-14: Link Training Ports

| Port Name | Direction | Clock Domain | Description and Notes |
|------------------------------|-----------|---------------|---|
| ctl_lt_training_enable | Input | tx_serdes_clk | Enables link training. When link training is disabled, all PCS lanes function in mission mode. |
| ctl_lt_restart_training | Input | tx_serdes_clk | This signal triggers a restart of link training regardless of the current state. |
| ctl_lt_rx_trained[4-1:0] | Input | tx_serdes_clk | This signal is asserted to indicate that the receiver FIR filter coefficients have all been set, and that the receiver portion of training is complete. |
| stat_lt_signal_detect[4-1:0] | Output | tx_serdes_clk | This signal indicates when the respective link training state machine has entered the SEND_DATA state, in which normal PCS operation can resume. |
| stat_lt_training[4-1:0] | Output | tx_serdes_clk | This signal indicates when the respective link training state machine is performing link training. |
| stat_lt_training_fail[4-1:0] | Output | tx_serdes_clk | This signal is asserted during link training if the corresponding link training state machine detects a time-out during the training period. |

Table 2-14: Link Training Ports (Cont'd)

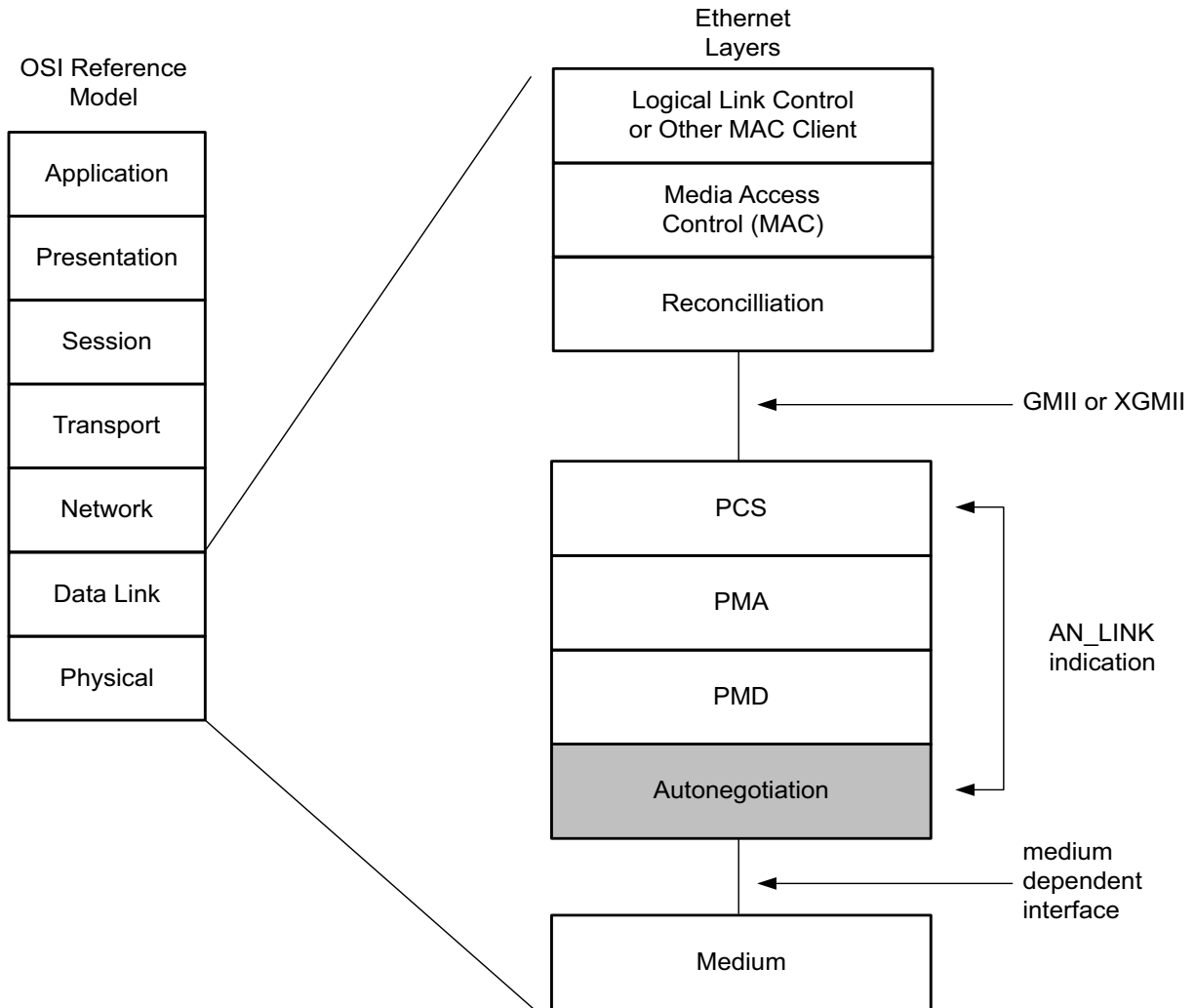
| Port Name | Direction | Clock Domain | Description and Notes |
|-----------------------------------|-----------|---------------|--|
| stat_lt_frame_lock[4-1:0] | Output | tx_serdes_clk | When link training has begun, these signals are asserted, for each PMD lane, when the corresponding link training receiver is able to establish a frame synchronization with the link partner. |
| stat_lt_preset_from_rx[4-1:0] | Output | rx_serdes_clk | This signal reflects the value of the preset control bit received in the control block from the link partner. |
| stat_lt_initialize_from_rx[4-1:0] | Output | rx_serdes_clk | This signal reflects the value of the initialize control bit received in the control block from the link partner. |
| stat_lt_k_p1_from_rx0[1:0] | Output | rx_serdes_clk | This 2-bit field indicates the update control bits for the k+1 coefficient, as received from the link partner in the control block |
| stat_lt_k0_from_rx0[1:0] | Output | rx_serdes_clk | This 2-bit field indicates the update control bits for the k0 coefficient, as received from the link partner in the control block. |
| stat_lt_k_m1_from_rx0[1:0] | Output | rx_serdes_clk | This 2-bit field indicates the update control bits for the k-1 coefficient, as received from the link partner in the control block. |
| stat_lt_stat_p1_from_rx0[1:0] | Output | rx_serdes_clk | This 2-bit field indicates the update status bits for the k+1 coefficient, as received from the link partner in the status block. |
| stat_lt_stat0_from_rx0[1:0] | Output | rx_serdes_clk | This 2-bit fields indicates the update status bits for the k0 coefficient, as received from the link partner in the status block. |
| stat_lt_stat_m1_from_rx0[1:0] | Output | rx_serdes_clk | This 2-bit field indicates the update status bits for the k-1 coefficient, as received from the link partner in the status block. |
| ctl_lt_pseudo_seed0[10:0] | Input | tx_serdes_clk | This 11- bit signal seeds the training pattern generator. The training pattern will not be correct if this seed is loaded with a value of zero. |
| ctl_lt_preset_to_tx[4-1:0] | Input | tx_serdes_clk | This signal is used to set the value of the preset bit that is transmitted to the link partner in the control block of the training frame. |
| ctl_lt_initialize_to_tx[4-1:0] | Input | tx_serdes_clk | This signal is used to set the value of the initialize bit that is transmitted to the link partner in the control block of the training frame. |
| ctl_lt_k_p1_to_tx0[1:0] | Input | tx_serdes_clk | This 2-bit field is used to set the value of the k+1 coefficient update field that is transmitted to the link partner in the control block of the training frame. |
| ctl_lt_k0_to_tx0[1:0] | Input | tx_serdes_clk | This 2-bit field is used to set the value of the k0 coefficient update field that is transmitted to the link partner in the control block of the training frame,. |

Table 2-14: Link Training Ports (Cont'd)

| Port Name | Direction | Clock Domain | Description and Notes |
|----------------------------|-----------|---------------|--|
| ctl_lt_k_m1_to_tx0[1:0] | Input | tx_serdes_clk | This 2-bit field is used to set the value of the k-1 coefficient update field that is transmitted to the link partner in the control block of the training frame. |
| ctl_lt_stat_p1_to_tx0[1:0] | Input | tx_serdes_clk | This 2-bit field is used to set the value of the k+1 coefficient update status that is transmitted to the link partner in the status block of the training frame. |
| ctl_lt_stat0_to_tx0[1:0] | Input | tx_serdes_clk | This 2-bit field is used to set the value of the k0 coefficient update status that is transmitted to the link partner in the status block of the training frame. |
| ctl_lt_stat_m1_to_tx0[1:0] | Input | tx_serdes_clk | This 2-bit field is used to set the value of the k-1 coefficient update status that is transmitted to the link partner in the status block of the training frame,. |
| stat_lt_rx_sof[4-1:0] | Output | rx_serdes_clk | This output is High for 1 RX SerDes clock cycle to indicate the start of the link training frame. |

Overview

Figure 2-9 as per IEEE P802.3 illustrates the position of the AN function in the OSI reference model.



X16606-032916

Figure 2-9: Auto-Negotiation in OSI Model

The Auto-Negotiation IP core implements the requirements as specified in Clause 73, IEEE Std 802.3-2012, including those amendments specified in IEEE P802.3by and Schedule 3 of the 25 GE Consortium.

The functions of the AN IP core are explicitly listed in clause 73, especially in Figure 73-11, Arbitration state diagram, of section 73.10.4, State diagrams.

During normal mission mode operation, with link control outputs set to (bin)11, the bit operating frequency of the SerDes input and output is typically 10.3125 or 25.78125 Gb/s. However, the Dual Manchester Encoding (DME) bit rate used on the lane during auto-negotiation is quite a bit different than the mission mode operation.

To accommodate this requirement, the AN IP core uses over-sampling and over-driving to match the 156.25 Mb/s auto-negotiation speed (DME clock frequency 312.5 MHz) with the mission mode 10.3125 or 25.78125 Gb/s physical lane speed.

Auto-Negotiation Description

autoneg_enable

When the `autoneg_enable` input signal is set to a 1, auto-negotiation begins automatically at power-up, or if the carrier signal is lost, or if the input `restart_negotiation` signal is cycled from a 0 to a 1. All of the 'ability' input signals as well as the two input signals `PAUSE` and `ASM_DIR` are tied Low or High to indicate the capability of the hardware. The `nonce_seed[7:0]` input must be set to a unique value for every instance of the auto-negotiator. The AN IP will not function if the `nonce_seed` is set to 0. This is important in order to guarantee that no deadlocks occur at power-up. If two link partners connected together attempt to auto-negotiate with their `nonce_seed[7:0]` inputs set to the same value, the auto-negotiation fails continuously. The `pseudo_sel` input is an arbitrary selection that is used to select the polynomial of the random bit generator in bit position 49 of the DME pages used during auto-negotiation. Any selection on this input is valid and will not result in any adverse behavior.

Link Control

When auto-negotiation has begun, then the various 'link control' signals are activated, depending on the disposition of the corresponding 'Ability' inputs for those links. Subsequently, the corresponding 'link status' signals are then monitored by the AN IP hardware for an indication of the state of the various links that can be connected. If particular links are unused, the corresponding link control outputs are unconnected, and the corresponding link-status inputs should be tied Low. During this time, the AN IP hardware sets up a communication link with the link partner and uses this link to negotiate the capabilities of the connection.

Autoneg Complete

When Auto-Negotiation is complete, the `autoneg_complete` output signal is asserted. In addition to this, the output signal `an_fec_enable` is asserted if the Forward Error Correction hardware is to be used; the output signal `tx_pause_en` is asserted if the transmitter hardware is allowed to generate PAUSE control packets, the output signal `rx_pause_en` is asserted if the receiver hardware is allowed to detect PAUSE control packets, and the output link control of the selected link is set to its mission mode value (bin)11.



IMPORTANT: *The autoneg complete signal is not asserted until rx_status is received from the PCS. That means that, where link training is included, the autoneg_complete output signal is not asserted until after link training has completed and rx_status is High.*

Link Training Description

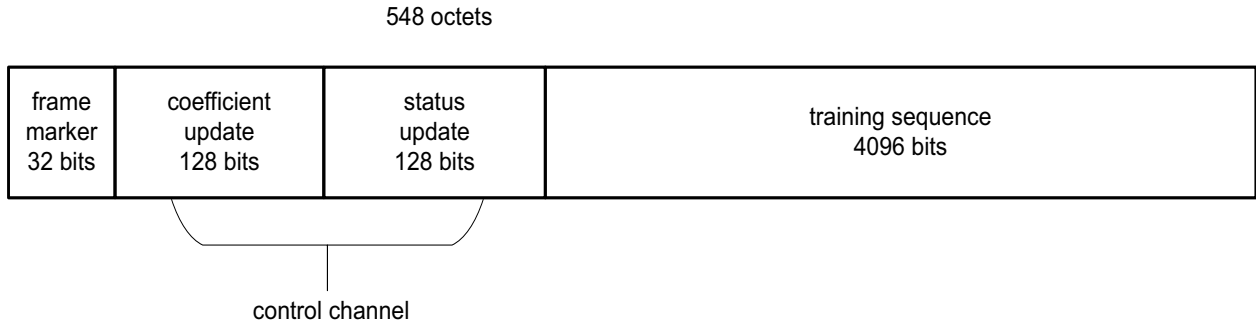
Overview

Link Training (LT) is performed after auto-negotiation (AN) converges to a backplane or copper technology. Technology selection can also be the result of a manual entry or parallel detection. Link training might be required due to frequency-dependent losses that can occur as digital signals traverse the backplane or a copper cable. The primary function of the LT IP core is to provide register information and a training sequence over the backplane link which is then analyzed by a receiving circuit that is not part of the IP core. The other function of the IP core is to communicate training feedback from the receiver to the corresponding transmitter so that its equalizer circuit (not part of the IP core) can be adjusted as required. The two circuits comprising the IP core are the receive Link Training block and the transmit Link Training block.

Note: The logic responsible for the adjustment of the transmitter pre-emphasis must be supplied external to this IP core.

Transmit

The LT transmit block constructs a 4,384-bit frame that contains a frame delimiter, control channel, and link training sequence. It is formatted as follows:



X16605-052516

Figure 2-10: Link Training Frame Structure

It is recommended that the control channel bits not be changed by the link training algorithm while the transmit state machine is in the process of transmitting them or they can be received incorrectly, possibly resulting in a DME error. This time will begin when `tx_SOF` is asserted and ends at least 288 bit times later, or approximately 30 nsec.

Note that although the coefficient and status contain 128 bit times at the line rate, the actual signaling rate for these two fields is reduced by a factor of 8. Therefore the DME clock rate is one quarter of the line rate.

Frame Marker

The frame marker consists of 16 consecutive 1s followed by 16 consecutive 0s. This pattern is not repeated in the remainder of the frame.

Coefficient and Status

Because the DME signaling rate for these two fields is reduced by a factor of 8, each coefficient and status transmission contain $128/8=16$ bits each numbered from 15:0. [Table 2-15](#) and [Table 2-16](#) define these bits in the order in which they are transmitted starting with bit 15 and ending with bit 0.

Table 2-15: Coefficient and Update Field Bit Definitions

| Bits | Name | Description |
|-------|-------------------------|---|
| 15:14 | Reserved | Transmitted as 0, ignored on reception. |
| 13 | Preset | 1 = Preset coefficients 0 = Normal operation |
| 12 | Initialize | 1 = Initialize coefficients 0 = Normal operation |
| 11:6 | Reserved | Transmitted as 0, ignored on reception. |
| 5:4 | Coefficient (+1) update | $\begin{matrix} 5 & 4 \\ 1 & 1 = \text{reserved} \\ 0 & 1 = \text{increment} \\ 1 & 0 = \text{decrement} \\ 0 & 0 = \text{hold} \end{matrix}$ |
| 3:2 | Coefficient (0) update | $\begin{matrix} 3 & 2 \\ 1 & 1 = \text{reserved} \\ 0 & 1 = \text{increment} \\ 1 & 0 = \text{decrement} \\ 0 & 0 = \text{hold} \end{matrix}$ |
| 1:0 | Coefficient (-1) update | $\begin{matrix} 1 & 0 \\ 1 & 1 = \text{reserved} \\ 0 & 1 = \text{increment} \\ 1 & 0 = \text{decrement} \\ 0 & 0 = \text{hold} \end{matrix}$ |

Table 2-16: Status Report Field Bit Definitions

| Bits | Name | Description |
|------|-------------------------|--|
| 15 | Receiver ready | <ul style="list-style-type: none"> • 1 = The local receiver has determined that training is complete and is prepared to receive data. • 0 = The local receiver is requesting that training continue. |
| 14:6 | Reserved | Transmitted as 0, ignored on reception. |
| 5:4 | Coefficient (+1) update | <u>5</u> <u>4</u> 1 1 = maximum 0 1 = minimum 1 0 = updated 0 0 = not_updated |
| 3:2 | Coefficient (0) update | <u>3</u> <u>2</u> 1 1 = maximum 0 1 = minimum 1 0 = updated 0 0 = not_updated |
| 1:0 | Coefficient (-1) update | <u>1</u> <u>0</u> 1 1 = maximum 0 1 = minimum 1 0 = updated 0 0 = not_updated |

The functions of each bit are defined in IEEE 802.3 Clause 72. Their purpose is to communicate the adjustments of the transmit equalizer during the process of link training. The corresponding signal names are defined in [Port Descriptions](#).

Training Sequence

The training sequence consists of a Pseudo Random Bit Sequence (PRBS) of 4,094 bits followed by two zeros, for a total of 4,096 bits. The PRBS is transmitted at the line rate of 10.3125 or 25.78125 Gb/s. The PRBS generator receives an 11-bit seed from an external source. Seed must be non-zero. Subsequent to the initial seed being loaded, the PRBS generator continues to run with no further intervention being required.

The PRBS generator itself is implemented with a circuit that corresponds to the following polynomial:

$$G(x) = 1 + x^9 + x^{11}$$

Receive

The receive block implements the frame alignment state diagram illustrated in IEEE 802.3 Clause 72 Figure 72-4.

Frame Lock State Machine

The frame lock state machine searches for the frame marker, consisting of 16 consecutive 1s followed by 16 consecutive 0s. This functionality is fully specified in IEEE 802.3 Clause 72 Figure 72-4. When frame lock has been achieved, the signal `frame_lock` is set to a value of TRUE.

Received Data

The receiver outputs the control channel with the bit definitions previously defined in [Table 2-15](#) and [Table 2-16](#) and signal names defined in [Port Descriptions](#).

If a `DME_error` has occurred during the reception of a particular DME frame, the control channel outputs are not updated but retain the value of the last received good DME frame and are updated when the next good DME frame is received.

Forward Error Correction (FEC)

The 40G/50G IP core has support for any one of three FEC modes of operation as defined in Schedule 3 of the 25G Consortium:

- No FEC
- Clause 74 FEC (shortened cyclic code (2112, 2080))
- Clause 91 FEC (Reed-Solomon (528,514)) Not supported in this version v1.1.

The FEC mode is communicated to the link partner during the auto-negotiation phase.

PCS Variant

A PCS-only variant of the 40G/50G Ethernet IP subsystem can be generated from the same wizard.

Features

- Designed to the Ethernet requirements for 50 Gb/s operation as defined in Schedule 3 of the 25G Ethernet Consortium [\[Ref 2\]](#)
- Designed to the Ethernet requirements for 40 Gb/s operation as defined in IEEE 802.3 Clause 82 [\[Ref 1\]](#)

- Includes complete PCS functions
- Standard XLGMII or LGMII system side interface
- Supports Base-R 64b/66b encoding and decoding
- Optional Base-KR FEC
- Optional Auto Negotiation and Link Training

Block Diagram

Figure 2-11 is a block diagram of the 50 Gb/s PCS-only variant.

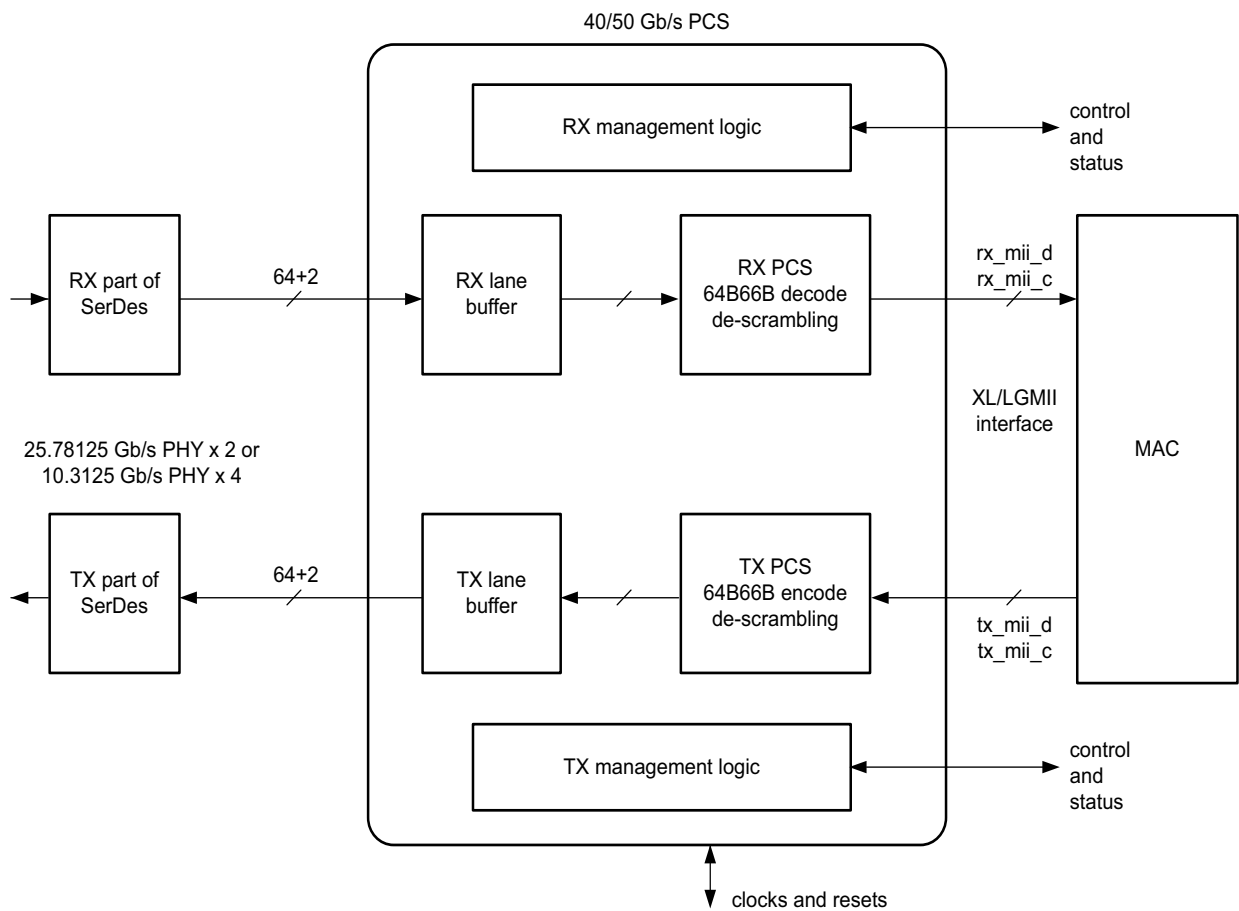


Figure 2-11: PCS Variant

Port List

The following table shows the 40G/50G PCS IP core ports. These are the ports when the PCS-only option is provided. There are no FCS functions and no AXI4-Stream-related ports.

The PCS does not contain the Pause and Flow Control ports. The system interface is LGMII instead of the AXI4-Stream.

These signals are found at the `*wrapper.v` hierarchy. Refer to [PCS Clocking](#) for clock domain definitions.

Table 2-17: PCS Variant Ports

| Name | Direction | Clock Domain | Description |
|-----------------------------------|-----------|---------------|--|
| Transceiver I/O | | | |
| rx_serdes_data_n0 | Input | rx_serdes_clk | Serial data from the line for lane 0; negative phase of the differential signal |
| rx_serdes_data_p0 | Input | rx_serdes_clk | Serial data from the line for lane 0; positive phase of the differential signal |
| tx_serdes_data_n0 | Output | tx_serdes_clk | Serial data to the line for lane 0; negative phase of the differential signal. |
| tx_serdes_data_p0 | Output | tx_serdes_clk | Serial data to the line for lane 0; positive phase of the differential signal. |
| rx_serdes_data_n1 | Input | rx_serdes_clk | Serial data from the line for lane 1; negative phase of the differential signal |
| rx_serdes_data_p1 | Input | rx_serdes_clk | Serial data from the line for lane 1; positive phase of the differential signal |
| tx_serdes_data_n1 | Output | tx_serdes_clk | Serial data to the line for lane 1; negative phase of the differential signal. |
| tx_serdes_data_p1 | Output | tx_serdes_clk | Serial data to the line for lane 1; positive phase of the differential signal. |
| GT_reset (ctl_gt_reset_all) | Input | async | Active-High reset for the transceiver startup FSM. Note that this signal also initiates the reset sequence for the entire IP core. |
| refclk_n0 | Input | | Differential reference clock input for the SerDes, negative phase. |
| refclk_p0 | Input | | Differential reference clock input for the SerDes, positive phase. |
| XL/LGMII Interface Signals | | | |
| rx_mii_d[127:0] | Output | rx_mii_clk | Receive XL/LGMII Data bus. |
| rx_mii_c[15:0] | Output | rx_mii_clk | Receive XL/LGMII Control bus. |
| rx_mii_clk | Input | | Receive XL/LGMII Clock input. |
| tx_mii_d[127:0] | Input | tx_mii_clk | Transmit XL/LGMII Data bus. |
| tx_mii_c[15:0] | Input | tx_mii_clk | XL/LGMII Control bus. |
| tx_mii_clk | Input | tx_mii_clk | Transmit XL/LGMII Clock input. |
| rx_mii_reset | Input | | Reset input for the rx XL/LGMII interface. |
| tx_mii_reset | Input | | Reset input for the tx XL/LGMII interface. |

Table 2-17: PCS Variant Ports (Cont'd)

| Name | Direction | Clock Domain | Description |
|---|-----------|---------------|---|
| rx_serdes_clk | Input | | Input clock signal used for clocking the core logic of the RX PCS. |
| tx_core_clk | Input | | Input clock signal used for clocking the core logic of the TX PCS. |
| rx_reset | Input | rx_serdes_clk | Reset associated with the rx_serdes_clk logic. Must be synchronous to rx_serdes_clk. |
| tx_reset | Input | tx_core_clk | Reset associated with the tx_core_clk logic. Must be synchronous to tx_core_clk. |
| LGMII Interface – Control/Status Signals | | | |
| ctl_rx_vl_length_minus1[15:0] | Input | static | Number of words in between PCS Lane markers minus one for RX. Default value, as defined in the IEEE 802.3, should be set to 16,383. This input should only be changed while the corresponding reset input is asserted. |
| ctl_tx_vl_length_minus1[15:0] | Input | static | Number of words in between PCS Lane markers minus one for TX. Default value, as defined in the IEEE 802.3, should be set to 16,383. This input should only be changed while the corresponding reset input is asserted. |
| ctl_rx_vl_marker_id0[63:0] | Input | static | PCS Lane marker for RX PCS lane0. For IEEE 802.3 default values, see RX and TX PCS Lane Marker Values. This input should only be changed while the corresponding reset input is asserted. |
| ctl_rx_vl_marker_id1[63:0] | Input | static | PCS Lane marker for RX PCS lane1. |
| ctl_rx_vl_marker_id2[63:0] | Input | static | PCS Lane marker for RX PCS lane2. |
| ctl_rx_vl_marker_id3[63:0] | Input | static | PCS Lane marker for RX PCS lane3. |
| ctl_tx_vl_marker_id0[63:0] | Input | static | PCS Lane marker for TX PCS lane0. For IEEE 802.3 default values, see RX and TX PCS Lane Marker Values. This input should only be changed while the corresponding reset input is asserted. |
| ctl_tx_vl_marker_id1[63:0] | Input | static | PCS Lane marker for TX PCS lane1. |
| ctl_tx_vl_marker_id2[63:0] | Input | static | PCS Lane marker for TX PCS lane2. |
| ctl_tx_vl_marker_id3[63:0] | Input | static | PCS Lane marker for TX PCS lane3. |
| ctl_rx_test_pattern | Input | rx_mii_clk | Test pattern enable for the RX core to receive scrambled idle pattern. Takes third precedence. |

Table 2-17: PCS Variant Ports (Cont'd)

| Name | Direction | Clock Domain | Description |
|-------------------------|-----------|---------------|---|
| ctl_tx_test_pattern | Input | tx_core_clk | Scrambled idle Test pattern generation enable for the TX core. A value of 1 enables test mode. Takes third precedence. |
| stat_rx_fifo_error | Output | rx_mii_clk | Receive clock compensation FIFO error indicator. A value of 1 indicates the clock compensation FIFO under or overflowed. This condition only occurs if the PPM difference between the recovered clock and the local reference clock is greater than ± 200 ppm. If this output is sampled as a 1 in any clock cycle, the corresponding port must be reset to resume proper operation. |
| stat_rx_local_fault | Output | rx_serdes_clk | A value of 1 indicates the receive decoder state machine is in the RX_INIT state. This output is level sensitive. |
| stat_rx_hi_ber | Output | rx_serdes_clk | High Bit Error Rate (BER) indicator. When set to 1, the BER is too high as defined by the IEEE 802.3. Corresponds to MDIO register bit 3.32.1 as defined in Clause 82.3. This output is level sensitive. |
| stat_rx_block_lock[3:0] | Output | rx_serdes_clk | Block lock status for each PCS lane. A value of 1 indicates the corresponding lane has achieved a block lock as defined in Clause 82. Corresponds to MDIO register bit 3.50.7:0 and 3.51.11:0 as defined in Clause 82.3. This output is level sensitive. |
| stat_rx_error[7:0] | Output | rx_serdes_clk | Test pattern mismatch increment. A non-zero value in any cycle indicates how many mismatches occurred for the test pattern in the RX core. This output is only active when <code>ctl_rx_test_pattern</code> is set to a 1. This output can be used to generate MDIO register 3.43.15:0 as defined in Clause 82.3. This output is pulsed for one clock cycle. |
| stat_rx_error_valid | Output | rx_serdes_clk | Increment valid indicator. If this signal is a 1 in any clock cycle, the value of <code>stat_rx_error</code> is valid. |

Table 2-17: PCS Variant Ports (Cont'd)

| Name | Direction | Clock Domain | Description |
|----------------------------|-----------|---------------|--|
| stat_rx_bad_code[1:0] | Output | rx_serdes_clk | Increment for 64B/66B code violations. This signal indicates the number of 64b/66b words received with an invalid block or if a wrong 64b/66b block sequence was detected. This output can be used to generate MDIO register 3.33:7:0 as defined in Clause 82.3. |
| stat_rx_bad_code_valid | Output | rx_serdes_clk | Increment valid indicator. If this signal is a 1 in any clock cycle, the value of stat_rx_bad_code is valid. |
| stat_rx_framing_err_0[3:0] | Output | rx_serdes_clk | Increment value for number of sync header errors detected for PCS lane 0. The value of this bus is only valid in the same cycle that stat_rx_framing_err_valid_0 is a 1. |
| stat_rx_framing_err_1[3:0] | Output | rx_serdes_clk | Increment value for number of sync header errors detected for PCS lane 1. |
| stat_rx_framing_err_2[3:0] | Output | rx_serdes_clk | Increment value for number of sync header errors detected for PCS lane 2. |
| stat_rx_framing_err_3[3:0] | Output | rx_serdes_clk | Increment value for number of sync header errors detected for PCS lane 3. |
| stat_rx_valid_0 | Output | rx_serdes_clk | Increment valid indicator for PCS lane 0. If this signal is a 1 in any clock cycle, the value of stat_rx_framing_err_0 is valid. |
| stat_rx_valid_1 | Output | rx_serdes_clk | Increment valid indicator for PCS lane 1. |
| stat_rx_valid_2 | Output | rx_serdes_clk | Increment valid indicator for PCS lane 2. |
| stat_rx_valid_3 | Output | rx_serdes_clk | Increment valid indicator for PCS lane 3. |
| stat_rx_aligned | Output | rx_serdes_clk | All PCS Lanes Aligned/Deskewed. This signal indicates whether or not all PCS lanes are aligned and deskewed. A value of 1 indicates all PCS lanes are aligned and deskewed. When this signal is a 1, the RX path is aligned and can receive packet data. When this signal is 0, a local fault condition exists. Also corresponds to MDIO register bit 3.50.12 as defined in Clause 82.3. This output is level sensitive. |
| stat_rx_aligned_err | Output | rx_serdes_clk | Loss of Lane Alignment/Deskew. This signal indicates an error occurred during PCS lane alignment or virtual lane alignment was lost. A value of 1 indicates an error occurred. This output is level sensitive. |

Table 2-17: PCS Variant Ports (Cont'd)

| Name | Direction | Clock Domain | Description |
|--------------------------|-----------|---------------|--|
| stat_rx_misaligned | Output | rx_serdes_clk | Alignment Error. This signal indicates that the lane aligner did not receive the expected PCS lane marker across all lanes. This signal is not asserted until the PCS lane marker has been received at least once across all lanes. This output is pulsed for one clock cycle to indicate an error condition. |
| stat_rx_status | Output | rx_serdes_clk | PCS status. A value of 1 indicates the PCS is aligned and not in hi_ber state. Corresponds to MDIO register bit 3.32.12 as defined in Clause 82.3. This output is level sensitive. |
| stat_rx_vl_demuxed[3:0] | Output | rx_serdes_clk | PCS Lane Marker found. If a signal of this bus is sampled as 1, it indicates that the receiver has properly de-muxed that PCS lane. This output is level sensitive. |
| stat_tx_local_fault | Output | tx_core_clk | A value of 1 indicates the transmit encoder state machine is in the TX_INIT state. This output is level sensitive. |
| stat_tx_fifo_error | Output | tx_mii_clk | Transmit clock compensation FIFO error indicator. A value of 1 indicates the clock compensation FIFO under or overflowed. This condition only occurs if the PPM difference between the transmitter clock and the local reference clock is greater than ± 200 ppm. If this output is sampled as a 1 in any clock cycle, the corresponding port must be reset to resume proper operation. |
| stat_rx_vl_number_0[1:0] | Output | rx_serdes_clk | The value of this bus indicates which physical lane appears on PCS lane 0. This bus is only valid when the corresponding bit of stat_rx_vl_synced[PCS_LANES-1:0] is a 1. These outputs are level sensitive. |
| stat_rx_vl_number_1[1:0] | Output | rx_serdes_clk | The value of this bus indicates which physical lane appears on PCS lane 1. |
| stat_rx_vl_number_2[1:0] | Output | rx_serdes_clk | The value of this bus indicates which physical lane appears on PCS lane 2. |
| stat_rx_vl_number_3[1:0] | Output | rx_serdes_clk | The value of this bus indicates which physical lane appears on PCS lane 3. |

Table 2-17: PCS Variant Ports (Cont'd)

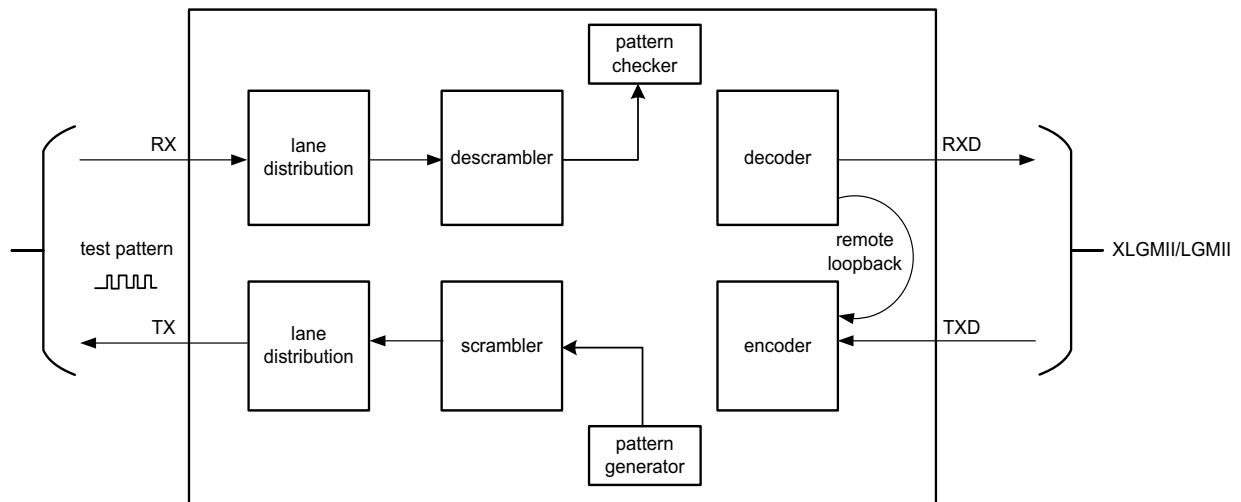
| Name | Direction | Clock Domain | Description |
|----------------------------|-----------|---------------|--|
| stat_rx_bip_err_0 | Output | rx_serdes_clk | BIP8 error indicator for PCS lane 0. A non-zero value indicates the BIP8 signature was in error. A non-zero value is pulsed for one clock cycle. This output is pulsed for one clock cycle to indicate an error condition. |
| stat_rx_bip_err_1 | Output | rx_serdes_clk | BIP8 error indicator for PCS lane 2. |
| stat_rx_bip_err_2 | Output | rx_serdes_clk | BIP8 error indicator for PCS lane 2. |
| stat_rx_bip_err_3 | Output | rx_serdes_clk | BIP8 error indicator for PCS lane 3. |
| stat_rx_synced[3:0] | Output | rx_serdes_clk | Word Boundary Synchronized. These signals indicate whether a PCS lane is word boundary synchronized. A value of 1 indicates the corresponding PCS lane has achieved word boundary synchronization and it has received a PCS lane marker. Corresponds to MDIO register bit 3.52.7:0 and 3.53.11:0 as defined in Clause 82.3. This output is level sensitive. |
| stat_rx_synced_err[3:0] | Output | rx_serdes_clk | Word Boundary Synchronization Error. These signals indicate whether an error occurred during word boundary synchronization in the respective PCS lane. A value of 1 indicates the corresponding PCS lane lost word boundary synchronization due to sync header framing bits errors or that a PCS lane marker was never received. This output is level sensitive. |
| stat_rx_mf_len_err[3:0] | Output | rx_serdes_clk | Virtual Lane Marker Length Error. These signals indicate whether a PCS Lane Marker length mismatch occurred in the respective lane (that is, PCS Lane Markers were received not every <code>ctl_rx_vl_length_minus1</code> words apart). A value of 1 indicates the corresponding lane is receiving PCS Lane Markers at wrong intervals. This output is pulsed for one clock cycle to indicate the error condition. |
| stat_rx_mf_repeat_err[3:0] | Output | rx_serdes_clk | PCS Lane Marker Consecutive Error. These signals indicate whether four consecutive PCS Lane Marker errors occurred in the respective lane. A value of 1 indicates an error in the corresponding lane. This output is pulsed for one clock cycle to indicate the error condition. |

Table 2-17: PCS Variant Ports (Cont'd)

| Name | Direction | Clock Domain | Description |
|---|-----------|---------------|---|
| stat_rx_mf_err | Output | rx_serdes_clk | PCS Lane Marker Word Error. These signals indicate that an incorrectly formed PCS Lane Marker Word was detected in the respective lane. A value of 1 indicates an error occurred. This output is pulsed for one clock cycle to indicate the error condition. |
| Miscellaneous Status/Control Signals | | | |
| dclk | Input | rx_serdes_clk | Dynamic reconfiguration port (DRP) clock input. The required frequency is indicated on the readme file for the release. |
| gt_loopback_in[12 6:0] | Input | async | GT loopback input signal for each transceiver. Refer to the GT user guide. 6-bit width for the 50G single core, 12-bit width for 40G single core/ 50G two cores. |

Testability

In addition to the local loopback available at the PMA, the 40G/50G PCS provides test pattern and remote loopback functions as illustrated in Figure 2-12.



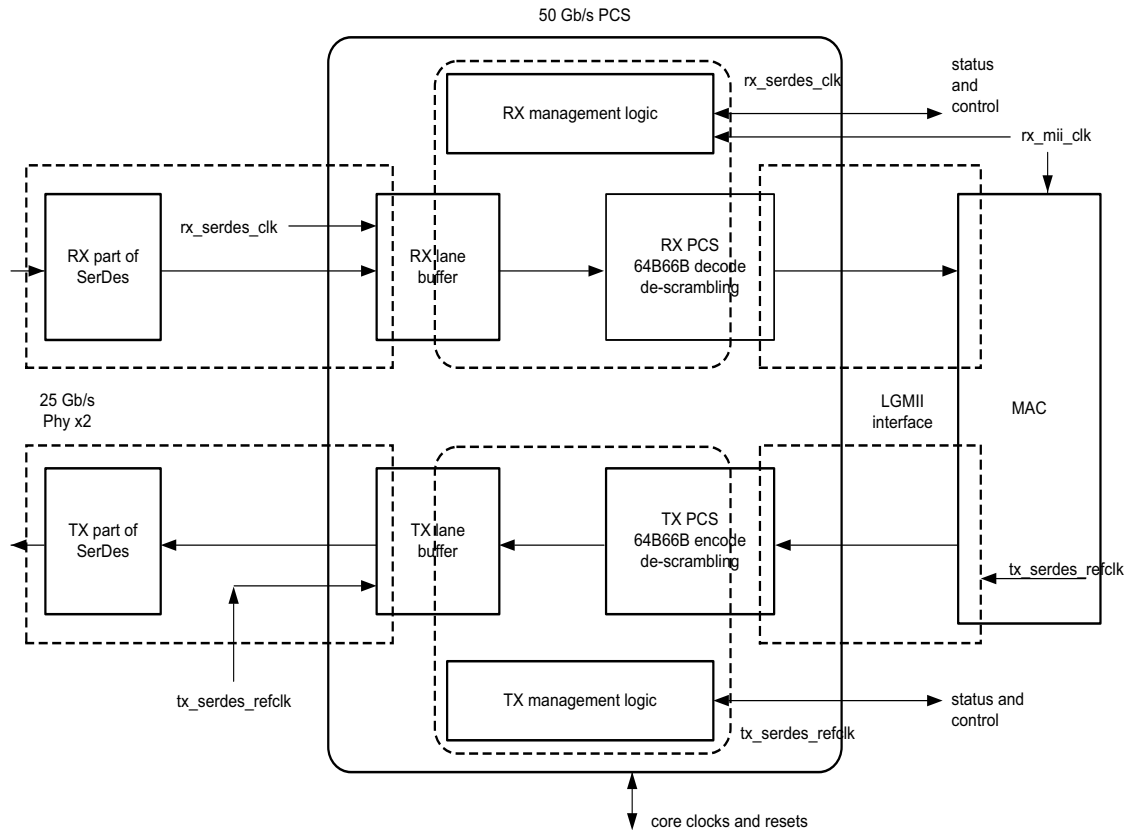
X16604-060716

Figure 2-12: PCS Loopback

When enabled, the transmitted test pattern is continuous idle characters, which are then scrambled.

PCS Clocking

The 40G/50G High Speed Ethernet Subsystem PCS uses separate RX and TX clock domains with RX and TX comprised of three clock domains each as illustrated in Figure 2-13.



X16601-033016

Figure 2-13: PCS Clocking

XLGMII/LGMII Clocks

These clocks drive logic for the XL/LGMII interfaces. The required clock frequency is determined by the Media Independent Interface (MII) bus width and the data rate. For example, for an LGMII interface (50 Gb/s) and a data bus width of 128 bits, the required clock frequency is $50e9/128 = 390.625$ MHz. For 40G the formula is $40e9/128 = 312.5$ MHz. This clock frequency is determined according to the IEEE Standard for Ethernet (IEEE Std 802.3-2012) specification.

SerDes Clocks

RX

Each SerDes lane has its own recovered clock. This clock is used for all of the logic for that SerDes lane. The 40G/50G High Speed Ethernet Subsystem synchronizes the received data from all of the SerDes to the RX core clock domains. There is one clock per SerDes lane. The SerDes clock frequency is equal to the data rate divided by the SerDes width. For example, at a data rate of 25.78125 Gb/s per lane and a 66-bit SerDes, the clock frequency is $25.78125 \times 10^9 / 66 = 390.625$ MHz.

TX

The TX SerDes domain is associated with the TX lane logic. Both TX transceivers must be clocked with the same frequency. The frequency is calculated in the same way as documented in [RX](#).

AXI4-Lite Register Space

The status and control signals of this Ethernet IP core can be optionally accessed by means of an AXI interface instead of the broadside bus. Detailed descriptions for each signal in the AXI register are found in the [Port List](#).

AXI Ports

[Table 2-18](#) describes the port list for the AXI processor interface.

Table 2-18: AXI Ports

| Signal | Direction | Description |
|--------------------|-----------|---|
| s_axi_aclk | In | AXI4-Lite clock. Range between 10 MHz and 300 MHz |
| s_axi_aresetn | In | Asynchronous active-Low reset |
| s_axi_awaddr[31:0] | In | Write address Bus |
| s_axi_awvalid | In | Write address valid |
| s_axi_awready | Out | Write address acknowledge |
| s_axi_wdata[31:0] | In | Write data bus |
| s_axi_wstrb[3:0] | In | Strobe signal for the data bus byte lane |
| s_axi_wvalid | Out | Write data valid |
| s_axi_wready | Out | Write data acknowledge |
| s_axi_bresp[1:0] | Out | Write transaction response |
| s_axi_bvalid | Out | Write response valid |

Table 2-18: AXI Ports (Cont'd)

| Signal | Direction | Description |
|--------------------|-----------|---|
| s_axi_bready | In | Write response acknowledge |
| s_axi_araddr[31:0] | In | Read address bus |
| s_axi_arvalid | In | Read address valid |
| s_axi_arready | Out | Read address acknowledge |
| s_axi_rdata[31:0] | Out | Read data output |
| s_axi_rresp[1:0] | Out | Read data response |
| s_axi_rvalid | Out | Read data/response valid |
| s_axi_rready | In | Read data acknowledge |
| pm_tick | In | Top level signal to read statistics counters; requires MODE_REG[30] to be set to 0. |

Additional information for the operation of the AXI4 bus is found in "Xilinx AXI Memory-Mapped Protocol Version 1.8".

As noted previously, the top-level signal `pm_tick` can be used to read statistics counters instead of the configuration register `TICK_REG`. In this case, configuration register `MODE_REG` bit 30 should be set to 0. If set to 1, `tick_reg` is used to read the statistics counters.

Base Pages

The Ethernet register map is divided into three sections as follows:

Table 2-19: Register Map

| Address Base | Address Space Name |
|--------------|----------------------------|
| 0x0000 | IP Configuration Registers |
| 0x0400 | Status Registers |
| 0x0500 | Statistics Counters |

All registers are 32 bits in size, and aligned on 32-bit addressing. The registers are designed such that the full 32b register is read/written (Byte write enables are ignored). In the below register space maps, any holes in the address space should be considered RESERVED and can cause an AXI-Ctl interface IP core to respond with an error if accessed.

When the AXI interface counters are selected, a "tick" register (`TICK_REG`) write/read is used to capture the statistics from the core clock domain on to the AXI clock domain, at the same time clearing the counters. After the "tick" is issued, the counters contain their updated value and can be read multiple times without destruction of this data.

The register reset signal is `s_axi_aresetn`, which is active-Low. This reset forces all registers to their default values as indicated in these tables.

Configuration Registers

The configuration space provides software with the ability to configure the IP core for various use cases. Certain features are optional (such as Auto-Negotiation, Link Training, and Flow Control), in which case the applicable registers are considered RESERVED.

For the programmed configuration to take effect, it is necessary to issue `tx_reset` and `rx_reset`, which are active-High.

Table 2-20: IP Configuration Registers

| Hex Address | Name/Link to Description | Note |
|-------------|--|-------------------------|
| 0x0000 | GT_RESET_REG: 0000 | . |
| 0x0004 | RESET_REG: 0004 | . |
| 0x0008 | MODE_REG: 0008 | . |
| 0x000C | CONFIGURATION_TX_REG1: 000C | . |
| 0x0014 | CONFIGURATION_RX_REG1: 0014 | . |
| 0x0018 | CONFIGURATION_RX_MTU: 0018 | Only in MAC+PCS variant |
| 0x001C | CONFIGURATION_VL_LENGTH_REG: 001C | |
| 0x0020 | TICK_REG: 0020 | |
| 0x0024 | CONFIGURATION_REVISION_REG: 0024 | |
| 0x0040 | CONFIGURATION_TX_FLOW_CONTROL_REG1: 0040 | Only in MAC+PCS variant |
| 0x0044 | CONFIGURATION_TX_FLOW_CONTROL_REFRESH_REG1: 0044 | Only in MAC+PCS variant |
| 0x0048 | CONFIGURATION_TX_FLOW_CONTROL_REFRESH_REG2: 0048 | Only in MAC+PCS variant |
| 0x004C | CONFIGURATION_TX_FLOW_CONTROL_REFRESH_REG3: 004C | Only in MAC+PCS variant |
| 0x0050 | CONFIGURATION_TX_FLOW_CONTROL_REFRESH_REG4: 0050 | Only in MAC+PCS variant |
| 0x0054 | CONFIGURATION_TX_FLOW_CONTROL_REFRESH_REG5: 0054 | Only in MAC+PCS variant |
| 0x0058 | CONFIGURATION_TX_FLOW_CONTROL_QUANTA_REG1: 0058 | Only in MAC+PCS variant |
| 0x005C | CONFIGURATION_TX_FLOW_CONTROL_QUANTA_REG2: 005C | Only in MAC+PCS variant |
| 0x0060 | CONFIGURATION_TX_FLOW_CONTROL_QUANTA_REG3: 0060 | Only in MAC+PCS variant |
| 0x0064 | CONFIGURATION_TX_FLOW_CONTROL_QUANTA_REG4: 0064 | Only in MAC+PCS variant |
| 0x0068 | CONFIGURATION_TX_FLOW_CONTROL_QUANTA_REG5: 0068 | Only in MAC+PCS variant |
| 0x006C | CONFIGURATION_TX_FLOW_CONTROL_PPP_ETYPE_OP_REG: 006C | Only in MAC+PCS variant |
| 0x0070 | CONFIGURATION_TX_FLOW_CONTROL_GPP_ETYPE_OP_REG: 0070 | Only in MAC+PCS variant |
| 0x0074 | CONFIGURATION_TX_FLOW_CONTROL_GPP_DA_REG_LSB: 0074 | Only in MAC+PCS variant |
| 0x0078 | CONFIGURATION_TX_FLOW_CONTROL_GPP_DA_REG_MSB: 0078 | Only in MAC+PCS variant |
| 0x007C | CONFIGURATION_TX_FLOW_CONTROL_GPP_SA_REG_LSB: 007C | Only in MAC+PCS variant |
| 0x0080 | CONFIGURATION_TX_FLOW_CONTROL_GPP_SA_REG_MSB: 0080 | Only in MAC+PCS variant |
| 0x0084 | CONFIGURATION_TX_FLOW_CONTROL_PPP_DA_REG_LSB: 0084 | Only in MAC+PCS variant |
| 0x0088 | CONFIGURATION_TX_FLOW_CONTROL_PPP_DA_REG_MSB: 0088 | Only in MAC+PCS variant |

Table 2-20: IP Configuration Registers (Cont'd)

| Hex Address | Name/Link to Description | Note |
|-------------|--|-------------------------|
| 0x008C | CONFIGURATION_TX_FLOW_CONTROL_PPP_SA_REG_LSB: 008C | Only in MAC+PCS variant |
| 0x0090 | CONFIGURATION_TX_FLOW_CONTROL_PPP_SA_REG_MSB: 0090 | Only in MAC+PCS variant |
| 0x0094 | CONFIGURATION_RX_FLOW_CONTROL_REG1: 0094 | Only in MAC+PCS variant |
| 0x0098 | CONFIGURATION_RX_FLOW_CONTROL_REG2: 0098 | Only in MAC+PCS variant |
| 0x009C | CONFIGURATION_RX_FLOW_CONTROL_PPP_ETYPE_OP_REG: 009C | Only in MAC+PCS variant |
| 0x00A0 | CONFIGURATION_RX_FLOW_CONTROL_GPP_ETYPE_OP_REG: 00A0 | Only in MAC+PCS variant |
| 0x00A4 | CONFIGURATION_RX_FLOW_CONTROL_GCP_PCP_TYPE_REG: 00A4 | Only in MAC+PCS variant |
| 0x00A8 | CONFIGURATION_RX_FLOW_CONTROL_PCP_OP_REG: 00A8 | Only in MAC+PCS variant |
| 0x00AC | CONFIGURATION_RX_FLOW_CONTROL_GCP_OP_REG: 00AC | Only in MAC+PCS variant |
| 0x00B0 | CONFIGURATION_RX_FLOW_CONTROL_DA_REG1_LSB: 00B0 | Only in MAC+PCS variant |
| 0x00B4 | CONFIGURATION_RX_FLOW_CONTROL_DA_REG1_MSB: 00B4 | Only in MAC+PCS variant |
| 0x00B8 | CONFIGURATION_RX_FLOW_CONTROL_DA_REG2_LSB: 00B8 | Only in MAC+PCS variant |
| 0x00BC | CONFIGURATION_RX_FLOW_CONTROL_DA_REG2_MSB: 00BC | Only in MAC+PCS variant |
| 0x00C0 | CONFIGURATION_RX_FLOW_CONTROL_SA_REG1_LSB: 00C0 | Only in MAC+PCS variant |
| 0x00C4 | CONFIGURATION_RX_FLOW_CONTROL_SA_REG1_MSB: 00C4 | Only in MAC+PCS variant |
| 0x00D4 | CONFIGURATION_FEC_REG: 00D4 | . |
| 0x00E0 | CONFIGURATION_AN_CONTROL_REG1: 00E0 | . |
| 0x00E4 | CONFIGURATION_AN_CONTROL_REG2: 00E4 | . |
| 0x00F8 | CONFIGURATION_AN_ABILITY: 00F8 | . |
| 0x0100 | CONFIGURATION_LT_CONTROL_REG1: 0100 | . |
| 0x0104 | CONFIGURATION_LT_TRAINED_REG: 0104 | . |
| 0x0108 | CONFIGURATION_LT_PRESET_REG: 0108 | . |
| 0x010C | CONFIGURATION_LT_INIT_REG: 010C | . |
| 0x0110 | CONFIGURATION_LT_SEED_REG0: 0110 | . |
| 0x0114 | CONFIGURATION_LT_SEED_REG1: 0114 | . |
| 0x0130 | CONFIGURATION_LT_COEFFICIENT_REG0: 0130 | . |
| 0x0134 | CONFIGURATION_LT_COEFFICIENT_REG1: 0134 | . |

Status Registers

The status registers provide an indication of the health of the system. These registers are read-only and a read operation clears the register.

Table 2-21: Status Registers

| Hex Address | Name/Link to Description | Notes |
|-------------|---|-------------------------|
| 0x0400 | STAT_TX_STATUS_REG1: 0400 | . |
| 0x0404 | STAT_RX_STATUS_REG1: 0404 | . |
| 0x0408 | STAT_STATUS_REG1: 0408 | . |
| 0x040C | STAT_RX_BLOCK_LOCK_REG: 040C | . |
| 0x0410 | STAT_RX_LANE_SYNC_REG: 0410 | . |
| 0x0414 | STAT_RX_LANE_SYNC_ERR_REG: 0414 | . |
| 0x0418 | STAT_RX_AM_ERR_REG: 0418 | . |
| 0x041C | STAT_RX_AM_LEN_ERR_REG: 041C | . |
| 0x0420 | STAT_RX_AM_REPEAT_ERR_REG: 0420 | . |
| 0x0424 | STAT_RX_LANE_DEMUXED: 0424 | . |
| 0x0428 | STAT_RX_PCS_LANE_NUM_REG1: 0428 | . |
| 0x0448 | STAT_RX_FEC_STATUS_REG: 0448 | . |
| 0x0450 | STAT_TX_FLOW_CONTROL_REG1: 0450 | Only in MAC+PCS variant |
| 0x0454 | STAT_RX_FLOW_CONTROL_REG1: 0454 | Only in MAC+PCS variant |
| 0x0458 | STAT_AN_STATUS: 0458 | . |
| 0x045C | STAT_AN_ABILITY: 045C | . |
| 0x0460 | STAT_AN_LINK_CTL: 0460 | . |
| 0x0464 | STAT_LT_STATUS_REG1: 0464 | . |
| 0x0468 | STAT_LT_STATUS_REG2: 0468 | . |
| 0x046C | STAT_LT_STATUS_REG3: 046C | . |
| 0x0470 | STAT_LT_STATUS_REG4: 0470 | . |
| 0x0474 | STAT_LT_COEFFICIENT0_REG: 0474 | . |
| 0x0478 | STAT_LT_COEFFICIENT1_REG: 0478 | . |

Statistics Counters

The statistics counters provide histograms of the classification of traffic and error counts. These counters can be read either by a 1 on `pm_tick` or by writing a 1 to `TICK_REG`, depending on the value of `MODE_REG[30]`.

The counters employ an internal accumulator. A write to the `TICK_REG` register causes the accumulated counts to be pushed to the readable `STAT_*_MSB/LSB` registers and simultaneously clear the accumulators. The `STAT_*_MSB/LSB` registers can then be read. In this way all values stored in the statistics counters represent a snapshot over the same time-interval.

The `STAT_CYCLE_COUNT_MSB/LSB` register contains a count of the number of RX core clock cycles between `TICK_REG` register writes. This allows for easy time-interval based statistics.

The counters have a default width of 48b. The counters saturate to 1s. The values in the counters are held until the next write to the `TICK_REG` register.

The first of the pair of addresses shown for the counters are the addresses of the LSB register, or bits 31:0 of the count. The MSB bits 47:32 of the counter are located at + 0x4 from the LSB.

Table 2-22: **Statistic Counters**

| Hex Address | Name/Link to Description | Notes |
|-------------|---|-------|
| 0x0500 | STATUS_CYCLE_COUNT_LSB: 0500 | |
| 0x0504 | STATUS_CYCLE_COUNT_MSB: 0504 | |
| 0x0508 | STAT_RX_BIP_ERR_0_LSB: 0508 | |
| 0x050C | STAT_RX_BIP_ERR_0_MSB: 050C | |
| 0x0510 | STAT_RX_BIP_ERR_1_LSB: 0510 | |
| 0x0514 | STAT_RX_BIP_ERR_1_MSB: 0514 | |
| 0x0518 | STAT_RX_BIP_ERR_2_LSB: 0518 | |
| 0x051C | STAT_RX_BIP_ERR_2_MSB: 051C | |
| 0x0520 | STAT_RX_BIP_ERR_3_LSB: 0520 | |
| 0x0524 | STAT_RX_BIP_ERR_3_MSB: 0524 | |
| 0x05A8 | STAT_RX_FRAMING_ERR_0_LSB: 05A8 | |
| 0x05AC | STAT_RX_FRAMING_ERR_0_MSB: 05AC | |
| 0x05B0 | STAT_RX_FRAMING_ERR_1_LSB: 05B0 | |
| 0x05B4 | STAT_RX_FRAMING_ERR_1_MSB: 05B4 | |
| 0x05B8 | STAT_RX_FRAMING_ERR_2_LSB: 05B8 | |
| 0x05BC | STAT_RX_FRAMING_ERR_2_MSB: 05BC | |
| 0x05C0 | STAT_RX_FRAMING_ERR_3_LSB: 05C0 | |
| 0x05C4 | STAT_RX_FRAMING_ERR_3_MSB: 05C4 | |

Table 2-22: Statistic Counters (Cont'd)

| Hex Address | Name/Link to Description | Notes |
|-------------|--|-------------------------|
| 0x0660 | STAT_RX_BAD_CODE_LSB: 0660 | |
| 0x0664 | STAT_RX_BAD_CODE_MSB: 0664 | |
| 0x0668 | STAT_RX_ERROR_LSB: 0668 | Only in PCS variant |
| 0x066C | STAT_RX_ERROR_MSB: 066C | Only in PCS variant |
| 0x06A0 | STAT_TX_FRAME_ERROR_LSB: 06A0 | Only in MAC+PCS variant |
| 0x06A4 | STAT_TX_FRAME_ERROR_MSB: 06A4 | Only in MAC+PCS variant |
| 0x0700 | STAT_TX_TOTAL_PACKETS_LSB: 0700 | Only in MAC+PCS variant |
| 0x0704 | STAT_TX_TOTAL_PACKETS_MSB: 0704 | Only in MAC+PCS variant |
| 0x0708 | STAT_TX_TOTAL_GOOD_PACKETS_LSB: 0708 | Only in MAC+PCS variant |
| 0x070C | STAT_TX_TOTAL_GOOD_PACKETS_MSB: 070C | Only in MAC+PCS variant |
| 0x0710 | STAT_TX_TOTAL_BYTES_LSB: 0710 | Only in MAC+PCS variant |
| 0x0714 | STAT_TX_TOTAL_BYTES_MSB: 0714 | Only in MAC+PCS variant |
| 0x0718 | STAT_TX_TOTAL_GOOD_BYTES_LSB: 0718 | Only in MAC+PCS variant |
| 0x071C | STAT_TX_TOTAL_GOOD_BYTES_MSB: 071C | Only in MAC+PCS variant |
| 0x0720 | STAT_TX_PACKET_64_BYTES_LSB: 0720 | Only in MAC+PCS variant |
| 0x0724 | STAT_TX_PACKET_64_BYTES_MSB: 0724 | Only in MAC+PCS variant |
| 0x0728 | STAT_TX_PACKET_65_127_BYTES_LSB: 0728 | Only in MAC+PCS variant |
| 0x072C | STAT_TX_PACKET_65_127_BYTES_MSB: 072C | Only in MAC+PCS variant |
| 0x0730 | STAT_TX_PACKET_128_255_BYTES_LSB: 0730 | Only in MAC+PCS variant |
| 0x0734 | STAT_TX_PACKET_128_255_BYTES_MSB: 0734 | Only in MAC+PCS variant |
| 0x0738 | STAT_TX_PACKET_256_511_BYTES_LSB: 0738 | Only in MAC+PCS variant |
| 0x073C | STAT_TX_PACKET_256_511_BYTES_MSB: 073C | Only in MAC+PCS variant |
| 0x0740 | STAT_TX_PACKET_512_1023_BYTES_LSB: 0740 | Only in MAC+PCS variant |
| 0x0744 | STAT_TX_PACKET_512_1023_BYTES_MSB: 0744 | Only in MAC+PCS variant |
| 0x0748 | STAT_TX_PACKET_1024_1518_BYTES_LSB: 0748 | Only in MAC+PCS variant |
| 0x074C | STAT_TX_PACKET_1024_1518_BYTES_MSB: 074C | Only in MAC+PCS variant |
| 0x0750 | STAT_TX_PACKET_1519_1522_BYTES_LSB: 0750 | Only in MAC+PCS variant |
| 0x0754 | STAT_TX_PACKET_1519_1522_BYTES_MSB: 0754 | Only in MAC+PCS variant |
| 0x0758 | STAT_TX_PACKET_1523_1548_BYTES_LSB: 0758 | Only in MAC+PCS variant |
| 0x075C | STAT_TX_PACKET_1523_1548_BYTES_MSB: 075C | Only in MAC+PCS variant |
| 0x0760 | STAT_TX_PACKET_1549_2047_BYTES_LSB: 0760 | Only in MAC+PCS variant |
| 0x0764 | STAT_TX_PACKET_1549_2047_BYTES_MSB: 0764 | Only in MAC+PCS variant |
| 0x0768 | STAT_TX_PACKET_2048_4095_BYTES_LSB: 0768 | Only in MAC+PCS variant |
| 0x076C | STAT_TX_PACKET_2048_4095_BYTES_MSB: 076C | Only in MAC+PCS variant |
| 0x0770 | STAT_TX_PACKET_4096_8191_BYTES_LSB: 0770 | Only in MAC+PCS variant |

Table 2-22: Statistic Counters (Cont'd)

| Hex Address | Name/Link to Description | Notes |
|-------------|--|-------------------------|
| 0x0774 | STAT_TX_PACKET_4096_8191_BYTES_MSB: 0774 | Only in MAC+PCS variant |
| 0x0778 | STAT_TX_PACKET_8192_9215_BYTES_LSB: 0778 | Only in MAC+PCS variant |
| 0x077C | STAT_TX_PACKET_8192_9215_BYTES_MSB: 077C | Only in MAC+PCS variant |
| 0x0780 | STAT_TX_PACKET_LARGE_LSB: 0780 | Only in MAC+PCS variant |
| 0x0784 | STAT_TX_PACKET_LARGE_MSB: 0784 | Only in MAC+PCS variant |
| 0x0788 | STAT_TX_PACKET_SMALL_LSB: 0788 | Only in MAC+PCS variant |
| 0x078C | STAT_TX_PACKET_SMALL_MSB: 078C | Only in MAC+PCS variant |
| 0x07B8 | STAT_TX_BAD_FCS_LSB: 07B8 | Only in MAC+PCS variant |
| 0x07BC | STAT_TX_BAD_FCS_MSB: 07BC | Only in MAC+PCS variant |
| 0x07D0 | STAT_TX_UNICAST_LSB: 07D0 | Only in MAC+PCS variant |
| 0x07D4 | STAT_TX_UNICAST_MSB: 07D4 | Only in MAC+PCS variant |
| 0x07D8 | STAT_TX_MULTICAST_LSB: 07D8 | Only in MAC+PCS variant |
| 0x07DC | STAT_TX_MULTICAST_MSB: 07DC | Only in MAC+PCS variant |
| 0x07E0 | STAT_TX_BROADCAST_LSB: 07E0 | Only in MAC+PCS variant |
| 0x07E4 | STAT_TX_BROADCAST_MSB: 07E4 | Only in MAC+PCS variant |
| 0x07E8 | STAT_TX_VLAN_LSB: 07E8 | Only in MAC+PCS variant |
| 0x07EC | STAT_TX_VLAN_MSB: 07EC | Only in MAC+PCS variant |
| 0x07F0 | STAT_TX_PAUSE_LSB: 07F0 | Only in MAC+PCS variant |
| 0x07F4 | STAT_TX_PAUSE_MSB: 07F4 | Only in MAC+PCS variant |
| 0x07F8 | STAT_TX_USER_PAUSE_LSB: 07F8 | Only in MAC+PCS variant |
| 0x07FC | STAT_TX_USER_PAUSE_MSB: 07FC | Only in MAC+PCS variant |
| 0x0808 | STAT_RX_TOTAL_PACKETS_LSB: 0808 | Only in MAC+PCS variant |
| 0x080C | STAT_RX_TOTAL_PACKETS_MSB: 080C | Only in MAC+PCS variant |
| 0x0810 | STAT_RX_TOTAL_GOOD_PACKETS_LSB: 0810 | Only in MAC+PCS variant |
| 0x0814 | STAT_RX_TOTAL_GOOD_PACKETS_MSB: 0814 | Only in MAC+PCS variant |
| 0x0818 | STAT_RX_TOTAL_BYTES_LSB: 0818 | Only in MAC+PCS variant |
| 0x081C | STAT_RX_TOTAL_BYTES_MSB: 081C | Only in MAC+PCS variant |
| 0x0820 | STAT_RX_TOTAL_GOOD_BYTES_LSB: 0820 | Only in MAC+PCS variant |
| 0x0824 | STAT_RX_TOTAL_GOOD_BYTES_MSB: 0824 | Only in MAC+PCS variant |
| 0x0828 | STAT_RX_PACKET_64_BYTES_LSB: 0828 | Only in MAC+PCS variant |
| 0x082C | STAT_RX_PACKET_64_BYTES_MSB: 082C | Only in MAC+PCS variant |
| 0x0830 | STAT_RX_PACKET_65_127_BYTES_LSB: 0830 | Only in MAC+PCS variant |
| 0x0834 | STAT_RX_PACKET_65_127_BYTES_MSB: 0834 | Only in MAC+PCS variant |
| 0x0838 | STAT_RX_PACKET_128_255_BYTES_LSB: 0838 | Only in MAC+PCS variant |
| 0x083C | STAT_RX_PACKET_128_255_BYTES_MSB: 083C | Only in MAC+PCS variant |

Table 2-22: Statistic Counters (Cont'd)

| Hex Address | Name/Link to Description | Notes |
|-------------|--|-------------------------|
| 0x0840 | STAT_RX_PACKET_256_511_BYTES_LSB: 0840 | Only in MAC+PCS variant |
| 0x0844 | STAT_RX_PACKET_256_511_BYTES_MSB: 0844 | Only in MAC+PCS variant |
| 0x0848 | STAT_RX_PACKET_512_1023_BYTES_LSB: 0848 | Only in MAC+PCS variant |
| 0x084C | STAT_RX_PACKET_512_1023_BYTES_MSB: 084C | Only in MAC+PCS variant |
| 0x0850 | STAT_RX_PACKET_1024_1518_BYTES_LSB: 0850 | Only in MAC+PCS variant |
| 0x0854 | STAT_RX_PACKET_1024_1518_BYTES_MSB: 0854 | Only in MAC+PCS variant |
| 0x0858 | STAT_RX_PACKET_1519_1522_BYTES_LSB: 0858 | Only in MAC+PCS variant |
| 0x085C | STAT_RX_PACKET_1519_1522_BYTES_MSB: 085C | Only in MAC+PCS variant |
| 0x0860 | STAT_RX_PACKET_1523_1548_BYTES_LSB: 0860 | Only in MAC+PCS variant |
| 0x0864 | STAT_RX_PACKET_1523_1548_BYTES_MSB: 0864 | Only in MAC+PCS variant |
| 0x0868 | STAT_RX_PACKET_1549_2047_BYTES_LSB: 0868 | Only in MAC+PCS variant |
| 0x086C | STAT_RX_PACKET_1549_2047_BYTES_MSB: 086C | Only in MAC+PCS variant |
| 0x0870 | STAT_RX_PACKET_2048_4095_BYTES_LSB: 0870 | Only in MAC+PCS variant |
| 0x0874 | STAT_RX_PACKET_2048_4095_BYTES_MSB: 0874 | Only in MAC+PCS variant |
| 0x0878 | STAT_RX_PACKET_4096_8191_BYTES_LSB: 0878 | Only in MAC+PCS variant |
| 0x087C | STAT_RX_PACKET_4096_8191_BYTES_MSB: 087C | Only in MAC+PCS variant |
| 0x0880 | STAT_RX_PACKET_8192_9215_BYTES_LSB: 0880 | Only in MAC+PCS variant |
| 0x0884 | STAT_RX_PACKET_8192_9215_BYTES_MSB: 0884 | Only in MAC+PCS variant |
| 0x0888 | STAT_RX_PACKET_LARGE_LSB: 0888 | Only in MAC+PCS variant |
| 0x088C | STAT_RX_PACKET_LARGE_MSB: 088C | Only in MAC+PCS variant |
| 0x0890 | STAT_RX_PACKET_SMALL_LSB: 0890 | Only in MAC+PCS variant |
| 0x0894 | STAT_RX_PACKET_SMALL_MSB: 0894 | Only in MAC+PCS variant |
| 0x0898 | STAT_RX_UNDERSIZE_LSB: 0898 | Only in MAC+PCS variant |
| 0x089C | STAT_RX_UNDERSIZE_MSB: 089C | Only in MAC+PCS variant |
| 0x08A0 | STAT_RX_FRAGMENT_LSB: 08A0 | Only in MAC+PCS variant |
| 0x08A4 | STAT_RX_FRAGMENT_MSB: 08A4 | Only in MAC+PCS variant |
| 0x08A8 | STAT_RX_OVERSIZE_LSB: 08A8 | Only in MAC+PCS variant |
| 0x08AC | STAT_RX_OVERSIZE_MSB: 08AC | Only in MAC+PCS variant |
| 0x08B0 | STAT_RX_TOOLONG_LSB: 08B0 | Only in MAC+PCS variant |
| 0x08B4 | STAT_RX_TOOLONG_MSB: 08B4 | Only in MAC+PCS variant |
| 0x08B8 | STAT_RX_JABBER_LSB: 08B8 | Only in MAC+PCS variant |
| 0x08BC | STAT_RX_JABBER_MSB: 08BC | Only in MAC+PCS variant |
| 0x08C0 | STAT_RX_BAD_FCS_LSB: 08C0 | Only in MAC+PCS variant |
| 0x08C4 | STAT_RX_BAD_FCS_MSB: 08C4 | Only in MAC+PCS variant |
| 0x08C8 | STAT_RX_PACKET_BAD_FCS_LSB: 08C8 | Only in MAC+PCS variant |

Table 2-22: Statistic Counters (Cont'd)

| Hex Address | Name/Link to Description | Notes |
|-------------|---|-------------------------|
| 0x08CC | STAT_RX_PACKET_BAD_FCS_MSB: 08CC | Only in MAC+PCS variant |
| 0x08D0 | STAT_RX_STOMPED_FCS_LSB: 08D0 | Only in MAC+PCS variant |
| 0x08D4 | STAT_RX_STOMPED_FCS_MSB: 08D4 | Only in MAC+PCS variant |
| 0x08D8 | STAT_RX_UNICAST_LSB: 08D8 | Only in MAC+PCS variant |
| 0x08DC | STAT_RX_UNICAST_MSB: 08DC | Only in MAC+PCS variant |
| 0x08E0 | STAT_RX_MULTICAST_LSB: 08E0 | Only in MAC+PCS variant |
| 0x08E4 | STAT_RX_MULTICAST_MSB: 08E4 | Only in MAC+PCS variant |
| 0x08E8 | STAT_RX_BROADCAST_LSB: 08E8 | Only in MAC+PCS variant |
| 0x08EC | STAT_RX_BROADCAST_MSB: 08EC | Only in MAC+PCS variant |
| 0x08F0 | STAT_RX_VLAN_LSB: 08F0 | Only in MAC+PCS variant |
| 0x08F4 | STAT_RX_VLAN_MSB: 08F4 | Only in MAC+PCS variant |
| 0x08F8 | STAT_RX_PAUSE_LSB: 08F8 | Only in MAC+PCS variant |
| 0x08FC | STAT_RX_PAUSE_MSB: 08FC | Only in MAC+PCS variant |
| 0x0900 | STAT_RX_USER_PAUSE_LSB: 0900 | Only in MAC+PCS variant |
| 0x0904 | STAT_RX_USER_PAUSE_MSB: 0904 | Only in MAC+PCS variant |
| 0x0908 | STAT_RX_INRANGEERR_LSB: 0908 | Only in MAC+PCS variant |
| 0x090C | STAT_RX_INRANGEERR_MSB: 090C | Only in MAC+PCS variant |
| 0x0910 | STAT_RX_TRUNCATED_LSB: 0910 | Only in MAC+PCS variant |
| 0x0914 | STAT_RX_TRUNCATED_MSB: 0914 | Only in MAC+PCS variant |
| 0x0918 | STAT_RX_TEST_PATTERN_MISMATCH_LSB: 0918 | Only in MAC+PCS variant |
| 0x091C | STAT_RX_TEST_PATTERN_MISMATCH_MSB: 091C | Only in MAC+PCS variant |
| 0x0920 | STAT_FEC_INC_CORRECT_COUNT_LSB: 0920 | |
| 0x0924 | STAT_FEC_INC_CORRECT_COUNT_MSB: 0924 | |
| 0x0928 | STAT_FEC_INC_CANT_CORRECT_COUNT_LSB: 0928 | |
| 0x092C | STAT_FEC_INC_CANT_CORRECT_COUNT_MSB: 092C | |

Register Definitions

Configuration Registers

This section contains descriptions of the configuration registers. In the cases where the features described in the bit fields are not present in the IP core, the bit field reverts to RESERVED. Reserved fields in the configuration registers do not accept any written value, and always return a 0 when read. Registers or bit fields within registers can be accessed for read-write (RW), write-only (WO), or read-only (RO). Default values shown are decimal values and take effect after reset.

A description of each signal is found in the port list section of this document.

GT_RESET_REG: 0000

Table 2-23: GT_RESET_REG: 0000

| Bits | Default | Type | Signal |
|------|---------|------|------------------|
| 0 | 0 | RW | ctl_gt_reset_all |
| 1 | 0 | RW | ctl_gt_rx_reset |
| 2 | 0 | RW | ctl_gt_tx_reset |

RESET_REG: 0004

Table 2-24: RESET_REG: 0004

| Bits | Default | Type | Signal |
|------|---------|------|-----------------|
| 3:0 | 0 | RW | rx_serdes_reset |
| 29 | 0 | RW | tx_serdes_reset |
| 30 | 0 | RW | rx_reset |
| 31 | 0 | RW | tx_reset |

MODE_REG: 0008

Table 2-25: MODE_REG: 0008

| Bits | Default | Type | Signal |
|------|---------|------|--------------------------|
| 30 | 1 | RW | tick_reg_mode_sel |
| 31 | 0 | RW | GT near-end PMA loopback |

CONFIGURATION_TX_REG1: 000C

Table 2-26: CONFIGURATION_TX_REG1: 000C

| Bits | Default | Type | Signal |
|-------|---------|------|--|
| 0 | 1 | RW | ctl_tx_enable ⁽¹⁾ |
| 1 | 1 | RW | ctl_tx_fcs_ins_enable ⁽¹⁾ |
| 2 | 0 | RW | ctl_tx_ignore_fcs ⁽¹⁾ |
| 3 | 0 | RW | ctl_tx_send_lfi ⁽¹⁾ |
| 4 | 0 | RW | ctl_tx_send_rfi ⁽¹⁾ |
| 5 | 0 | RW | ctl_tx_send_idle ⁽¹⁾ |
| 13:10 | 12 | RW | ctl_tx_ipg_value ⁽¹⁾ |
| 14 | 0 | RW | ctl_tx_test_pattern |
| 18 | 0 | RW | ctl_tx_custom_preamble_enable ⁽¹⁾ |

Notes:

1. Only in MAC+PCS variant

CONFIGURATION_RX_REG1: 0014

Table 2-27: CONFIGURATION_RX_REG1: 0014

| Bits | Default | Type | Signal |
|------|---------|------|--|
| 0 | 1 | RW | ctl_rx_enable ⁽¹⁾ |
| 1 | 1 | RW | ctl_rx_delete_fcs ⁽¹⁾ |
| 2 | 0 | RW | ctl_rx_ignore_fcs ⁽¹⁾ |
| 3 | 0 | RW | ctl_rx_process_lfi ⁽¹⁾ |
| 4 | 1 | RW | ctl_rx_check_sfd ⁽¹⁾ |
| 5 | 1 | RW | ctl_rx_check_preamble ⁽¹⁾ |
| 6 | 0 | RW | ctl_rx_force_resync ⁽¹⁾ |
| 7 | 0 | RW | ctl_rx_test_pattern |
| 18 | 0 | RW | ctl_rx_custom_preamble_enable ⁽¹⁾ |

Notes:

1. Only in MAC+PCS variant

CONFIGURATION_RX_MTU: 0018

Table 2-28: CONFIGURATION_RX_MTU: 0018

| Bits | Default | Type | Signal |
|-------|---------|------|-----------------------|
| 7:0 | 64 | RW | ctl_rx_min_packet_len |
| 30:16 | 9600 | RW | ctl_rx_max_packet_len |

CONFIGURATION_VL_LENGTH_REG: 001C

Table 2-29: CONFIGURATION_VL_LENGTH_REG: 001C

| Bits | Default | Type | Signal |
|-------|---------|------|-------------------------|
| 15:0 | 16383 | RW | ctl_tx_vl_length_minus1 |
| 31:16 | 16383 | RW | ctl_rx_vl_length_minus1 |

TICK_REG: 0020

Table 2-30: TICK_REG: 0020

| Bits | Default | Type | Signal |
|------|---------|------|----------|
| 0 | 0 | WO | tick_reg |

CONFIGURATION_REVISION_REG: 0024

Table 2-31: CONFIGURATION_REVISION_REG: 0024

| Bits | Default | Type | Signal |
|-------|---------|------|-----------|
| 7:0 | 1 | RO | major_rev |
| 15:8 | 1 | RO | minor_rev |
| 31:24 | 0 | RO | patch_rev |

CONFIGURATION_TX_FLOW_CONTROL_REG1: 0040

Table 2-32: CONFIGURATION_TX_FLOW_CONTROL_REG1: 0040

| Bits | Default | Type | Signal |
|------|---------|------|---------------------|
| 8:0 | 0 | RW | ctl_tx_pause_enable |

CONFIGURATION_TX_FLOW_CONTROL_REFRESH_REG1: 0044

Table 2-33: CONFIGURATION_TX_FLOW_CONTROL_REFRESH_REG1: 0044

| Bits | Default | Type | Signal |
|-------|---------|------|-----------------------------|
| 15:0 | 0 | RW | ctl_tx_pause_refresh_timer0 |
| 31:16 | 0 | RW | ctl_tx_pause_refresh_timer1 |

CONFIGURATION_TX_FLOW_CONTROL_REFRESH_REG2: 0048

Table 2-34: CONFIGURATION_TX_FLOW_CONTROL_REFRESH_REG2: 0048

| Bits | Default | Type | Signal |
|-------|---------|------|-----------------------------|
| 15:0 | 0 | RW | ctl_tx_pause_refresh_timer2 |
| 31:16 | 0 | RW | ctl_tx_pause_refresh_timer3 |

CONFIGURATION_TX_FLOW_CONTROL_REFRESH_REG3: 004C

Table 2-35: CONFIGURATION_TX_FLOW_CONTROL_REFRESH_REG3: 004C

| Bits | Default | Type | Signal |
|-------|---------|------|-----------------------------|
| 15:0 | 0 | RW | ctl_tx_pause_refresh_timer4 |
| 31:16 | 0 | RW | ctl_tx_pause_refresh_timer5 |

CONFIGURATION_TX_FLOW_CONTROL_REFRESH_REG4: 0050

Table 2-36: CONFIGURATION_TX_FLOW_CONTROL_REFRESH_REG4: 0050

| Bits | Default | Type | Signal |
|-------|---------|------|-----------------------------|
| 15:0 | 0 | RW | ctl_tx_pause_refresh_timer6 |
| 31:16 | 0 | RW | ctl_tx_pause_refresh_timer7 |

CONFIGURATION_TX_FLOW_CONTROL_REFRESH_REG5: 0054

Table 2-37: CONFIGURATION_TX_FLOW_CONTROL_REFRESH_REG5: 0054

| Bits | Default | Type | Signal |
|------|---------|------|-----------------------------|
| 15:0 | 0 | RW | ctl_tx_pause_refresh_timer8 |

CONFIGURATION_TX_FLOW_CONTROL_QUANTA_REG1: 0058

Table 2-38: CONFIGURATION_TX_FLOW_CONTROL_QUANTA_REG1: 0058

| Bits | Default | Type | Signal |
|-------|---------|------|----------------------|
| 15:0 | 0 | RW | ctl_tx_pause_quanta0 |
| 31:16 | 0 | RW | ctl_tx_pause_quanta1 |

CONFIGURATION_TX_FLOW_CONTROL_QUANTA_REG2: 005C

Table 2-39: CONFIGURATION_TX_FLOW_CONTROL_QUANTA_REG2: 005C

| Bits | Default | Type | Signal |
|-------|---------|------|----------------------|
| 15:0 | 0 | RW | ctl_tx_pause_quanta2 |
| 31:16 | 0 | RW | ctl_tx_pause_quanta3 |

CONFIGURATION_TX_FLOW_CONTROL_QUANTA_REG3: 0060

Table 2-40: CONFIGURATION_TX_FLOW_CONTROL_QUANTA_REG3: 0060

| Bits | Default | Type | Signal |
|-------|---------|------|----------------------|
| 15:0 | 0 | RW | ctl_tx_pause_quanta4 |
| 31:16 | 0 | RW | ctl_tx_pause_quanta5 |

CONFIGURATION_TX_FLOW_CONTROL_QUANTA_REG4: 0064

Table 2-41: CONFIGURATION_TX_FLOW_CONTROL_QUANTA_REG4: 0064

| Bits | Default | Type | Signal |
|-------|---------|------|----------------------|
| 15:0 | 0 | RW | ctl_tx_pause_quanta6 |
| 31:16 | 0 | RW | ctl_tx_pause_quanta7 |

CONFIGURATION_TX_FLOW_CONTROL_QUANTA_REG5: 0068

Table 2-42: CONFIGURATION_TX_FLOW_CONTROL_QUANTA_REG5: 0068

| Bits | Default | Type | Signal |
|------|---------|------|----------------------|
| 15:0 | 0 | RW | ctl_tx_pause_quanta8 |

CONFIGURATION_TX_FLOW_CONTROL_PPP_ETYPE_OP_REG: 006C

Table 2-43: CONFIGURATION_TX_FLOW_CONTROL_PPP_ETYPE_OP_REG: 006C

| Bits | Default | Type | Signal |
|-------|---------|------|----------------------|
| 15:0 | 34824 | RW | ctl_tx_ethertype_ppp |
| 31:16 | 257 | RW | ctl_tx_opcode_ppp |

CONFIGURATION_TX_FLOW_CONTROL_GPP_ETYPE_OP_REG: 0070

Table 2-44: CONFIGURATION_TX_FLOW_CONTROL_GPP_ETYPE_OP_REG: 0070

| Bits | Default | Type | Signal |
|-------|---------|------|----------------------|
| 15:0 | 34824 | RW | ctl_tx_ethertype_gpp |
| 31:16 | 1 | RW | ctl_tx_opcode_gpp |

CONFIGURATION_TX_FLOW_CONTROL_GPP_DA_REG_LSB: 0074

Table 2-45: CONFIGURATION_TX_FLOW_CONTROL_GPP_DA_REG_LSB: 0074

| Bits | Default | Type | Signal |
|------|---------|------|---------------------|
| 31:0 | 0 | RW | ctl_tx_da_gpp[31:0] |

CONFIGURATION_TX_FLOW_CONTROL_GPP_DA_REG_MSB: 0078

Table 2-46: CONFIGURATION_TX_FLOW_CONTROL_GPP_DA_REG_MSB: 0078

| Bits | Default | Type | Signal |
|------|---------|------|----------------------|
| 15:0 | 0 | RW | ctl_tx_da_gpp[47:32] |

CONFIGURATION_TX_FLOW_CONTROL_GPP_SA_REG_LSB: 007C

Table 2-47: CONFIGURATION_TX_FLOW_CONTROL_GPP_SA_REG_LSB: 007C

| Bits | Default | Type | Signal |
|------|---------|------|---------------------|
| 31:0 | 0 | RW | ctl_tx_sa_gpp[31:0] |

CONFIGURATION_TX_FLOW_CONTROL_GPP_SA_REG_MSB: 0080

Table 2-48: CONFIGURATION_TX_FLOW_CONTROL_GPP_SA_REG_MSB: 0080

| Bits | Default | Type | Signal |
|------|---------|------|----------------------|
| 15:0 | 0 | RW | ctl_tx_sa_gpp[47:32] |

CONFIGURATION_TX_FLOW_CONTROL_PPP_DA_REG_LSB: 0084

Table 2-49: CONFIGURATION_TX_FLOW_CONTROL_PPP_DA_REG_LSB: 0084

| Bits | Default | Type | Signal |
|------|---------|------|---------------------|
| 31:0 | 0 | RW | ctl_tx_da_ppp[31:0] |

CONFIGURATION_TX_FLOW_CONTROL_PPP_DA_REG_MSB: 0088

Table 2-50: CONFIGURATION_TX_FLOW_CONTROL_PPP_DA_REG_MSB: 0088

| Bits | Default | Type | Signal |
|------|---------|------|----------------------|
| 15:0 | 0 | RW | ctl_tx_da_ppp[47:32] |

CONFIGURATION_TX_FLOW_CONTROL_PPP_SA_REG_LSB: 008C

Table 2-51: CONFIGURATION_TX_FLOW_CONTROL_PPP_SA_REG_LSB: 008C

| Bits | Default | Type | Signal |
|------|---------|------|---------------------|
| 31:0 | 0 | RW | ctl_tx_sa_ppp[31:0] |

CONFIGURATION_TX_FLOW_CONTROL_PPP_SA_REG_MSB: 0090

Table 2-52: CONFIGURATION_TX_FLOW_CONTROL_PPP_SA_REG_MSB: 0090

| Bits | Default | Type | Signal |
|------|---------|------|----------------------|
| 15:0 | 0 | RW | ctl_tx_sa_ppp[47:32] |

CONFIGURATION_RX_FLOW_CONTROL_REG1: 0094

Table 2-53: CONFIGURATION_RX_FLOW_CONTROL_REG1: 0094

| Bits | Default | Type | Signal |
|------|---------|------|------------------------|
| 8:0 | 0 | RW | ctl_rx_pause_enable |
| 9 | 0 | RW | ctl_rx_forward_control |
| 10 | 0 | RW | ctl_rx_enable_gcp |
| 11 | 0 | RW | ctl_rx_enable_pcp |
| 12 | 0 | RW | ctl_rx_enable_gpp |
| 13 | 0 | RW | ctl_rx_enable_ppp |
| 14 | 0 | RW | ctl_rx_check_ack |

CONFIGURATION_RX_FLOW_CONTROL_REG2: 0098

Table 2-54: CONFIGURATION_RX_FLOW_CONTROL_REG2: 0098

| Bits | Default | Type | Signal |
|------|---------|------|-------------------------|
| 0 | 0 | RW | ctl_rx_check_mcast_gcp |
| 1 | 0 | RW | ctl_rx_check_ucast_gcp |
| 2 | 0 | RW | ctl_rx_check_sa_gcp |
| 3 | 0 | RW | ctl_rx_check_etype_gcp |
| 4 | 0 | RW | ctl_rx_check_opcode_gcp |
| 5 | 0 | RW | ctl_rx_check_mcast_pcp |
| 6 | 0 | RW | ctl_rx_check_ucast_pcp |
| 7 | 0 | RW | ctl_rx_check_sa_pcp |
| 8 | 0 | RW | ctl_rx_check_etype_pcp |
| 9 | 0 | RW | ctl_rx_check_opcode_pcp |
| 10 | 0 | RW | ctl_rx_check_mcast_gpp |
| 11 | 0 | RW | ctl_rx_check_ucast_gpp |
| 12 | 0 | RW | ctl_rx_check_sa_gpp |
| 13 | 0 | RW | ctl_rx_check_etype_gpp |
| 14 | 0 | RW | ctl_rx_check_opcode_gpp |
| 15 | 0 | RW | ctl_rx_check_mcast_ppp |
| 16 | 0 | RW | ctl_rx_check_ucast_ppp |
| 17 | 0 | RW | ctl_rx_check_sa_ppp |
| 18 | 0 | RW | ctl_rx_check_etype_ppp |
| 19 | 0 | RW | ctl_rx_check_opcode_ppp |

CONFIGURATION_RX_FLOW_CONTROL_PPP_ETYPE_OP_REG: 009C

Table 2-55: CONFIGURATION_RX_FLOW_CONTROL_PPP_ETYPE_OP_REG: 009C

| Bits | Default | Type | Signal |
|-------|---------|------|-------------------|
| 15:0 | 34,824 | RW | ctl_rx_etype_ppp |
| 31:16 | 257 | RW | ctl_rx_opcode_ppp |

CONFIGURATION_RX_FLOW_CONTROL_GPP_ETYPE_OP_REG: 00A0

Table 2-56: RATION_RX_FLOW_CONTROL_GPP_ETYPE_OP_REG: 00A0

| Bits | Default | Type | Signal |
|-------|---------|------|-------------------|
| 15:0 | 34824 | RW | ctl_rx_etype_gpp |
| 31:16 | 1 | RW | ctl_rx_opcode_gpp |

CONFIGURATION_RX_FLOW_CONTROL_GCP_PCP_TYPE_REG: 00A4

Table 2-57: CONFIGURATION_RX_FLOW_CONTROL_GCP_PCP_TYPE_REG: 00A4

| Bits | Default | Type | Signal |
|-------|---------|------|------------------|
| 15:0 | 34,824 | RW | ctl_rx_etype_gcp |
| 31:16 | 34,824 | RW | ctl_rx_etype_pcp |

CONFIGURATION_RX_FLOW_CONTROL_PCP_OP_REG: 00A8

Table 2-58: CONFIGURATION_RX_FLOW_CONTROL_PCP_OP_REG: 00A8

| Bits | Default | Type | Signal |
|-------|---------|------|-----------------------|
| 15:0 | 257 | RW | ctl_rx_opcode_min_pcp |
| 31:16 | 257 | RW | ctl_rx_opcode_max_pcp |

CONFIGURATION_RX_FLOW_CONTROL_GCP_OP_REG: 00AC

Table 2-59: CONFIGURATION_RX_FLOW_CONTROL_GCP_OP_REG: 00AC

| Bits | Default | Type | Signal |
|-------|---------|------|-----------------------|
| 15:0 | 1 | RW | ctl_rx_opcode_min_gcp |
| 31:16 | 6 | RW | ctl_rx_opcode_max_gcp |

CONFIGURATION_RX_FLOW_CONTROL_DA_REG1_LSB: 00B0

Table 2-60: CONFIGURATION_RX_FLOW_CONTROL_DA_REG1_LSB: 00B0

| Bits | Default | Type | Signal |
|------|---------|------|-----------------------------|
| 31:0 | 0 | RW | ctl_rx_pause_da_ucast[31:0] |

CONFIGURATION_RX_FLOW_CONTROL_DA_REG1_MSB: 00B4

Table 2-61: CONFIGURATION_RX_FLOW_CONTROL_DA_REG1_MSB: 00B4

| Bits | Default | Type | Signal |
|------|---------|------|------------------------------|
| 15:0 | 0 | RW | ctl_rx_pause_da_ucast[47:32] |

CONFIGURATION_RX_FLOW_CONTROL_DA_REG2_LSB: 00B8

Table 2-62: CONFIGURATION_RX_FLOW_CONTROL_DA_REG2_LSB: 00B8

| Bits | Default | Type | Signal |
|------|---------|------|-----------------------------|
| 31:0 | 0 | RW | ctl_rx_pause_da_mcast[31:0] |

CONFIGURATION_RX_FLOW_CONTROL_DA_REG2_MSB: 00BC

Table 2-63: CONFIGURATION_RX_FLOW_CONTROL_DA_REG2_MSB: 00BC

| Bits | Default | Type | Signal |
|------|---------|------|------------------------------|
| 15:0 | 0 | RW | ctl_rx_pause_da_mcast[47:32] |

CONFIGURATION_RX_FLOW_CONTROL_SA_REG1_LSB: 00C0

Table 2-64: CONFIGURATION_RX_FLOW_CONTROL_SA_REG1_LSB: 00C0

| Bits | Default | Type | Signal |
|------|---------|------|-----------------------|
| 31:0 | 0 | RW | ctl_rx_pause_sa[31:0] |

CONFIGURATION_RX_FLOW_CONTROL_SA_REG1_MSB: 00C4

Table 2-65: CONFIGURATION_RX_FLOW_CONTROL_SA_REG1_MSB: 00C4

| Bits | Default | Type | Signal |
|------|---------|------|------------------------|
| 15:0 | 0 | RW | ctl_rx_pause_sa[47:32] |

CONFIGURATION_FEC_REG: 00D4

Table 2-66: CONFIGURATION_FEC_REG: 00D4

| Bits | Default | Type | Signal |
|------|---------|------|-----------------------------|
| 0 | 0 | RW | ctl_fec_rx_enable |
| 1 | 0 | RW | ctl_fec_tx_enable |
| 2 | 0 | RW | ctl_fec_enable_error_to_pcs |

CONFIGURATION_AN_CONTROL_REG1: 00E0

Table 2-67: CONFIGURATION_AN_CONTROL_REG1: 00E0

| Bits | Default | Type | Signal |
|------|---------|------|-------------------------|
| 0 | 0 | RW | ctl_autoneg_enable |
| 1 | 1 | RW | ctl_autoneg_bypass |
| 9:2 | 0 | RW | ctl_an_nonce_seed |
| 10 | 0 | RW | ctl_an_pseudo_sel |
| 11 | 0 | RW | ctl_restart_negotiation |
| 12 | 0 | RW | ctl_an_local_fault |

CONFIGURATION_AN_CONTROL_REG2: 00E4

Table 2-68: CONFIGURATION_AN_CONTROL_REG2: 00E4

| Bits | Default | Type | Signal |
|------|---------|------|-----------------------------|
| 0 | 0 | RW | ctl_an_pause |
| 1 | 0 | RW | ctl_an_asmdir |
| 16 | 0 | RW | ctl_an_fec_10g_request |
| 17 | 0 | RW | ctl_an_fec_ability_override |
| 18 | 0 | RW | ctl_an_cl91_fec_request |
| 19 | 0 | RW | ctl_an_cl91_fec_ability |

Table 2-68: CONFIGURATION_AN_CONTROL_REG2: 00E4 (Cont'd)

| Bits | Default | Type | Signal |
|------|---------|------|------------------------------|
| 20 | 0 | RW | ctl_an_fec_25g_rs_request |
| 21 | 0 | RW | ctl_an_fec_25g_baser_request |

CONFIGURATION_AN_ABILITY: 00F8

Table 2-69: CONFIGURATION_AN_ABILITY: 00F8

| Bits | Default | Type | Signal |
|------|---------|------|-------------------------------|
| 0 | 0 | RW | ctl_an_ability_1000base_kx |
| 1 | 0 | RW | ctl_an_ability_10gbase_kx4 |
| 2 | 0 | RW | ctl_an_ability_10gbase_kr |
| 3 | 0 | RW | ctl_an_ability_40gbase_kr4 |
| 4 | 0 | RW | ctl_an_ability_40gbase_cr4 |
| 5 | 0 | RW | ctl_an_ability_100gbase_cr10 |
| 6 | 0 | RW | ctl_an_ability_100gbase_kp4 |
| 7 | 0 | RW | ctl_an_ability_100gbase_kr4 |
| 8 | 0 | RW | ctl_an_ability_100gbase_cr4 |
| 9 | 0 | RW | ctl_an_ability_25gbase_krcr_s |
| 10 | 0 | RW | ctl_an_ability_25gbase_krcr |
| 11 | 0 | RW | ctl_an_ability_25gbase_kr1 |
| 12 | 0 | RW | ctl_an_ability_25gbase_cr1 |
| 13 | 0 | RW | ctl_an_ability_50gbase_kr2 |
| 14 | 0 | RW | ctl_an_ability_50gbase_cr2 |

CONFIGURATION_LT_CONTROL_REG1: 0100

Table 2-70: CONFIGURATION_LT_CONTROL_REG1: 0100

| Bits | Default | Type | Signal |
|------|---------|------|-------------------------|
| 0 | 0 | RW | ctl_lt_training_enable |
| 1 | 0 | RW | ctl_lt_restart_training |

CONFIGURATION_LT_TRAINED_REG: 0104

Table 2-71: CONFIGURATION_LT_TRAINED_REG: 0104

| Bits | Default | Type | Signal |
|------|---------|------|-------------------|
| 3:0 | 0 | RW | ctl_lt_rx_trained |

CONFIGURATION_LT_PRESET_REG: 0108

Table 2-72: CONFIGURATION_LT_PRESET_REG: 0108

| Bits | Default | Type | Signal |
|------|---------|------|---------------------|
| 3:0 | 0 | RW | ctl_lt_preset_to_tx |

CONFIGURATION_LT_INIT_REG: 010C

Table 2-73: CONFIGURATION_LT_INIT_REG: 010C

| Bits | Default | Type | Signal |
|------|---------|------|-------------------------|
| 3:0 | 0 | RW | ctl_lt_initialize_to_tx |

CONFIGURATION_LT_SEED_REG0: 0110

Table 2-74: CONFIGURATION_LT_SEED_REG0: 0110

| Bits | Default | Type | Signal |
|-------|---------|------|---------------------|
| 10:0 | 0 | RW | ctl_lt_pseudo_seed0 |
| 26:16 | 0 | RW | ctl_lt_pseudo_seed1 |

CONFIGURATION_LT_SEED_REG1: 0114

Table 2-75: CONFIGURATION_LT_SEED_REG1: 0114

| Bits | Default | Type | Signal |
|-------|---------|------|---------------------|
| 10:0 | 0 | RW | ctl_lt_pseudo_seed2 |
| 26:16 | 0 | RW | ctl_lt_pseudo_seed3 |

CONFIGURATION_LT_COEFFICIENT_REG0: 0130

Table 2-76: CONFIGURATION_LT_COEFFICIENT_REG0: 0130

| Bits | Default | Type | Signal |
|-------|---------|------|-----------------------|
| 1:0 | 0 | RW | ctl_lt_k_p1_to_tx0 |
| 3:2 | 0 | RW | ctl_lt_k0_to_tx0 |
| 5:4 | 0 | RW | ctl_lt_k_m1_to_tx0 |
| 7:6 | 0 | RW | ctl_lt_stat_p1_to_tx0 |
| 9:8 | 0 | RW | ctl_lt_stat0_to_tx0 |
| 11:10 | 0 | RW | ctl_lt_stat_m1_to_tx0 |
| 17:16 | 0 | RW | ctl_lt_k_p1_to_tx1 |
| 19:18 | 0 | RW | ctl_lt_k0_to_tx1 |
| 21:20 | 0 | RW | ctl_lt_k_m1_to_tx1 |

Table 2-76: CONFIGURATION_LT_COEFFICIENT_REG0: 0130 (Cont'd)

| Bits | Default | Type | Signal |
|-------|---------|------|-----------------------|
| 23:22 | 0 | RW | ctl_lt_stat_p1_to_tx1 |
| 25:24 | 0 | RW | ctl_lt_stat0_to_tx1 |
| 27:26 | 0 | RW | ctl_lt_stat_m1_to_tx1 |

CONFIGURATION_LT_COEFFICIENT_REG1: 0134

Table 2-77: CONFIGURATION_LT_COEFFICIENT_REG1: 0134

| Bits | Default | Type | Signal |
|-------|---------|------|-----------------------|
| 1:0 | 0 | RW | ctl_lt_k_p1_to_tx2 |
| 3:2 | 0 | RW | ctl_lt_k0_to_tx2 |
| 5:4 | 0 | RW | ctl_lt_k_m1_to_tx2 |
| 7:6 | 0 | RW | ctl_lt_stat_p1_to_tx2 |
| 9:8 | 0 | RW | ctl_lt_stat0_to_tx2 |
| 11:10 | 0 | RW | ctl_lt_stat_m1_to_tx2 |
| 17:16 | 0 | RW | ctl_lt_k_p1_to_tx3 |
| 19:18 | 0 | RW | ctl_lt_k0_to_tx3 |
| 21:20 | 0 | RW | ctl_lt_k_m1_to_tx3 |
| 23:22 | 0 | RW | ctl_lt_stat_p1_to_tx3 |
| 25:24 | 0 | RW | ctl_lt_stat0_to_tx3 |
| 27:26 | 0 | RW | ctl_lt_stat_m1_to_tx3 |

Status Registers

The following tables describe the status registers for this Ethernet IP core.

Some bits are sticky, that is, latching their value High or Low once set. This is indicated by the type LH (latched High) or LL (latched Low).

STAT_TX_STATUS_REG1: 0400

Table 2-78: STAT_TX_STATUS_REG1: 0400

| Bits | Default | Type | Signal |
|------|---------|-------|---------------------|
| 0 | 0 | RO LH | stat_tx_local_fault |

STAT_RX_STATUS_REG1: 0404

Table 2-79: STAT_RX_STATUS_REG1: 0404

| Bits | Default | Type | Signal |
|------|---------|-------|---|
| 0 | 0 | RO LL | stat_rx_status |
| 1 | 0 | RO LL | stat_rx_aligned |
| 2 | 0 | RO LH | stat_rx_misaligned |
| 3 | 0 | RO LH | stat_rx_aligned_err |
| 4 | 0 | RO LH | stat_rx_hi_ber |
| 5 | 0 | RO LH | stat_rx_remote_fault ⁽¹⁾ |
| 6 | 0 | RO LH | stat_rx_local_fault |
| 7 | 0 | RO LH | stat_rx_internal_local_fault ⁽¹⁾ |
| 8 | 0 | RO LH | stat_rx_received_local_fault ⁽¹⁾ |
| 9 | 0 | RO LH | stat_rx_bad_preamble ⁽¹⁾ |
| 10 | 0 | RO LH | stat_rx_bad_sfd ⁽¹⁾ |
| 11 | 0 | RO LH | stat_rx_got_signal_os ⁽¹⁾ |

Notes:

1. Only in MAC+PCS variant

STAT_STATUS_REG1: 0408

Table 2-80: STAT_STATUS_REG1: 0408

| Bits | Default | Type | Signal |
|------|---------|-------|---|
| 0 | 0 | RO LH | stat_tx_fifo_error |
| 4 | 0 | RO LH | stat_tx_ptp_fifo_read_error ⁽¹⁾ |
| 5 | 0 | RO LH | stat_tx_ptp_fifo_write_error ⁽¹⁾ |

Notes:

1. Only in MAC+PCS variant

STAT_RX_BLOCK_LOCK_REG: 040C

Table 2-81: STAT_RX_BLOCK_LOCK_REG: 040C

| Bits | Default | Type | Signal |
|------|---------|-------|--------------------|
| 3:0 | 0 | RO LL | stat_rx_block_lock |

STAT_RX_LANE_SYNC_REG: 0410

Table 2-82: STAT_RX_LANE_SYNC_REG: 0410

| Bits | Default | Type | Signal |
|------|---------|-------|----------------|
| 3:0 | 0 | RO LL | stat_rx_synced |

STAT_RX_LANE_SYNC_ERR_REG: 0414

Table 2-83: STAT_RX_LANE_SYNC_ERR_REG: 0414

| Bits | Default | Type | Signal |
|------|---------|-------|--------------------|
| 3:0 | 0 | RO LH | stat_rx_synced_err |

STAT_RX_AM_ERR_REG: 0418

Table 2-84: STAT_RX_AM_ERR_REG: 0418

| Bits | Default | Type | Signal |
|------|---------|-------|----------------|
| 3:0 | 0 | RO LH | stat_rx_mf_err |

STAT_RX_AM_LEN_ERR_REG: 041C

Table 2-85: STAT_RX_AM_LEN_ERR_REG: 041C

| Bits | Default | Type | Signal |
|------|---------|-------|--------------------|
| 3:0 | 0 | RO LH | stat_rx_mf_len_err |

STAT_RX_AM_REPEAT_ERR_REG: 0420

Table 2-86: STAT_RX_AM_REPEAT_ERR_REG: 0420

| Bits | Default | Type | Signal |
|------|---------|-------|-----------------------|
| 3:0 | 0 | RO LH | stat_rx_mf_repeat_err |

STAT_RX_LANE_DEMUXED: 0424

Table 2-87: STAT_RX_LANE_DEMUXED: 0424

| Bits | Default | Type | Signal |
|------|---------|------|--------------------|
| 3:0 | 0 | RO | stat_rx_vl_demuxed |

STAT_RX_PCS_LANE_NUM_REG1: 0428

Table 2-88: STAT_RX_PCS_LANE_NUM_REG1: 0428

| Bits | Default | Type | Signal |
|-------|---------|------|---------------------|
| 1:0 | 0 | RO | stat_rx_vl_number_0 |
| 6:5 | 0 | RO | stat_rx_vl_number_1 |
| 11:10 | 0 | RO | stat_rx_vl_number_2 |
| 16:15 | 0 | RO | stat_rx_vl_number_3 |

STAT_RX_FEC_STATUS_REG: 0448

Table 2-89: STAT_RX_FEC_STATUS_REG: 0448

| Bits | Default | Type | Signal |
|-------|---------|-------|---------------------|
| 3:0 | 0 | RO LL | stat_fec_rx_lock |
| 19:16 | 0 | RO LL | stat_fec_lock_error |

STAT_TX_FLOW_CONTROL_REG1: 0450

Table 2-90: STAT_TX_FLOW_CONTROL_REG1: 0450

| Bits | Default | Type | Signal |
|------|---------|-------|---------------------|
| 8:0 | 0 | RO LH | stat_tx_pause_valid |

STAT_RX_FLOW_CONTROL_REG1: 0454

Table 2-91: STAT_RX_FLOW_CONTROL_REG1: 0454

| Bits | Default | Type | Signal |
|------|---------|-------|---------------------|
| 8:0 | 0 | RO LH | stat_rx_pause_req |
| 17:9 | 0 | RO LH | stat_rx_pause_valid |

STAT_AN_STATUS: 0458

Table 2-92: STAT_AN_STATUS: 0458

| Bits | Default | Type | Signal |
|-------|---------|-------|-----------------------------------|
| 0 | 0 | RO | stat_an_fec_enable |
| 1 | 0 | RO | stat_an_rs_fec_enable |
| 2 | 0 | RO | stat_an_autoneg_complete |
| 3 | 0 | RO | stat_an_parallel_detection_fault |
| 4 | 0 | RO | stat_an_tx_pause_enable |
| 5 | 0 | RO | stat_an_rx_pause_enable |
| 6 | 0 | RO LH | stat_an_lp_ability_valid |
| 7 | 0 | RO | stat_an_lp_autoneg_able |
| 8 | 0 | RO | stat_an_lp_pause |
| 9 | 0 | RO | stat_an_lp_asm_dir |
| 10 | 0 | RO | stat_an_lp_rf |
| 11 | 0 | RO | stat_an_lp_fec_10g_ability |
| 12 | 0 | RO | stat_an_lp_fec_10g_request |
| 13 | 0 | RO LH | stat_an_lp_extended_ability_valid |
| 17:14 | 0 | RO | stat_an_lp_ability_extended_fec |

Table 2-92: STAT_AN_STATUS: 0458 (Cont'd)

| Bits | Default | Type | Signal |
|------|---------|------|----------------------------------|
| 18 | 0 | RO | stat_an_lp_fec_25g_rs_request |
| 19 | 0 | RO | stat_an_lp_fec_25g_baser_request |

STAT_AN_ABILITY: 045C

Table 2-93: STAT_AN_ABILITY: 045C

| Bits | Default | Type | Signal |
|------|---------|------|-----------------------------------|
| 0 | 0 | RO | stat_an_lp_ability_1000base_kx |
| 1 | 0 | RO | stat_an_lp_ability_10gbase_kx4 |
| 2 | 0 | RO | stat_an_lp_ability_10gbase_kr |
| 3 | 0 | RO | stat_an_lp_ability_40gbase_kr4 |
| 4 | 0 | RO | stat_an_lp_ability_40gbase_cr4 |
| 5 | 0 | RO | stat_an_lp_ability_100gbase_cr10 |
| 6 | 0 | RO | stat_an_lp_ability_100gbase_kp4 |
| 7 | 0 | RO | stat_an_lp_ability_100gbase_kr4 |
| 8 | 0 | RO | stat_an_lp_ability_100gbase_cr4 |
| 9 | 0 | RO | stat_an_lp_ability_25gbase_krcr_s |
| 10 | 0 | RO | stat_an_lp_ability_25gbase_krcr |
| 11 | 0 | RO | stat_an_lp_ability_25gbase_kr1 |
| 12 | 0 | RO | stat_an_lp_ability_25gbase_cr1 |
| 13 | 0 | RO | stat_an_lp_ability_50gbase_kr2 |
| 14 | 0 | RO | stat_an_lp_ability_50gbase_cr2 |

STAT_AN_LINK_CTL: 0460

Table 2-94: STAT_AN_LINK_CTL: 0460

| Bits | Default | Type | Signal |
|-------|---------|------|----------------------------------|
| 1:0 | 0 | RO | stat_an_link_cntl_1000base_kx |
| 3:2 | 0 | RO | stat_an_link_cntl_10gbase_kx4 |
| 5:4 | 0 | RO | stat_an_link_cntl_10gbase_kr |
| 7:6 | 0 | RO | stat_an_link_cntl_40gbase_kr4 |
| 9:8 | 0 | RO | stat_an_link_cntl_40gbase_cr4 |
| 11:10 | 0 | RO | stat_an_link_cntl_100gbase_cr10 |
| 13:12 | 0 | RO | stat_an_link_cntl_100gbase_kp4 |
| 15:14 | 0 | RO | stat_an_link_cntl_100gbase_kr4 |
| 17:16 | 0 | RO | stat_an_link_cntl_100gbase_cr4 |
| 19:18 | 0 | RO | stat_an_link_cntl_25gbase_krcr_s |

Table 2-94: STAT_AN_LINK_CTL: 0460 (Cont'd)

| Bits | Default | Type | Signal |
|-------|---------|------|--------------------------------|
| 21:20 | 0 | RO | stat_an_link_cntl_25gbase_krcr |
| 23:22 | 0 | RO | stat_an_link_cntl_25gbase_kr1 |
| 25:24 | 0 | RO | stat_an_link_cntl_25gbase_cr1 |
| 27:26 | 0 | RO | stat_an_link_cntl_50gbase_kr2 |
| 29:28 | 0 | RO | stat_an_link_cntl_50gbase_cr2 |

STAT_LT_STATUS_REG1: 0464

Table 2-95: STAT_LT_STATUS_REG1: 0464

| Bits | Default | Type | Signal |
|-------|---------|------|----------------------------|
| 3:0 | 0 | RO | stat_lt_initialize_from_rx |
| 19:16 | 0 | RO | stat_lt_preset_from_rx |

STAT_LT_STATUS_REG2: 0468

Table 2-96: STAT_LT_STATUS_REG2: 0468

| Bits | Default | Type | Signal |
|-------|---------|------|--------------------|
| 3:0 | 0 | RO | stat_lt_training |
| 19:16 | 0 | RO | stat_lt_frame_lock |

STAT_LT_STATUS_REG3: 046C

Table 2-97: STAT_LT_STATUS_REG3: 046C

| Bits | Default | Type | Signal |
|-------|---------|------|-----------------------|
| 3:0 | 0 | RO | stat_lt_signal_detect |
| 19:16 | 0 | RO | stat_lt_training_fail |

STAT_LT_STATUS_REG4: 0470

Table 2-98: STAT_LT_STATUS_REG4: 0470

| Bits | Default | Type | Signal |
|------|---------|-------|----------------|
| 3:0 | 0 | RO LH | stat_lt_rx_sof |

STAT_LT_COEFFICIENT0_REG: 0474

Table 2-99: STAT_LT_COEFFICIENT0_REG: 0474

| Bits | Default | Type | Signal |
|------|---------|------|-----------------------|
| 1:0 | 0 | RO | stat_lt_k_p1_from_rx0 |
| 3:2 | 0 | RO | stat_lt_k0_from_rx0 |
| 5:4 | 0 | RO | stat_lt_k_m1_from_rx0 |

Table 2-99: STAT_LT_COEFFICIENT0_REG: 0474 (Cont'd)

| Bits | Default | Type | Signal |
|-------|---------|------|--------------------------|
| 7:6 | 0 | RO | stat_lt_stat_p1_from_rx0 |
| 9:8 | 0 | RO | stat_lt_stat0_from_rx0 |
| 11:10 | 0 | RO | stat_lt_stat_m1_from_rx0 |
| 17:16 | 0 | RO | stat_lt_k_p1_from_rx1 |
| 19:18 | 0 | RO | stat_lt_k0_from_rx1 |
| 21:20 | 0 | RO | stat_lt_k_m1_from_rx1 |
| 23:22 | 0 | RO | stat_lt_stat_p1_from_rx1 |
| 25:24 | 0 | RO | stat_lt_stat0_from_rx1 |
| 27:26 | 0 | RO | stat_lt_stat_m1_from_rx1 |

STAT_LT_COEFFICIENT1_REG: 0478

Table 2-100: STAT_LT_COEFFICIENT1_REG: 0478

| Bits | Default | Type | Signal |
|-------|---------|------|--------------------------|
| 1:0 | 0 | RO | stat_lt_k_p1_from_rx2 |
| 3:2 | 0 | RO | stat_lt_k0_from_rx2 |
| 5:4 | 0 | RO | stat_lt_k_m1_from_rx2 |
| 7:6 | 0 | RO | stat_lt_stat_p1_from_rx2 |
| 9:8 | 0 | RO | stat_lt_stat0_from_rx2 |
| 11:10 | 0 | RO | stat_lt_stat_m1_from_rx2 |
| 17:16 | 0 | RO | stat_lt_k_p1_from_rx3 |
| 19:18 | 0 | RO | stat_lt_k0_from_rx3 |
| 21:20 | 0 | RO | stat_lt_k_m1_from_rx3 |
| 23:22 | 0 | RO | stat_lt_stat_p1_from_rx3 |
| 25:24 | 0 | RO | stat_lt_stat0_from_rx3 |
| 27:26 | 0 | RO | stat_lt_stat_m1_from_rx3 |

Statistics Counters

Counters are 48 bits and require two 32-bit address spaces. The following tables provide the first address for each counter corresponding to the 32 LSBs and the next address contains the remaining 16 bits of MSBs in bits [15:0].

The default value for all counters is 0.

Counters are cleared when read by `tick_reg` (or `pm_tick` if so selected), but the register retains its value.

Each counter saturates at FFFFFFFF (hex).

STATUS_CYCLE_COUNT_LSB: 0500

Table 2-101: STATUS_CYCLE_COUNT_LSB: 0500

| Bits | Default | Type | Signal |
|------|---------|---------|------------------------|
| 31:0 | 0 | RO HIST | stat_cycle_count[31:0] |

STATUS_CYCLE_COUNT_MSB: 0504

Table 2-102: STATUS_CYCLE_COUNT_MSB: 0504

| Bits | Default | Type | Signal |
|------|---------|---------|-------------------------|
| 15:0 | 0 | RO HIST | stat_cycle_count[47:32] |

STAT_RX_BIP_ERR_0_LSB: 0508

Table 2-103: STAT_RX_BIP_ERR_0_LSB: 0508

| Bits | Default | Type | Signal |
|------|---------|------|-------------------------------|
| 31:0 | 0 | HIST | stat_rx_bip_err_0_count[31:0] |

STAT_RX_BIP_ERR_0_MSB: 050C

Table 2-104: STAT_RX_BIP_ERR_0_MSB: 050C

| Bits | Default | Type | Signal |
|------|---------|------|----------------------------------|
| 15:0 | 0 | HIST | stat_rx_bip_err_0_count[48-1:32] |

STAT_RX_BIP_ERR_1_LSB: 0510

Table 2-105: STAT_RX_BIP_ERR_1_LSB: 0510

| Bits | Default | Type | Signal |
|------|---------|------|-------------------------------|
| 31:0 | 0 | HIST | stat_rx_bip_err_1_count[31:0] |

STAT_RX_BIP_ERR_1_MSB: 0514

Table 2-106: STAT_RX_BIP_ERR_1_MSB: 0514

| Bits | Default | Type | Signal |
|------|---------|------|----------------------------------|
| 15:0 | 0 | HIST | stat_rx_bip_err_1_count[48-1:32] |

STAT_RX_BIP_ERR_2_LSB: 0518

Table 2-107: STAT_RX_BIP_ERR_2_LSB: 0518

| Bits | Default | Type | Signal |
|------|---------|------|-------------------------------|
| 31:0 | 0 | HIST | stat_rx_bip_err_2_count[31:0] |

STAT_RX_BIP_ERR_2_MSB: 051C

Table 2-108: STAT_RX_BIP_ERR_2_MSB: 051C

| Bits | Default | Type | Signal |
|------|---------|------|----------------------------------|
| 15:0 | 0 | HIST | stat_rx_bip_err_2_count[48-1:32] |

STAT_RX_BIP_ERR_3_LSB: 0520

Table 2-109: STAT_RX_BIP_ERR_3_LSB: 0520

| Bits | Default | Type | Signal |
|------|---------|------|-------------------------------|
| 31:0 | 0 | HIST | stat_rx_bip_err_3_count[31:0] |

STAT_RX_BIP_ERR_3_MSB: 0524

Table 2-110: STAT_RX_BIP_ERR_3_MSB: 0524

| Bits | Default | Type | Signal |
|------|---------|------|----------------------------------|
| 15:0 | 0 | HIST | stat_rx_bip_err_3_count[48-1:32] |

STAT_RX_FRAMING_ERR_0_LSB: 05A8

Table 2-111: STAT_RX_FRAMING_ERR_0_LSB: 05A8

| Bits | Default | Type | Signal |
|------|---------|------|-----------------------------------|
| 31:0 | 0 | HIST | stat_rx_framing_err_0_count[31:0] |

STAT_RX_FRAMING_ERR_0_MSB: 05AC

Table 2-112: STAT_RX_FRAMING_ERR_0_MSB: 05AC

| Bits | Default | Type | Signal |
|------|---------|------|--------------------------------------|
| 15:0 | 0 | HIST | stat_rx_framing_err_0_count[48-1:32] |

STAT_RX_FRAMING_ERR_1_LSB: 05B0

Table 2-113: STAT_RX_FRAMING_ERR_1_LSB: 05B0

| Bits | Default | Type | Signal |
|------|---------|------|-----------------------------------|
| 31:0 | 0 | HIST | stat_rx_framing_err_1_count[31:0] |

STAT_RX_FRAMING_ERR_1_MSB: 05B4

Table 2-114: STAT_RX_FRAMING_ERR_1_MSB: 05B4

| Bits | Default | Type | Signal |
|------|---------|------|--------------------------------------|
| 15:0 | 0 | HIST | stat_rx_framing_err_1_count[48-1:32] |

STAT_RX_FRAMING_ERR_2_LSB: 05B8

Table 2-115: STAT_RX_FRAMING_ERR_2_LSB: 05B8

| Bits | Default | Type | Signal |
|------|---------|------|-----------------------------------|
| 31:0 | 0 | HIST | stat_rx_framing_err_2_count[31:0] |

STAT_RX_FRAMING_ERR_2_MSB: 05BC

Table 2-116: STAT_RX_FRAMING_ERR_2_MSB: 05BC

| Bits | Default | Type | Signal |
|------|---------|------|--------------------------------------|
| 15:0 | 0 | HIST | stat_rx_framing_err_2_count[48-1:32] |

STAT_RX_FRAMING_ERR_3_LSB: 05C0

Table 2-117: STAT_RX_FRAMING_ERR_3_LSB: 05C0

| Bits | Default | Type | Signal |
|------|---------|------|-----------------------------------|
| 31:0 | 0 | HIST | stat_rx_framing_err_3_count[31:0] |

STAT_RX_FRAMING_ERR_3_MSB: 05C4

Table 2-118: STAT_RX_FRAMING_ERR_3_MSB: 05C4

| Bits | Default | Type | Signal |
|------|---------|------|--------------------------------------|
| 15:0 | 0 | HIST | stat_rx_framing_err_3_count[48-1:32] |

STAT_RX_BAD_CODE_LSB: 0660

Table 2-119: STAT_RX_BAD_CODE_LSB: 0660

| Bits | Default | Type | Signal |
|------|---------|------|------------------------------|
| 31:0 | 0 | HIST | stat_rx_bad_code_count[31:0] |

STAT_RX_BAD_CODE_MSB: 0664

Table 2-120: STAT_RX_BAD_CODE_MSB: 0664

| Bits | Default | Type | Signal |
|------|---------|------|---------------------------------|
| 15:0 | 0 | HIST | stat_rx_bad_code_count[48-1:32] |

STAT_RX_ERROR_LSB: 0668

Table 2-121: STAT_RX_ERROR_LSB: 0668

| Bits | Default | Type | Signal |
|------|---------|------|---------------------------|
| 31:0 | 0 | HIST | stat_rx_error_count[31:0] |

STAT_RX_ERROR_MSB: 066C

Table 2-122: STAT_RX_ERROR_MSB: 066C

| Bits | Default | Type | Signal |
|------|---------|------|----------------------------|
| 15:0 | 0 | HIST | stat_rx_error_count[47:32] |

STAT_TX_FRAME_ERROR_LSB: 06A0

Table 2-123: STAT_TX_FRAME_ERROR_LSB: 06A0

| Bits | Default | Type | Signal |
|------|---------|------|---------------------------------|
| 31:0 | 0 | HIST | stat_tx_frame_error_count[31:0] |

STAT_TX_FRAME_ERROR_MSB: 06A4

Table 2-124: STAT_TX_FRAME_ERROR_MSB: 06A4

| Bits | Default | Type | Signal |
|------|---------|------|------------------------------------|
| 15:0 | 0 | HIST | stat_tx_frame_error_count[48-1:32] |

STAT_TX_TOTAL_PACKETS_LSB: 0700

Table 2-125: STAT_TX_TOTAL_PACKETS_LSB: 0700

| Bits | Default | Type | Signal |
|------|---------|------|-----------------------------------|
| 31:0 | 0 | HIST | stat_tx_total_packets_count[31:0] |

STAT_TX_TOTAL_PACKETS_MSB: 0704

Table 2-126: STAT_TX_TOTAL_PACKETS_MSB: 0704

| Bits | Default | Type | Signal |
|------|---------|------|--------------------------------------|
| 15:0 | 0 | HIST | stat_tx_total_packets_count[48-1:32] |

STAT_TX_TOTAL_GOOD_PACKETS_LSB: 0708

Table 2-127: STAT_TX_TOTAL_GOOD_PACKETS_LSB: 0708

| Bits | Default | Type | Signal |
|------|---------|------|--|
| 31:0 | 0 | HIST | stat_tx_total_good_packets_count[31:0] |

STAT_TX_TOTAL_GOOD_PACKETS_MSB: 070C

Table 2-128: STAT_TX_TOTAL_GOOD_PACKETS_MSB: 070C

| Bits | Default | Type | Signal |
|------|---------|------|---|
| 15:0 | 0 | HIST | stat_tx_total_good_packets_count[48-1:32] |

STAT_TX_TOTAL_BYTES_LSB: 0710

Table 2-129: STAT_TX_TOTAL_BYTES_LSB: 0710

| Bits | Default | Type | Signal |
|------|---------|------|---------------------------------|
| 31:0 | 0 | HIST | stat_tx_total_bytes_count[31:0] |

STAT_TX_TOTAL_BYTES_MSB: 0714

Table 2-130: STAT_TX_TOTAL_BYTES_MSB: 0714

| Bits | Default | Type | Signal |
|------|---------|------|------------------------------------|
| 15:0 | 0 | HIST | stat_tx_total_bytes_count[48-1:32] |

STAT_TX_TOTAL_GOOD_BYTES_LSB: 0718

Table 2-131: STAT_TX_TOTAL_GOOD_BYTES_LSB: 0718

| Bits | Default | Type | Signal |
|------|---------|------|--------------------------------------|
| 31:0 | 0 | HIST | stat_tx_total_good_bytes_count[31:0] |

STAT_TX_TOTAL_GOOD_BYTES_MSB: 071C

Table 2-132: STAT_TX_TOTAL_GOOD_BYTES_MSB: 071C

| Bits | Default | Type | Signal |
|------|---------|------|---|
| 15:0 | 0 | HIST | stat_tx_total_good_bytes_count[48-1:32] |

STAT_TX_PACKET_64_BYTES_LSB: 0720

Table 2-133: STAT_TX_PACKET_64_BYTES_LSB: 0720

| Bits | Default | Type | Signal |
|------|---------|------|-------------------------------------|
| 31:0 | 0 | HIST | stat_tx_packet_64_bytes_count[31:0] |

STAT_TX_PACKET_64_BYTES_MSB: 0724

Table 2-134: STAT_TX_PACKET_64_BYTES_MSB: 0724

| Bits | Default | Type | Signal |
|------|---------|------|--|
| 15:0 | 0 | HIST | stat_tx_packet_64_bytes_count[48-1:32] |

STAT_TX_PACKET_65_127_BYTES_LSB: 0728

Table 2-135: STAT_TX_PACKET_65_127_BYTES_LSB: 0728

| Bits | Default | Type | Signal |
|------|---------|------|---|
| 31:0 | 0 | HIST | stat_tx_packet_65_127_bytes_count[31:0] |

STAT_TX_PACKET_65_127_BYTES_MSB: 072C

Table 2-136: STAT_TX_PACKET_65_127_BYTES_MSB: 072C

| Bits | Default | Type | Signal |
|------|---------|------|--|
| 15:0 | 0 | HIST | stat_tx_packet_65_127_bytes_count[48-1:32] |

STAT_TX_PACKET_128_255_BYTES_LSB: 0730

Table 2-137: STAT_TX_PACKET_128_255_BYTES_LSB: 0730

| Bits | Default | Type | Signal |
|------|---------|------|--|
| 31:0 | 0 | HIST | stat_tx_packet_128_255_bytes_count[31:0] |

STAT_TX_PACKET_128_255_BYTES_MSB: 0734

Table 2-138: STAT_TX_PACKET_128_255_BYTES_MSB: 0734

| Bits | Default | Type | Signal |
|------|---------|------|---|
| 15:0 | 0 | HIST | stat_tx_packet_128_255_bytes_count[48-1:32] |

STAT_TX_PACKET_256_511_BYTES_LSB: 0738

Table 2-139: STAT_TX_PACKET_256_511_BYTES_LSB: 0738

| Bits | Default | Type | Signal |
|------|---------|------|--|
| 31:0 | 0 | HIST | stat_tx_packet_256_511_bytes_count[31:0] |

STAT_TX_PACKET_256_511_BYTES_MSB: 073C

Table 2-140: STAT_TX_PACKET_256_511_BYTES_MSB: 073C

| Bits | Default | Type | Signal |
|------|---------|------|---|
| 15:0 | 0 | HIST | stat_tx_packet_256_511_bytes_count[48-1:32] |

STAT_TX_PACKET_512_1023_BYTES_LSB: 0740

Table 2-141: STAT_TX_PACKET_512_1023_BYTES_LSB: 0740

| Bits | Default | Type | Signal |
|------|---------|------|---|
| 31:0 | 0 | HIST | stat_tx_packet_512_1023_bytes_count[31:0] |

STAT_TX_PACKET_512_1023_BYTES_MSB: 0744

Table 2-142: STAT_TX_PACKET_512_1023_BYTES_MSB: 0744

| Bits | Default | Type | Signal |
|------|---------|------|--|
| 15:0 | 0 | HIST | stat_tx_packet_512_1023_bytes_count[48-1:32] |

STAT_TX_PACKET_1024_1518_BYTES_LSB: 0748

Table 2-143: STAT_TX_PACKET_1024_1518_BYTES_LSB: 0748

| Bits | Default | Type | Signal |
|------|---------|------|--|
| 31:0 | 0 | HIST | stat_tx_packet_1024_1518_bytes_count[31:0] |

STAT_TX_PACKET_1024_1518_BYTES_MSB: 074C

Table 2-144: STAT_TX_PACKET_1024_1518_BYTES_MSB: 074C

| Bits | Default | Type | Signal |
|------|---------|------|---|
| 15:0 | 0 | HIST | stat_tx_packet_1024_1518_bytes_count[48-1:32] |

STAT_TX_PACKET_1519_1522_BYTES_LSB: 0750

Table 2-145: STAT_TX_PACKET_1519_1522_BYTES_LSB: 0750

| Bits | Default | Type | Signal |
|------|---------|------|--|
| 31:0 | 0 | HIST | stat_tx_packet_1519_1522_bytes_count[31:0] |

STAT_TX_PACKET_1519_1522_BYTES_MSB: 0754

Table 2-146: STAT_TX_PACKET_1519_1522_BYTES_MSB: 0754

| Bits | Default | Type | Signal |
|------|---------|------|---|
| 15:0 | 0 | HIST | stat_tx_packet_1519_1522_bytes_count[48-1:32] |

STAT_TX_PACKET_1523_1548_BYTES_LSB: 0758

Table 2-147: STAT_TX_PACKET_1523_1548_BYTES_LSB: 0758

| Bits | Default | Type | Signal |
|------|---------|------|--|
| 31:0 | 0 | HIST | stat_tx_packet_1523_1548_bytes_count[31:0] |

STAT_TX_PACKET_1523_1548_BYTES_MSB: 075C

Table 2-148: STAT_TX_PACKET_1523_1548_BYTES_MSB: 075C

| Bits | Default | Type | Signal |
|------|---------|------|---|
| 15:0 | 0 | HIST | stat_tx_packet_1523_1548_bytes_count[48-1:32] |

STAT_TX_PACKET_1549_2047_BYTES_LSB: 0760

Table 2-149: STAT_TX_PACKET_1549_2047_BYTES_LSB: 0760

| Bits | Default | Type | Signal |
|------|---------|------|--|
| 31:0 | 0 | HIST | stat_tx_packet_1549_2047_bytes_count[31:0] |

STAT_TX_PACKET_1549_2047_BYTES_MSB: 0764

Table 2-150: STAT_TX_PACKET_1549_2047_BYTES_MSB: 0764

| Bits | Default | Type | Signal |
|------|---------|------|---|
| 15:0 | 0 | HIST | stat_tx_packet_1549_2047_bytes_count[48-1:32] |

STAT_TX_PACKET_2048_4095_BYTES_LSB: 0768

Table 2-151: STAT_TX_PACKET_2048_4095_BYTES_LSB: 0768

| Bits | Default | Type | Signal |
|------|---------|------|--|
| 31:0 | 0 | HIST | stat_tx_packet_2048_4095_bytes_count[31:0] |

STAT_TX_PACKET_2048_4095_BYTES_MSB: 076C

Table 2-152: STAT_TX_PACKET_2048_4095_BYTES_MSB: 076C

| Bits | Default | Type | Signal |
|------|---------|------|---|
| 15:0 | 0 | HIST | stat_tx_packet_2048_4095_bytes_count[48-1:32] |

STAT_TX_PACKET_4096_8191_BYTES_LSB: 0770

Table 2-153: STAT_TX_PACKET_4096_8191_BYTES_LSB: 0770

| Bits | Default | Type | Signal |
|------|---------|------|--|
| 31:0 | 0 | HIST | stat_tx_packet_4096_8191_bytes_count[31:0] |

STAT_TX_PACKET_4096_8191_BYTES_MSB: 0774

Table 2-154: STAT_TX_PACKET_4096_8191_BYTES_MSB: 0774

| Bits | Default | Type | Signal |
|------|---------|------|---|
| 15:0 | 0 | HIST | stat_tx_packet_4096_8191_bytes_count[48-1:32] |

STAT_TX_PACKET_8192_9215_BYTES_LSB: 0778

Table 2-155: STAT_TX_PACKET_8192_9215_BYTES_LSB: 0778

| Bits | Default | Type | Signal |
|------|---------|------|--|
| 31:0 | 0 | HIST | stat_tx_packet_8192_9215_bytes_count[31:0] |

STAT_TX_PACKET_8192_9215_BYTES_MSB: 077C

Table 2-156: STAT_TX_PACKET_8192_9215_BYTES_MSB: 077C

| Bits | Default | Type | Signal |
|------|---------|------|---|
| 15:0 | 0 | HIST | stat_tx_packet_8192_9215_bytes_count[48-1:32] |

STAT_TX_PACKET_LARGE_LSB: 0780

Table 2-157: STAT_TX_PACKET_LARGE_LSB: 0780

| Bits | Default | Type | Signal |
|------|---------|------|----------------------------------|
| 31:0 | 0 | HIST | stat_tx_packet_large_count[31:0] |

STAT_TX_PACKET_LARGE_MSB: 0784

Table 2-158: STAT_TX_PACKET_LARGE_MSB: 0784

| Bits | Default | Type | Signal |
|------|---------|------|-------------------------------------|
| 15:0 | 0 | HIST | stat_tx_packet_large_count[48-1:32] |

STAT_TX_PACKET_SMALL_LSB: 0788

Table 2-159: STAT_TX_PACKET_SMALL_LSB: 0788

| Bits | Default | Type | Signal |
|------|---------|------|----------------------------------|
| 31:0 | 0 | HIST | stat_tx_packet_small_count[31:0] |

STAT_TX_PACKET_SMALL_MSB: 078C

Table 2-160: STAT_TX_PACKET_SMALL_MSB: 078C

| Bits | Default | Type | Signal |
|------|---------|------|-------------------------------------|
| 15:0 | 0 | HIST | stat_tx_packet_small_count[48-1:32] |

STAT_TX_BAD_FCS_LSB: 07B8

Table 2-161: STAT_TX_BAD_FCS_LSB: 07B8

| Bits | Default | Type | Signal |
|------|---------|------|-----------------------------|
| 31:0 | 0 | HIST | stat_tx_bad_fcs_count[31:0] |

STAT_TX_BAD_FCS_MSB: 07BC

Table 2-162: STAT_TX_BAD_FCS_MSB: 07BC

| Bits | Default | Type | Signal |
|------|---------|------|--------------------------------|
| 15:0 | 0 | HIST | stat_tx_bad_fcs_count[48-1:32] |

STAT_TX_UNICAST_LSB: 07D0

Table 2-163: STAT_TX_UNICAST_LSB: 07D0

| Bits | Default | Type | Signal |
|------|---------|------|-----------------------------|
| 31:0 | 0 | HIST | stat_tx_unicast_count[31:0] |

STAT_TX_UNICAST_MSB: 07D4

Table 2-164: STAT_TX_UNICAST_MSB: 07D4

| Bits | Default | Type | Signal |
|------|---------|------|--------------------------------|
| 15:0 | 0 | HIST | stat_tx_unicast_count[48-1:32] |

STAT_TX_MULTICAST_LSB: 07D8

Table 2-165: STAT_TX_MULTICAST_LSB: 07D8

| Bits | Default | Type | Signal |
|------|---------|------|-------------------------------|
| 31:0 | 0 | HIST | stat_tx_multicast_count[31:0] |

STAT_TX_MULTICAST_MSB: 07DC

Table 2-166: STAT_TX_MULTICAST_MSB: 07DC

| Bits | Default | Type | Signal |
|------|---------|------|----------------------------------|
| 15:0 | 0 | HIST | stat_tx_multicast_count[48-1:32] |

STAT_TX_BROADCAST_LSB: 07E0

Table 2-167: STAT_TX_BROADCAST_LSB: 07E0

| Bits | Default | Type | Signal |
|------|---------|------|-------------------------------|
| 31:0 | 0 | HIST | stat_tx_broadcast_count[31:0] |

STAT_TX_BROADCAST_MSB: 07E4

Table 2-168: STAT_TX_BROADCAST_MSB: 07E4

| Bits | Default | Type | Signal |
|------|---------|------|----------------------------------|
| 15:0 | 0 | HIST | stat_tx_broadcast_count[48-1:32] |

STAT_TX_VLAN_LSB: 07E8

Table 2-169: STAT_TX_VLAN_LSB: 07E8

| Bits | Default | Type | Signal |
|------|---------|------|--------------------------|
| 31:0 | 0 | HIST | stat_tx_vlan_count[31:0] |

STAT_TX_VLAN_MSB: 07EC

Table 2-170: STAT_TX_VLAN_MSB: 07EC

| Bits | Default | Type | Signal |
|------|---------|------|-----------------------------|
| 15:0 | 0 | HIST | stat_tx_vlan_count[48-1:32] |

STAT_TX_PAUSE_LSB: 07F0

Table 2-171: STAT_TX_PAUSE_LSB: 07F0

| Bits | Default | Type | Signal |
|------|---------|------|---------------------------|
| 31:0 | 0 | HIST | stat_tx_pause_count[31:0] |

STAT_TX_PAUSE_MSB: 07F4

Table 2-172: STAT_TX_PAUSE_MSB: 07F4

| Bits | Default | Type | Signal |
|------|---------|------|------------------------------|
| 15:0 | 0 | HIST | stat_tx_pause_count[48-1:32] |

STAT_TX_USER_PAUSE_LSB: 07F8

Table 2-173: STAT_TX_USER_PAUSE_LSB: 07F8

| Bits | Default | Type | Signal |
|------|---------|------|--------------------------------|
| 31:0 | 0 | HIST | stat_tx_user_pause_count[31:0] |

STAT_TX_USER_PAUSE_MSB: 07FC

Table 2-174: STAT_TX_USER_PAUSE_MSB: 07FC

| Bits | Default | Type | Signal |
|------|---------|------|-----------------------------------|
| 15:0 | 0 | HIST | stat_tx_user_pause_count[48-1:32] |

STAT_RX_TOTAL_PACKETS_LSB: 0808

Table 2-175: STAT_RX_TOTAL_PACKETS_LSB: 0808

| Bits | Default | Type | Signal |
|------|---------|------|-----------------------------------|
| 31:0 | 0 | HIST | stat_rx_total_packets_count[31:0] |

STAT_RX_TOTAL_PACKETS_MSB: 080C

Table 2-176: STAT_RX_TOTAL_PACKETS_MSB: 080C

| Bits | Default | Type | Signal |
|------|---------|------|--------------------------------------|
| 15:0 | 0 | HIST | stat_rx_total_packets_count[48-1:32] |

STAT_RX_TOTAL_GOOD_PACKETS_LSB: 0810

Table 2-177: STAT_RX_TOTAL_GOOD_PACKETS_LSB: 0810

| Bits | Default | Type | Signal |
|------|---------|------|--|
| 31:0 | 0 | HIST | stat_rx_total_good_packets_count[31:0] |

STAT_RX_TOTAL_GOOD_PACKETS_MSB: 0814

Table 2-178: STAT_RX_TOTAL_GOOD_PACKETS_MSB: 0814

| Bits | Default | Type | Signal |
|------|---------|------|---|
| 15:0 | 0 | HIST | stat_rx_total_good_packets_count[48-1:32] |

STAT_RX_TOTAL_BYTES_LSB: 0818

Table 2-179: STAT_RX_TOTAL_BYTES_LSB: 0818

| Bits | Default | Type | Signal |
|------|---------|------|---------------------------------|
| 31:0 | 0 | HIST | stat_rx_total_bytes_count[31:0] |

STAT_RX_TOTAL_BYTES_MSB: 081C

Table 2-180: STAT_RX_TOTAL_BYTES_MSB: 081C

| Bits | Default | Type | Signal |
|------|---------|------|------------------------------------|
| 15:0 | 0 | HIST | stat_rx_total_bytes_count[48-1:32] |

STAT_RX_TOTAL_GOOD_BYTES_LSB: 0820

Table 2-181: STAT_RX_TOTAL_GOOD_BYTES_LSB: 0820

| Bits | Default | Type | Signal |
|------|---------|------|--------------------------------------|
| 31:0 | 0 | HIST | stat_rx_total_good_bytes_count[31:0] |

STAT_RX_TOTAL_GOOD_BYTES_MSB: 0824

Table 2-182: STAT_RX_TOTAL_GOOD_BYTES_MSB: 0824

| Bits | Default | Type | Signal |
|------|---------|------|---|
| 15:0 | 0 | HIST | stat_rx_total_good_bytes_count[48-1:32] |

STAT_RX_PACKET_64_BYTES_LSB: 0828

Table 2-183: STAT_RX_PACKET_64_BYTES_LSB: 0828

| Bits | Default | Type | Signal |
|------|---------|------|-------------------------------------|
| 31:0 | 0 | HIST | stat_rx_packet_64_bytes_count[31:0] |

STAT_RX_PACKET_64_BYTES_MSB: 082C

Table 2-184: STAT_RX_PACKET_64_BYTES_MSB: 082C

| Bits | Default | Type | Signal |
|------|---------|------|--|
| 15:0 | 0 | HIST | stat_rx_packet_64_bytes_count[48-1:32] |

STAT_RX_PACKET_65_127_BYTES_LSB: 0830

Table 2-185: STAT_RX_PACKET_65_127_BYTES_LSB: 0830

| Bits | Default | Type | Signal |
|------|---------|------|---|
| 31:0 | 0 | HIST | stat_rx_packet_65_127_bytes_count[31:0] |

STAT_RX_PACKET_65_127_BYTES_MSB: 0834

Table 2-186: STAT_RX_PACKET_65_127_BYTES_MSB: 0834

| Bits | Default | Type | Signal |
|------|---------|------|--|
| 15:0 | 0 | HIST | stat_rx_packet_65_127_bytes_count[48-1:32] |

STAT_RX_PACKET_128_255_BYTES_LSB: 0838

Table 2-187: STAT_RX_PACKET_128_255_BYTES_LSB: 0838

| Bits | Default | Type | Signal |
|------|---------|------|--|
| 31:0 | 0 | HIST | stat_rx_packet_128_255_bytes_count[31:0] |

STAT_RX_PACKET_128_255_BYTES_MSB: 083C

Table 2-188: STAT_RX_PACKET_128_255_BYTES_MSB: 083C

| Bits | Default | Type | Signal |
|------|---------|------|---|
| 15:0 | 0 | HIST | stat_rx_packet_128_255_bytes_count[48-1:32] |

STAT_RX_PACKET_256_511_BYTES_LSB: 0840

Table 2-189: STAT_RX_PACKET_256_511_BYTES_LSB: 0840

| Bits | Default | Type | Signal |
|------|---------|------|--|
| 31:0 | 0 | HIST | stat_rx_packet_256_511_bytes_count[31:0] |

STAT_RX_PACKET_256_511_BYTES_MSB: 0844

Table 2-190: STAT_RX_PACKET_256_511_BYTES_MSB: 0844

| Bits | Default | Type | Signal |
|------|---------|------|---|
| 15:0 | 0 | HIST | stat_rx_packet_256_511_bytes_count[48-1:32] |

STAT_RX_PACKET_512_1023_BYTES_LSB: 0848

Table 2-191: STAT_RX_PACKET_512_1023_BYTES_LSB: 0848

| Bits | Default | Type | Signal |
|------|---------|------|---|
| 31:0 | 0 | HIST | stat_rx_packet_512_1023_bytes_count[31:0] |

STAT_RX_PACKET_512_1023_BYTES_MSB: 084C

Table 2-192: STAT_RX_PACKET_512_1023_BYTES_MSB: 084C

| Bits | Default | Type | Signal |
|------|---------|------|--|
| 15:0 | 0 | HIST | stat_rx_packet_512_1023_bytes_count[48-1:32] |

STAT_RX_PACKET_1024_1518_BYTES_LSB: 0850

Table 2-193: STAT_RX_PACKET_1024_1518_BYTES_LSB: 0850

| Bits | Default | Type | Signal |
|------|---------|------|--|
| 31:0 | 0 | HIST | stat_rx_packet_1024_1518_bytes_count[31:0] |

STAT_RX_PACKET_1024_1518_BYTES_MSB: 0854

Table 2-194: STAT_RX_PACKET_1024_1518_BYTES_MSB: 0854

| Bits | Default | Type | Signal |
|------|---------|------|---|
| 15:0 | 0 | HIST | stat_rx_packet_1024_1518_bytes_count[48-1:32] |

STAT_RX_PACKET_1519_1522_BYTES_LSB: 0858

Table 2-195: STAT_RX_PACKET_1519_1522_BYTES_LSB: 0858

| Bits | Default | Type | Signal |
|------|---------|------|--|
| 31:0 | 0 | HIST | stat_rx_packet_1519_1522_bytes_count[31:0] |

STAT_RX_PACKET_1519_1522_BYTES_MSB: 085C

Table 2-196: STAT_RX_PACKET_1519_1522_BYTES_MSB: 085C

| Bits | Default | Type | Signal |
|------|---------|------|---|
| 15:0 | 0 | HIST | stat_rx_packet_1519_1522_bytes_count[48-1:32] |

STAT_RX_PACKET_1523_1548_BYTES_LSB: 0860

Table 2-197: STAT_RX_PACKET_1523_1548_BYTES_LSB: 0860

| Bits | Default | Type | Signal |
|------|---------|------|--|
| 31:0 | 0 | HIST | stat_rx_packet_1523_1548_bytes_count[31:0] |

STAT_RX_PACKET_1523_1548_BYTES_MSB: 0864

Table 2-198: STAT_RX_PACKET_1523_1548_BYTES_MSB: 0864

| Bits | Default | Type | Signal |
|------|---------|------|---|
| 15:0 | 0 | HIST | stat_rx_packet_1523_1548_bytes_count[48-1:32] |

STAT_RX_PACKET_1549_2047_BYTES_LSB: 0868

Table 2-199: STAT_RX_PACKET_1549_2047_BYTES_LSB: 0868

| Bits | Default | Type | Signal |
|------|---------|------|--|
| 31:0 | 0 | HIST | stat_rx_packet_1549_2047_bytes_count[31:0] |

STAT_RX_PACKET_1549_2047_BYTES_MSB: 086C

Table 2-200: STAT_RX_PACKET_1549_2047_BYTES_MSB: 086C

| Bits | Default | Type | Signal |
|------|---------|------|---|
| 15:0 | 0 | HIST | stat_rx_packet_1549_2047_bytes_count[48-1:32] |

STAT_RX_PACKET_2048_4095_BYTES_LSB: 0870

Table 2-201: STAT_RX_PACKET_2048_4095_BYTES_LSB: 0870

| Bits | Default | Type | Signal |
|------|---------|------|--|
| 31:0 | 0 | HIST | stat_rx_packet_2048_4095_bytes_count[31:0] |

STAT_RX_PACKET_2048_4095_BYTES_MSB: 0874

Table 2-202: STAT_RX_PACKET_2048_4095_BYTES_MSB: 0874

| Bits | Default | Type | Signal |
|------|---------|------|---|
| 15:0 | 0 | HIST | stat_rx_packet_2048_4095_bytes_count[48-1:32] |

STAT_RX_PACKET_4096_8191_BYTES_LSB: 0878

Table 2-203: STAT_RX_PACKET_4096_8191_BYTES_LSB: 0878

| Bits | Default | Type | Signal |
|------|---------|------|--|
| 31:0 | 0 | HIST | stat_rx_packet_4096_8191_bytes_count[31:0] |

STAT_RX_PACKET_4096_8191_BYTES_MSB: 087C

Table 2-204: STAT_RX_PACKET_4096_8191_BYTES_MSB: 087C

| Bits | Default | Type | Signal |
|------|---------|------|---|
| 15:0 | 0 | HIST | stat_rx_packet_4096_8191_bytes_count[48-1:32] |

STAT_RX_PACKET_8192_9215_BYTES_LSB: 0880

Table 2-205: STAT_RX_PACKET_8192_9215_BYTES_LSB: 0880

| Bits | Default | Type | Signal |
|------|---------|------|--|
| 31:0 | 0 | HIST | stat_rx_packet_8192_9215_bytes_count[31:0] |

STAT_RX_PACKET_8192_9215_BYTES_MSB: 0884

Table 2-206: STAT_RX_PACKET_8192_9215_BYTES_MSB: 0884

| Bits | Default | Type | Signal |
|------|---------|------|---|
| 15:0 | 0 | HIST | stat_rx_packet_8192_9215_bytes_count[48-1:32] |

STAT_RX_PACKET_LARGE_LSB: 0888

Table 2-207: STAT_RX_PACKET_LARGE_LSB: 0888

| Bits | Default | Type | Signal |
|------|---------|------|----------------------------------|
| 31:0 | 0 | HIST | stat_rx_packet_large_count[31:0] |

STAT_RX_PACKET_LARGE_MSB: 088C

Table 2-208: STAT_RX_PACKET_LARGE_MSB: 088C

| Bits | Default | Type | Signal |
|------|---------|------|-------------------------------------|
| 15:0 | 0 | HIST | stat_rx_packet_large_count[48-1:32] |

STAT_RX_PACKET_SMALL_LSB: 0890

Table 2-209: STAT_RX_PACKET_SMALL_LSB: 0890

| Bits | Default | Type | Signal |
|------|---------|------|----------------------------------|
| 31:0 | 0 | HIST | stat_rx_packet_small_count[31:0] |

STAT_RX_PACKET_SMALL_MSB: 0894

Table 2-210: STAT_RX_PACKET_SMALL_MSB: 0894

| Bits | Default | Type | Signal |
|------|---------|------|-------------------------------------|
| 15:0 | 0 | HIST | stat_rx_packet_small_count[48-1:32] |

STAT_RX_UNDERSIZE_LSB: 0898

Table 2-211: STAT_RX_UNDERSIZE_LSB: 0898

| Bits | Default | Type | Signal |
|------|---------|------|-------------------------------|
| 31:0 | 0 | HIST | stat_rx_undersize_count[31:0] |

STAT_RX_UNDERSIZE_MSB: 089C

Table 2-212: STAT_RX_UNDERSIZE_MSB: 089C

| Bits | Default | Type | Signal |
|------|---------|------|----------------------------------|
| 15:0 | 0 | HIST | stat_rx_undersize_count[48-1:32] |

STAT_RX_FRAGMENT_LSB: 08A0

Table 2-213: STAT_RX_FRAGMENT_LSB: 08A0

| Bits | Default | Type | Signal |
|------|---------|------|------------------------------|
| 31:0 | 0 | HIST | stat_rx_fragment_count[31:0] |

STAT_RX_FRAGMENT_MSB: 08A4

Table 2-214: STAT_RX_FRAGMENT_MSB: 08A4

| Bits | Default | Type | Signal |
|------|---------|------|---------------------------------|
| 15:0 | 0 | HIST | stat_rx_fragment_count[48-1:32] |

STAT_RX_OVERSIZE_LSB: 08A8

Table 2-215: STAT_RX_OVERSIZE_LSB: 08A8

| Bits | Default | Type | Signal |
|------|---------|------|------------------------------|
| 31:0 | 0 | HIST | stat_rx_oversize_count[31:0] |

STAT_RX_OVERSIZE_MSB: 08AC

Table 2-216: STAT_RX_OVERSIZE_MSB: 08AC

| Bits | Default | Type | Signal |
|------|---------|------|---------------------------------|
| 15:0 | 0 | HIST | stat_rx_oversize_count[48-1:32] |

STAT_RX_TOOLONG_LSB: 08B0

Table 2-217: STAT_RX_TOOLONG_LSB: 08B0

| Bits | Default | Type | Signal |
|------|---------|------|-----------------------------|
| 31:0 | 0 | HIST | stat_rx_toolong_count[31:0] |

STAT_RX_TOOLONG_MSB: 08B4

Table 2-218: STAT_RX_TOOLONG_MSB: 08B4

| Bits | Default | Type | Signal |
|------|---------|------|--------------------------------|
| 15:0 | 0 | HIST | stat_rx_toolong_count[48-1:32] |

STAT_RX_JABBER_LSB: 08B8

Table 2-219: STAT_RX_JABBER_LSB: 08B8

| Bits | Default | Type | Signal |
|------|---------|------|----------------------------|
| 31:0 | 0 | HIST | stat_rx_jabber_count[31:0] |

STAT_RX_JABBER_MSB: 08BC

Table 2-220: STAT_RX_JABBER_MSB: 08BC

| Bits | Default | Type | Signal |
|------|---------|------|-------------------------------|
| 15:0 | 0 | HIST | stat_rx_jabber_count[48-1:32] |

STAT_RX_BAD_FCS_LSB: 08C0

Table 2-221: STAT_RX_BAD_FCS_LSB: 08C0

| Bits | Default | Type | Signal |
|------|---------|------|-----------------------------|
| 31:0 | 0 | HIST | stat_rx_bad_fcs_count[31:0] |

STAT_RX_BAD_FCS_MSB: 08C4

Table 2-222: STAT_RX_BAD_FCS_MSB: 08C4

| Bits | Default | Type | Signal |
|------|---------|------|--------------------------------|
| 15:0 | 0 | HIST | stat_rx_bad_fcs_count[48-1:32] |

STAT_RX_PACKET_BAD_FCS_LSB: 08C8

Table 2-223: STAT_RX_PACKET_BAD_FCS_LSB: 08C8

| Bits | Default | Type | Signal |
|------|---------|------|------------------------------------|
| 31:0 | 0 | HIST | stat_rx_packet_bad_fcs_count[31:0] |

STAT_RX_PACKET_BAD_FCS_MSB: 08CC

Table 2-224: STAT_RX_PACKET_BAD_FCS_MSB: 08CC

| Bits | Default | Type | Signal |
|------|---------|------|---------------------------------------|
| 15:0 | 0 | HIST | stat_rx_packet_bad_fcs_count[48-1:32] |

STAT_RX_STOMPED_FCS_LSB: 08D0

Table 2-225: STAT_RX_STOMPED_FCS_LSB: 08D0

| Bits | Default | Type | Signal |
|------|---------|------|---------------------------------|
| 31:0 | 0 | HIST | stat_rx_stomped_fcs_count[31:0] |

STAT_RX_STOMPED_FCS_MSB: 08D4

Table 2-226: STAT_RX_STOMPED_FCS_MSB: 08D4

| Bits | Default | Type | Signal |
|------|---------|------|------------------------------------|
| 15:0 | 0 | HIST | stat_rx_stomped_fcs_count[48-1:32] |

STAT_RX_UNICAST_LSB: 08D8

Table 2-227: STAT_RX_UNICAST_LSB: 08D8

| Bits | Default | Type | Signal |
|------|---------|------|-----------------------------|
| 31:0 | 0 | HIST | stat_rx_unicast_count[31:0] |

STAT_RX_UNICAST_MSB: 08DC

Table 2-228: STAT_RX_UNICAST_MSB: 08DC

| Bits | Default | Type | Signal |
|------|---------|------|--------------------------------|
| 15:0 | 0 | HIST | stat_rx_unicast_count[48-1:32] |

STAT_RX_MULTICAST_LSB: 08E0

Table 2-229: STAT_RX_MULTICAST_LSB: 08E0

| Bits | Default | Type | Signal |
|------|---------|------|-------------------------------|
| 31:0 | 0 | HIST | stat_rx_multicast_count[31:0] |

STAT_RX_MULTICAST_MSB: 08E4

Table 2-230: STAT_RX_MULTICAST_MSB: 08E4

| Bits | Default | Type | Signal |
|------|---------|------|----------------------------------|
| 15:0 | 0 | HIST | stat_rx_multicast_count[48-1:32] |

STAT_RX_BROADCAST_LSB: 08E8

Table 2-231: STAT_RX_BROADCAST_LSB: 08E8

| Bits | Default | Type | Signal |
|------|---------|------|-------------------------------|
| 31:0 | 0 | HIST | stat_rx_broadcast_count[31:0] |

STAT_RX_BROADCAST_MSB: 08EC

Table 2-232: STAT_RX_BROADCAST_MSB: 08EC

| Bits | Default | Type | Signal |
|------|---------|------|----------------------------------|
| 15:0 | 0 | HIST | stat_rx_broadcast_count[48-1:32] |

STAT_RX_VLAN_LSB: 08F0

Table 2-233: STAT_RX_VLAN_LSB: 08F0

| Bits | Default | Type | Signal |
|------|---------|------|--------------------------|
| 31:0 | 0 | HIST | stat_rx_vlan_count[31:0] |

STAT_RX_VLAN_MSB: 08F4

Table 2-234: STAT_RX_VLAN_MSB: 08F4

| Bits | Default | Type | Signal |
|------|---------|------|-----------------------------|
| 15:0 | 0 | HIST | stat_rx_vlan_count[48-1:32] |

STAT_RX_PAUSE_LSB: 08F8

Table 2-235: STAT_RX_PAUSE_LSB: 08F8

| Bits | Default | Type | Signal |
|------|---------|------|---------------------------|
| 31:0 | 0 | HIST | stat_rx_pause_count[31:0] |

STAT_RX_PAUSE_MSB: 08FC

Table 2-236: STAT_RX_PAUSE_MSB: 08FC

| Bits | Default | Type | Signal |
|------|---------|------|------------------------------|
| 15:0 | 0 | HIST | stat_rx_pause_count[48-1:32] |

STAT_RX_USER_PAUSE_LSB: 0900

Table 2-237: STAT_RX_USER_PAUSE_LSB: 0900

| Bits | Default | Type | Signal |
|------|---------|------|--------------------------------|
| 31:0 | 0 | HIST | stat_rx_user_pause_count[31:0] |

STAT_RX_USER_PAUSE_MSB: 0904

Table 2-238: STAT_RX_USER_PAUSE_MSB: 0904

| Bits | Default | Type | Signal |
|------|---------|------|-----------------------------------|
| 15:0 | 0 | HIST | stat_rx_user_pause_count[48-1:32] |

STAT_RX_INRANGEERR_LSB: 0908

Table 2-239: STAT_RX_INRANGEERR_LSB: 0908

| Bits | Default | Type | Signal |
|------|---------|------|--------------------------------|
| 31:0 | 0 | HIST | stat_rx_inrangeerr_count[31:0] |

STAT_RX_INRANGEERR_MSB: 090C

Table 2-240: STAT_RX_INRANGEERR_MSB: 090C

| Bits | Default | Type | Signal |
|------|---------|------|-----------------------------------|
| 15:0 | 0 | HIST | stat_rx_inrangeerr_count[48-1:32] |

STAT_RX_TRUNCATED_LSB: 0910

Table 2-241: STAT_RX_TRUNCATED_LSB: 0910

| Bits | Default | Type | Signal |
|------|---------|------|-------------------------------|
| 31:0 | 0 | HIST | stat_rx_truncated_count[31:0] |

STAT_RX_TRUNCATED_MSB: 0914

Table 2-242: STAT_RX_TRUNCATED_MSB: 0914

| Bits | Default | Type | Signal |
|------|---------|------|----------------------------------|
| 15:0 | 0 | HIST | stat_rx_truncated_count[48-1:32] |

STAT_RX_TEST_PATTERN_MISMATCH_LSB: 0918

Table 2-243: STAT_RX_TEST_PATTERN_MISMATCH_LSB: 0918

| Bits | Default | Type | Signal |
|------|---------|------|---|
| 31:0 | 0 | HIST | stat_rx_test_pattern_mismatch_count[31:0] |

STAT_RX_TEST_PATTERN_MISMATCH_MSB: 091C

Table 2-244: STAT_RX_TEST_PATTERN_MISMATCH_MSB: 091C

| Bits | Default | Type | Signal |
|------|---------|------|--|
| 15:0 | 0 | HIST | stat_rx_test_pattern_mismatch_count[48-1:32] |

STAT_FEC_INC_CORRECT_COUNT_LSB: 0920

Table 2-245: STAT_FEC_INC_CORRECT_COUNT_LSB: 0920

| Bits | Default | Type | Signal |
|------|---------|------|--|
| 31:0 | 0 | HIST | stat_fec_inc_correct_count_count[31:0] |

STAT_FEC_INC_CORRECT_COUNT_MSB: 0924

Table 2-246: STAT_FEC_INC_CORRECT_COUNT_MSB: 0924

| Bits | Default | Type | Signal |
|------|---------|------|---|
| 15:0 | 0 | HIST | stat_fec_inc_correct_count_count[48-1:32] |

STAT_FEC_INC_CANT_CORRECT_COUNT_LSB: 0928

Table 2-247: STAT_FEC_INC_CANT_CORRECT_COUNT_LSB: 0928

| Bits | Default | Type | Signal |
|------|---------|------|---|
| 31:0 | 0 | HIST | stat_fec_inc_cant_correct_count_count[31:0] |

STAT_FEC_INC_CANT_CORRECT_COUNT_MSB: 092C

Table 2-248: STAT_FEC_INC_CANT_CORRECT_COUNT_MSB: 092C

| Bits | Default | Type | Signal |
|------|---------|------|--|
| 15:0 | 0 | HIST | stat_fec_inc_cant_correct_count_count[48-1:32] |

Designing with the Core

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

General Design Guidelines

Use the Example Design as a Starting Point

Each release is delivered as a complete reference design that includes a sample test bench for simulation. Transceivers are included, targeted to the particular Xilinx device requested for that release. In most cases, you need to re-assign the transceivers according to the device pinout specific to your board layout. You might also wish to generate new custom transceivers using the Vivado® Design Suite with characteristics suited to your board.

Know the Degree of Difficulty

Xilinx 40G/50G High Speed Ethernet Subsystem designs are challenging to implement in any technology, and the degree of difficulty is further influenced by:

- Maximum system clock frequency
- Targeted device architecture
- Nature of your application

All 40G/50G High Speed Ethernet Subsystem implementations need careful attention to system performance requirements. Pipelining, logic mapping, placement constraints, and logic duplication are all methods that help boost system performance.

Keep it Registered

To simplify timing and increase system performance in an FPGA design, keep all inputs and outputs registered between your application and the core. This means that all inputs and outputs from your application should come from, or connect to a flip-flop. While registering signals cannot be possible for all paths, it simplifies timing analysis and makes it easier for the Vivado Design Suite to place and route the design.

Recognize Timing Critical Signals

The timing constraints file that is provided with the example design for the core identifies the critical signals and the timing constraints that should be applied.

Make Only Allowed Modifications

The 40G/50G High Speed Ethernet Subsystem is not user-modifiable. Do not make any modifications because these modifications can have adverse effects on system timing and protocol functionality. You can submit supported user configurations of the 40G/50G High Speed Ethernet Subsystem to Xilinx Technical Support for implementation.

You are encouraged to modify the transceivers included with the example design. Use the latest GT Wizard which is part of the Vivado Design Suite. Some features that might need to be customized are the transceiver placement, reference clocks, and optional ports, among others.

Clocking

This section describes the clocking for all the 40G/50G configurations at the component support wrapper layer. There are three fundamentally different clocking architectures depending on the functionality and options:

- [PCS/PMA Only Clocking](#)
- [40G/50G MAC with PCS/PMA Clocking](#)
- [Low Latency 40G/50G MAC with PCS/PMA Clocking](#)

Also described is [Auto-Negotiation and Link Training Clocking](#).

PCS/PMA Only Clocking

The clocking architecture for the 40G/50G PCS is illustrated below. There are three clock domains in the datapath, as illustrated by the dashed lines in [Figure 3-1](#).

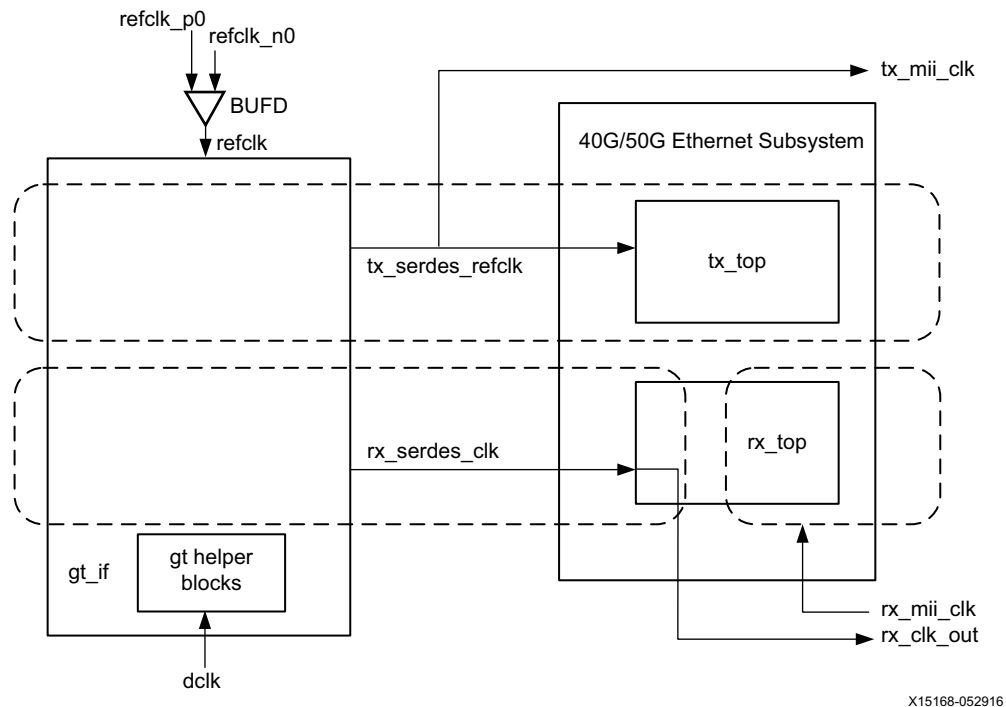


Figure 3-1: PCS/PMA Clocking

refclk_p0, refclk_n0, tx_serdes_refclk

The `refclk` differential pair is required to be an input to the FPGA. The example design includes a buffer to convert this clock to a single-ended signal `refclk`, which is used as the reference clock for the GT block. The `tx_serdes_refclk` is directly derived from `refclk`. Note that `refclk` must be chosen so that the `tx_mii_clk` meets the requirements of 802.3, which is within 100 ppm of 312.5 MHz for 40G and 390.625 MHz for 50G.

tx_mii_clk

The `tx_mii_clk` is an output which is the same as the `tx_serdes_refclk`. The entire TX path is driven by this clock. You must synchronize the TX path `mii` bus to this clock output. All TX control and status signals are referenced to this clock.

rx_serdes_clk

The `rx_serdes_clk` is derived from the incoming data stream within the GT block. The incoming data stream is processed by the RX core in this clock domain.

rx_clk_out

The `rx_clk_out` output signal is presented as a reference for the RX control and status signals processed by the RX core. It is the same frequency as the `rx_serdes_clk`.

rx_mii_clk

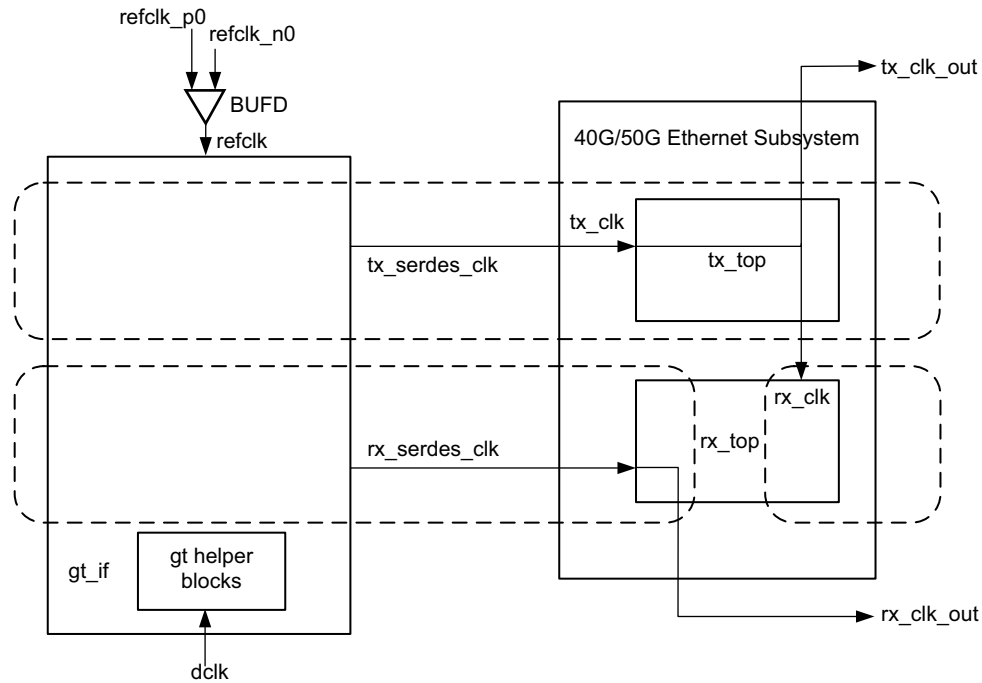
The `rx_mii_clk` input is required to be synchronized to the RX LGMII data bus. This clock and the RX LGMII bus must be within 100 ppm of the required frequency, which is 312.5 MHz for 40G and 390.625 MHz for 50G.

dclk

The `dclk` signal must be a convenient stable clock. It is used as a reference frequency for the GT helper blocks which initiate the GT itself. In the example design, a typical value is 75 MHz, which is readily derived from the 300 MHz clock available on the VCU107 evaluation board. Note that the actual frequency must be known to the GT helper blocks for proper operation.

40G/50G MAC with PCS/PMA Clocking

The clocking architecture for the 40/50G MAC with PCS/PMA clocking is illustrated below. This version of the subsystem includes FIFOs in the RX and TX. There are three clock domains in the data path, as illustrated by the dashed lines in [Figure 3-2](#).



X15167-052916

Figure 3-2: 40G/50G MAC with PCS/PMA Clocking

refclk_p0, refclk_n0, tx_serdes_refclk

The `refclk` differential pair is required to be an input to the FPGA. The example design includes a buffer to convert this clock to a single-ended signal `refclk`, which is used as the reference clock for the GT block. The `tx_serdes_refclk` is directly derived from `refclk`. Note that `refclk` must be chosen so that the `tx_serdes_refclk` meets the requirements of 802.3, which is within 100 ppm of 312.5 MHz for 40G and 390.625 MHz for 50G.

tx_clk_out

This clock is used for clocking data into the TX AXI4-Stream Interface and it is also the reference clock for the TX control and status signals. It is the same frequency as `tx_serdes_refclk`.

rx_clk_out

The `rx_clk_out` output signal is presented as a reference for the RX control and status signals processed by the RX core. It is the same frequency as the `rx_serdes_clk`.

rx_clk

The `rx_clk` input to the RX core is not presented in the example design. Instead, it is connected to the `tx_clk` which also drives the TX core. When connected in this manner, the RX AXI4-Stream Interface and the TX AXI4-Stream Interface are on the same clock domain, which in most cases is the preferred mode of operation for the system side datapath. If desired, you can disconnect the `rx_clk` input from the `rx_top` module and drive the RX AXI4-Stream Interface with a different clock than the TX AXI4-Stream Interface. In this case, the frequency of the `rx_clk` must be equal to or greater than the `tx_clk`.

dclk

The `dclk` signal must be a convenient stable clock. It is used as a reference frequency for the GT helper blocks which initiate the GT itself. In the example design, a typical value is 75 MHz, which is readily derived from the 300 MHz clock available on the VCU107 evaluation board.

Note: The actual frequency must be known to the GT helper blocks for proper operation.

Low Latency 40G/50G MAC with PCS/PMA Clocking

The clocking architecture for the Low Latency 40/50G MAC with PCS/PMA clocking is illustrated in [Figure 3-3](#). Low latency is achieved by omitting the RX and TX FIFOs, which results in different clocking arrangement. There are two clock domains in the datapath, as illustrated by the dashed lines in [Figure 3-3](#).

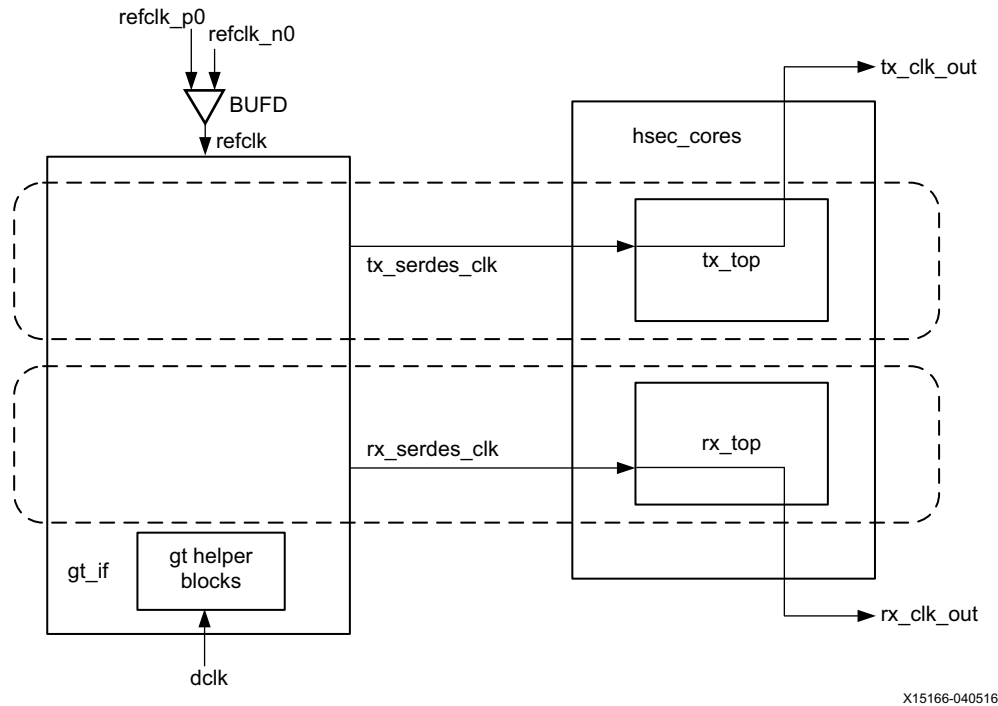


Figure 3-3: Low Latency 40G/50G MAC with PCS/PMA Cloning

refclk_p0, refclk_n0, tx_serdes_refclk

The `refclk` differential pair is required to be an input to the FPGA. The example design includes a buffer to convert this clock to a single-ended signal `refclk`, which is used as the reference clock for the GT block. The `tx_serdes_refclk` is directly derived from `refclk`. Note that `refclk` must be chosen so that the `tx_serdes_refclk` meets the requirements of 802.3, which is within 100 ppm of 312.5 MHz for 40G, and 390.625 MHz for 50G.

tx_clk_out

This clock is used for clocking data into the TX AXI4-Stream Interface and it is also the reference clock for the TX control and status signals. It is the same frequency as `tx_serdes_refclk`. Because there is no TX FIFO, you must respond immediately to the `tx_axis_tready` signal.

rx_clk_out

The `rx_clk_out` output signal is presented as a reference for the RX control and status signals processed by the RX core. It is the same frequency as the `rx_serdes_clk`. Because there is no RX FIFO, this is also the clock which drives the RX AXI4-Stream Interface. In this arrangement, `rx_clk_out` and `tx_clk_out` are different frequencies and have no defined phase relationship to each other.

dclk

The `dclk` signal must be a convenient stable clock. It is used as a reference frequency for the GT helper blocks which initiate the GT itself. In the example design, a typical value is 75 MHz, which is readily derived from the 300 MHz clock available on the VCU107 evaluation board. Note that the actual frequency must be known to the GT helper blocks for proper operation.

Auto-Negotiation and Link Training Clocking

The clocking architecture for the Auto-Negotiation and Link Training blocks are illustrated in Figure 3-4. Note that these blocks are not included unless the 50GBASE-KR or 50GBASE-CR feature is selected.

The Auto-Negotiation and Link Training blocks function independently from the MAC and PCS, and therefore they are on different clock domains.

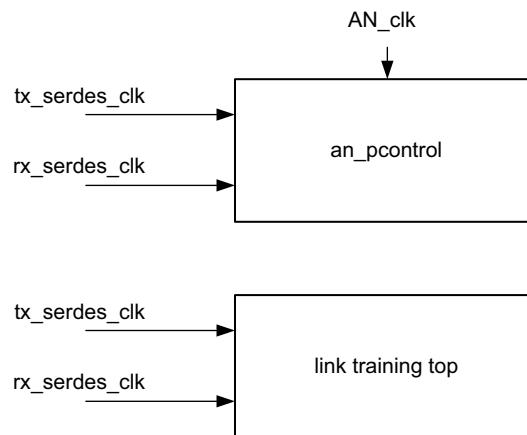


Figure 3-4: Auto-Negotiation and Link Training Clocking

tx_serdes_clk

The `tx_serdes_clk` drives the TX line side logic for the Auto-Negotiation and Link Training. The DME frame is generated on this clock domain.

rx_serdes_clk

The `rx_serdes_clk` drives the RX line side logic for the Auto-Negotiation and Link Training.

AN_clk

The `AN_clk` drives the Auto-Negotiation state machine. All ability signals are on this clock domain. The `AN_clk` can be any convenient frequency. In the example design, `AN_clk` is connected to the `dc1k` input, which has a typical frequency of 75 MHz. The `AN_clk` frequency must be known to the Auto-Negotiation state machine because it is the reference for all timers.

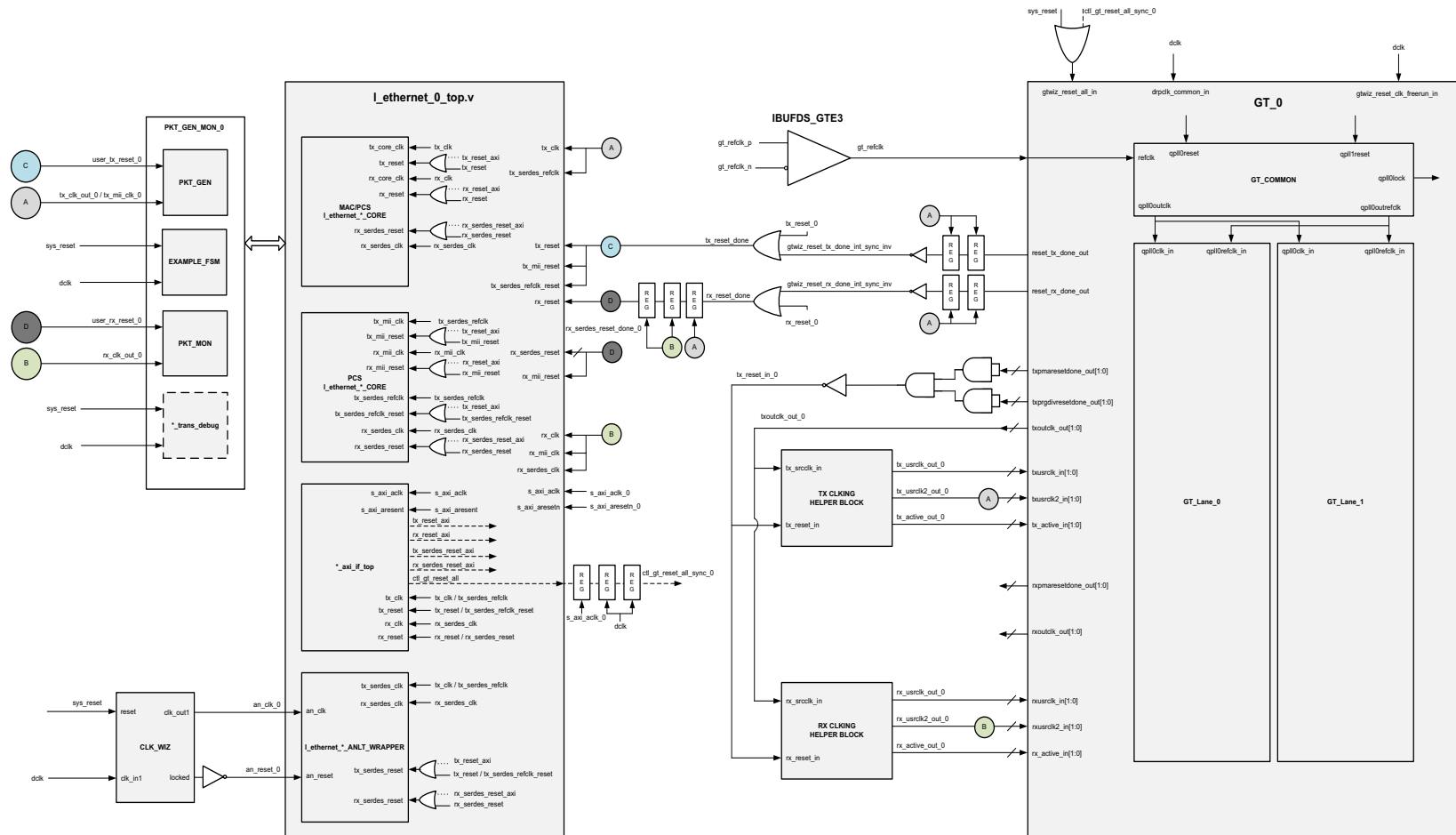
LogiCORE Example Design Clocking and Resets

The 40G/50G High Speed Ethernet Subsystem has separate reset inputs for the RX and TX paths that can be asserted independently. Within the RX and TX paths, there are resets for each of the various clock domains. The reset procedure is simple and the only requirement is that a reset must be asserted when the corresponding clock is not stable.

The 40G/50G High Speed Ethernet Subsystem takes care of ensuring that the different resets properly interact with each other internally and the interface operates properly (that is, there is no order required for asserting/deasserting different resets). It is left up to you to ensure a reset is held until the corresponding clock is fully stable.

Note: Some of the control inputs to the 40G/50G High Speed Ethernet Subsystem can only be modified while the core is held in reset. If one of these inputs needs changing, the appropriate RX or TX AXI4-Stream reset input (`rx_reset` or `tx_reset`) must be asserted until the control input is stabilized. See [Table 2-1](#) for a list of these inputs. Currently, all resets are synchronous to their corresponding clocks. That is, there must be a 0-1 transition on the corresponding clock while the reset is asserted High in order for the reset to be performed.

[Figure 3-6](#) through [Figure 3-9](#), illustrate the clocking and reset structure when you implement the Example Design using the Vivado tools.



X16607-040316

Figure 3-5: Detailed Diagram of Single Core - Synchronous Clock Mode

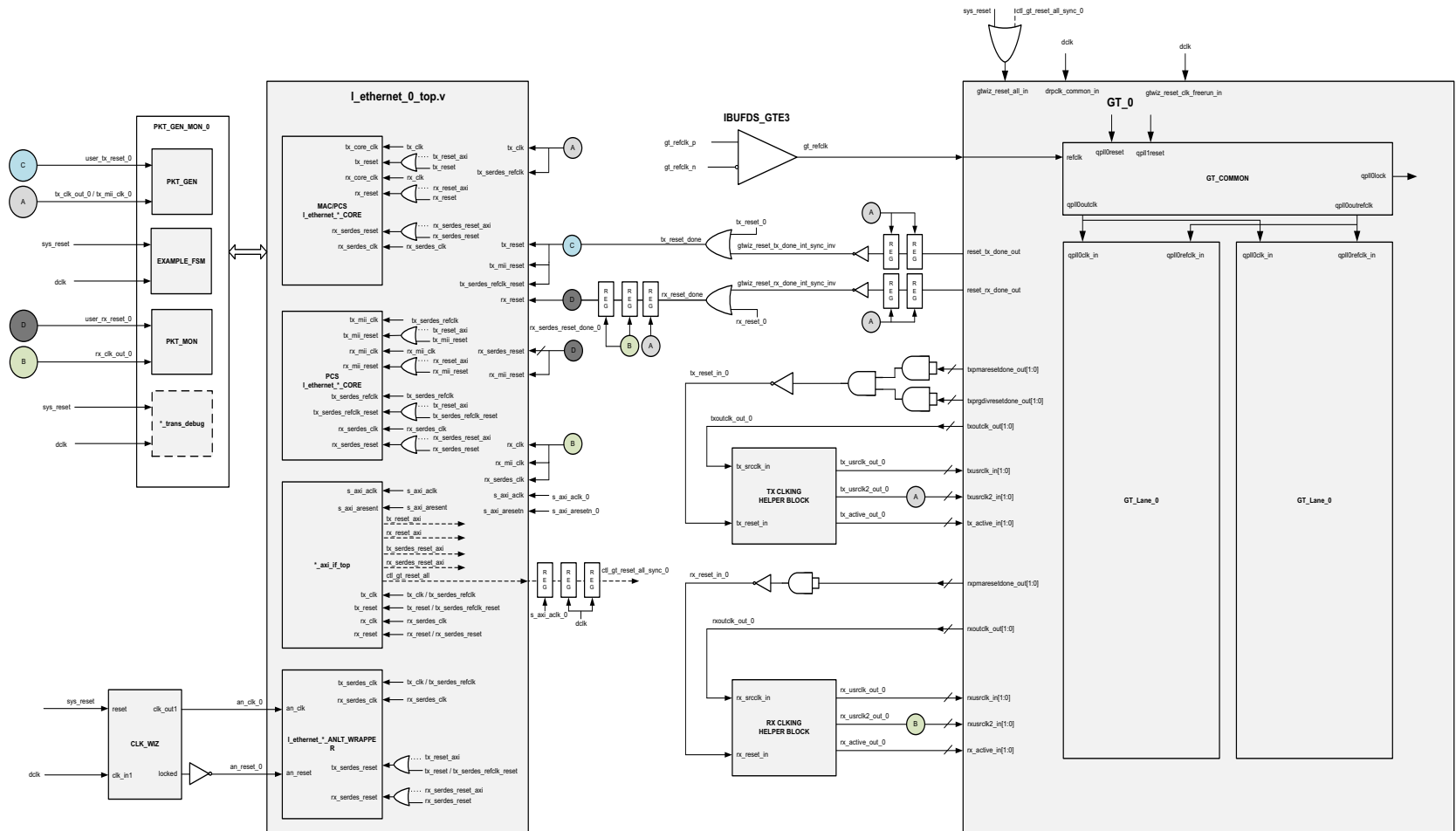
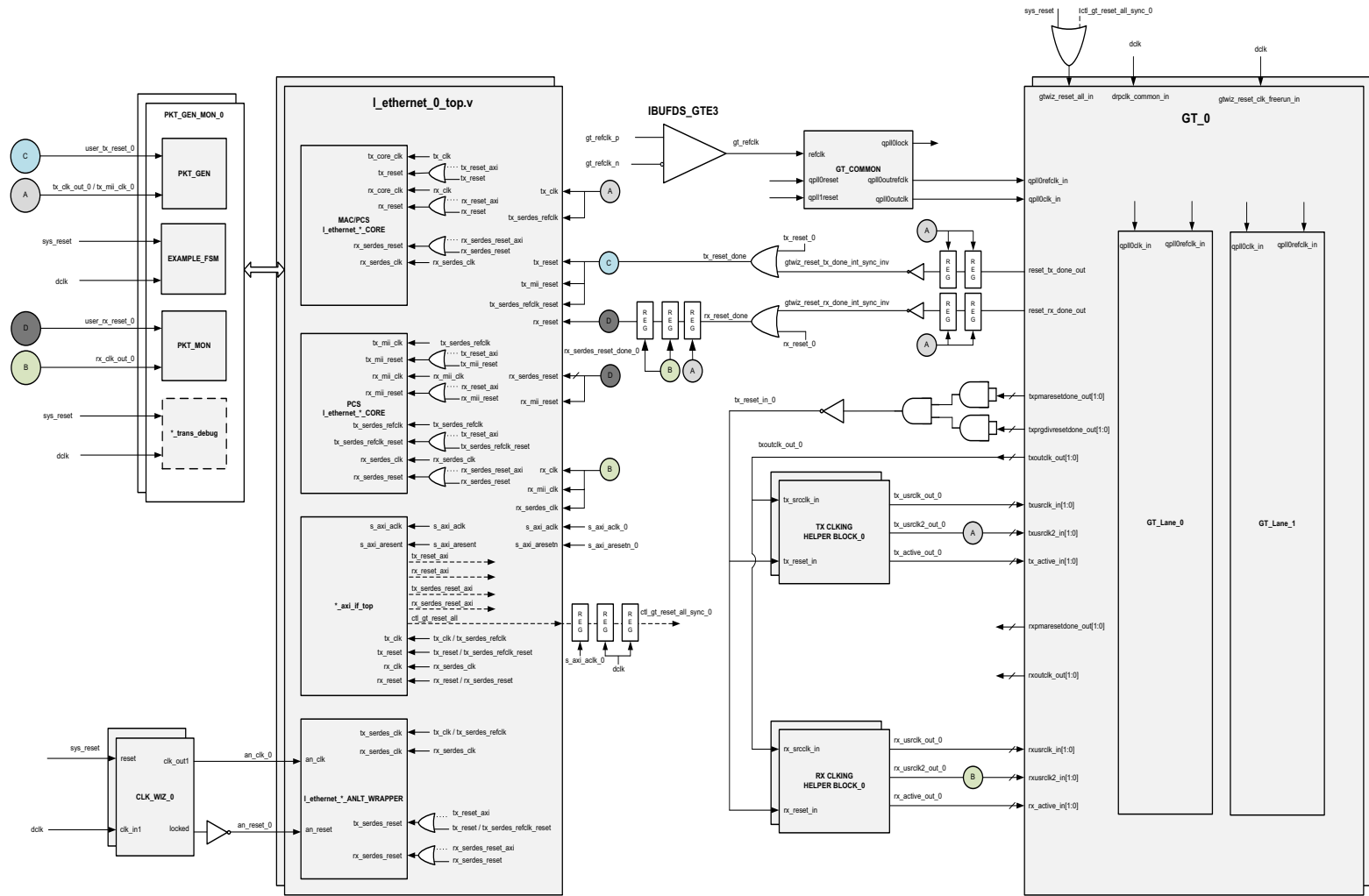
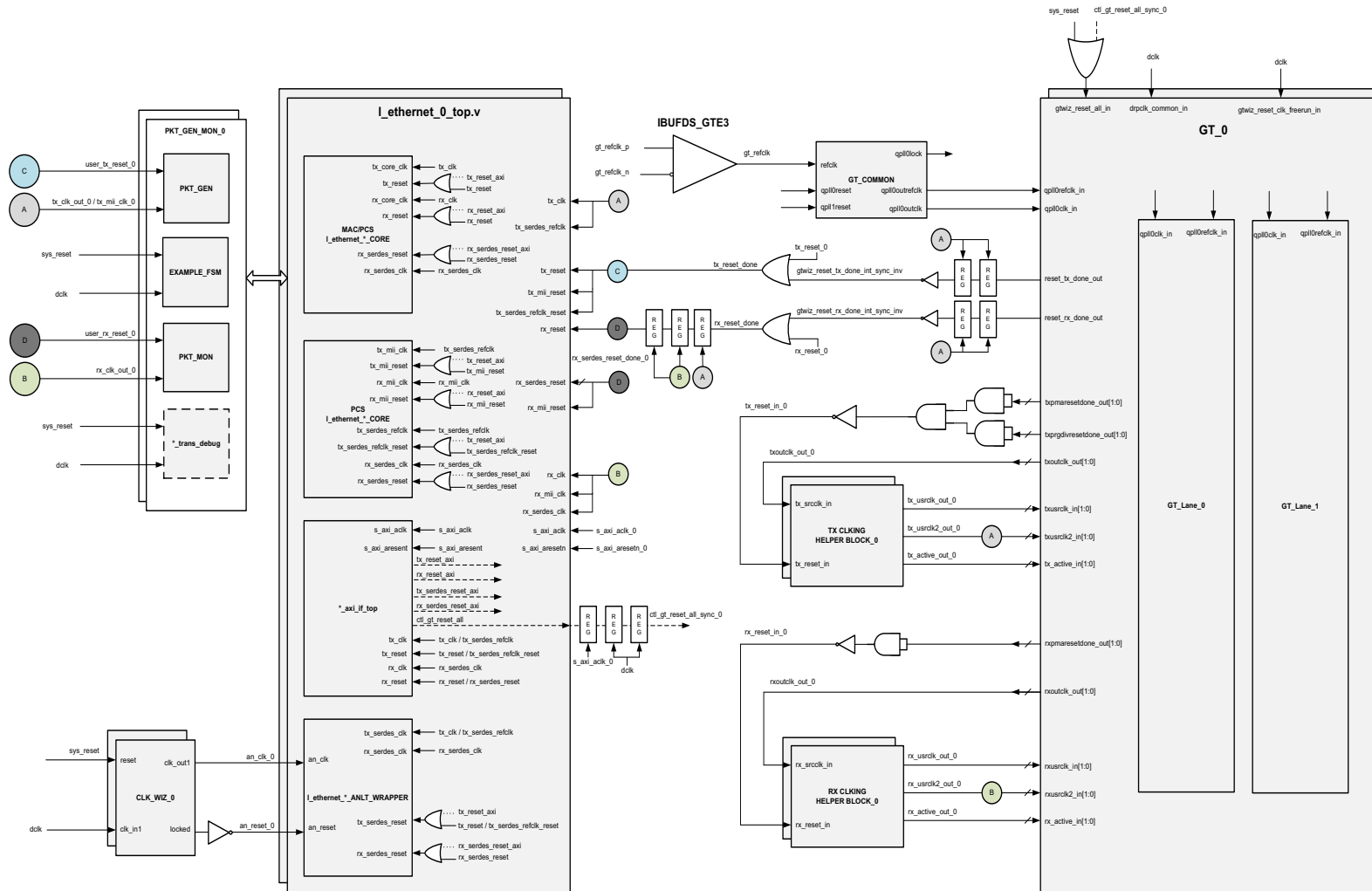


Figure 3-6: Detailed Diagram of Single Core - Asynchronous Clock Mode



X16609-022916

Figure 3-7: Detailed Diagram of Multiple Cores - Synchronous Clock Mode



X16810-02/2016

Figure 3-8: Detailed Diagram of Multiple Cores - Asynchronous Clock Mode

IEEE PTP 1588v2 for 40G/50G Subsystem

Overview

This section details the packet timestamping function of the 40G/50G Ethernet subsystem when the MAC layer is included. The timestamping option must be specified at the time of generating the subsystem from the IP catalog or ordering the IP core asynchronously. This feature presently supports two-step only IEEE PTP 1588 functionality. One-step operation is described in this appendix for reference in anticipation of its availability in future releases.

Note: 1-step is not supported in v1.1.

Ethernet frames are timestamped at both ingress and egress. The option can be used for implementing all kinds of IEEE 1588 clocks: Ordinary, Transparent, and Boundary. It can also be used for the generic timestamping of packets at the ingress and egress ports of a system. While this feature can be used for a variety of packet timestamping applications, the rest of this appendix assumes that you are also implementing the IEEE 1588 Precision Time Protocol (PTP).

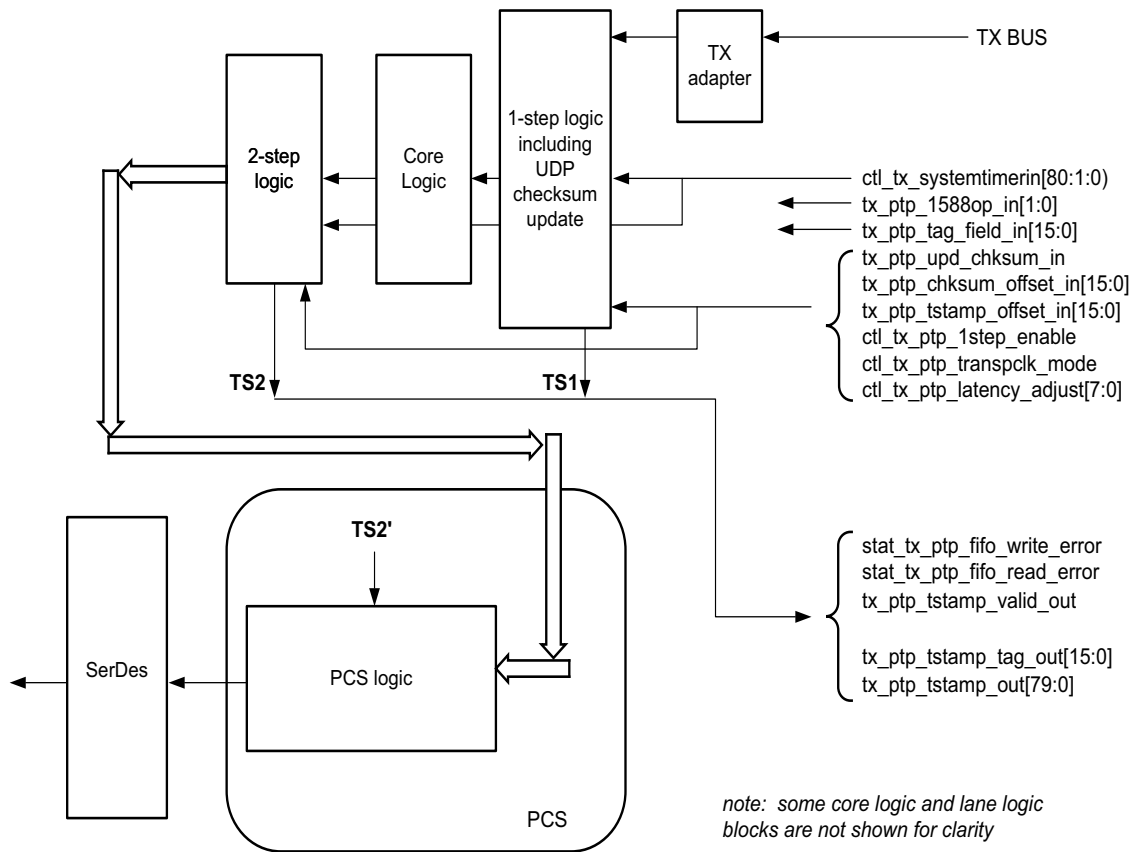
IEEE 1588 defines a protocol for performing timing synchronization across a network. A 1588 network has a single master clock timing reference, usually selected through a best master clock algorithm. Periodically, this master samples its system timer reference counter, and transmits this sampled time value across the network using defined packet formats. This timer should be sampled (a timestamp) when the start of a 1588 timing packet is transmitted. Therefore, to achieve high synchronization accuracy over the network, accurate timestamps are required. If this sampled timer value (the timestamp) is placed into the packet that triggered the timestamp, then this is known as 1-step operation. Alternatively, the timestamp value can be placed into a follow up packet; this is known as 2-step operation.

Other timing slave devices on the network receive these timing reference packets from the network timing master and attempt to synchronize their own local timer references to it. This mechanism relies on these Ethernet ports also taking timestamps (samples of their own local timer) when the 1588 timing packets are received. Further explanation of the operation of 1588 is out of the scope of this document. It is assumed that the reader is familiar with the IEEE 1588 specification for the rest of this section.

The 1588 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 an unsigned 48-bit second field and a 32-bit nanosecond field.
- Correction Field format: IEEE 1588-2008 numerical format consisting of a 64-bit signed field representing nanoseconds multiplied by 216 (see IEEE 1588 clause 13.3.2.7). This timer should count from 0 through the full range up to 264 -1 before wrapping around.

Egress



X16339-030916

Figure 3-9: Egress

As seen on the diagram, timestamping logic exists in two locations depending on whether 1-step or 2-step operation is desired. 1-step operation requires user datagram protocol (UDP) checksum and FCS updates and therefore the FCS core logic is used.

The TS references are defined as follows:

- TS1: The output timestamp signal when a 1-step operation is selected.
- TS2: The output timestamp signal when a 2-step operation is selected.
- TS2': The plane to which both timestamps are corrected.

TS2 always has a correction applied so that it is referenced to the TS2' plane. TS1 might or might not have the TS2' correction applied, depending on the value of the signal `ctl_tx_ptp_latency_adjust[10:0]`. The default value of this signal is determined when the subsystem is generated.

On the transmit side, a command field is provided by the client to the subsystem in parallel with the frame sent for transmission. This indicates, on a frame-by-frame basis, the 1588 function to perform (either no-operation, 1-step, or 2-step) and also indicates, for 1-step frames, whether there is a UDP checksum field to update.

If using the ToD format, then for both 1-step and 2-step operation, the full captured 80-bit ToD timestamp is returned to the client logic using the additional ports defined in [Table 3-1](#). If using the Correction Field format, then for both 1-step and 2-step operation, the full captured 64-bit timestamp is returned to the client logic using the additional ports defined in [Table 3-1](#) (with the upper bits of data set to zero as defined in the table).

If using the ToD format, then for 1-step operation, the full captured 80-bit ToD timestamp is inserted into the frame. If using the Correction Field format, then for 1-step operation, the captured 64-bit timestamp is summed with the existing Correction Field contained within the frame and the summed result is overwritten into the original Correction Field of the frame. Supported frame types for 1-step timestamping are:

- Raw Ethernet
- UDP/IPv4
- UDP/IPv6

For 1-step UDP frame types, the UDP checksum is updated in accordance with IETF RFC 1624. For all 1-step frames, the Ethernet Frame Check Sequence (FCS) field is calculated after all frame modifications have been completed. For 2-step transmit operation, all Precision Time Protocol (PTP) frame types are supported.

Frame-by-Frame Timestamping Operation

The operational mode of the egress timestamping function is determined by the settings on the 1588 command port. The information contained within the command port indicates one of the following:

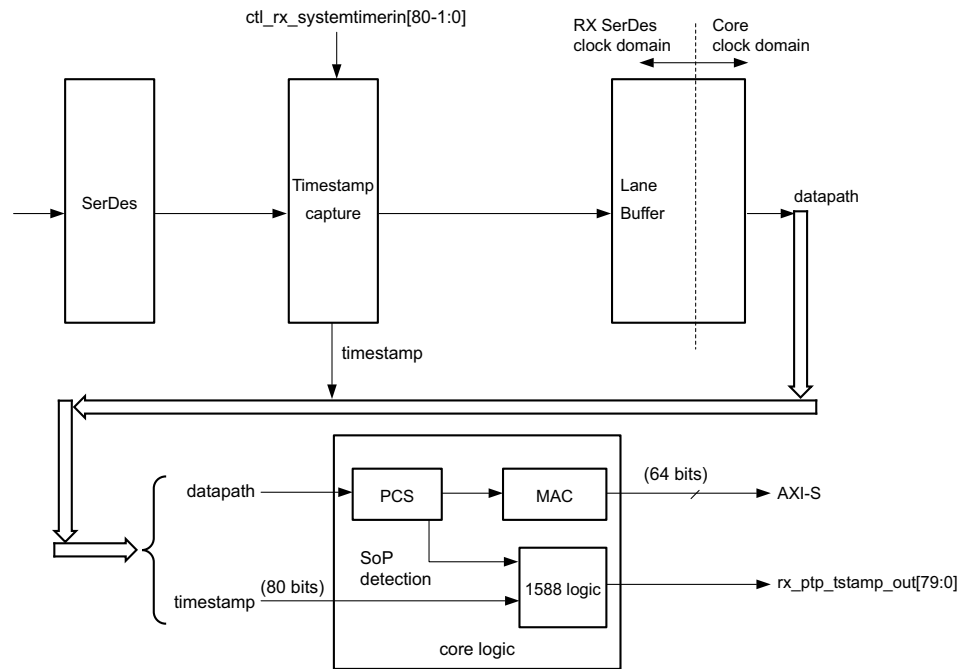
- No operation: the frame is not a PTP frame and no timestamp action should be taken.
- Two-step operation is required and a tag value (user-sequence ID) is provided as part of the command field; the frame should be timestamped, and the timestamp made available to the client logic, along with the provided tag value for the frame. The additional MAC transmitter ports 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 port; 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 port; the frame should be timestamped, 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, following the frame modification, the CRC value of the frame should also be updated/recalculated. For UDP IPv4 and IPv6 PTP formatted frames, the checksum value in the header of the frame needs to be updated/recalculated.

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

Ingress



X16340-030916

Figure 3-10: Ingress

The ingress logic does not parse the ingress packets to search for 1588 (PTP) frames. Instead, it takes a timestamp for every received frame and outputs this value to the user logic. The feature is always enabled, but the timestamp output can be ignored if you are not requiring this function.

Timestamps are filtered after the PCS decoder to retain only those timestamps corresponding to an SOP. These 80-bit timestamps are output on the system side. The timestamp is valid during the SOP cycle and when `ena_out = 1`.

Port Descriptions

The following table details the additional signals present when the packet timestamping feature is included.

Table 3-1: 1588v2 Port List and Descriptions

| Signal | Direction | Description | Clock Domain |
|--------------------------------------|-----------|---|---------------|
| IEEE 1588 Interface – TX Path | | | |
| ctl_tx_systemtimerin[80-1:0] | Input | System timer input for the TX. In normal clock mode, the 32 LSBs carry nsec and the 48 MSBs carry seconds. In transparent clock mode, bits 62:16 carry nanoseconds, and bits 15:0 carry fractional nanoseconds. Refer to IEEE 1588v2 for the representational definitions. This input must be in the TX SerDes clock domain. | tx_serdes_clk |
| tx_ptp_tstamp_valid_out | Output | This bit indicates that a valid timestamp is being presented on the TX system interface. | tx_clk_out |
| tx_ptp_tstamp_tag_out[15:0] | Output | Tag output corresponding to tx_ptp_tag_field_in[15:0] | tx_clk_out |
| tx_ptp_tstamp_out[80-1:0] | Output | Time stamp for the transmitted packet SOP corresponding to the time at which it passed the capture plane. Time format same as timer input. | tx_clk_out |
| tx_ptp_1588op_in[1:0] | Input | This signal should be valid on the first cycle of the packet. 2'b00 – No operation: no timestamp will be taken and the frame will not be 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 using the additional ports of 2-step operation. The frame itself will not be modified. 2'b11 – Reserved: act as No operation | tx_clk_out |
| ctl_tx_ptp_1step_enable | Input | When set to 1, this bit enables 1-step operation. | tx_clk_out |
| ctl_ptp_transpclk_mode | Input | When set to 1, this input places the timestamping logic into transparent clock mode. In this mode, the system timer input is interpreted as a correction value. The TX will add the correction value to the TX timestamp according to the process defined in IEEE 1588v2. It is expected that the corresponding incoming PTP packet correction field has already been adjusted with the proper RX timestamp. | tx_clk_out |

Table 3-1: 1588v2 Port List and Descriptions (Cont'd)

| Signal | Direction | Description | Clock Domain |
|---------------------------------|-----------|---|--------------|
| tx_ptp_tag_field_in[15:0] | Input | <p>The usage of this field is dependent on the 1588 operation. This signal should be valid on the first cycle of the packet.</p> <ul style="list-style-type: none"> •For No operation, this field is ignored. •For 1-step and 2-step this field is a tag field. This tag value will be returned to the client with the timestamp for the current frame using the additional ports of 2-step operation. 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. | tx_clk_out |
| ctl_tx_ptp_latency_adjust[10:0] | Input | <p>This bus can be used to adjust the 1-step TX timestamp with respect to the 2-step timestamp. The units of bits [10:3] are nanoseconds and bits [2:0] are fractional nanoseconds. In transparent clock mode the value of 802 decimal is recommended.</p> | tx_clk_out |
| stat_tx_ptp_fifo_write_error | Output | <p>Transmit PTP FIFO write error. A value of 1 on this status indicates that an error occurred during the PTP Tag write. A TX Path reset is required to clear the error.</p> | tx_clk_out |
| stat_tx_ptp_fifo_read_error | Output | <p>Transmit PTP FIFO read error. A value of 1 on this status indicates that an error occurred during the PTP Tag read. A TX Path reset is required to clear the error.</p> | tx_clk_out |
| tx_ptp_rxtstamp_in | Input | <p>The usage of this field is dependent on the 1588 operation.</p> <ul style="list-style-type: none"> • For "No operation", "2-step", or "1-step" in Ordinary Clock Mode, this field is ignored. • For "1-step" in Transparent Clock mode, this field carries a PTP packet Rx timestamp. Bit 63 is expected to be zero, bits 62:16 carry nanoseconds, and bits 15:0 carry fractional nanoseconds. Refer to IEEE 1588v2 for the representational definitions. The difference between this Rx timestamp and the PTP packet Tx timestamp corresponds to its residence time and will be added in a two's complement operation to the contents of the PTP packet Correction field. | tx_clk_out |

Table 3-1: 1588v2 Port List and Descriptions (*Cont'd*)

| Signal | Direction | Description | Clock Domain |
|--------------------------------------|-----------|---|---------------|
| IEEE 1588 Interface – RX Path | | | |
| ctl_rx_systemtimerin[80-1:0] | Input | System timer input for the RX. Same time format as the TX. This input must be in the same clock domain as the RX SerDes. | rx_serdes_clk |
| rx_ptp_tstamp_out[80-1:0] | Output | Time stamp for the received packet SOP corresponding to the time at which it passed the capture plane. The signal will be valid on the first cycle of the packet. | rx_clk_out |

IEEE PTP 1588v2 Functional Description

The IEEE 1588 feature of the 40G/50G subsystem provides accurate timestamping of Ethernet frames at the hardware level for both the ingress and egress directions.

Timestamps are captured according to the input clock source (system timer) defined previously. However, it is required that this time source be in the same clock domain as the SerDes. You might be required to re-time using an external circuit.

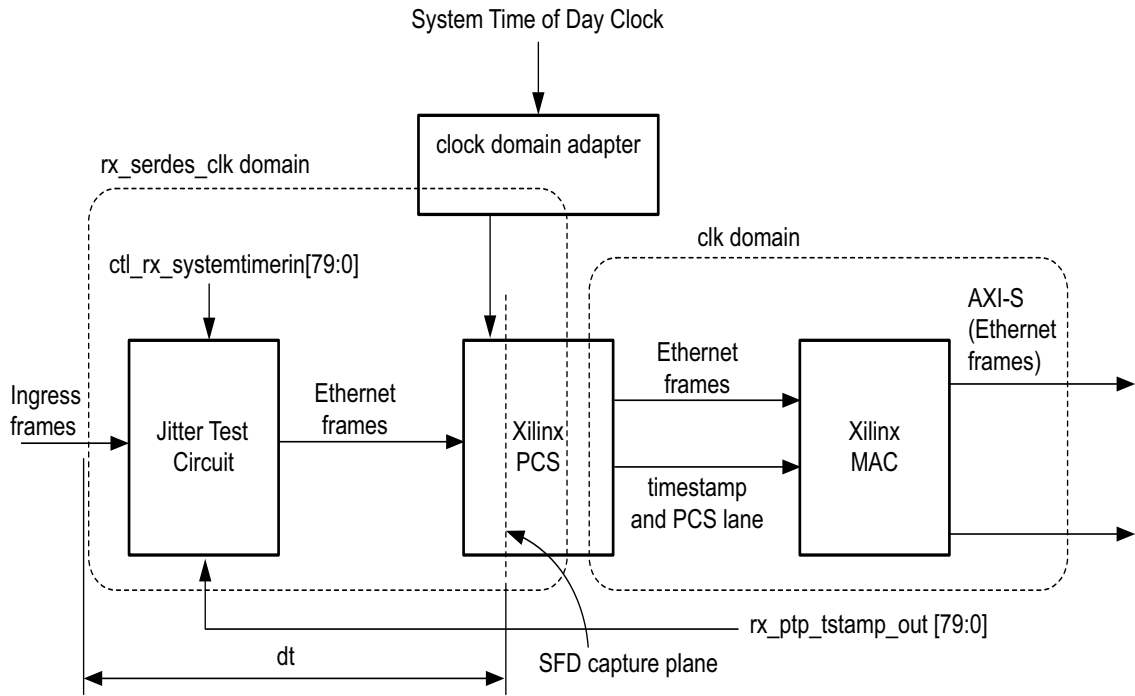
In a typical application, the PTP algorithm (or servo, not part of this IP) will remove timestamp errors over the course of time (many packet samples). It is advantageous for the error to be as small as possible in order to minimize the convergence time as well as minimizing slave clock drift. PTP packets are typically transmitted about 10 times per second.

All ingress frames receive a timestamp. It is up to you to interpret the received frames and determine whether a particular frame contains PTP information (by means of its Ethertype) and if the timestamp needs to be retained or discarded.

Egress frames are timestamped if they are tagged as PTP frames. The timestamps of egress frames are matched to their user-supplied tags.

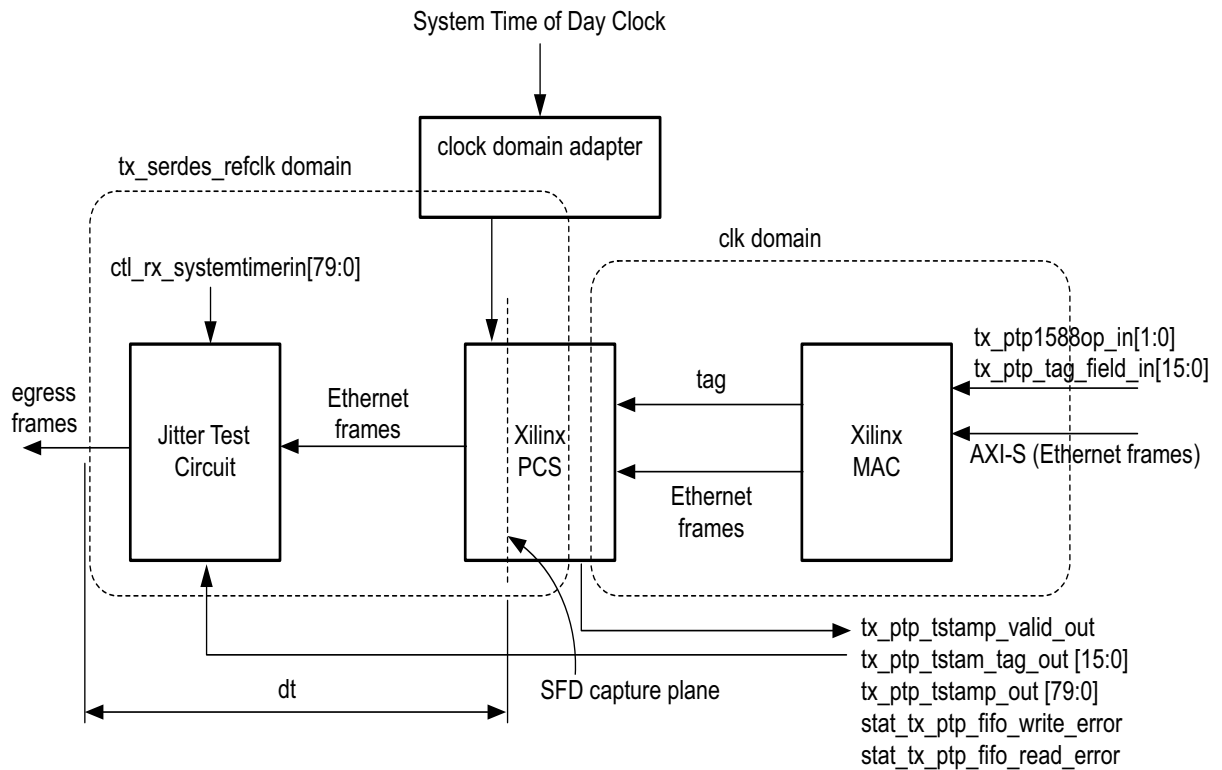
Timestamps for incoming frames are presented at the user interface in parallel with the AXI4-Stream cycle corresponding to the start of packet. You can then append the timestamp to the packet as required.

By definition, a timestamp is captured coincident with the passing of the SOP through the capture plane within the 40G/50G High Speed Ethernet Subsystem. This is illustrated in the following schematic diagrams:



X16341-030916

Figure 3-11: Receive



X16342-030916

Figure 3-12: Transmit

Performance

Performance of the timestamping logic is tested as shown in the previously referenced diagrams. On the RX side, the jitter test circuit takes note of the system timer at exactly the time when a start-of-frame packet enters the test circuit. Some time later, the timestamp is captured by the RX PCS and is eventually output on the system side AXI4-Stream interface (`rx_ptp_tstamp_out[79:0]`). The variation of the difference between these two time captures, dt , is defined as the "jitter" performance of the timestamping logic. The TX test is similar for `tx_ptp_tstamp_out[79:0]`.

The 40G/50G subsystem timestamping logic is theoretically capable of determining the time of crossing the SOP capture plane to within the granularity of the 80-bit system timer input. Therefore, if the system timer has a 1 nsec period, the timestamp will be accurate to within a jitter of 1 nsec. 1 nsec is also the granularity of the least significant bit of the 80-bit field as defined by IEEE 1588.

In practice, additional factors will limit the accuracy achievable in a real system.

Clock Domain

In a practical sense, the system timer input is required to have a granularity of the SerDes clock. Therefore the clock domain crossing of the system timer input should be taken into account. For example, if the SerDes clock has a frequency of 390 MHz, the system timer will have an actual granularity of 2.56 ns, which is also the clock which captures the timestamp. Hence an additional variation of 2.56 ns can be expected.

Transceiver

The addition of a SerDes in the datapath does not impact the jitter performance of the 40G/50G subsystem but might result in asymmetry.

In a 1588 clock application, the RX + TX SerDes latency becomes part of the loop delay and is therefore measured by the 1588 protocol. For maximum accuracy of the slave clock it is desirable to take loop asymmetry into account (the difference between the RX and TX SerDes latencies). Xilinx can provide the transceiver latency for various settings of the SerDes specific to your device. You need to contact the vendor of the other transceiver in a datapath if necessary for its characteristics, if you wish to take asymmetry into account in your PTP system.

UltraScale™ and UltraScale+™ transceivers have the ability to report the RX latency. Variation of the RX transceiver latency is mainly due to the fill level of its internal elastic buffer. Refer to the transceiver guide for more details.

Forward Error Correction

Forward Error Correction (for both Clause 74 and Clause 91) takes place on the line side of the timestamp capture. Therefore the addition of FEC will not impact the accuracy of the timestamp capture in a 40G/50G subsystem. Similar to the SerDes case discussed previously, the additional total (RX + TX) latency of the FEC will be measured by the 1588 protocol.

(Note that the SOP is not visible in a transmitted FEC frame until it has been decoded by the RX FEC function.)

For maximum 1588 slave clock accuracy, it is useful to know the asymmetry of the FEC latency in the RX and TX directions. Contact Xilinx for RX and TX latency of the specific FEC and its configuration. You might also need to obtain this information from the vendor of the link partner in the PTP system if it is not a Xilinx FEC implementation.

Receive Skew Correction

In a multi-lane system such as 40G and 50G, the packet corresponding to the SOP can occur on any lane. Furthermore, lanes can have skew relative to each other. The 40G/50G subsystem provides the ability to take the arrival lane of an SOP frame into account by reporting the SOP lane and its skew. Correction may be performed by hardware or software. The recommended steps are as follows.

Consider the example cases below, for 40G and 50G. The procedure is the same for both except that the number of PMD lanes is 2 and 4 respectively.

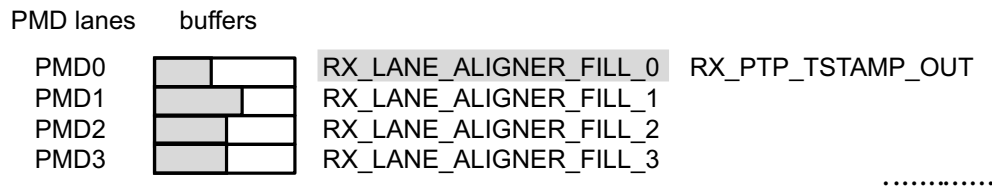


Figure 3-13: 40G Fill Level Correction Example

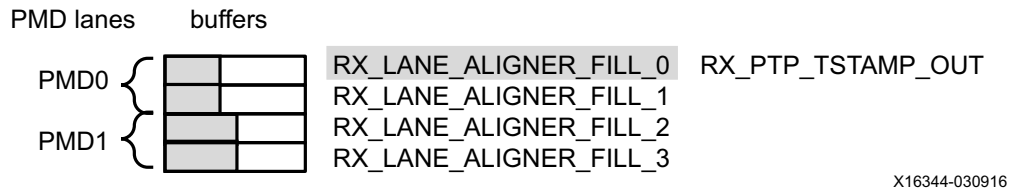


Figure 3-14: 50G Fill Level Correction Example

Example: RX_PTP_PCSLANE_OUT = 2

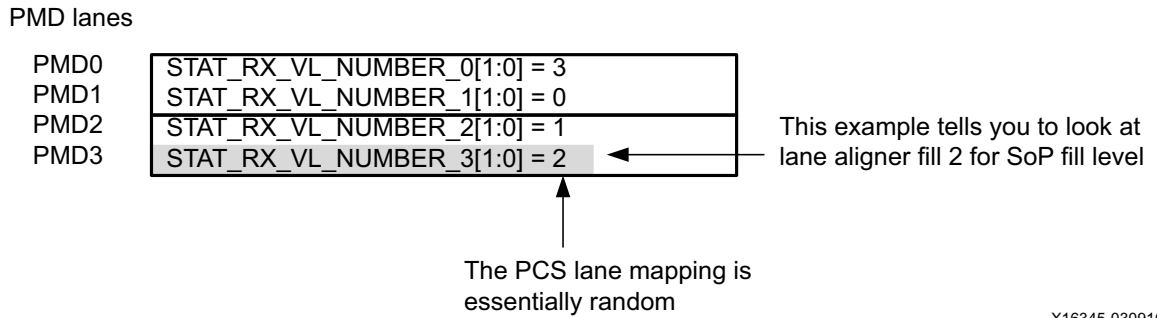


Figure 3-15: 40G Skew Correction Example

Example: RX_PTP_PCSLANE_OUT = 1

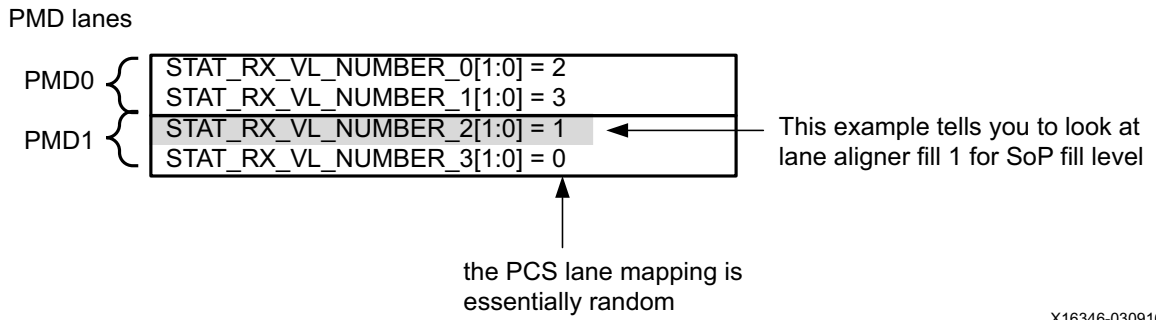
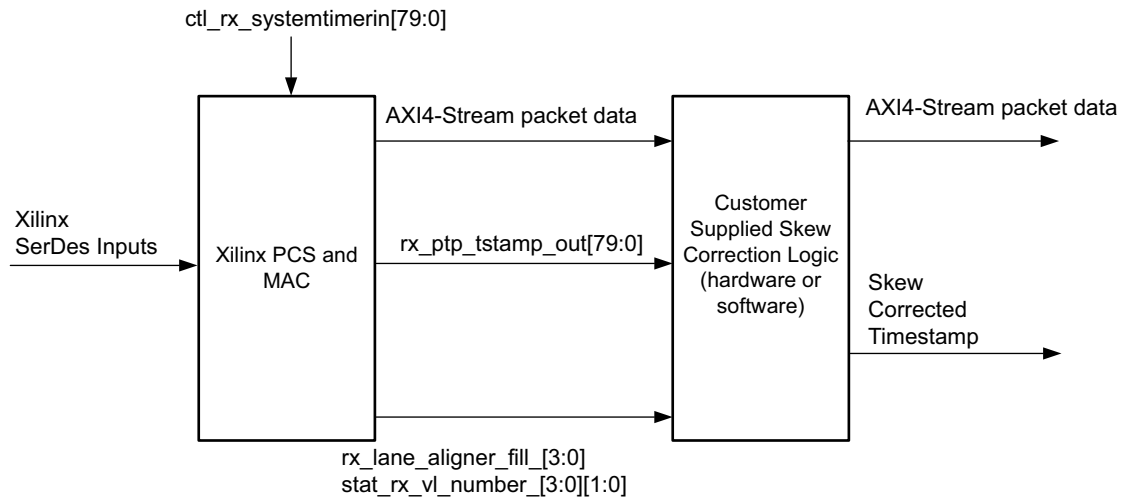


Figure 3-16: 50G Skew Correction Example



X16347-0309

Figure 3-17: Timestamp Skew Correction Logic

The first step is to take a time average of the alignment buffer fill levels because the granularity of these signals is one SerDes clock cycle. While the skew remains relatively constant over time (for example, minutes or hours), the alignment buffer levels have short-term fluctuations of SerDes clock cycles due to sampling quantization. Therefore the actual skew can be obtained to a high degree of accuracy (for example, sub nanoseconds) by taking a time average of each of the fill levels.

Assuming the fill levels have been accurately determined as above, the following formula is used to correct for skew:

```
correction = ((RX_LANE_ALIGNER_FILL_n) - (RX_LANE_ALIGNER_FILL_0)) * SerDes clock period
corrected timestamp = RX_PTP_TSTAMP_OUT + correction
```

Where:

corrected timestamp is the skew-corrected timestamp, which is required to be kept in step with the corresponding packet data

RX_PTP_TSTAMP_OUT is the captured timestamp.

RX_LANE_ALIGNER_FILL_0 is the alignment buffer fill level for the lane on which the timestamp was taken, usually lane 0 (check with Xilinx technical sales support for updates).

RX_LANE_ALIGNER_FILL_n is the alignment buffer fill level for the lane containing the SOF.

The units of all the calculations need to be consistent. Because fill levels are provided in terms of clock cycles, they may have to be converted to nanoseconds or whatever units are consistent with the calculation.

For additional information, see the IEEE Standard 1588-2008, "IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems" (standards.ieee.org/findstds/standard/1588-2008.html)

Status/Control Interface

The Status/Control interface allows you to set up the 40G/50G High Speed Ethernet Subsystem configuration and to monitor the status of the 40G/50G High Speed Ethernet Subsystem. The following subsections describe in more detail some of the various Status and Control signals.

RX and TX PCS Lane Marker Values

The IEEE Std 802.3 defines the PCS Lane marker values, shown in [Table 3-2](#).

Table 3-2: 40G/50G Marker Definitions

| Input Signal Name | Value |
|--|-----------------------------|
| ctl_rx_vl_marker_id[0][63:0] ctl_tx_vl_marker_id[0][63:0] | 64'h90_76_47_00_6f_89_b8_00 |
| ctl_rx_vl_marker_id[1][63:0] ctl_tx_vl_marker_id[1][63:0] | 64'hf0_c4_e6_00_0f_3b_19_00 |
| ctl_rx_vl_marker_id[2][63:0] ctl_tx_vl_marker_id[2][63:0] | 64'hc5_65_9b_00_3a_9a_64_00 |
| ctl_rx_vl_marker_id[3][63:0] ctl_tx_vl_marker_id[3][63:0] | 64'ha2_79_3d_00_5d_86_c2_00 |

RX PCS Lane Alignment Status

The 40G/50G High Speed Ethernet Subsystem provides status bits to indicate the state of word boundary synchronization and PCS lane alignment. All signals are synchronous with the rising-edge of `c1k` and a detailed description of each signal follows.

stat_rx_synced[3:0]

When a bit of this bus is 0, it indicates that word boundary synchronization of the corresponding lane is not complete or that an error has occurred as identified by another status bit.

When a bit of this bus is 1, it indicates that the corresponding lane is word boundary synchronized and is receiving PCS Lane Marker Words as expected.

stat_rx_synced_err[3:0]

When a bit of this bus is 1, it indicates one of several possible failures on the corresponding lane:

- Word boundary synchronization in the lane was not possible using Framing bits [65:64]
- After word boundary synchronization in the lane was achieved, errors were detected on Framing bits [65:64]
- After word boundary synchronization in the lane was achieved, a valid PCS Lane Marker Word was never received

The bits of the bus remain asserted until word boundary synchronization occurs or until some other error/failure is signaled for the corresponding lane.

stat_rx_mf_len_err[3:0]

When a bit of this bus is 1, it indicates that PCS Lane Marker Words are being received but not at the expected rate in the corresponding lane. The transmitter and receiver must be reconfigured with the same Meta Frame length.

The bits of the bus remain asserted until word boundary synchronization occurs or until some other error/failure is signaled for the corresponding lane.

stat_rx_mf_repeat_err[3:0]

After word boundary synchronization is achieved in a lane, if a bit of this bus is a 1, it indicates that four consecutive invalid PCS Lane Marker Words were detected in the corresponding lane.

The bits of the bus remain asserted until resynchronization occurs or until some other error/failure is signaled for the corresponding lane.

stat_rx_mf_err[3:0]

When a bit of this bus is 1, it indicates that an invalid PCS Lane Marker Word was received on the corresponding lane. This bit is only asserted after word boundary synchronization is achieved. This output is asserted for one clock period each time an invalid Meta Packet Synchronization Word is detected.

stat_rx_aligned

When `stat_rx_aligned` is a value of 1, all of the lanes are aligned/deskewed and the receiver is ready to receive packet data.

stat_rx_aligned_err

When `stat_rx_aligned_err` is a value of 1, one of two things occurred:

- Lane alignment failed after several attempts.
- Lane alignment was lost (`stat_rx_aligned` was asserted and then it was negated).

stat_rx_misaligned

When `stat_rx_misaligned` is a value of 1, a valid PCS Lane Marker Word was not received on all PCS lanes simultaneously.

This output is asserted for one clock period each time this error condition is detected.

stat_rx_framing_err_[3:0][3:0] and stat_rx_framing_err_valid_[3:0]

This set of buses is intended to be used to keep track of sync header errors. There is a pair of outputs for each PCS Lane. The `stat_rx_framing_err_[PCSL_LANES-3:0]` output bus indicates how many sync header errors were received and it is qualified (that is, the value is only valid) when the corresponding `stat_rx_framing_err_valid_[PCSL_LANES-3:0]` is sampled as a 1.

stat_rx_vl_number[3:0][1:0]

Each bus indicates which PCS lane has its status reflected on specific status pins. For example, `stat_rx_vlane_number_0` indicates which PCS lane has its status reflected on pin 0 of the other status signals.

These buses can be used to detect if a PCS lane has not been found or if one has been mapped to multiple status pins.

stat_rx_vl_demuxed[3:0]

After word boundary synchronization is achieved on each lane, if a bit of this bus is 1 it indicates that the corresponding PCS lane was properly found and de-muxed.

stat_rx_block_lock[3:0]

Each bit indicates that the corresponding PCS lane has achieved sync header lock as defined by the IEEE Std 802.3-2012. A value of 1 indicates block lock is achieved.

stat_rx_status

This output is set to a 1 when `stat_rx_aligned` is a 1 and `stat_rx_hi_ber` is a 0. This is as defined by the IEEE Std 802.3-2012.

stat_rx_local_fault

This output is set to a 1 when `stat_rx_received_local_fault` or `stat_rx_internal_local_fault` is asserted. This output is level sensitive.

RX Error Status

The 40G/50G High Speed Ethernet Subsystem provides status signals to identify 64b/66b words and sequences violations and CRC32 checking failures.

All signals are synchronous with the rising-edge of `clk` and a detailed description of each signal follows.

stat_rx_bad_fcs

When this signal is a value of 1, it indicates that the error detection logic has identified a mismatch between the expected and received value of CRC32 in the received packet.

When a CRC32 error is detected, the received packet is marked as containing an error and is sent with `rx_errout` asserted during the last transfer (the cycle with `rx_eopout` asserted) unless `ctl_rx_ignore_fcs` is asserted.

This signal is asserted for one clock period each time a CRC32 error is detected.

stat_rx_bad_code[1:0]

This signal indicates how many cycles the RX PCS receive state machine is in the RX_E state as defined by the IEEE Std 802.3-2012.

Design Flow Steps

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

- *Vivado Design Suite User Guide: Designing IP Subsystems using IP Integrator* (UG994) [Ref 5]
- *Vivado Design Suite User Guide: Designing with IP* (UG896) [Ref 6]
- *Vivado Design Suite User Guide: Getting Started* (UG910) [Ref 7]
- *Vivado Design Suite User Guide: Logic Simulation* (UG900) [Ref 8]

Customizing and Designing the Core

Configuration Tab

The Configuration tab (Figure 4-1) provides the basic core configuration options.

Default values are pre-populated in all tabs.

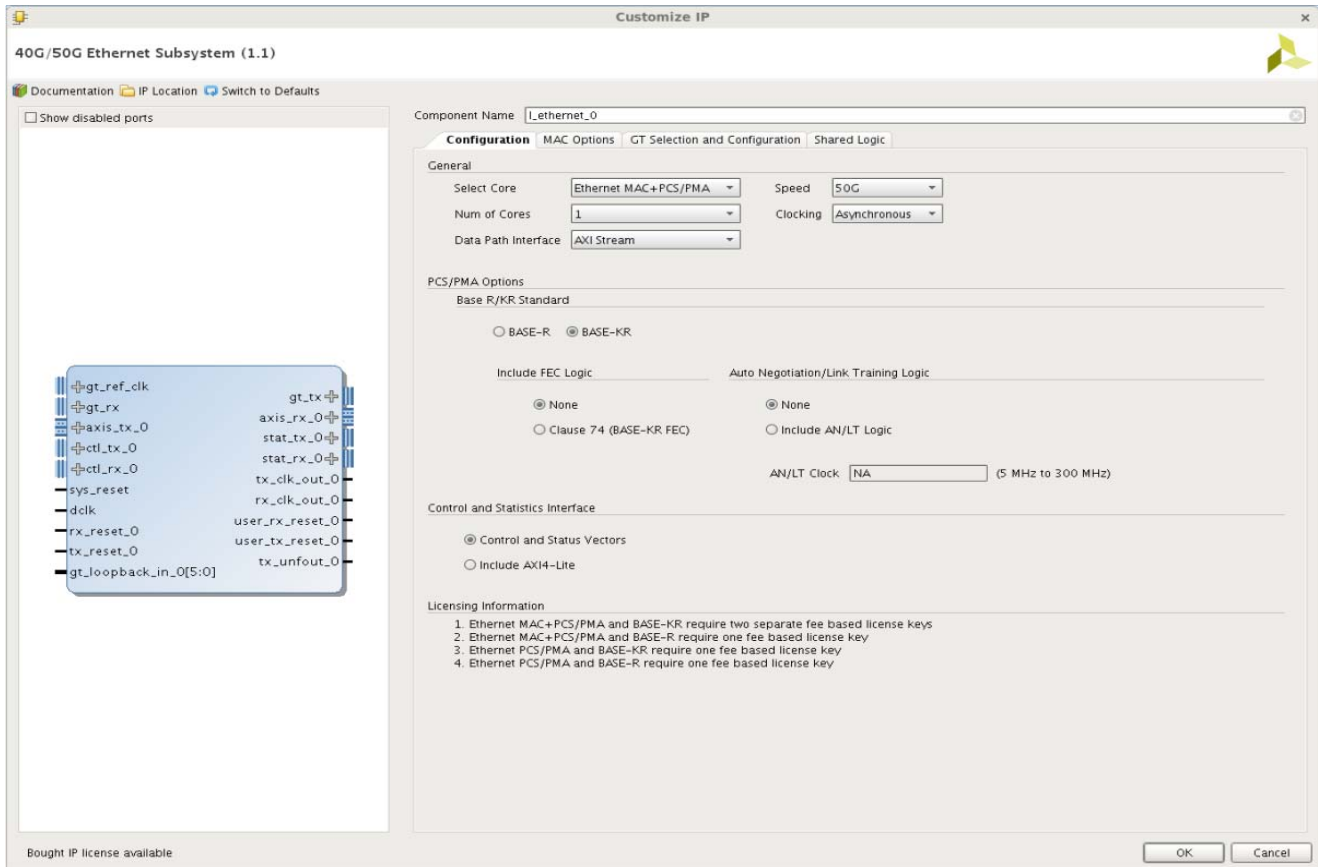


Figure 4-1: Configuration Tab

Table 4-1: Configuration Options

| Option | Values | Default |
|---|---|----------------------------|
| General | | |
| Select Core | Ethernet MAC+PCS/PMA Ethernet PCS/PMA | Ethernet MAC+PCS/PMA |
| Speed | 50G 40G | 50G |
| Num of Cores | 1 2 | 1 |
| Clocking | Synchronous Asynchronous | Asynchronous |
| Data Path Interface | AXI Stream ⁽¹⁾ MII ⁽²⁾ | AXI Stream |
| PCS/PMA Options | | |
| Base-R Base-KR | Base-R Base-KR | Base-KR |
| Include FEC Logic | None Clause 74 (BASE-KR FEC) | None |
| Auto-negotiation/Link Training Logic | None Include AN/LT Logic | None |
| AN/LT Clock | 5 MHz to 300 MHz | 75 MHz |
| Control and Statistics Interface | | |
| Control and Statistics interface | Control and Status Vectors Include AXI4-Lite | Control and Status Vectors |

Notes:

1. The AXI4-Stream interface is visible and is the only option for the Ethernet MAC+PCS/PMA core.
2. The MII interface is visible and is the only option for the Ethernet PCS/PMA core.

MAC Options Tab

The MAC Options tab (Figure 4-2) provides additional core configuration options.

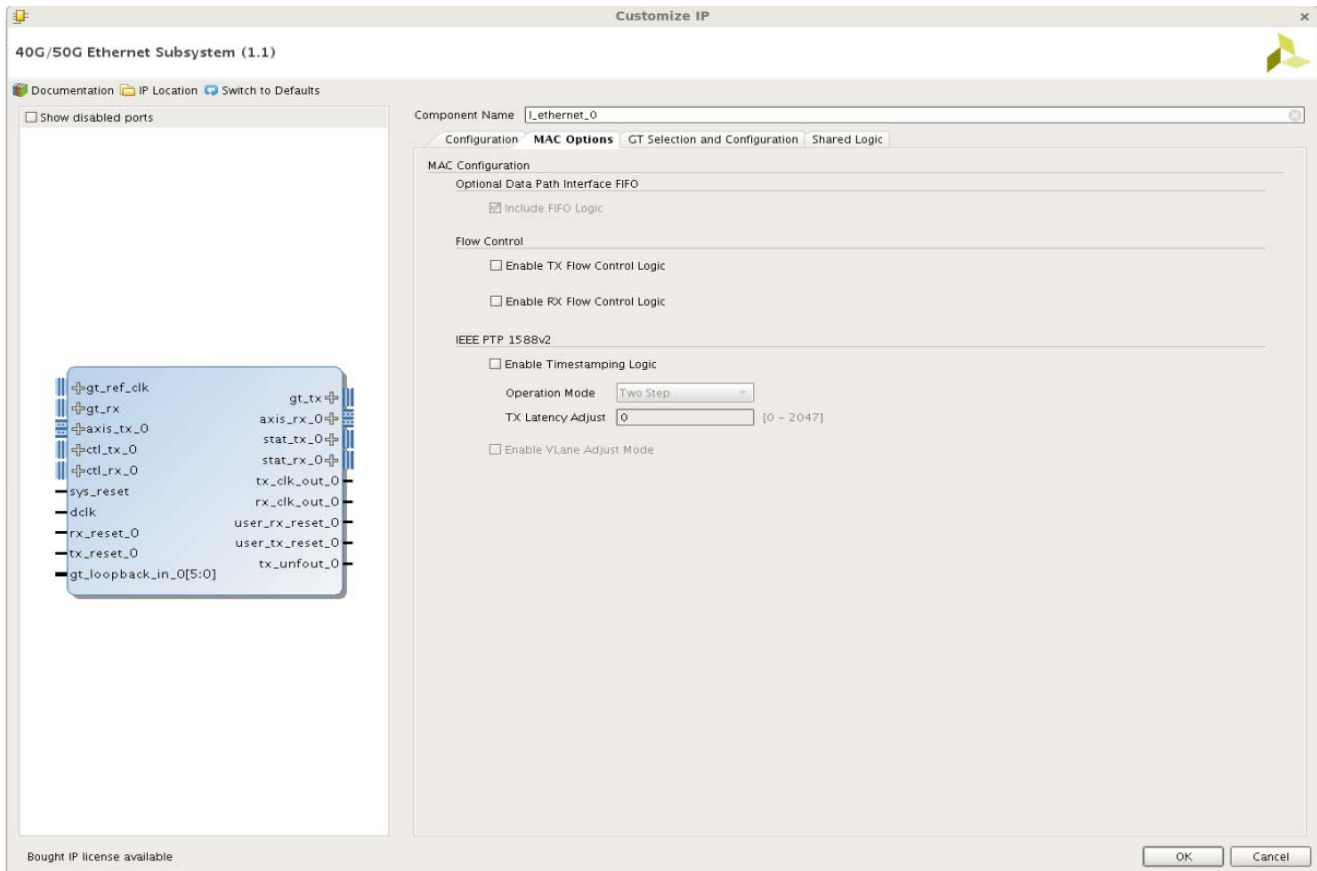


Figure 4-2: MAC Options Tab

Table 4-2: MAC Configuration

| Option | Values | Default |
|--|-----------|----------|
| Optional Data Path Interface FIFO | | |
| Include FIFO Logic | 0, 1 | 1 |
| Flow Control | | |
| Enable TX Flow Control Logic | 0, 1 | 0 |
| Enable RX Flow Control Logic | 0, 1 | 0 |
| IEEE PTP 1588v2 | | |
| Enable Timestamping Logic | 0, 1 | 0 |
| Operation Mode | Two Step | Two Step |
| TX Latency Adjust | 0 to 2047 | 0 |
| Enable V Lane Adjust Mode | 0, 1 | 0 |

GT Selection and Configuration Tab

The GT Selection and Configuration tab (Figure 4-3) enables you to configure the serial transceiver features of the core.

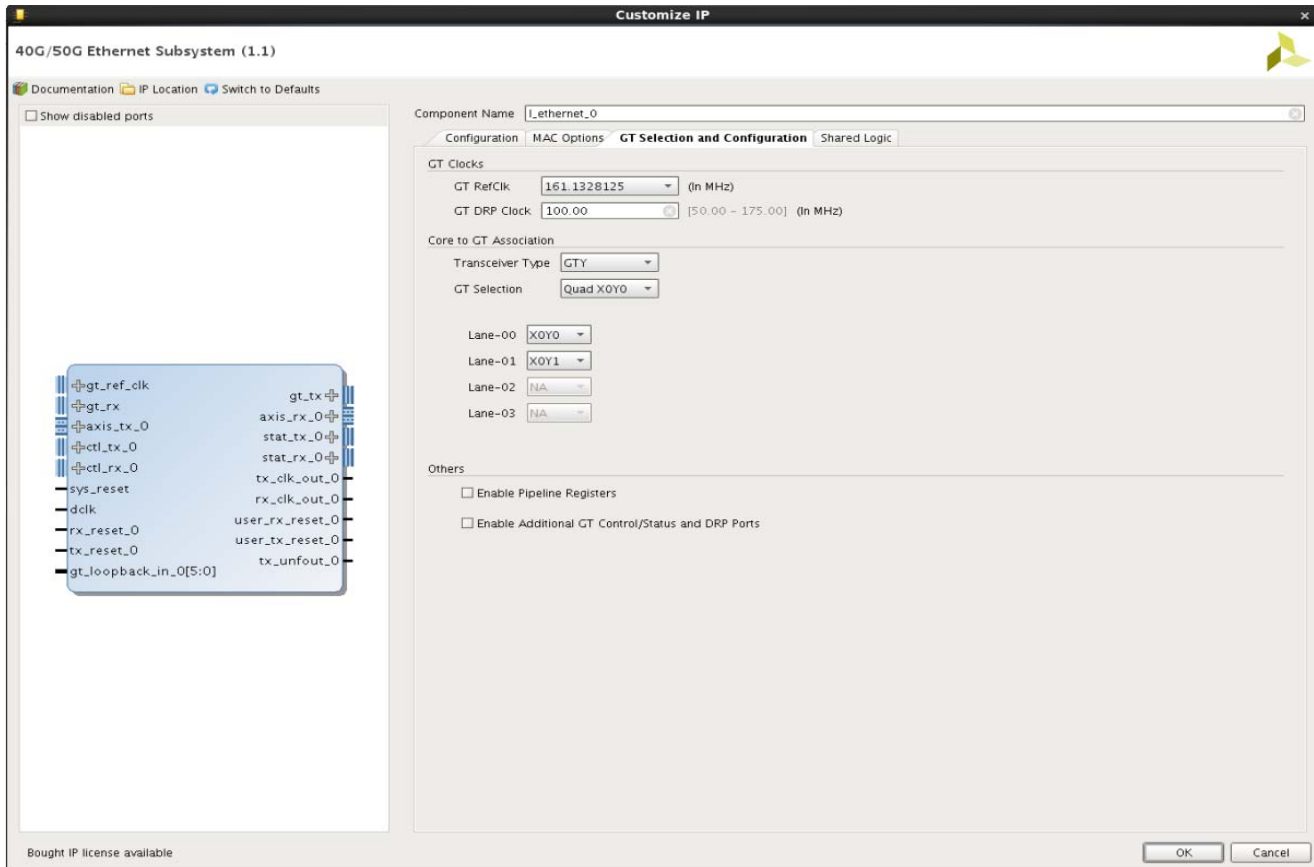


Figure 4-3: GT Selection and Configuration Tab

Table 4-3: GT Clocks Options

| Option | Values | Default |
|---|--|-------------|
| GT Clocks | | |
| GT RefClk (In MHz) | 161.1328125 195.3125 201.4160156 257.8125 322.265625 | 161.1328125 |
| GT DRP Clock (In MHz) | 50.00 – 175.00 MHz | 100.00 |
| Core to Transceiver Association | | |
| Transceiver Type | GTH GTY | GTY |
| GT Selection | Options based on device/package Quad groups. For example: Quad X0Y1 Quad X0Y2 Quad X0Y3 ... | Quad X0Y1 |
| Lane-00 to Lane-03 | Auto filled based on device/package. For example, if Speed = 50G and Num of Cores = 2 (or Speed = 40G and Num of Cores = 1) and GT selection = Quad X0Y1, then the four GT lanes are: X0Y4 X0Y5 X0Y6 X0Y7 | |
| Others | | |
| Enable Pipeline Register | 0, 1 | 0 |
| Enable Additional GT Control and Status Ports | 0, 1 | 0 |

Shared Logic Tab

The Shared Logic tab (Figure 4-4) enables you to use shared logic in either the core or the example design.

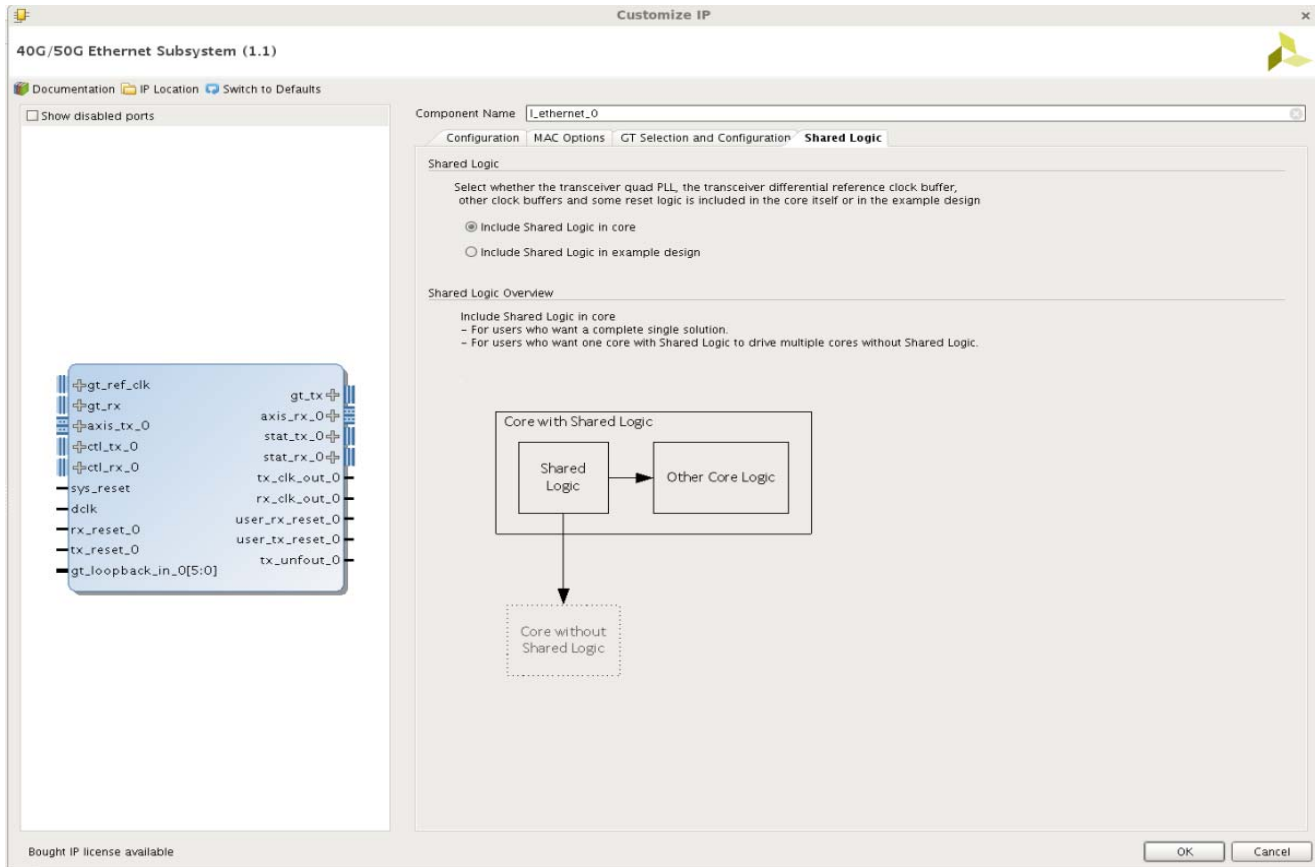


Figure 4-4: Shared Logic Tab

Table 4-4: Shared Logic Options

| Options | Default |
|--|------------------------------|
| Include Shared Logic in Core | Include Shared Logic in Core |
| Include Shared Logic in example design | |

Configuration Spreadsheet

In addition to the IP catalog in Vivado, the 40G/50G Ethernet Subsystem can be requested for a specific configuration using a customer-provided spreadsheet. You must submit the completed spreadsheet to Xilinx Technical Support to obtain the target netlist. See [Synthesis and Implementation](#) for a description of what is delivered and how to proceed with the deliverables.

For a complete understanding of the core, consult the document corresponding to the core and the device selected, such as the *UltraScale Architecture GTH Transceivers User Guide* (UG576) [\[Ref 3\]](#)

[Table 4-5](#) shows the typical input data required to specify the core.

Table 4-5: Parameter Values

| Parameter | Value | User Action (Enter Value, or N/A ⁽¹⁾) |
|--|---------------|---|
| Core Name (select from list) | 50 G MAC | Enter |
| Xilinx IP Core Part Number | TBD | N/A |
| Target FPGA (for example, UltraScale™) | ultrascale | Enter |
| Target Device (for example, VU 095) | VU 095 | Enter |
| Target Package (for example, FFG1157) | FFG1928 | Enter |
| Target Speed Grade | -2 | Enter |
| ISE/Vivado version (latest is default) | latest | Enter |
| Number of SerDes Lanes | 2 | N/A |
| SerDes speed (GHz) | 25.78125 | N/A |
| Ethernet Line Rate (Gb/s) | 51.5625 | N/A |
| User Interface AXI4-Stream Width (bits) | 256 | Enter |
| User Interface Protocol (Segmented, Non-Segmented)** | Non-segmented | N/A |
| AXI4-Stream Clock Frequency (MHz) | 216.00 | N/A |
| Internal Bus Width | 208 | Enter |
| Minimum Recommended clk Frequency (MHz)* | 204 | N/A |
| clk Clock Frequency (MHz) *** | 216 | Enter |
| Auto-Negotiation and FEC requirements (only valid for 40GBASE-KR4 MAC) | AN only | Enter |
| Pause Processing Included | Yes | N/A |
| 1588 Timestamping included | No | N/A |
| 1588 2-step included | No | N/A |
| 1588 1-step included | No | N/A |
| Evaluation Only | No | Enter |
| Default SerDes REFCLK0 (MHz) | 161.13 | N/A |
| SerDes Width (bits) | 64 | N/A |
| Provide any additional requirements here | | Enter |

Notes:

1. N/A denotes that the value is populated by the tool (either auto-detected or generated).

You must enter data into the spreadsheet in the shaded cells, as indicated in [Table 4-5](#). The white cells are calculated by the tools. The data entered (shaded rows) is done using either pull-down menus where values are pre-defined, or cells where text can be entered. Due to the dependencies between some cells, Xilinx recommends that cells be completed in the order in which they occur.

If you attempt to enter data that is beyond the scope of the technology, error messages guide you to a correct value. An example is entering a clock frequency which is either too slow or too fast.

Configuration Parameters

The following are some notes for the configuration parameters.

Core Name

Select the core name from the pull-down list. The corresponding Xilinx part number is automatically filled in.

Xilinx IP Core Part Number

This is determined by the IP core selected.

Device

40G/50G Ethernet core is supported by UltraScale+ and UltraScale devices.

Device Part Number

Enter the part number exactly as it appears in the product guide. If no part number is entered, you are prompted to provide one.

Package

Enter the package exactly as it appears in the product guide. If no package is entered, you are prompted to provide one.

Speed Grade

Refer to the product guide for available speed grades for the chosen device.

Tool Version

The IP is verified using the latest supported tool versions unless otherwise specified.

Number of Lanes

The number of lanes is determined by the selected Ethernet protocol according to the 25G and 40G/50G Ethernet Consortium specification.

Lane Rate

The SerDes lane rate is determined by the selected Ethernet protocol according to the 25G and 50G Ethernet Consortium specification.

Line Rate

This is the Ethernet line rate as determined by the selected protocol.

System Interface Bus Width

Using the pull-down menu, select the datapath bus width. For MAC configurations, it is the AXI4-Stream, and for PCS only, it is LGMII.

AXI4-Stream Protocol

The AXI4-Stream protocol is only applicable to MAC configurations. It is determined by the selected configuration.

AXI4-Stream or MII Clock

This is the system interface clock frequency as determined by the selected IP configuration.

Internal Bus Width

Using the pull-down menu, select an internal bus width from those available for this configuration. The lower the width, the higher the required clock frequency.

Note: For PCS, this number is determined by the LGMII width. If the current value is stale (for example, from a previous entry), you are prompted to select a new number.

Minimum Core Clock Frequency

The minimum core clock is determined by the chosen bus widths with some margin required for clock tolerance.

Core Clock

Enter the desired core clock frequency, which must be greater than or equal to the minimum recommended. For MAC only, the core clock is the same as the AXI4-Stream clock. The IP is configured using this frequency. Timing might not be met if the frequency is too high. If the value entered is too low or too high, you are prompted to update it.

Note: The final timing analysis needs to be performed by the Xilinx tools; the value specified here is only a starting point.

AN, FEC Options

Auto-Negotiation and/or Forward Error Correction are supported by some configurations. Contact Xilinx Technical Support for utilization information.

Pause Processing

Pause Processing for Priority and Global pause frames is included by default.

IEEE 1588 Options

Please contact Xilinx Technical Support for the latest information regarding support of IEEE 1588.

Evaluation Only

Using the pull-down menu, select Yes to generate IP that is *not* licensed for production. The IP stops working after the evaluation period.

REFCLK

This is the default SerDes reference clock frequency. If a different frequency is required, discuss your requirements with Xilinx Technical Support. The frequency must be available in the GT Wizard.

SerDes Width

The SerDes width is determined by the configuration parameters.

Additional Requirements

The Xilinx High-Speed Ethernet IP core is very configurable. If there are special requirements, discuss them with Xilinx Technical Support and list them here.

Constraining the Core

Each release includes one or more XDC files specific to your configuration and its clocking. Additional constraints might be required when you incorporate the IP into your design. If more information is required, contact Xilinx Technical Support.

Required Constraints

This section is not applicable for this core.

Device, Package, and Speed Grade Selections

This section is not applicable for this core.

Clock Frequencies

This section is not applicable for this core.

Clock Management

This section is not applicable for this core.

Clock Placement

This section is not applicable for this core.

Banking

This section is not applicable for this core.

Transceiver Placement

This section is not applicable for this core.

I/O Standard and Placement

This section is not applicable for this core.

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) [Ref 8].

A demonstration simulation test bench is part of each release. Simulation is performed on the included encrypted RTL. The test bench consists of a loopback from the TX side of the user interface, through the TX circuit, looping back to the RX circuit, and checking the received packets at the RX side of the user interface.

The loopback simulation includes a path through the transceiver. The simulation is run using the provided Linux scripts for several common industry-standard simulators.

For more information, see [Chapter 6, Test Bench](#).

Synthesis and Implementation

If using, use the core generated through spreadsheet submission, instead of the Vivado Catalog. For details about synthesis and implementation, see the *Vivado Design Suite User Guide: Designing with IP* (UG896) [Ref 6].

Example Design

This chapter provides a brief explanation of the 40G/50G Ethernet Subsystem LogiCORE™ example design.

Example Design Hierarchy

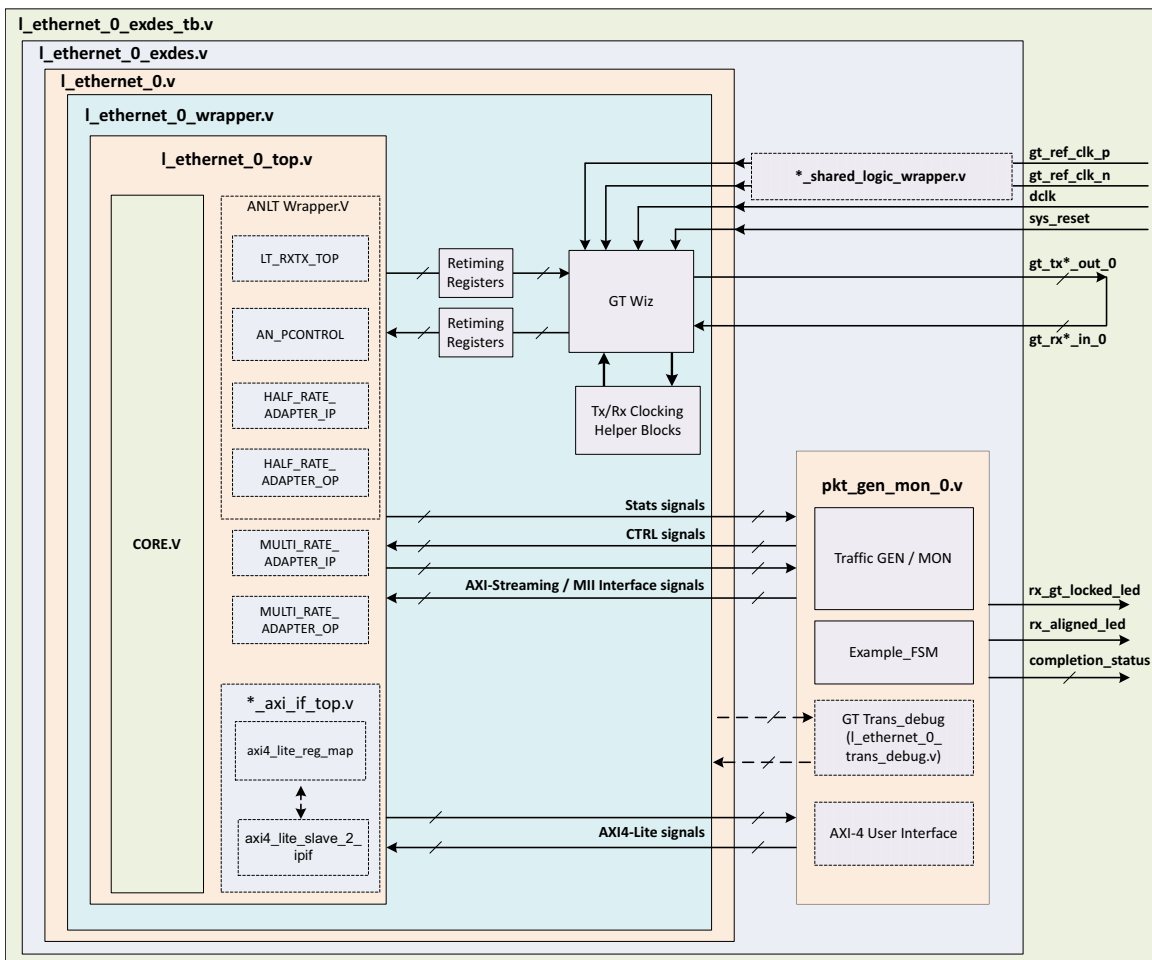


Figure 5-1: Single Core Example Design Hierarchy

Figure 5-1 shows the instantiation of various modules and their hierarchy for a single core configuration of the `l_etherenet_0` example design.

The top module `l_etherenet_0_exdes.v` instantiates the Device Under Test (DUT), that is, `l_etherenet_0.v`, `pkt_gen_mon_0.v` and a few optional modules like `l_etherenet_0_comon_wrapper.v`.

The `l_etherenet_0.v` module instantiates the `l_etherenet_0_wrapper.v` module that has `l_etherenet_0_top.v` and the GT (serial transceiver) along with various helper blocks. Retiming registers are used for the synchronization of data between the core and the GT. Clocking helper blocks are used to generate the required clock frequency for the core.

Following are the user interfaces available for different configurations

- MAC/PCS configuration:
 - AXI4-Stream for data path interface
 - AXI4-Lite for control and statistics interface
- PCS configuration:
 - MII for data path interface
 - AXI4-Lite for control and statistics interface

The `l_etherenet_0_pkt_gen_mon` module is used to generate the data packets for sanity testing. The packet generation and checking is controlled by a Finite State Machine (FSM) module.

Descriptions of optional modules are as follows:

- `l_etherenet_0_trans_debug`

This module is present in the example design inside the `l_etherenet_0_pkt_gen_mon` module when you enable the **Additional GT Control and Status Ports** check box from the GT Selection and Configuration tab of Vivado® IDE. This module brings out all the GT DRP ports and a few control and status ports of the transceiver module out of the `l_etherenet` core.

- `l_etherenet_0_shared_logic_wrapper`

This module is present in the example design when you select the **Include Shared Logic in Example Design** from the Shared Logic tab. This module brings all the logic to the example design that can be shared between multiple cores, like the transceiver common module and the reset modules out of the core.

- Retiming registers:

This module instantiates two-stage registering for the signals between the GT and the core, using `gt_txusrclk2` and `gt_rxusrclk2` for TX and RX paths respectively.

When you select the **Enable Pipeline Register** option from GT Selection and Configuration tab, it includes one more single stage pipeline register between the core and the GT to ease timing, using `gt_txusrclk2` and `gt_rxusrclk2` for TX and RX paths respectively.

Figure 5-2 shows the instantiation of various modules and their hierarchy for the multiple core configuration of the `I_ethernet_0` example design.

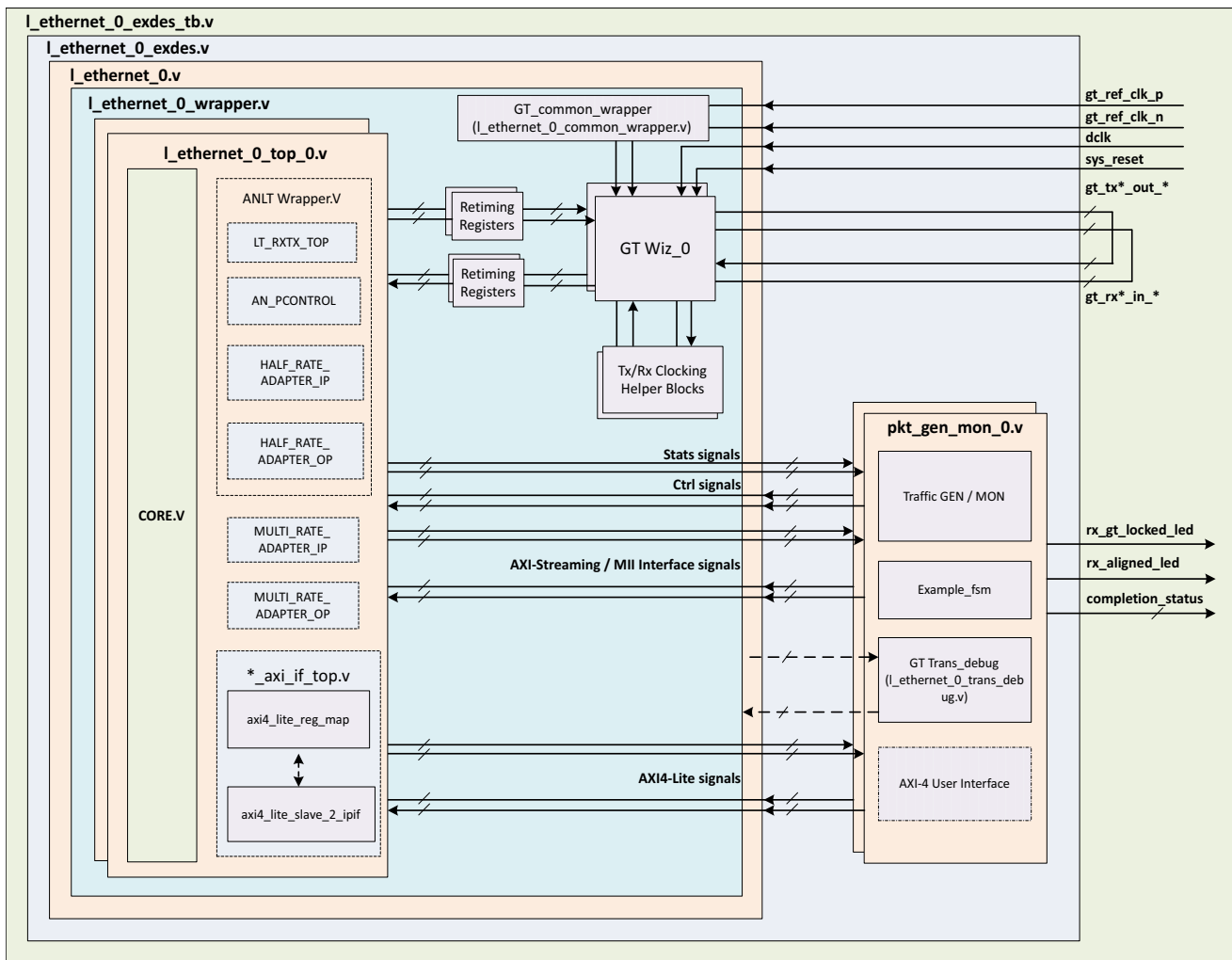


Figure 5-2: Multiple Core Example Design Hierarchy

User Interface

GPIOs have been provided to control the example design:

Table 5-1: Table 1. User Input / Output Ports

| Name | Size | Direction | Description |
|-------------------|------|-----------|---|
| sys_reset | 1 | Input | Reset for l_ethernet core |
| gt_refclk_p | 1 | Input | Differential input clk to GT |
| gt_refclk_n | 1 | Input | Differential input clk to GT |
| dclk | 1 | Input | Stable/free running input clk to GT |
| rx_gt_locked_led | 1 | Output | Indicates that GT has been locked. |
| rx_aligned_led | 1 | Output | Indicates RX aligned has been achieved |
| completion_status | | | <p>This signal represents the test status/result.</p> <ul style="list-style-type: none"> • 5'd0: Test did not run. • 5'd1: PASSED 50GE/40GE CORE TEST SUCCESSFULLY COMPLETED • 5'd2: No block lock on any lanes. • 5'd3: Not all lanes achieved block lock. • 5'd4: Some lanes lost block lock after achieving block lock. • 5'd5: No lane sync on any lanes. • 5'd6: Not all lanes achieved sync. • 5'd7: Some lanes lost sync after achieving sync. • 5'd8: No alignment status or rx_status was achieved. • 5'd9: Loss of alignment status or rx_status after both were achieved. • 5'd10: TX timed out. • 5'd11: No tx data was sent. • 5'd12: Number of packets received did not equal the number of packets sent. • 5'd13: Total number of bytes received did not equal the total number of bytes sent. • 5'd14: An protocol error was detected. • 5'd15: Bit errors were detected in the received packets. • 5'd31: Test is stuck in reset. |

Core xci Top Level Port List

In the following table an asterisk (*) represents the core number, having a value of 0 and 1.

Example: Port_NAME_*

- Port_NAME_0: for first core
- Port_NAME_1: for second core (is present when you select number of cores 2)

Table 5-2: Top Level Port List

| Name | Size | Direction | Description |
|-----------------------------------|------|-----------|--|
| Common Clock/Reset Signals | | | |
| sys_reset | 1 | Input | Reset for core. |
| dclk | 1 | Input | Stable/free running input clk to GT |
| gt_refclk_p | 1 | Input | Differential input clk to GT. Note: This port is available when the Include Shared Logic in core option is selected in the Shared Logic tab. |
| gt_refclk_n | 1 | Input | Differential input clk to GT. Note: This port is available when the Include Shared Logic in core option is selected in the Shared Logic tab. |
| qpll0clk_in | 2/4 | Input | QPLL0 clock input. (QPPL is quad phase-locked loop) Note: This port is available when the Include Shared Logic in example design option is selected in the Shared Logic tab. Port width: 2 bits for the 50G single core and 4 bit for 40G one core and 50G two cores. |
| qpll0refclk_in | 2/4 | Input | QPLL0 ref clock input. Note: This port is available when the Include Shared Logic in example design option is selected in the Shared Logic tab. Port width: 2 bits for the 50G single core and 4 bit for 40G one core and 50G two cores. |
| qpll1clk_in | 2/4 | Input | QPLL1 clock input. Note: This port is available when the Include Shared Logic in example design option is selected in the Shared Logic tab. Port width: 2 bits for the 50G single core and 4 bits for 40G one core and 50G two cores. |

Table 5-2: Top Level Port List (Cont'd)

| Name | Size | Direction | Description |
|----------------------------|------|-----------|---|
| qpll1refclk_in | 2/4 | Input | QPLL1 ref clock input. Note: This port is available when the Include Shared Logic in example design option is selected in the Shared Logic tab. Port width: 2 bits for the 50G single core and 4 bits for 40G one core and 50G two cores. |
| gtwiz_reset_qpll0lock_in | 1 | Input | QPLL0 lock reset input to the GT. This port is available when the Include Shared Logic in example design option is selected in the Shared Logic tab. |
| gtwiz_reset_qpll0reset_out | 1 | Output | QPLL0 lock reset output from the GT. This port is available when the Include Shared Logic in example design option is selected in the Shared Logic tab. |
| tx_clk_out_* | 1 | Output | TX user clock output from GT. Note: This port is available when the core type is Ethernet MAC+PCS/PMA. |
| tx_mii_clk_* | 1 | Output | TX user clock output from GT. Note: This port is available when the core type is Ethernet PCS/PMA. |
| rx_clk_out_* | 1 | Output | RX user clock output from GT. |
| tx_reset_* | 1 | Input | TX reset input to the core. |
| user_tx_reset_* | 1 | Output | TX reset output for the user logic. Note: This port is available when the Include Shared Logic in core option is selected in the Shared Logic tab. |
| gt_reset_tx_done_out_* | 1 | Output | TX reset done signal from the GT. Note: This port is available when the Include Shared Logic in example design option is selected in the Shared Logic tab. |
| rx_reset_* | 1 | Input | RX reset input to the core. |
| user_rx_reset_* | 1 | Output | RX reset output for the user logic. Note: This port is available when the Include Shared Logic in core option is selected in the Shared Logic tab. |
| gt_reset_rx_done_out_* | 1 | Output | RX reset done signal from the GT. Note: This port is available when the Include Shared Logic in example design option is selected in the Shared Logic tab. |
| rx_serdes_reset_* | 1 | Input | RX serdes reset signal. Note: This port is available when the Include Shared Logic in example design option is selected in the Shared Logic tab. |

Table 5-2: Top Level Port List (Cont'd)

| Name | Size | Direction | Description |
|--------------------|------|-----------|--|
| ctl_gt_reset_all_* | 1 | Output | gt_reset_all signal from the AXI4-Lite register map. Note: This port is available when Include AXI4-Lite is selected from the Configuration tab and the Include Shared Logic in example design option is selected in the Shared Logic tab. |
| ctl_gt_tx_reset_* | 1 | Output | gt_tx_reset signal from the AXI4 lite register map. Note: This port is available when Include AXI4-Lite is selected from the Configuration tab and the Include Shared Logic in example design option is selected in the Shared Logic tab. |
| ctl_gt_rx_reset_* | 1 | Output | gt_rx_reset signal from the AXI4-Lite register map. Note: This port is available when Include AXI4-Lite is selected from the Configuration tab and the Include Shared Logic in example design option is selected in the Shared Logic tab. |
| gt_reset_all_in_* | 1 | Input | gt_reset_all signal from the reset_wrapper of shared logic wrapper. Note: This port is available when the Include Shared Logic in example design option is selected in the Shared Logic tab. |
| gt_tx_reset_in_* | 1 | Input | gt_tx_reset_in signal from reset_wrapper of shared logic wrapper. Note: This port is available when the Include Shared Logic in example design option is selected in the Shared Logic tab. |
| gt_rx_reset_in_* | 1 | Input | gt_rx_reset_in signal from reset_wrapper of shared logic wrapper. Note: This port is available when the Include Shared Logic in example design option is selected in the Shared Logic tab. |

Table 5-2: Top Level Port List (Cont'd)

| Name | Size | Direction | Description |
|---------------------------------|------|-----------|---|
| Common Transceiver Ports | | | |
| gt_loopback_in | 6/12 | Input | GT loopback input signal. Refer to the GT user guide. Note: 6-bit width for the 50G single core, 12-bit width for 40G single core/ 50G two cores) |
| gt_rxp_in_0 | 1 | Input | Differential serial GT RX input for lane 0. |
| gt_rxn_in_0 | 1 | Input | Differential serial GT RX input for lane 0. |
| gt_rxp_in_1 | 1 | Input | Differential serial GT RX input for lane 1. |
| gt_rxn_in_1 | 1 | Input | Differential serial GT RX input for lane 1. |
| gt_rxp_in_2 | 1 | Input | Differential serial GT RX input for lane 2. Note: This port is available for 40G and 50G two cores. |
| gt_rxn_in_2 | 1 | Input | Differential serial GT RX input for lane 2. Note: This port is available for 40G and 50G two cores. |
| gt_rxp_in_3 | 1 | Input | Differential serial GT RX input for lane 3. Note: This port is available for 40G and 50G two cores. |
| gt_rxn_in_3 | 1 | Input | Differential serial GT RX input for lane 3. Note: This port is available for 40G and 50G two cores. |
| gt_txp_out_0 | 1 | Output | Differential serial GT TX output for lane 0. |
| gt_txn_out_0 | 1 | Output | Differential serial GT TX output for lane 0. |
| gt_txp_out_1 | 1 | Output | Differential serial GT TX output for lane 1. |
| gt_txn_out_1 | 1 | Output | Differential serial GT TX output for lane 1. |
| gt_txp_out_2 | 1 | Output | Differential serial GT TX output for lane 2. Note: This port is available for 40G and 50G two cores. |
| gt_txn_out_2 | 1 | Output | Differential serial GT TX output for lane 2. Note: This port is available for 40G and 50G two core. |
| gt_txp_out_3 | 1 | Output | Differential serial GT TX output for lane 3. Note: This port is available for 40G and 50G two cores. |
| gt_txn_out_3 | 1 | Output | Differential serial GT TX output for lane 3. Note: This port is available for 40G and 50G two cores. |

Table 5-2: Top Level Port List (Cont'd)

| Name | Size | Direction | Description |
|---|-------|-----------|---|
| Transceiver control and status debug ports | | | |
| Ports under this section will be available when Enable Additional GT Control/Status and DRP Ports is selected from the GT Selection and Configuration tab. | | | |
| gt_dmonitorout_* | 34/68 | Output | Refer to the GT user guide for the port description. Note: Port width: 34 bits for the 50G single core and 68 bits for 40G. |
| gt_eyes candataerror_* | 2/4 | Output | Refer to the GT user guide for the port description. Note: Port width: 2 bits for the 50G single core and 4 bits for 40G. |
| gt_eyes canreset_* | 2/4 | Input | Refer to the GT user guide for the port description. Note: Port width: 2 bits for the 50G single core and 4 bits for 40G. |
| gt_eyes cantrigger_* | 2/4 | Input | Refer to the GT user guide for the port description. Note: Port width: 2 bits for the 50G single core and 4 bits for 40G. |
| gt_pcs rsvdin_* | 32/64 | Input | Refer to the GT user guide for the port description. Note: Port width: 32 bits for the 50G single core and 64 bits for 40G. |
| gt_rxbufreset_* | 2/4 | Input | Refer to the GT user guide for the port description. Note: Port width: 2 bits for the 50G single core and 4 bits for 40G. |
| gt_rxbufstatus_* | 6/12 | Output | Refer to the GT user guide for the port description. Note: Port width: 6-bit for the 50G single core and 12 bits for 40G. |
| gt_rxcdrhold_* | 2/4 | Input | Refer to the GT user guide for the port description. Note: Port width: 2 bits for the 50G single core and 4 bits for 40G. |
| gt_rxcommadeten_* | 2/4 | Input | Refer to the GT user guide for the port description. Note: Port width: 2 bits for the 50G single core and 4 bits for 40G. |
| gt_rxdfeagchold_* | 2/4 | Input | Refer to the GT user guide for the port description. Note: Port width: 2 bits for the 50G single core and 4 bits for 40G. |

Table 5-2: Top Level Port List (Cont'd)

| Name | Size | Direction | Description |
|---------------------|------|-----------|---|
| gt_rxdfelpmreset_* | 2/4 | Input | Refer to the GT user guide for the port description. Note: Port width: 2 bits for the 50G single core and 4 bits for 40G. |
| gt_rxlataclk_* | 2/4 | Input | Refer to the GT user guide for the port description. Note: Port width: 2 bits for the 50G single core and 4 bits for 40G. |
| gt_rxlpmen_* | 2/4 | Input | Refer to the GT user guide for the port description. Note: Port width: 2 bits for the 50G single core and 4 bits for 40G. |
| gt_rxpcsreset_* | 2/4 | Input | Refer to the GT user guide for the port description. Note: Port width: 2 bits for the 50G single core and 4 bits for 40G. |
| gt_rxpmareset_* | 2/4 | Input | Refer to the GT user guide for the port description. Note: Port width: 2 bits for the 50G single core and 4 bits for 40G. |
| gt_rxpolarity_* | 2/4 | Input | Refer to the GT user guide for the port description. Note: Port width: 2 bits for the 50G single core and 4 bits for 40G. |
| gt_rxprbscntreset_* | 2/4 | Input | Refer to the GT user guide for the port description. Note: Port width: 2 bits for the 50G single core and 4 bits for 40G. |
| gt_rxprbserr_* | 2/4 | Input | Refer to the GT user guide for the port description. Note: Port width: 2 bits for the 50G single core and 4 bits for 40G. |
| gt_rxprbsssel_* | 8/16 | Input | Refer to the GT user guide for the port description. Note: Port width: 8-bit for the 50G single core and 16 bits for 40G. |
| gt_rxrate_* | 6/12 | Input | Refer to the GT user guide for the port description. Note: Port width: 6-bit for the 50G single core and 12 bits for 40G. |
| gt_rxslide_in_* | 2/4 | Input | Refer to the GT user guide for the port description. Note: Port width: 2 bits for the 50G single core and 4 bits for 40G. |

Table 5-2: Top Level Port List (Cont'd)

| Name | Size | Direction | Description |
|---------------------|-------|-----------|--|
| gt_rxstartofseq_* | 4/8 | Output | Refer to the GT user guide for the port description. Note: Port width: 4-bit for the 50G single core and 8 bits for 40G. |
| gt_txbufstatus_* | 4/8 | Output | Refer to the GT user guide for the port description. Note: Port width: 4-bit for the 50G single core and 8 bits for 40G. |
| gt_txdiffctrl_* | 10/20 | Input | Refer to the GT user guide for the port description. Note: Port width: 10-bit for the 50G single core and 20 bits for 40G. |
| gt_txinhibit_* | 2/4 | Input | Refer to the GT user guide for the port description. Note: Port width: 2 bits for the 50G single core and 4 bits for 40G. |
| gt_txlatclk_* | 2/4 | Input | Refer to the GT user guide for the port description. Note: Port width: 2 bits for the 50G single core and 4 bits for 40G. |
| gt_txmaincursor_* | 14/28 | Input | Refer to the GT user guide for the port description. Note: Port width: 14-bit for the 50G single core and 28 bits for 40G. |
| gt_txpcsreset_* | 2/4 | Input | Refer to the GT user guide for the port description. Note: Port width: 2 bits for the 50G single core and 4 bits for 40G. |
| gt_txpmareset_* | 2/4 | Input | Refer to the GT user guide for the port description. Note: Port width: 2 bits for the 50G single core and 4 bits for 40G. |
| gt_txpolarity_* | 2/4 | Input | Refer to the GT user guide for the port description. Note: Port width: 2 bits for the 50G single core and 4 bits for 40G. |
| gt_txpostcursor_* | 10/20 | Input | Refer to the GT user guide for the port description. Note: Port width: 10-bit for the 50G single core and 20 bits for 40G. |
| gt_txprbsforceerr_* | 2/4 | Input | Refer to the GT user guide for the port description. Note: Port width: 2 bits for the 50G single core and 4 bits for 40G. |

Table 5-2: Top Level Port List (Cont'd)

| Name | Size | Direction | Description |
|---------------------------|-------|-----------|--|
| gt_txprbssel_* | 8/16 | Input | Refer to the GT user guide for the port description. Note: Port width: 8-bit for the 50G single core and 16 bits for 40G. |
| gt_txprecursor_* | 10/20 | Input | Refer to the GT user guide for the port description. Note: Port width: 10-bit for the 50G single core and 20 bits for 40G. |
| gtwiz_reset_tx_datapath_* | 2/4 | Input | Refer to the GT user guide for the port description. Note: Port width: 2 bits for the 50G single core and 4 bits for 40G. |
| gtwiz_reset_rx_datapath_* | 2/4 | Input | Refer to the GT user guide for the port description. Note: Port width: 2 bits for the 50G single core and 4 bits for 40G. |
| gt_common_drpclk | 1 | Input | Refer to the GT user guide for the port description. Note: This port is available when the Include Shared Logic in core option is selected in the Shared Logic tab. |
| gt_common_drpdo | 16 | Output | Refer to the GT user guide for the port description. Note: This port is available when the Include Shared Logic in core option is selected in the Shared Logic tab. |
| gt_common_drprdy | 1 | Output | Refer to the GT user guide for the port description. Note: This port is available when the Include Shared Logic in core option is selected in the Shared Logic tab. |
| gt_common_drpen | 1 | Input | Refer to the GT user guide for the port description. Note: This port is available when the Include Shared Logic in core option is selected in the Shared Logic tab. |
| gt_common_drpwe | 1 | Input | Refer to the GT user guide for the port description. Note: This port is available when the Include Shared Logic in core option is selected in the Shared Logic tab. |
| gt_common_drpaddr | 10 | Input | Refer to the GT user guide for the port description. Note: This port is available when the Include Shared Logic in core option is selected in the Shared Logic tab. |

Table 5-2: Top Level Port List (Cont'd)

| Name | Size | Direction | Description |
|-----------------|------|-----------|--|
| gt_common_drpdi | 16 | Input | Refer to the GT user guide for the port description. Note: This port is available when the Include Shared Logic in core option is selected in the Shared Logic tab. |
| gt_ch_drpclk_0 | 1 | Input | Refer to the GT user guide for the port description. |
| gt_ch_drpdo_0 | 16 | Output | Refer to the GT user guide for the port description. |
| gt_ch_drprdy_0 | 1 | Output | Refer to the GT user guide for the port description. |
| gt_ch_drpen_0 | 1 | Input | Refer to the GT user guide for the port description. |
| gt_ch_drpwe_0 | 1 | Input | Refer to the GT user guide for the port description. |
| gt_ch_drpaddr_0 | 10 | Input | Refer to the GT user guide for the port description. |
| gt_ch_drpdi_0 | 16 | Input | Refer to the GT user guide for the port description. |
| gt_ch_drpclk_1 | 1 | Input | Refer to the GT user guide for the port description. |
| gt_ch_drpdo_1 | 16 | Output | Refer to the GT user guide for the port description. |
| gt_ch_drprdy_1 | 1 | Output | Refer to the GT user guide for the port description. |
| gt_ch_drpen_1 | 1 | Input | Refer to the GT user guide for the port description. |
| gt_ch_drpwe_1 | 1 | Input | Refer to the GT user guide for the port description. |
| gt_ch_drpaddr_1 | 10 | Input | Refer to the GT user guide for the port description. |
| gt_ch_drpdi_1 | 16 | Input | Refer to the GT user guide for the port description. |
| gt_ch_drpclk_2 | 1 | Input | Refer to the GT user guide for the port description. Note: This port is available when core speed 40G / speed 50G with two cores. |
| gt_ch_drpdo_2 | 16 | Output | Refer to the GT user guide for the port description. Note: This port is available when core speed 40G / speed 50G with two cores. |

Table 5-2: Top Level Port List (Cont'd)

| Name | Size | Direction | Description |
|-----------------|------|-----------|---|
| gt_ch_drprdy_2 | 1 | Output | Refer to the GT user guide for the port description. Note: This port is available when core speed 40G / speed 50G with two cores. |
| gt_ch_drpen_2 | 1 | Input | Refer to the GT user guide for the port description. Note: This port is available when core speed 40G / speed 50G with two cores. |
| gt_ch_drpwe_2 | 1 | Input | Refer to the GT user guide for the port description. Note: This port is available when core speed 40G / speed 50G with two cores. |
| gt_ch_drpaddr_2 | 10 | Input | Refer to the GT user guide for the port description. Note: This port is available when core speed 40G / speed 50G with two cores. |
| gt_ch_drpdi_2 | 16 | Input | Refer to the GT user guide for the port description. Note: This port is available when core speed 40G / speed 50G with two cores. |
| gt_ch_drpcl_3 | 1 | Input | Refer to the GT user guide for the port description. Note: This port is available when core speed 40G / speed 50G with two cores. |
| gt_ch_drpdo_3 | 16 | Output | Refer to the GT user guide for the port description. Note: This port is available when core speed 40G / speed 50G with two cores. |
| gt_ch_drprdy_3 | 1 | Output | Refer to the GT user guide for the port description. Note: This port is available when core speed 40G / speed 50G with two cores. |
| gt_ch_drpen_3 | 1 | Input | Refer to the GT user guide for the port description. Note: This port is available when core speed 40G / speed 50G with two cores. |
| gt_ch_drpwe_3 | 1 | Input | Refer to the GT user guide for the port description. Note: This port is available when core speed 40G / speed 50G with two cores. |
| gt_ch_drpaddr_3 | 10 | Input | Refer to the GT user guide for the port description. Note: This port is available when core speed 40G / speed 50G with two cores. |

Table 5-2: Top Level Port List (Cont'd)

| Name | Size | Direction | Description |
|--|------|-----------|--|
| gt_ch_drpd_3 | 16 | Input | Refer to the GT user guide for the port description. Note: This port is available when core speed 40G / speed 50G with two cores. |
| AXI4-Lite Interface Ports | | | |
| Ports under this section will be available when Include AXI4-Lite is selected from the Configuration tab. | | | |
| s_axi_aclk_* | 1 | Input | AXI clock signal |
| s_axi_aresetn_* | 1 | Input | AXI reset signal |
| pm_tick_* | 1 | Input | PM tick user input |
| s_axi_awaddr_* | 32 | Input | AXI write address |
| s_axi_awvalid_* | 1 | Input | AXI write address valid |
| s_axi_awready_* | 1 | Output | AXI write address ready |
| s_axi_wdata_* | 32 | Input | AXI write data |
| s_axi_wstrb_* | 4 | Input | AXI write strobe. This signal indicates which byte lanes hold valid data. |
| s_axi_wvalid_* | 1 | Input | AXI write data valid. This signal indicates that valid write data and strobes are available. |
| s_axi_wready_* | 1 | Output | AXI write data ready |
| s_axi_bresp_* | 2 | Output | AXI write response. This signal indicates the status of the write transaction. 'b00 = OKAY 'b01 = EXOKAY 'b10 = SLVERR 'b11 = DECERR |
| s_axi_bvalid_* | 1 | Output | AXI write response valid. This signal indicates that the channel is signaling a valid write response. |
| s_axi_bready_* | 1 | Input | AXI write response ready. |
| s_axi_araddr_* | 32 | Input | AXI read address |
| s_axi_arvalid_* | 1 | Input | AXI read address valid |
| s_axi_arready_* | 1 | Output | AXI read address ready |
| s_axi_rdata_* | 32 | Output | AXI read data issued by slave |

Table 5-2: Top Level Port List (Cont'd)

| Name | Size | Direction | Description | | | | | | | | | | | | | | | | | | | | | | |
|---|-------------------------|-----------|--|----|--------------------|-------|-------------------------|----|--------------------|----|--------------------|----|--------------------|----|--------------------|-------|-------------------------|----|--------------------|----|--------------------|----|--------------------|------|---------------|
| s_axi_rresp_* | 2 | Output | AXI read response. This signal indicates the status of the read transfer. 'b00 = OKAY 'b01 = EXOKAY 'b10 = SLVERR 'b11 = DECERR | | | | | | | | | | | | | | | | | | | | | | |
| s_axi_rvalid_* | 1 | Output | AXI read data valid | | | | | | | | | | | | | | | | | | | | | | |
| s_axi_rready_* | 1 | Input | AXI read ready. This signal indicates the user/master can accept the read data and response information. | | | | | | | | | | | | | | | | | | | | | | |
| AXI4-Stream User Interface Signals | | | | | | | | | | | | | | | | | | | | | | | | | |
| Ports under this section will be available when Ethernet MAC+PCS/PMA is selected from the Configuration tab. | | | | | | | | | | | | | | | | | | | | | | | | | |
| tx_unfout_* | 1 | Output | Underflow signal for TX path from core. | | | | | | | | | | | | | | | | | | | | | | |
| tx_axis_tready_* | 1 | Output | TX path ready signal from core. | | | | | | | | | | | | | | | | | | | | | | |
| tx_axis_tvalid_* | 1 | Input | Transmit AXI Stream Data valid. | | | | | | | | | | | | | | | | | | | | | | |
| tx_axis_tdata_* | 128 | Input | Transmit AXI Stream Data bus. | | | | | | | | | | | | | | | | | | | | | | |
| tx_axis_tuser_* | 70 | Input | TX segment and packet information signal. tx_axis_tuser_0[69:0] <table border="1" style="margin-left: 20px;"> <tr><td>69</td><td>tx_axis_tuser_err1</td></tr> <tr><td>68:66</td><td>tx_axis_tuser_mty1[2:0]</td></tr> <tr><td>65</td><td>tx_axis_tuser_eop1</td></tr> <tr><td>64</td><td>tx_axis_tuser_sop1</td></tr> <tr><td>63</td><td>tx_axis_tuser_ena1</td></tr> <tr><td>62</td><td>tx_axis_tuser_err0</td></tr> <tr><td>61:59</td><td>tx_axis_tuser_mty0[2:0]</td></tr> <tr><td>58</td><td>tx_axis_tuser_eop0</td></tr> <tr><td>57</td><td>tx_axis_tuser_sop0</td></tr> <tr><td>56</td><td>tx_axis_tuser_ena0</td></tr> <tr><td>55:0</td><td>tx_preamblein</td></tr> </table> | 69 | tx_axis_tuser_err1 | 68:66 | tx_axis_tuser_mty1[2:0] | 65 | tx_axis_tuser_eop1 | 64 | tx_axis_tuser_sop1 | 63 | tx_axis_tuser_ena1 | 62 | tx_axis_tuser_err0 | 61:59 | tx_axis_tuser_mty0[2:0] | 58 | tx_axis_tuser_eop0 | 57 | tx_axis_tuser_sop0 | 56 | tx_axis_tuser_ena0 | 55:0 | tx_preamblein |
| 69 | tx_axis_tuser_err1 | | | | | | | | | | | | | | | | | | | | | | | | |
| 68:66 | tx_axis_tuser_mty1[2:0] | | | | | | | | | | | | | | | | | | | | | | | | |
| 65 | tx_axis_tuser_eop1 | | | | | | | | | | | | | | | | | | | | | | | | |
| 64 | tx_axis_tuser_sop1 | | | | | | | | | | | | | | | | | | | | | | | | |
| 63 | tx_axis_tuser_ena1 | | | | | | | | | | | | | | | | | | | | | | | | |
| 62 | tx_axis_tuser_err0 | | | | | | | | | | | | | | | | | | | | | | | | |
| 61:59 | tx_axis_tuser_mty0[2:0] | | | | | | | | | | | | | | | | | | | | | | | | |
| 58 | tx_axis_tuser_eop0 | | | | | | | | | | | | | | | | | | | | | | | | |
| 57 | tx_axis_tuser_sop0 | | | | | | | | | | | | | | | | | | | | | | | | |
| 56 | tx_axis_tuser_ena0 | | | | | | | | | | | | | | | | | | | | | | | | |
| 55:0 | tx_preamblein | | | | | | | | | | | | | | | | | | | | | | | | |
| rx_axis_tvalid_* | 1 | Output | Receive AXI4-Stream Data valid. | | | | | | | | | | | | | | | | | | | | | | |
| rx_axis_tdata_* | 128 | Output | Receive AXI4-Stream Data bus. | | | | | | | | | | | | | | | | | | | | | | |

Table 5-2: Top Level Port List (Cont'd)

| Name | Size | Direction | Description | |
|---|---------------|-----------|---|-------------------------|
| rx_axis_tuser | 70 | Output | RX segment and packet information signal. rx_axis_tuser_0[69:0] | |
| | | | 69 | rx_axis_tuser_err1 |
| | | | 68:66 | rx_axis_tuser_mty1[2:0] |
| | | | 65 | rx_axis_tuser_eop1 |
| | | | 64 | rx_axis_tuser_sop1 |
| | | | 63 | rx_axis_tuser_ena1 |
| | | | 62 | rx_axis_tuser_err0 |
| | | | 61:59 | rx_axis_tuser_mty0[2:0] |
| | | | 58 | rx_axis_tuser_eop0 |
| | | | 57 | rx_axis_tuser_sop0 |
| | | | 56 | rx_axis_tuser_ena0 |
| 55:0 | rx_preamblein | | | |
| MII User Interface Signals | | | | |
| Ports under this section will be available when Ethernet PCS/PMA is selected from the Configuration tab. | | | | |
| tx_mii_d_* | 128 | Input | Transmit LGMII Data bus. | |
| tx_mii_c_* | 16 | Input | LGMII Control bus. | |
| rx_mii_d_* | 128 | Output | Receive LGMII Data bus. | |
| rx_mii_c_* | 16 | Output | Receive LGMII Control bus. | |
| TX Path Control / Status / Statistics Signals | | | | |
| ctl_tx_vl_length_minus1_* | 16 | Input | Number of words in between PCS Lane markers minus one. Default value, as defined in IEEE Std 802.3-2012, should be set to 16,383. This input should only be changed while the corresponding reset input is asserted. Note: This port is available when the AXI4-Lite interface is not selected. | |
| ctl_tx_test_pattern_* | 1 | Input | Test pattern generation enable for the TX core. A value of 1 enables test mode as defined in Clause 82.2.10. Corresponds to MDIO register bit 3.42.3 as defined in Clause 82.3. Generates a scrambled idle pattern. Note: This port is available when the AXI4-Lite interface is not selected. | |

Table 5-2: Top Level Port List (Cont'd)

| Name | Size | Direction | Description |
|-------------------------|------|-----------|--|
| ctl_tx_enable_* | 1 | Input | <p>TX Enable. This signal is used to enable the transmission of data when it is sampled as a 1. When sampled as a 0, only idles are transmitted by the CORE</p> <p>Note: This port is available when the AXI4-Lite interface is not selected and core type is Ethernet MAC+PCS/PMA.</p> |
| ctl_tx_fcs_ins_enable_* | 1 | Input | <p>Enable FCS insertion by the TX core. If this bit is set to 0, the 40G/50G High Speed Ethernet Subsystem does not add FCS to packet. If this bit is set to 1, the 40G/50G High Speed Ethernet Subsystem calculates and adds the FCS to the packet. This input cannot be changed dynamically between packets.</p> <p>Note: This port is available when the AXI4-Lite interface is not selected and core type is Ethernet MAC+PCS/PMA.</p> |
| ctl_tx_ipg_value_* | 4 | Input | <p>This signal may be optionally present. The <code>ctl_tx_ipg_value</code> defines the target average minimum Inter Packet Gap (IPG, in bytes) inserted between <code>rx_serdes_clk</code> packets. Valid values are 8 to 12. The <code>ctl_tx_ipg_value</code> can also be programmed to a value in the 0 to 7 range, but in that case, it is interpreted as meaning "minimal IPG", so only Terminate code word IPG is inserted; no Idles are ever added in that case and that produces an average IPG of around 4 bytes when random-size packets are transmitted.</p> <p>Note: This port is available when the AXI4-Lite interface is not selected and core type is Ethernet MAC+PCS/PMA and Include FIFO Logic is disabled.</p> |
| ctl_tx_send_lfi_* | 1 | Input | <p>Transmit Local Fault Indication (LFI) code word. Takes precedence over RFI.</p> <p>Note: This port is available when core type is Ethernet MAC+PCS/PMA.</p> |
| ctl_tx_send_rfi_* | 1 | Input | <p>Transmit Remote Fault Indication (RFI) code word.</p> <p>Note: This port is available when core type is Ethernet MAC+PCS/PMA.</p> |
| ctl_tx_send_idle_* | 1 | Input | <p>Transmit Idle code words. If this input is sampled as a 1, the TX path only transmits Idle code words. This input should be set to 1 when the partner device is sending Remote Fault Indication (RFI) code words.</p> <p>Note: This port is available when core type is Ethernet MAC+PCS/PMA.</p> |

Table 5-2: Top Level Port List (Cont'd)

| Name | Size | Direction | Description |
|---------------------------------|------|-----------|---|
| ctl_tx_custom_preamble_enable_* | 1 | Input | When asserted, this signal enables the use of tx_preamblein as a custom preamble instead of inserting a standard preamble. Note: This port is available when the AXI4-Lite interface is not selected and core type is Ethernet MAC+PCS/PMA and Include FIFO Logic is disabled |
| ctl_tx_ignore_fcs_* | 1 | Input | Enable FCS error checking at the AXI4-Stream interface by the TX core. This input only has effect when ctl_tx_fcs_ins_enable is Low. If this input is Low and a packet with bad FCS is being transmitted, it is not binned as good. If this input is High, a packet with bad FCS is binned as good. The error is flagged on the signals stat_tx_bad_fcs and stomped_fcs, and the packet is transmitted as it was received. Note: Statistics are reported as if there was no FCS error. Note: This port is available when the AXI4-Lite interface is not selected and core type is Ethernet MAC+PCS. |
| ctl_tx_vl_marker_id0_* | 64 | Input | These inputs set the PCS Lane markers for each PCS lane. For 802.3 default values, see the IEEE Std 802.3-2012. This input should only be changed while the corresponding reset input is asserted. |
| ctl_tx_vl_marker_id1_* | 64 | Input | These inputs set the PCS Lane markers for each PCS lane. For 802.3 default values, see the IEEE Std 802.3-2012. This input should only be changed while the corresponding reset input is asserted. |
| ctl_tx_vl_marker_id2_* | 64 | Input | These inputs set the PCS Lane markers for each PCS lane. For 802.3 default values, see the IEEE Std 802.3-2012. This input should only be changed while the corresponding reset input is asserted. |
| ctl_tx_vl_marker_id3_* | 64 | Input | These inputs set the PCS Lane markers for each PCS lane. For 802.3 default values, see the IEEE Std 802.3-2012. This input should only be changed while the corresponding reset input is asserted. |
| stat_tx_total_packets_* | 1 | Output | Increment for the total number of packets transmitted. Note: This port is available when core type is Ethernet MAC+PCS/PMA. |
| stat_tx_total_bytes_* | 5 | Output | Increment for the total number of bytes transmitted. Note: This port is available when core type is Ethernet MAC+PCS/PMA. |

Table 5-2: Top Level Port List (Cont'd)

| Name | Size | Direction | Description |
|----------------------------------|------|-----------|--|
| stat_tx_total_good_packets_* | 1 | Output | Increment for the total number of good packets transmitted. Note: This port is available when core type is Ethernet MAC+PCS/PMA. |
| stat_tx_total_good_bytes_* | 14 | Output | Increment for the total number of good bytes transmitted. This value is only non-zero when a packet is transmitted completely and contains no errors. Note: This port is available when core type is Ethernet MAC+PCS/PMA. |
| stat_tx_packet_64_bytes_* | 1 | Output | Increment for good and bad packets transmitted that contain 64 bytes. Note: This port is available when core type is Ethernet MAC+PCS/PMA. |
| stat_tx_packet_65_127_bytes_* | 1 | Output | Increment for good and bad packets transmitted that contain 65 to 127 bytes. Note: This port is available when core type is Ethernet MAC+PCS/PMA. |
| stat_tx_packet_128_255_bytes_* | 1 | Output | Increment for good and bad packets transmitted that contain 128 to 255 bytes. Note: This port is available when core type is Ethernet MAC+PCS/PMA. |
| stat_tx_packet_256_511_bytes_* | 1 | Output | Increment for good and bad packets transmitted that contain 256 to 511 bytes. Note: This port is available when core type is Ethernet MAC+PCS/PMA. |
| stat_tx_packet_512_1023_bytes_* | 1 | Output | Increment for good and bad packets transmitted that contain 512 to 1,023 bytes. Note: This port is available when core type is Ethernet MAC+PCS/PMA. |
| stat_tx_packet_1024_1518_bytes_* | 1 | Output | Increment for good and bad packets transmitted that contain 1,024 to 1,518 bytes. Note: This port is available when core type is Ethernet MAC+PCS/PMA. |
| stat_tx_packet_1519_1522_bytes_* | 1 | Output | Increment for good and bad packets transmitted that contain 1,519 to 1,522 bytes. Note: This port is available when core type is Ethernet MAC+PCS/PMA. |
| stat_tx_packet_1523_1548_bytes_* | 1 | Output | Increment for good and bad packets transmitted that contain 1,523 to 1,548 bytes. Note: This port is available when core type is Ethernet MAC+PCS/PMA. |

Table 5-2: Top Level Port List (Cont'd)

| Name | Size | Direction | Description |
|----------------------------------|------|-----------|---|
| stat_tx_packet_1549_2047_bytes_* | 1 | Output | Increment for good and bad packets transmitted that contain 1,549 to 2,047 bytes. Note: This port is available when core type is Ethernet MAC+PCS/PMA. |
| stat_tx_packet_2048_4095_bytes_* | 1 | Output | Increment for good and bad packets transmitted that contain 2,048 to 4,095 bytes. Note: This port is available when core type is Ethernet MAC+PCS/PMA. |
| stat_tx_packet_4096_8191_bytes_* | 1 | Output | Increment for good and bad packets transmitted that contain 4,096 to 8,191 bytes. Note: This port is available when core type is Ethernet MAC+PCS/PMA. |
| stat_tx_packet_8192_9215_bytes_* | 1 | Output | Increment for good and bad packets transmitted that contain 8,192 to 9,215 bytes. Note: This port is available when core type is Ethernet MAC+PCS/PMA. |
| stat_tx_packet_small_* | 1 | Output | Increment for all packets that are less than 64 bytes long. Packets that are less than 64 bytes are not transmitted. Note: This port is available when core type is Ethernet MAC+PCS/PMA. |
| stat_tx_packet_large_* | 1 | Output | Increment for all packets that are more than 9,215 bytes long. Note: This port is available when core type is Ethernet MAC+PCS/PMA. |
| stat_tx_bad_fcs_* | 1 | Output | Increment for packets greater than 64 bytes that have FCS errors. Note: This port is available when core type is Ethernet MAC+PCS/PMA. |
| stat_tx_frame_error_* | 1 | Output | Increment for packets with tx_errin set to indicate an EOP abort. Note: This port is available when core type is Ethernet MAC+PCS/PMA. |
| stat_tx_local_fault_* | 1 | Output | A value of 1 indicates the receive decoder state machine is in the TX_INIT state. This output is level sensitive. |

Table 5-2: Top Level Port List (Cont'd)

| Name | Size | Direction | Description |
|--|------|-----------|--|
| stat_tx_fifo_error_* | 1 | Output | <p>Transmit clock compensation First In First Out (FIFO) error indicator. A value of 1 indicates the clock compensation FIFO under or overflowed. This condition only occurs if the PPM difference between the transmitter clock and the local reference clock is greater than ± 200 ppm.</p> <p>If this output is sampled as a 1 in any clock cycle, the corresponding port must be reset to resume proper operation.</p> <p>Note: This port is available when core type is Ethernet PCS/PMA.</p> |
| RX Path Control / Status / Statistics Signals | | | |
| ctl_rx_vl_length_minus1_* | 16 | Input | <p>Number of words in between PCS Lane markers minus one for RX. Default value, as defined in the 802.3, should be set to 16,383.</p> <p>This input should only be changed while the corresponding reset input is asserted.</p> <p>Note: This port is available when the AXI4-Lite interface is not selected.</p> |
| ctl_rx_test_pattern_* | 1 | Input | <p>Test pattern checking enable for the RX core. A value of 1 enables test mode as defined in Clause 82.2.17. Corresponds to MDIO register bit 3.42.2 as defined in Clause 82.3. Checks for scrambled idle pattern.</p> <p>Note: This port is available when the AXI4-Lite interface is not selected.</p> |
| ctl_rx_enable_* | 1 | Input | <p>RX Enable. For normal operation, this input must be set to 1. When this input is set to 0, after the RX completes the reception of the current packet (if any), it stops receiving packets by keeping the PCS from decoding incoming data. In this mode, there are no statistics reported and the user interface is idle.</p> <p>Note: This port is available when the AXI4-Lite interface is not selected and core type is Ethernet MAC+PCS/PMA.</p> |

Table 5-2: Top Level Port List (Cont'd)

| Name | Size | Direction | Description |
|-------------------------|------|-----------|---|
| ctl_rx_delete_fcs_* | 1 | Input | <p>Enable FCS removal by the RX core. If this bit is set to 0, the 40G/50G High Speed Ethernet Subsystem does not remove the FCS of the incoming packet. If this bit is set to 1, the 40G/50G High Speed Ethernet Subsystem deletes the FCS to the received packet. FCS is not deleted for packets that are ≤ 8 bytes long. This input should only be changed while the corresponding reset input is asserted.</p> <p>Note: This port is available when the AXI4-Lite interface is not selected and core type is Ethernet MAC+PCS/PMA.</p> |
| ctl_rx_ignore_fcs_* | 1 | Input | <p>Enable FCS error checking at the user interface by the RX core. If this bit is set to 0, a packet received with an FCS error is sent with the rx_errout pin asserted during the last transfer (rx_eopout and rx_enaout sampled 1). If this bit is set to 1, the 40G/50G High Speed Ethernet Subsystem does not flag an FCS error at the user interface.</p> <p>Note: The statistics are reported as if the packet is good. The stat_rx_bad_fcs signal, however, reports the error.</p> <p>Note: This port is available when the AXI4-Lite interface is not selected and core type is Ethernet MAC+PCS/PMA.</p> |
| ctl_rx_max_packet_len_* | 15 | Input | <p>Any packet longer than this value is considered to be oversized. If a packet has a size greater than this value, the packet is truncated to this value and the rx_errout signal is asserted along with the rx_eopout signal. Packets less than 64 bytes are dropped. The allowed value for this bus can range from 64 to 16,383.</p> <p>ctl_rx_max_packet_len[14] is reserved and must be set to 0.</p> <p>Note: This port is available when the AXI4-Lite interface is not selected and core type is Ethernet MAC+PCS/PMA.</p> |
| ctl_rx_min_packet_len_* | 8 | Input | <p>Any packet shorter than this value is considered to be undersized. If a packet has a size less than this value, the rx_errout signal is asserted during the rx_eopout asserted cycle. Packets that are less than 64 bytes are dropped.</p> <p>Note: This port is available when the AXI4-Lite interface is not selected and core type is Ethernet MAC+PCS/PMA.</p> |

Table 5-2: Top Level Port List (Cont'd)

| Name | Size | Direction | Description |
|---------------------------------|------|-----------|---|
| ctl_rx_custom_preamble_enable_* | 1 | Input | When asserted, this signal causes the preamble to be presented on rx_preambleout. Note: This port is available when the AXI4-Lite interface is not selected and core type is Ethernet MAC+PCS/PMA and Include FIFO Logic is disabled) |
| ctl_rx_check_sfd_* | 1 | Input | When asserted, this input causes the Ethernet MAC to check the start of frame Delimiter of the received frame. Note: This port is available when the AXI4-Lite interface is not selected and core type is Ethernet MAC+PCS/PMA. |
| ctl_rx_check_preamble_* | 1 | Input | When asserted, this input causes the Ethernet MAC to check the preamble of the received frame. Note: This port is available when the AXI4-Lite interface is not selected and core type is Ethernet MAC+PCS/PMA. |
| ctl_rx_process_lfi_* | 1 | Input | When this input is set to 1, the RX core expects and processes LF control codes coming in from the SerDes. When set to 0, the RX core ignores LF control codes coming in from the SerDes. Note: This port is available when the AXI4-Lite interface is not selected and core type is Ethernet MAC+PCS/PMA. |
| ctl_rx_force_resync_* | 1 | Input | RX force resynchronization input. This signal is used to force the RX path to reset, re-synchronize, and realign. A value of 1 forces the reset operation. A value of 0 allows normal operation. Note: This input should normally be Low and should only be pulsed (1 cycle minimum pulse) to force realignment. Note: This port is available when the AXI4-Lite interface is not selected and core type is Ethernet MAC+PCS/PMA. |
| ctl_rx_vl_marker_id0_* | 64 | Input | These inputs set the PCS Lane markers for each PCS lane. These inputs should be set to the values as defined in the IEEE Std 802.3-2012. This input should only be changed while the corresponding reset input is asserted. |
| ctl_rx_vl_marker_id1_* | 64 | Input | These inputs set the PCS Lane markers for each PCS lane. These inputs should be set to the values as defined in the IEEE Std 802.3-2012. This input should only be changed while the corresponding reset input is asserted. |

Table 5-2: Top Level Port List (Cont'd)

| Name | Size | Direction | Description |
|-------------------------------|------|-----------|--|
| ctl_rx_vl_marker_id2_* | 64 | Input | These inputs set the PCS Lane markers for each PCS lane. These inputs should be set to the values as defined in the IEEE Std 802.3-2012. This input should only be changed while the corresponding reset input is asserted. |
| ctl_rx_vl_marker_id3_* | 64 | Input | These inputs set the PCS Lane markers for each PCS lane. These inputs should be set to the values as defined in the IEEE Std 802.3-2012. This input should only be changed while the corresponding reset input is asserted. |
| stat_rx_block_lock_* | 4 | Output | Block lock status for each PCS lane. A value of 1 indicates that the corresponding lane has achieved block lock as defined in Clause 82. Corresponds to MDIO register bit 3.50.7:0 and 3.51.11:0 as defined in Clause 82.3. This output is level sensitive. |
| stat_rx_framing_err_valid_0_* | 1 | Output | Valid indicator for stat_rx_framing_err_0. When 1 stat_rx_framing_err_0 is valid. |
| stat_rx_framing_err_0_* | 3 | Output | RX sync header bits framing error. Each PCS Lane has a four-bit bus that indicates how many sync header errors were received for that PCS Lane. The value of the bus is only valid when the corresponding stat_rx_framing_err_valid_0 is a 1. The values on these buses can be updated at any time and are intended to be used as increment values for sync header error counters. |
| stat_rx_framing_err_valid_1_* | 1 | Output | Valid indicator for stat_rx_framing_err_1. When 1 stat_rx_framing_err_1 is valid. |
| stat_rx_framing_err_1_* | 3 | Output | RX sync header bits framing error. Each PCS Lane has a four-bit bus that indicates how many sync header errors were received for that PCS Lane. The value of the bus is only valid when the corresponding stat_rx_framing_err_valid_1 is a 1. The values on these buses can be updated at any time and are intended to be used as increment values for sync header error counters. |
| stat_rx_framing_err_valid_2_* | 1 | Output | Valid indicator for stat_rx_framing_err_2. When 1 stat_rx_framing_err_2 is valid. |
| stat_rx_framing_err_2_* | 3 | Output | RX sync header bits framing error. Each PCS Lane has a four-bit bus that indicates how many sync header errors were received for that PCS Lane. The value of the bus is only valid when the corresponding stat_rx_framing_err_valid_2 is a 1. The values on these buses can be updated at any time and are intended to be used as increment values for sync header error counters. |

Table 5-2: Top Level Port List (Cont'd)

| Name | Size | Direction | Description |
|-------------------------------|------|-----------|--|
| stat_rx_framing_err_valid_3_* | 1 | Output | Valid indicator for stat_rx_framing_err_3. When 1 stat_rx_framing_err_3 is valid. |
| stat_rx_framing_err_3_* | 3 | Output | RX sync header bits framing error. Each PCS Lane has a four-bit bus that indicates how many sync header errors were received for that PCS Lane. The value of the bus is only valid when the corresponding stat_rx_framing_err_valid_3 is a 1. The values on these buses can be updated at any time and are intended to be used as increment values for sync header error counters. |
| stat_rx_vl_demuxed_* | 4 | Output | PCS Lane Marker found. If a signal of this bus is sampled as 1, it indicates that the receiver has properly de-muxed that PCS lane. This output is level sensitive. |
| stat_rx_vl_number_0_* | 2 | Output | The value of this bus indicates which physical lane appears on PCS lane 0. This bus is only valid when the corresponding bit of stat_rx_vl_synced[PCS_LANES-1:0] is a 1. These outputs are level sensitive. |
| stat_rx_vl_number_1_* | 2 | Output | The value of this bus indicates which physical lane appears on PCS lane 1. |
| stat_rx_vl_number_2_* | 2 | Output | The value of this bus indicates which physical lane appears on PCS lane 2. |
| stat_rx_vl_number_3_* | 2 | Output | The value of this bus indicates which physical lane appears on PCS lane 3. |
| stat_rx_synced_* | 4 | Output | Word Boundary Synchronized. These signals indicate whether a PCS lane is word boundary synchronized. A value of 1 indicates the corresponding PCS lane has achieved word boundary synchronization and it has received a PCS lane marker. Corresponds to MDIO register bit 3.52.7:0 and 3.53.11:0 as defined in Clause 82.3. This output is level sensitive. |
| stat_rx_synced_err_* | 4 | Output | Word Boundary Synchronization Error. These signals indicate whether an error occurred during word boundary synchronization in the respective PCS lane. A value of 1 indicates that the corresponding PCS lane lost word boundary synchronization due to sync header framing bits errors or that a PCS lane marker was never received. This output is level sensitive. |

Table 5-2: Top Level Port List (Cont'd)

| Name | Size | Direction | Description |
|-------------------------|------|-----------|--|
| stat_rx_mf_len_err_* | 4 | Output | PCS Lane Marker Length Error. These signals indicate whether a PCS Lane Marker length mismatch occurred in the respective lane (that is, PCS Lane Markers were received not every <code>ctl_rx_vl_length_minus1</code> words apart). A value of 1 indicates that the corresponding lane is receiving PCS Lane Markers at wrong intervals. This remains High until the error condition is removed. |
| stat_rx_mf_repeat_err_* | 4 | Output | PCS Lane Marker Consecutive Error. These signals indicate whether four consecutive PCS Lane Marker errors occurred in the respective lane. A value of 1 indicates an error in the corresponding lane. This output remains High until the error condition is removed. |
| stat_rx_mf_err_* | 4 | Output | PCS Lane Marker Word Error. These signals indicate that an incorrectly formed PCS Lane Marker Word was detected in the respective lane. A value of 1 indicates an error occurred. This output is pulsed for one clock cycle to indicate the error condition. Pulses can occur in back-to-back cycles. |
| stat_rx_misaligned_* | 1 | Output | Alignment Error. This signal indicates that the lane aligner did not receive the expected PCS lane marker across all lanes. This signal is not asserted until the PCS lane marker has been received at least once across all lanes and at least one incorrect lane marker has been received. This occurs one metaframe after the error. This signal is not asserted if the lane markers have never been received correctly. Lane marker errors are indicated by the corresponding <code>stat_rx_mf_err</code> signal. This output is pulsed for one clock cycle to indicate an error condition. Pulses can occur in back-to-back cycles. |
| stat_rx_aligned_err_* | 1 | Output | Loss of Lane Alignment/Deskew. This signal indicates that an error occurred during PCS lane alignment or PCS lane alignment was lost. A value of 1 indicates an error occurred. This output is level sensitive. |
| stat_rx_bip_err_0_* | 1 | Output | BIP8 error indicator for PCS lane 0. A non-zero value indicates the BIP8 signature was in error. A non-zero value is pulsed for one clock cycle. This output is pulsed for one clock cycle to indicate an error condition. |
| stat_rx_bip_err_1_* | 1 | Output | BIP8 error indicator for PCS lane 1. |

Table 5-2: Top Level Port List (Cont'd)

| Name | Size | Direction | Description |
|--------------------------|------|-----------|---|
| stat_rx_bip_err_2_* | 1 | Output | BIP8 error indicator for PCS lane 2. |
| stat_rx_bip_err_3_* | 1 | Output | BIP8 error indicator for PCS lane 3. |
| stat_rx_aligned_* | 1 | Output | All PCS Lanes Aligned/Deskewed. This signal indicates whether or not all PCS lanes are aligned and deskewed. A value of 1 indicates all PCS lanes are aligned and deskewed. When this signal is a 1, the RX path is aligned and can receive packet data. When this signal is 0, a local fault condition exists. This also corresponds to MDIO register bit 3.50.12 as defined in Clause 82.3. This output is level sensitive. |
| stat_rx_hi_ber_* | 1 | Output | High Bit Error Rate (BER) indicator. When set to 1, the BER is too high as defined by IEEE Std 802.3-2012. Corresponds to MDIO register bit 3.32.1 as defined in Clause 82.3. This output is level sensitive. |
| stat_rx_status_* | 1 | Output | PCS status. A value of 1 indicates that the PCS is aligned and not in hi_ber state. Corresponds to Management Data Input/ Output (MDIO) register bit 3.32.12 as defined in Clause 82.3. This output is level sensitive. |
| stat_rx_bad_code_* | 2 | Output | Increment for 64B/66B code violations. This signal indicates that the RX PCS receive state machine is in the RX_E state as specified by the IEEE Std 802.3-2012. This output can be used to generate MDIO register 3.33:7:0 as defined in Clause 82.3. |
| stat_rx_bad_code_valid_* | 1 | Output | Indicates when stat_rx_bad_code is valid. Note: This port is available when core type is Ethernet PCS/PMA. |
| stat_rx_error_valid_* | 1 | Output | Indicates when stat_rx_error is valid. Note: This port is available when core type is Ethernet PCS/PMA. |
| stat_rx_error_* | 8 | Output | Test pattern mismatch increment. A non-zero value in any cycle indicates a mismatch occurred for the test pattern in the RX core. This output is only active when ctl_rx_test_pattern is set to a 1. This output is pulsed for one clock cycle. Note: This port is available when core type is Ethernet PCS/PMA. |
| stat_rx_fifo_error_* | 1 | Output | Indicates when RX FIFO goes into an underflow or overflow condition. Note: This port is available when core type is Ethernet PCS/PMA. |

Table 5-2: Top Level Port List (Cont'd)

| Name | Size | Direction | Description |
|------------------------------|------|-----------|--|
| stat_rx_total_packets_* | 2 | Output | Increment for the total number of packets received. Note: This port is available when core type is Ethernet MAC+PCS/PMA. |
| stat_rx_total_good_packets_* | 1 | Output | Increment for the total number of good packets received. This value is only non-zero when a packet is received completely and contains no errors. Note: This port is available when core type is Ethernet MAC+PCS/PMA. |
| stat_rx_total_bytes_* | 6 | Output | Increment for the total number of bytes received. Note: This port is available when core type is Ethernet MAC+PCS/PMA. |
| stat_rx_total_good_bytes_* | 14 | Output | Increment for the total number of good bytes received. This value is only non-zero when a packet is received completely and contains no errors. Note: This port is available when core type is Ethernet MAC+PCS/PMA. |
| stat_rx_packet_small_* | 2 | Output | Increment for all packets that are less than 64 bytes long. Packets that are less than 64 bytes are dropped. Note: This port is available when core type is Ethernet MAC+PCS/PMA. |
| stat_rx_jabber_* | 1 | Output | Increment for packets longer than <code>ctl_rx_max_packet_len</code> with bad FCS. Note: This port is available when core type is Ethernet MAC+PCS/PMA. |
| stat_rx_packet_large_* | 1 | Output | Increment for all packets that are more than 9,215 bytes long. Note: This port is available when core type is Ethernet MAC+PCS/PMA. |
| stat_rx_oversize_* | 1 | Output | Increment for packets longer than <code>ctl_rx_max_packet_len</code> with good FCS. Note: This port is available when core type is Ethernet MAC+PCS/PMA. |
| stat_rx_undersize_* | 2 | Output | Increment for packets shorter than <code>stat_rx_min_packet_len</code> with good FCS. Note: This port is available when core type is Ethernet MAC+PCS/PMA. |
| stat_rx_toolong_* | 1 | Output | Increment for packets longer than <code>ctl_rx_max_packet_len</code> with good and bad FCS. Note: This port is available when core type is Ethernet MAC+PCS/PMA. |

Table 5-2: Top Level Port List (Cont'd)

| Name | Size | Direction | Description |
|----------------------------------|------|-----------|---|
| stat_rx_fragment_* | 2 | Output | Increment for packets shorter than stat_rx_min_packet_len with bad FCS. Note: This port is available when core type is Ethernet MAC+PCS/PMA. |
| stat_rx_packet_64_bytes_* | 1 | Output | Increment for good and bad packets received that contain 64 bytes. Note: This port is available when core type is Ethernet MAC+PCS/PMA. |
| stat_rx_packet_65_127_bytes_* | 1 | Output | Increment for good and bad packets received that contain 65 to 127 bytes. Note: This port is available when core type is Ethernet MAC+PCS/PMA. |
| stat_rx_packet_128_255_bytes_* | 1 | Output | Increment for good and bad packets received that contain 128 to 255 bytes. Note: This port is available when core type is Ethernet MAC+PCS/PMA. |
| stat_rx_packet_256_511_bytes_* | 1 | Output | Increment for good and bad packets received that contain 256 to 511 bytes. Note: This port is available when core type is Ethernet MAC+PCS/PMA. |
| stat_rx_packet_512_1023_bytes_* | 1 | Output | Increment for good and bad packets received that contain 512 to 1,023 bytes. Note: This port is available when core type is Ethernet MAC+PCS/PMA. |
| stat_rx_packet_1024_1518_bytes_* | 1 | Output | Increment for good and bad packets received that contain 1,024 to 1,518 bytes. Note: This port is available when core type is Ethernet MAC+PCS/PMA. |
| stat_rx_packet_1519_1522_bytes_* | 1 | Output | Increment for good and bad packets received that contain 1,519 to 1,522 bytes. Note: This port is available when core type is Ethernet MAC+PCS/PMA. |
| stat_rx_packet_1523_1548_bytes_* | 1 | Output | Increment for good and bad packets received that contain 1,523 to 1,548 bytes. Note: This port is available when core type is Ethernet MAC+PCS/PMA. |
| stat_rx_packet_1549_2047_bytes_* | 1 | Output | Increment for good and bad packets received that contain 1,549 to 2,047 bytes. Note: This port is available when core type is Ethernet MAC+PCS/PMA. |
| stat_rx_packet_2048_4095_bytes_* | 1 | Output | Increment for good and bad packets received that contain 2,048 to 4,095 bytes. Note: This port is available when core type is Ethernet MAC+PCS/PMA. |

Table 5-2: Top Level Port List (Cont'd)

| Name | Size | Direction | Description |
|----------------------------------|------|-----------|--|
| stat_rx_packet_4096_8191_bytes_* | 1 | Output | Increment for good and bad packets received that contain 4,096 to 8,191 bytes. Note: This port is available when core type is Ethernet MAC+PCS/PMA. |
| stat_rx_packet_8192_9215_bytes_* | 1 | Output | Increment for good and bad packets received that contain 8,192 to 9,215 bytes. Note: This port is available when core type is Ethernet MAC+PCS/PMA. |
| stat_rx_bad_fcs_* | 2 | Output | Bad FCS indicator. The value on this bus indicates packets received with a bad FCS, but not a stomped FCS. A stomped FCS is defined as the bitwise inverse of the expected good FCS. This output is pulsed for one clock cycle to indicate an error condition. Pulses can occur in back-to-back cycles. Note: This port is available when core type is Ethernet MAC+PCS/PMA. |
| stat_rx_packet_bad_fcs_* | 1 | Output | Increment for packets between 64 and ctl_rx_max_packet_len bytes that have FCS errors. Note: This port is available when core type is Ethernet MAC+PCS/PMA. |
| stat_rx_stomped_fcs_* | 2 | Output | Stomped FCS indicator. The value on this bus indicates packets were received with a stomped FCS. A stomped FCS is defined as the bitwise inverse of the expected good FCS. This output is pulsed for one clock cycle to indicate the stomped condition. Pulses can occur in back-to-back cycles. Note: This port is available when core type is Ethernet MAC+PCS/PMA. |
| stat_rx_bad_preamble_* | 1 | Output | Increment bad preamble. This signal indicates if the Ethernet packet received was preceded by a valid preamble. A value of 1 indicates that an invalid preamble was received. Note: This port is available when core type is Ethernet MAC+PCS/PMA. |
| stat_rx_bad_sfd_* | 1 | Output | Increment bad SFD. This signal indicates if the Ethernet packet received was preceded by a valid SFD. A value of 1 indicates that an invalid SFD was received. Note: This port is available when core type is Ethernet MAC+PCS/PMA. |

Table 5-2: Top Level Port List (Cont'd)

| Name | Size | Direction | Description |
|---------------------------------|------|-----------|---|
| stat_rx_got_signal_os_* | 1 | Output | Signal OS indication. If this bit is sampled as a 1, it indicates that a Signal OS word was received. Note: Signal OS should not be received in an Ethernet network. Note: This port is available when core type is Ethernet MAC+PCS/PMA. |
| stat_rx_test_pattern_mismatch_* | 2 | Output | Test pattern mismatch increment. A nonzero value in any cycle indicates how many mismatches occurred for the test pattern in the RX core. This output is only active when ctl_rx_test_pattern is set to a 1. This output can be used to generate MDIO register 3.43.15:0 as defined in Clause 82.3. This output is pulsed for one clock cycle. Note: This port is available when core type is Ethernet MAC+PCS/PMA. |
| stat_rx_truncated_* | 1 | Output | Packet truncation indicator. A value of 1 indicates that the current packet in flight is truncated due to its length exceeding ctl_rx_max_packet_len[14:0]. This output is pulsed for one clock cycle to indicate the truncated condition. Pulses can occur in back-to-back cycles. Note: This port is available when core type is Ethernet MAC+PCS/PMA. |
| stat_rx_local_fault_* | 1 | Output | This output is High when stat_rx_internal_local_fault or stat_rx_received_local_fault is asserted. This output is level sensitive. |
| stat_rx_remote_fault_* | 1 | Output | Remote fault indication status. If this bit is sampled as a 1, it indicates a remote fault condition was detected. If this bit is sampled as a 0, a remote fault condition does not exist. This output is level sensitive. Note: This port is available when core type is Ethernet MAC+PCS/PMA. |
| stat_rx_internal_local_fault_* | 1 | Output | This signal goes High when an internal local fault is generated due to any one of the following: test pattern generation, bad lane alignment, or high bit error rate. This signal remains High as long as the fault condition persists. Note: This port is available when core type is Ethernet MAC+PCS/PMA. |

Table 5-2: Top Level Port List (Cont'd)

| Name | Size | Direction | Description |
|---|------|-----------|--|
| stat_rx_received_local_fault_* | 1 | Output | This signal goes High when enough local fault words are received from the link partner to trigger a fault condition as specified by the IEEE fault state machine. This signal remains High as long as the fault condition persists. Note: This port is available when core type is Ethernet MAC+PCS/PMA. |
| TX Pause Interface Control / Status / Statistics Signals | | | |
| Ports under this section will be available when Enable TX Flow Control Logic is selected from the MAC Options tab and the CORE type is Ethernet MAC+PCS/PMA. | | | |
| ctl_tx_pause_req_* | 9 | Input | If a bit of this bus is set to 1, the CORE transmits a pause packet using the associated quanta value on the ctl_tx_pause_quanta[8:0][15:0] bus. If bit[8] is set to 1, a global pause packet is transmitted. All other bits cause a priority pause packet to be transmitted. |
| ctl_tx_pause_enable_* | 9 | Input | TX pause enable signal. This input is used to enable the processing of the pause quanta for the corresponding priority. This signal gates transmission of pause packets. Note: This port is available when the AXI4-Lite interface is not selected. |
| ctl_tx_resend_pause_* | 1 | Input | Re-transmit pending pause packets. When this input is sampled as 1, all pending pause packets are retransmitted as soon as possible (that is, after the current packet in flight is completed) and the retransmit counters are reset. This input should be pulsed to 1 for one cycle at a time. |
| ctl_tx_pause_quanta0_* | 16 | Input | These buses indicate the quanta to be transmitted for each of the eight priorities in priority-based pause operation and the global pause operation. The value for ctl_tx_pause_quanta[8] is used for global pause operation. All other values are used for priority pause operation. Note: This port is available when the AXI4-Lite interface is not selected. |
| ctl_tx_pause_quanta1_* | 16 | Input | These buses indicate the quanta to be transmitted for each of the eight priorities in priority-based pause operation and the global pause operation. The value for ctl_tx_pause_quanta[8] is used for global pause operation. All other values are used for priority pause operation. Note: This port is available when the AXI4-Lite interface is not selected. |

Table 5-2: Top Level Port List (Cont'd)

| Name | Size | Direction | Description |
|------------------------|------|-----------|--|
| ctl_tx_pause_quanta2_* | 16 | Input | <p>These buses indicate the quanta to be transmitted for each of the eight priorities in priority-based pause operation and the global pause operation. The value for <code>ctl_tx_pause_quanta[8]</code> is used for global pause operation. All other values are used for priority pause operation.</p> <p>Note: This port is available when the AXI4-Lite interface is not selected.</p> |
| ctl_tx_pause_quanta3_* | 16 | Input | <p>These buses indicate the quanta to be transmitted for each of the eight priorities in priority-based pause operation and the global pause operation. The value for <code>ctl_tx_pause_quanta[8]</code> is used for global pause operation. All other values are used for priority pause operation.</p> <p>Note: This port is available when the AXI4-Lite interface is not selected.</p> |
| ctl_tx_pause_quanta4_* | 16 | Input | <p>These buses indicate the quanta to be transmitted for each of the eight priorities in priority-based pause operation and the global pause operation. The value for <code>ctl_tx_pause_quanta[8]</code> is used for global pause operation. All other values are used for priority pause operation.</p> <p>Note: This port is available when the AXI4-Lite interface is not selected.</p> |
| ctl_tx_pause_quanta5_* | 16 | Input | <p>These buses indicate the quanta to be transmitted for each of the eight priorities in priority-based pause operation and the global pause operation. The value for <code>ctl_tx_pause_quanta[8]</code> is used for global pause operation. All other values are used for priority pause operation.</p> <p>Note: This port is available when the AXI4-Lite interface is not selected.</p> |
| ctl_tx_pause_quanta6_* | 16 | Input | <p>These buses indicate the quanta to be transmitted for each of the eight priorities in priority-based pause operation and the global pause operation. The value for <code>ctl_tx_pause_quanta[8]</code> is used for global pause operation. All other values are used for priority pause operation.</p> <p>Note: This port is available when the AXI4-Lite interface is not selected.</p> |
| ctl_tx_pause_quanta7_* | 16 | Input | <p>These buses indicate the quanta to be transmitted for each of the eight priorities in priority-based pause operation and the global pause operation. The value for <code>ctl_tx_pause_quanta[8]</code> is used for global pause operation. All other values are used for priority pause operation.</p> <p>Note: This port is available when the AXI4-Lite interface is not selected.</p> |

Table 5-2: Top Level Port List (Cont'd)

| Name | Size | Direction | Description |
|-------------------------------|------|-----------|--|
| ctl_tx_pause_quanta8_* | 16 | Input | <p>These buses indicate the quanta to be transmitted for each of the eight priorities in priority-based pause operation and the global pause operation. The value for <code>ctl_tx_pause_quanta[8]</code> is used for global pause operation. All other values are used for priority pause operation.</p> <p>Note: This port is available when the AXI4-Lite interface is not selected.</p> |
| ctl_tx_pause_refresh_timer0_* | 16 | Input | <p>This bus sets the retransmission time of pause packets for each of the eight priorities in priority-based pause operation and the global pause operation. The value for <code>ctl_tx_pause_refresh_timer[8]</code> is used for global pause operation. All other values are used for priority pause operation.</p> <p>Note: This port is available when the AXI4-Lite interface is not selected.</p> |
| ctl_tx_pause_refresh_timer1_* | 16 | Input | <p>This bus sets the retransmission time of pause packets for each of the eight priorities in priority-based pause operation and the global pause operation. The value for <code>ctl_tx_pause_refresh_timer[8]</code> is used for global pause operation. All other values are used for priority pause operation.</p> <p>Note: This port is available when the AXI4-Lite interface is not selected.</p> |
| ctl_tx_pause_refresh_timer2_* | 16 | Input | <p>This bus sets the retransmission time of pause packets for each of the eight priorities in priority-based pause operation and the global pause operation. The value for <code>ctl_tx_pause_refresh_timer[8]</code> is used for global pause operation. All other values are used for priority pause operation.</p> <p>Note: This port is available when the AXI4-Lite interface is not selected.</p> |
| ctl_tx_pause_refresh_timer3_* | 16 | Input | <p>This bus sets the retransmission time of pause packets for each of the eight priorities in priority-based pause operation and the global pause operation. The value for <code>ctl_tx_pause_refresh_timer[8]</code> is used for global pause operation. All other values are used for priority pause operation.</p> <p>Note: This port is available when the AXI4-Lite interface is not selected.</p> |

Table 5-2: Top Level Port List (Cont'd)

| Name | Size | Direction | Description |
|-------------------------------|------|-----------|--|
| ctl_tx_pause_refresh_timer4_* | 16 | Input | <p>This bus sets the retransmission time of pause packets for each of the eight priorities in priority-based pause operation and the global pause operation. The value for <code>ctl_tx_pause_refresh_timer[8]</code> is used for global pause operation. All other values are used for priority pause operation.</p> <p>Note: This port is available when the AXI4-Lite interface is not selected.</p> |
| ctl_tx_pause_refresh_timer5_* | 16 | Input | <p>This bus sets the retransmission time of pause packets for each of the eight priorities in priority-based pause operation and the global pause operation. The value for <code>ctl_tx_pause_refresh_timer[8]</code> is used for global pause operation. All other values are used for priority pause operation.</p> <p>Note: This port is available when the AXI4-Lite interface is not selected.</p> |
| ctl_tx_pause_refresh_timer6_* | 16 | Input | <p>This bus sets the retransmission time of pause packets for each of the eight priorities in priority-based pause operation and the global pause operation. The value for <code>ctl_tx_pause_refresh_timer[8]</code> is used for global pause operation. All other values are used for priority pause operation.</p> <p>Note: This port is available when the AXI4-Lite interface is not selected.</p> |
| ctl_tx_pause_refresh_timer7_* | 16 | Input | <p>This bus sets the retransmission time of pause packets for each of the eight priorities in priority-based pause operation and the global pause operation. The value for <code>ctl_tx_pause_refresh_timer[8]</code> is used for global pause operation. All other values are used for priority pause operation.</p> <p>Note: This port is available when the AXI4-Lite interface is not selected.</p> |
| ctl_tx_pause_refresh_timer8_* | 16 | Input | <p>This bus sets the retransmission time of pause packets for each of the eight priorities in priority-based pause operation and the global pause operation. The value for <code>ctl_tx_pause_refresh_timer[8]</code> is used for global pause operation. All other values are used for priority pause operation.</p> <p>Note: This port is available when the AXI4-Lite interface is not selected.</p> |

Table 5-2: Top Level Port List (Cont'd)

| Name | Size | Direction | Description |
|------------------------|------|-----------|--|
| ctl_tx_da_gpp_* | 48 | Input | Destination address for transmitting global pause packets. Note: This port is available when the AXI4-Lite interface is not selected. |
| ctl_tx_sa_gpp_* | 48 | Input | Source address for transmitting global pause packets. Note: This port is available when the AXI4-Lite interface is not selected. |
| ctl_tx_ethertype_gpp_* | 16 | Input | Ethertype for transmitting global pause packets. Note: This port is available when the AXI4-Lite interface is not selected. |
| ctl_tx_opcode_gpp_* | 16 | Input | Opcode for transmitting global pause packets. Note: This port is available when the AXI4-Lite interface is not selected. |
| ctl_tx_da_ppp_* | 48 | Input | Destination address for transmitting priority pause packets. Note: This port is available when the AXI4-Lite interface is not selected. |
| ctl_tx_sa_ppp_* | 48 | Input | Source address for transmitting priority pause packets. Note: This port is available when the AXI4-Lite interface is not selected. |
| ctl_tx_ethertype_ppp_* | 16 | Input | Ethertype for transmitting priority pause packets. Note: This port is available when the AXI4-Lite interface is not selected. |
| ctl_tx_opcode_ppp_* | 16 | Input | Opcode for transmitting priority pause packets. Note: This port is available when the AXI4-Lite interface is not selected. |
| stat_tx_pause_valid_* | 9 | Output | If a bit of this bus is set to 1, the 40G/50G High Speed Ethernet Subsystem has transmitted a pause packet. If bit[8] is set to 1, a global pause packet is transmitted. All other bits cause a priority pause packet to be transmitted. |
| stat_tx_unicast_* | 1 | Output | Increment for good unicast packets. |
| stat_tx_multicast_* | 1 | Output | Increment for good multicast packets. |
| stat_tx_broadcast_* | 1 | Output | Increment for good broadcast packets. |
| stat_tx_vlan_* | 1 | Output | Increment for good 802.1Q tagged VLAN packets. |
| stat_tx_pause_* | 1 | Output | Increment for 802.3x Ethernet MAC Pause packet with good FCS. |

Table 5-2: Top Level Port List (Cont'd)

| Name | Size | Direction | Description |
|---|------|-----------|--|
| stat_tx_user_pause_* | 1 | Output | Increment for priority-based pause packets with good FCS. |
| RX Pause Interface Control / Status / Statistics Signals | | | |
| Ports under this section will be available when Enable RX Flow Control Logic is selected from the MAC Options tab and CORE type is Ethernet MAC+PCS/PMA. | | | |
| ctl_rx_forward_control_* | 1 | Input | A value of 1 indicates that the core forwards control packets to you. A value of 0 causes the core to drop control packets. Note: This port is available when the AXI4-Lite interface is not selected. |
| ctl_rx_pause_ack_* | 9 | Input | Pause acknowledge signal. This bus is used to acknowledge the receipt of the pause frame from the user logic. |
| ctl_rx_check_ack_* | 1 | Input | Wait for acknowledge. If this input is set to 1, the CORE uses the ctl_rx_pause_ack[8:0] bus for pause processing. If this input is set to 0, ctl_rx_pause_ack[8:0] is not used. Note: This port is available when the AXI4-Lite interface is not selected. |
| ctl_rx_pause_enable_* | 9 | Input | RX pause enable signal. This input is used to enable the processing of the pause quanta for the corresponding priority. Note: This signal only affects the RX user interface, not the pause processing logic. Note: This port is available when the AXI4-Lite interface is not selected. |
| ctl_rx_enable_gcp_* | 1 | Input | A value of 1 enables global control packet processing. Note: This port is available when the AXI4-Lite interface is not selected. |
| ctl_rx_check_mcast_gcp_* | 1 | Input | A value of 1 enables global control multicast destination address processing. Note: This port is available when the AXI4-Lite interface is not selected. |
| ctl_rx_check_ucast_gcp_* | 1 | Input | A value of 1 enables global control unicast destination address processing. Note: This port is available when the AXI4-Lite interface is not selected. |
| ctl_rx_pause_da_ucast_* | 48 | Input | Unicast destination address for pause processing. Note: This port is available when the AXI4-Lite interface is not selected. |

Table 5-2: Top Level Port List (Cont'd)

| Name | Size | Direction | Description |
|---------------------------|------|-----------|--|
| ctl_rx_check_sa_gcp_* | 1 | Input | A value of 1 enables global control source address processing. Note: This port is available when the AXI4-Lite interface is not selected. |
| ctl_rx_pause_sa_* | 48 | Input | Source address for pause processing. Note: This port is available when the AXI4-Lite interface is not selected. |
| ctl_rx_check_etype_gcp_* | 1 | Input | A value of 1 enables global control ethertype processing. Note: This port is available when the AXI4-Lite interface is not selected. |
| ctl_rx_etype_gcp_* | 16 | Input | Ethertype field for global control processing. Note: This port is available when the AXI4-Lite interface is not selected. |
| ctl_rx_check_opcode_gcp_* | 1 | Input | A value of 1 enables global control opcode processing. Note: This port is available when the AXI4-Lite interface is not selected. |
| ctl_rx_opcode_min_gcp_* | 16 | Input | Minimum global control opcode value. Note: This port is available when the AXI4-Lite interface is not selected. |
| ctl_rx_opcode_max_gcp_* | 16 | Input | Maximum global control opcode value. Note: This port is available when the AXI4-Lite interface is not selected. |
| ctl_rx_enable_pcp_* | 1 | Input | A value of 1 enables priority control packet processing. Note: This port is available when the AXI4-Lite interface is not selected. |
| ctl_rx_check_mcast_pcp_* | 1 | Input | A value of 1 enables priority control multicast destination address processing. Note: This port is available when the AXI4-Lite interface is not selected. |
| ctl_rx_check_ucast_pcp_* | 1 | Input | A value of 1 enables priority control unicast destination address processing. Note: This port is available when the AXI4-Lite interface is not selected. |
| ctl_rx_pause_da_mcast_* | 48 | Input | Multicast destination address for pause processing. Note: This port is available when the AXI4-Lite interface is not selected. |

Table 5-2: Top Level Port List (Cont'd)

| Name | Size | Direction | Description |
|---------------------------|------|-----------|--|
| ctl_rx_check_sa_pcp_* | 1 | Input | A value of 1 enables priority control source address processing. Note: This port is available when the AXI4-Lite interface is not selected. |
| ctl_rx_check_etype_pcp_* | 1 | Input | A value of 1 enables priority control ethertype processing. Note: This port is available when the AXI4-Lite interface is not selected. |
| ctl_rx_etype_pcp_* | 16 | Input | Ethertype field for priority control processing. Note: This port is available when the AXI4-Lite interface is not selected. |
| ctl_rx_check_opcode_pcp_* | 1 | Input | A value of 1 enables priority control opcode processing. Note: This port is available when the AXI4-Lite interface is not selected. |
| ctl_rx_opcode_min_pcp_* | 16 | Input | Minimum priority control opcode value. Note: This port is available when the AXI4-Lite interface is not selected. |
| ctl_rx_opcode_max_pcp_* | 16 | Input | Maximum priority control opcode value. Note: This port is available when the AXI4-Lite interface is not selected. |
| ctl_rx_enable_gpp_* | 1 | Input | A value of 1 enables global pause packet processing. Note: This port is available when the AXI4-Lite interface is not selected. |
| ctl_rx_check_mcast_gpp_* | 1 | Input | A value of 1 enables global pause multicast destination address processing. Note: This port is available when the AXI4-Lite interface is not selected. |
| ctl_rx_check_ucast_gpp_* | 1 | Input | A value of 1 enables global pause unicast destination address processing. Note: This port is available when the AXI4-Lite interface is not selected. |
| ctl_rx_check_sa_gpp_* | 1 | Input | A value of 1 enables global pause source address processing. Note: This port is available when the AXI4-Lite interface is not selected. |
| ctl_rx_check_etype_gpp_* | 1 | Input | A value of 1 enables global pause ethertype processing. Note: This port is available when the AXI4-Lite interface is not selected. |

Table 5-2: Top Level Port List (Cont'd)

| Name | Size | Direction | Description |
|---------------------------|------|-----------|--|
| ctl_rx_etype_gpp_* | 16 | Input | Ethertype field for global pause processing. Note: This port is available when the AXI4-Lite interface is not selected. |
| ctl_rx_check_opcode_gpp_* | 1 | Input | A value of 1 enables global pause opcode processing. Note: This port is available when the AXI4-Lite interface is not selected. |
| ctl_rx_opcode_gpp_* | 16 | Input | Global pause opcode value. Note: This port is available when the AXI4-Lite interface is not selected. |
| ctl_rx_enable_ppp_* | 1 | Input | A value of 1 enables priority pause packet processing. Note: This port is available when the AXI4-Lite interface is not selected. |
| ctl_rx_check_mcast_ppp_* | 1 | Input | A value of 1 enables priority pause multicast destination address processing. Note: This port is available when the AXI4-Lite interface is not selected. |
| ctl_rx_check_ucast_ppp_* | 1 | Input | A value of 1 enables priority pause unicast destination address processing. Note: This port is available when the AXI4-Lite interface is not selected. |
| ctl_rx_check_sa_ppp_* | 1 | Input | A value of 1 enables priority pause source address processing. Note: This port is available when the AXI4-Lite interface is not selected. |
| ctl_rx_check_etype_ppp_* | 1 | Input | A value of 1 enables priority pause ethernet processing. Note: This port is available when the AXI4-Lite interface is not selected. |
| ctl_rx_etype_ppp_* | 16 | Input | Ethertype field for priority pause processing. Note: This port is available when the AXI4-Lite interface is not selected. |
| ctl_rx_check_opcode_ppp_* | 1 | Input | A value of 1 enables priority pause opcode processing. Note: This port is available when the AXI4-Lite interface is not selected. |
| ctl_rx_opcode_ppp_* | 16 | Input | Priority pause opcode value. Note: This port is available when the AXI4-Lite interface is not selected. |
| stat_rx_unicast_* | 1 | Output | Increment for good unicast packets. |
| stat_rx_multicast_* | 1 | Output | Increment for good multicast packets. |

Table 5-2: Top Level Port List (Cont'd)

| Name | Size | Direction | Description |
|-------------------------|------|-----------|--|
| stat_rx_broadcast_* | 1 | Output | Increment for good broadcast packets. |
| stat_rx_vlan_* | 1 | Output | Increment for good 802.1Q tagged VLAN packets. |
| stat_rx_pause_* | 1 | Output | Increment for 802.3x Ethernet MAC Pause packet with good FCS. |
| stat_rx_user_pause_* | 1 | Output | Increment for priority-based pause packets with good FCS. |
| stat_rx_inrangeerr_* | 1 | Output | Increment for packets with Length field error but with good FCS. |
| stat_rx_pause_valid_* | 9 | Output | This bus indicates that a pause packet was received and the associated quanta on the stat_rx_pause_quanta[8:0][15:0] bus is valid and must be used for pause processing. If an 802.3x Ethernet MAC Pause packet is received, bit[8] is set to 1. |
| stat_rx_pause_quanta0_* | 16 | Output | These buses indicate the quanta received for each of the eight priorities in priority-based pause operation and global pause operation. If an 802.3x Ethernet MAC Pause packet is received, the quanta are placed in value [8]. |
| stat_rx_pause_quanta1_* | 16 | Output | These buses indicate the quanta received for each of the eight priorities in priority-based pause operation and global pause operation. If an 802.3x Ethernet MAC Pause packet is received, the quanta are placed in value [8]. |
| stat_rx_pause_quanta2_* | 16 | Output | These buses indicate the quanta received for each of the eight priorities in priority-based pause operation and global pause operation. If an 802.3x Ethernet MAC Pause packet is received, the quanta are placed in value [8]. |
| stat_rx_pause_quanta3_* | 16 | Output | These buses indicate the quanta received for each of the eight priorities in priority-based pause operation and global pause operation. If an 802.3x Ethernet MAC Pause packet is received, the quanta are placed in value [8]. |
| stat_rx_pause_quanta4_* | 16 | Output | These buses indicate the quanta received for each of the eight priorities in priority-based pause operation and global pause operation. If an 802.3x Ethernet MAC Pause packet is received, the quanta are placed in value [8]. |
| stat_rx_pause_quanta5_* | 16 | Output | These buses indicate the quanta received for each of the eight priorities in priority-based pause operation and global pause operation. If an 802.3x Ethernet MAC Pause packet is received, the quanta are placed in value [8]. |

Table 5-2: Top Level Port List (Cont'd)

| Name | Size | Direction | Description |
|---|------|-----------|---|
| stat_rx_pause_quanta6_* | 16 | Output | These buses indicate the quanta received for each of the eight priorities in priority-based pause operation and global pause operation. If an 802.3x Ethernet MAC Pause packet is received, the quanta are placed in value [8]. |
| stat_rx_pause_quanta7_* | 16 | Output | These buses indicate the quanta received for each of the eight priorities in priority-based pause operation and global pause operation. If an 802.3x Ethernet MAC Pause packet is received, the quanta are placed in value [8]. |
| stat_rx_pause_quanta8_* | 16 | Output | These buses indicate the quanta received for each of the eight priorities in priority-based pause operation and global pause operation. If an 802.3x Ethernet MAC Pause packet is received, the quanta are placed in value [8]. |
| stat_rx_pause_req_* | 9 | Output | Pause request signal. When the RX receives a valid pause frame, it sets the corresponding bit of this bus to a 1 and keep it at 1 until the pause packet has been processed. |
| IEEE 1588 TX/RX Interface Control / Status / Statistics Signals | | | |
| Ports under this section will be available when Enable_Time_Stamping is selected from the MAC Options tab and CORE type is Ethernet MAC+PCS/PMA. | | | |
| ctl_tx_systemtimerin_* | 80 | Input | System timer input for the TX. In normal clock mode, the time format is according to the IEEE 1588 format, with 48 bits for seconds and 32 bits for nanoseconds. In transparent clock mode, bit 63 is expected to be zero, bits 62:16 carry nanoseconds, and bits 15:0 carry fractional nanoseconds. Refer to IEEE 1588v2 for the representational definitions. This input must be in the TX clock domain. |
| ctl_rx_systemtimerin_* | 80 | Input | System timer input for the RX. In normal clock mode, the time format is according to the IEEE 1588 format, with 48 bits for seconds and 32 bits for nanoseconds. In transparent clock mode, bit 63 is expected to be zero, bits 62:16 carry nanoseconds, and bits 15:0 carry fractional nanoseconds. Refer to IEEE 1588v2 for the representational definitions. This input must be in the same clock domain as the lane 0 RX SerDes. |
| stat_tx_ptp_fifo_read_error_* | 1 | Output | Transmit PTP FIFO write error. A 1 on this status indicates that an error occurred during the PTP Tag write. A TX Path reset is required to clear the error. |

Table 5-2: Top Level Port List (Cont'd)

| Name | Size | Direction | Description |
|--------------------------------|------|-----------|---|
| stat_tx_ptp_fifo_write_error_* | 1 | Output | Transmit PTP FIFO read error. A 1 on this status indicates that an error occurred during the PTP Tag read. A TX Path reset is required to clear the error. |
| tx_ptp_1588op_in_* | 2 | Input | 2'b00 – No operation: no timestamp will be taken and the frame will not be 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 using the additional ports of 2-step operation. The frame itself will not be modified. 2'b11 – Reserved: act as No operation. |
| tx_ptp_tag_field_in_* | 16 | Input | The usage of this field is dependent on the 1588 operation. <ul style="list-style-type: none"> • For No operation, this field will be ignored. • For 1-step and 2-step this field is a tag field. This tag value will be returned to the client with the timestamp for the current frame using the additional ports of 2-step operation. 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. |
| tx_ptp_tstamp_valid_out_* | 1 | Output | This bit indicates that a valid timestamp is being presented on the tx. |
| tx_ptp_tstamp_tag_out_* | 16 | Output | Tag output corresponding to tx_ptp_tag_field_in[15:0] |
| tx_ptp_tstamp_out_* | 80 | Output | Time stamp for the transmitted packet SOP corresponding to the time at which it passed the capture plane. The representation of the bits contained in this bus is the same as the timer input. |
| rx_ptp_tstamp_out_* | 80 | Output | Time stamp for the received packet SOP corresponding to the time at which it passed the capture plane. Note that this signal will be valid starting at the same clock cycle during which the SOP is asserted for one of the segments. The representation of the bits contained in this bus is the same as the timer input. |
| tx_ptp_pcslane_out_* | 2 | Output | This bus identifies which of the PCS lanes that the SOP was detected on for the corresponding timestamp. Note that this signal will be valid starting at the same clock cycle during which the SOP is asserted for one of the segments. |

Table 5-2: Top Level Port List (Cont'd)

| Name | Size | Direction | Description |
|--|------|-----------|--|
| rx_ptp_pcslane_out_* | 2 | Output | This bus identifies which of the PCS lanes that the SOP was detected on for the corresponding timestamp. Note that this signal will be valid starting at the same clock cycle during which the SOP is asserted for one of the segments. |
| rx_lane_aligner_fill_0_* | 7 | Output | This output indicates the fill level of the alignment buffer for PCS lane0. This information can be used by the PTP application, together with the signal rx_ptp_pcslane_out_*, to adjust for the lane skew of the arriving SOP. The units are SerDes clock cycles. |
| rx_lane_aligner_fill_1_* | 7 | Output | This output indicates the fill level of the alignment buffer for PCS lane1. |
| rx_lane_aligner_fill_2_* | 7 | Output | This output indicates the fill level of the alignment buffer for PCS lane2. |
| rx_lane_aligner_fill_3_* | 7 | Output | This output indicates the fill level of the alignment buffer for PCS lane3. |
| AN and LT Interface Control / Status / Statistics Signals | | | |
| Ports under this section will be available when Include AN/LT Logic is selected from the Configuration tab. | | | |
| an_clk_* | 1 | Input | Input Clock for the Auto-Negotiation circuit. |
| an_reset_* | 1 | Input | Asynchronous active-High reset corresponding to an_clk domain. |
| an_loc_np_data_* | 48 | Input | Local Next Page codeword. This is the 48 bit codeword used if the loc_np input is set. In this data field, the bits NP, ACK, & T, bit positions 15, 14, 12, and 11, are not transferred as part of the next page codeword. These bits are generated in the AN IP. However, the Message Protocol bit, MP, in bit position 13, is transferred. |
| an_lp_np_data_* | 48 | Output | Link Partner Next Page Data. This 48 bit word is driven by the AN IP with the 48 bit next page codeword from the remote link partner. |
| lt_tx_sof_* | 4 | Output | This is a link training signal that is asserted for one tx_serdes_clk period at the start of each training frame. It is provided for applications that need to count training frames or synchronize events to the output of the training frames. |
| ctl_autoneg_enable_* | 1 | Input | Enable signal for auto-negotiation. Note: This port is available when the AXI4-Lite interface is not selected. |

Table 5-2: Top Level Port List (Cont'd)

| Name | Size | Direction | Description |
|---------------------------|------|-----------|---|
| ctl_autoneg_bypass_* | 1 | Input | Input to disable auto-negotiation and bypass the auto-negotiation function. If this input is asserted, then auto-negotiation is turned off, but the PCS is connected to the output to allow operation. Note: This port is available when the AXI4-Lite interface is not selected. |
| ctl_an_nonce_seed_* | 8 | Input | 8-bit seed to initialize the nonce field polynomial generator. Note: This port is available when the AXI4-Lite interface is not selected. |
| ctl_an_pseudo_sel_* | 1 | Input | Selects the polynomial generator for the bit 49 random bit generator. If this input is 1, then the polynomial is x^7+x^6+1 . If this input is zero, then the polynomial is x^7+x^3+1 . Note: This port is available when the AXI4-Lite interface is not selected. |
| ctl_restart_negotiation_* | 1 | Input | This input is used to trigger a restart of the auto-negotiation, regardless of what state the circuit is currently in. Note: This port is available when the AXI4-Lite interface is not selected. |
| ctl_an_local_fault_* | 1 | Input | This input signal is used to set the local_fault bit of the transmit link codeword. Note: This port is available when the AXI4-Lite interface is not selected. |
| ctl_an_pause_* | 1 | Input | This input signal is used to set the PAUSE bit, (C0), of the transmit link codeword. This signal may not be present if the core does not support pause. Note: This port is available when the AXI4-Lite interface is not selected. |
| ctl_an_asmdir_* | 1 | Input | This input signal is used to set the ASMDIR bit, (C1), of the transmit link codeword. This signal may not be present if the core does not support pause. Note: This port is available when the AXI4-Lite interface is not selected. |
| ctl_an_fec_10g_request_* | 1 | Input | This signal is used to signal the link partner that the local station is requesting clause 74 FEC on the 10Gb/s lane protocols. Note: This port is available when the AXI4-Lite interface is not selected. |

Table 5-2: Top Level Port List (Cont'd)

| Name | Size | Direction | Description |
|--------------------------------|------|-----------|--|
| ctl_an_fec_25g_rs_request_* | 1 | Input | This signal is used to signal the link partner that the local station is requesting rs FEC (clause 91 or 108) on the 25Gb/s lane protocols. Note: This port is available when the AXI4-Lite interface is not selected. |
| ctl_an_fec_25g_baser_request_* | 1 | Input | This signal is used to signal the link partner that the local station is requesting clause 74 FEC on the 25Gb/s lane protocols. Note: This port is available when the AXI4-Lite interface is not selected. |
| ctl_an_fec_ability_override_* | 1 | Input | Used to set the clause 74 FEC ability bit in the transmit link codeword. If this input is set, the FEC ability bit in the transmit link codeword is cleared. This signal may not be present if the IP core does not support clause 74 FEC. Note: This port is available when the AXI4-Lite interface is not selected. |
| ctl_an_loc_np_* | 1 | Input | Local Next Page indicator. If this bit is a '1', the AN IP transfers the next page word at input loc_np_data to the remote link partner. If this bit is '0', the AN IP does not initiate the next page protocol. If the link partner has next pages to send and the 'loc_np' bit is clear, the AN IP transfers null message pages. |
| ctl_an_lp_np_ack_* | 1 | Input | Link Partner Next Page Acknowledge. This is used to signal the AN IP that the next page data from the remote link partner at output pin 'lp_np_data' has been read by the local host. When this signal goes High, the AN IP acknowledges reception of the next page codeword to the remote link partner and initiate transfer of the next codeword. During this time, the AN IP removes the 'lp_np' signal until the new next page information is available. |
| ctl_an_cl91_fec_request_* | 1 | Input | This bit is used to request clause 91 FEC. Note: This port is available when the AXI4-Lite interface is not selected. |
| ctl_an_cl91_fec_ability_* | 1 | Input | This bit is used to set clause 91 FEC ability. Note: This port is available when the AXI4-Lite interface is not selected. |

Table 5-2: Top Level Port List (Cont'd)

| Name | Size | Direction | Description |
|---------------------------------|------|-----------|--|
| ctl_an_ability_1000base_kx_* | 1 | Input | These inputs identify the Ethernet protocol abilities that are advertised in the transmit link codeword to the link partner. A value of 1 indicates that the interface advertises that it supports the protocol. |
| ctl_an_ability_10gbase_kx4_* | 1 | Input | |
| ctl_an_ability_10gbase_kr_* | 1 | Input | |
| ctl_an_ability_40gbase_kr4_* | 1 | Input | |
| ctl_an_ability_40gbase_cr4_* | 1 | Input | |
| ctl_an_ability_100gbase_cr10_* | 1 | Input | |
| ctl_an_ability_100gbase_kp4_* | 1 | Input | |
| ctl_an_ability_100gbase_kr4_* | 1 | Input | |
| ctl_an_ability_100gbase_cr4_* | 1 | Input | |
| ctl_an_ability_25gbase_krcr_s_* | 1 | Input | |
| ctl_an_ability_25gbase_krcr_* | 1 | Input | |
| ctl_an_ability_25gbase_kr1_* | 1 | Input | |
| ctl_an_ability_25gbase_cr1_* | 1 | Input | |
| ctl_an_ability_50gbase_kr2_* | 1 | Input | |
| ctl_an_ability_50gbase_cr2_* | 1 | Input | |
| ctl_lt_training_enable_* | 1 | Input | Enables link training. When link training is disabled, all PCS lanes function in mission mode. Note: This port is available when the AXI4-Lite interface is not selected. |
| ctl_lt_restart_training_* | 1 | Input | This signal triggers a restart of link training regardless of the current state. Note: This port is available when the AXI4-Lite interface is not selected. |
| ctl_lt_rx_trained_* | 4 | Input | This signal is asserted to indicate that the receiver FIR filter coefficients have all been set, and that the receiver portion of training is complete. Note: This port is available when the AXI4-Lite interface is not selected. |

Table 5-2: Top Level Port List (Cont'd)

| Name | Size | Direction | Description |
|---------------------------|------|-----------|--|
| ctl_lt_preset_to_tx_* | 4 | Input | This signal is used to set the value of the preset bit that is transmitted to the link partner in the control block of the training frame. Note: This port is available when the AXI4-Lite interface is not selected. |
| ctl_lt_initialize_to_tx_* | 4 | Input | This signal is used to set the value of the initialize bit that is transmitted to the link partner in the control block of the training frame. Note: This port is available when the AXI4-Lite interface is not selected. |
| ctl_lt_pseudo_seed0_* | 11 | Input | This 11-bit signal seeds the training pattern generator. Note: This port is available when the AXI4-Lite interface is not selected. |
| ctl_lt_k_p1_to_tx0_* | 2 | Input | This 2-bit field is used to set the value of the k+1 coefficient update field that is transmitted to the link partner in the control block of the training frame. Note: This port is available when the AXI4-Lite interface is not selected. |
| ctl_lt_k0_to_tx0_* | 2 | Input | This 2-bit field is used to set the value of the k0 coefficient update field that is transmitted to the link partner in the control block of the training frame. Note: This port is available when the AXI4-Lite interface is not selected. |
| ctl_lt_k_m1_to_tx0_* | 2 | Input | This 2-bit field is used to set the value of the k-1 coefficient update field that is transmitted to the link partner in the control block of the training frame. Note: This port is available when the AXI4-Lite interface is not selected. |
| ctl_lt_stat_p1_to_tx0_* | 2 | Input | This 2-bit field is used to set the value of the k+1 coefficient update status that is transmitted to the link partner in the status block of the training frame. Note: This port is available when the AXI4-Lite interface is not selected. |
| ctl_lt_stat0_to_tx0_* | 2 | Input | This 2-bit field is used to set the value of the k0 coefficient update status that is transmitted to the link partner in the status block of the training frame. Note: This port is available when the AXI4-Lite interface is not selected. |

Table 5-2: Top Level Port List (Cont'd)

| Name | Size | Direction | Description |
|-------------------------|------|-----------|--|
| ctl_lt_stat_m1_to_tx0_* | 2 | Input | This 2-bit field is used to set the value of the k-1 coefficient update status that is transmitted to the link partner in the status block of the training frame. Note: This port is available when the AXI4-Lite interface is not selected. |
| ctl_lt_pseudo_seed1_* | 11 | Input | This 11-bit signal seeds the training pattern generator. Note: This port is available when the AXI4-Lite interface is not selected. |
| ctl_lt_k_p1_to_tx1_* | 2 | Input | This 2-bit field is used to set the value of the k+1 coefficient update field that is transmitted to the link partner in the control block of the training frame. Note: This port is available when the AXI4-Lite interface is not selected. |
| ctl_lt_k0_to_tx1_* | 2 | Input | This 2-bit field is used to set the value of the k0 coefficient update field that is transmitted to the link partner in the control block of the training frame. Note: This port is available when the AXI4-Lite interface is not selected. |
| ctl_lt_k_m1_to_tx1_* | 2 | Input | This 2-bit field is used to set the value of the k-1 coefficient update field that is transmitted to the link partner in the control block of the training frame. Note: This port is available when the AXI4-Lite interface is not selected. |
| ctl_lt_stat_p1_to_tx1_* | 2 | Input | This 2-bit field is used to set the value of the k+1 coefficient update status that is transmitted to the link partner in the status block of the training frame. Note: This port is available when the AXI4-Lite interface is not selected. |
| ctl_lt_stat0_to_tx1_* | 2 | Input | This 2-bit field is used to set the value of the k0 coefficient update status that is transmitted to the link partner in the status block of the training frame. Note: This port is available when the AXI4-Lite interface is not selected. |
| ctl_lt_stat_m1_to_tx1_* | 2 | Input | This 2-bit field is used to set the value of the k-1 coefficient update status that is transmitted to the link partner in the status block of the training frame. Note: This port is available when the AXI4-Lite interface is not selected. |

Table 5-2: Top Level Port List (Cont'd)

| Name | Size | Direction | Description |
|-------------------------|------|-----------|--|
| ctl_lt_pseudo_seed2_* | 11 | Input | This 11-bit signal seeds the training pattern generator. Note: This port is available when the AXI4-Lite interface is not selected. |
| ctl_lt_k_p1_to_tx2_* | 2 | Input | This 2-bit field is used to set the value of the k+1 coefficient update field that is transmitted to the link partner in the control block of the training frame. Note: This port is available when the AXI4-Lite interface is not selected. |
| ctl_lt_k0_to_tx2_* | 2 | Input | This 2-bit field is used to set the value of the k0 coefficient update field that is transmitted to the link partner in the control block of the training frame. Note: This port is available when the AXI4-Lite interface is not selected. |
| ctl_lt_k_m1_to_tx2_* | 2 | Input | This 2-bit field is used to set the value of the k-1 coefficient update field that is transmitted to the link partner in the control block of the training frame. Note: This port is available when the AXI4-Lite interface is not selected. |
| ctl_lt_stat_p1_to_tx2_* | 2 | Input | This 2-bit field is used to set the value of the k+1 coefficient update status that is transmitted to the link partner in the status block of the training frame. Note: This port is available when the AXI4-Lite interface is not selected. |
| ctl_lt_stat0_to_tx2_* | 2 | Input | This 2-bit field is used to set the value of the k0 coefficient update status that is transmitted to the link partner in the status block of the training frame. Note: This port is available when the AXI4-Lite interface is not selected. |
| ctl_lt_stat_m1_to_tx2_* | 2 | Input | This 2-bit field is used to set the value of the k-1 coefficient update status that is transmitted to the link partner in the status block of the training frame. Note: This port is available when the AXI4-Lite interface is not selected. |
| ctl_lt_pseudo_seed3_* | 11 | Input | This 11-bit signal seeds the training pattern generator. Note: This port is available when the AXI4-Lite interface is not selected. |

Table 5-2: Top Level Port List (Cont'd)

| Name | Size | Direction | Description |
|-------------------------|------|-----------|--|
| ctl_lt_k_p1_to_tx3_* | 2 | Input | This 2-bit field is used to set the value of the k+1 coefficient update field that is transmitted to the link partner in the control block of the training frame. Note: This port is available when the AXI4-Lite interface is not selected. |
| ctl_lt_k0_to_tx3_* | 2 | Input | This 2-bit field is used to set the value of the k0 coefficient update field that is transmitted to the link partner in the control block of the training frame. Note: This port is available when the AXI4-Lite interface is not selected. |
| ctl_lt_k_m1_to_tx3_* | 2 | Input | This 2-bit field is used to set the value of the k-1 coefficient update field that is transmitted to the link partner in the control block of the training frame. Note: This port is available when the AXI4-Lite interface is not selected. |
| ctl_lt_stat_p1_to_tx3_* | 2 | Input | This 2-bit field is used to set the value of the k+1 coefficient update status that is transmitted to the link partner in the status block of the training frame. Note: This port is available when the AXI4-Lite interface is not selected. |
| ctl_lt_stat0_to_tx3_* | 2 | Input | This 2-bit field is used to set the value of the k0 coefficient update status that is transmitted to the link partner in the status block of the training frame. Note: This port is available when the AXI4-Lite interface is not selected. |
| ctl_lt_stat_m1_to_tx3_* | 2 | Input | This 2-bit field is used to set the value of the k-1 coefficient update status that is transmitted to the link partner in the status block of the training frame. Note: This port is available when the AXI4-Lite interface is not selected. |

Table 5-2: Top Level Port List (Cont'd)

| Name | Size | Direction | Description |
|------------------------------------|------|-----------|---|
| stat_an_link_cntl_1000base_kx_* | 2 | Output | Link Control outputs from the auto-negotiation controller for the various Ethernet protocols. Settings are as follows: <ul style="list-style-type: none"> • 00: DISABLE; PCS is disconnected; • 01: SCAN_FOR_CARRIER; RX is connected to PCS; • 11: ENABLE; PCS is connected for mission mode operation. • 10: not used |
| stat_an_link_cntl_10gbase_kx4_* | 2 | Output | |
| stat_an_link_cntl_10gbase_kr_* | 2 | Output | |
| stat_an_link_cntl_40gbase_kr4_* | 2 | Output | |
| stat_an_link_cntl_40gbase_cr4_* | 2 | Output | |
| stat_an_link_cntl_100gbase_cr10_* | 2 | Output | |
| stat_an_link_cntl_100gbase_kp4_* | 2 | Output | |
| stat_an_link_cntl_100gbase_kr4_* | 2 | Output | |
| stat_an_link_cntl_100gbase_cr4_* | 2 | Output | |
| stat_an_link_cntl_25gbase_krcr_s_* | 2 | Output | |
| stat_an_link_cntl_25gbase_krcr_* | 2 | Output | |
| stat_an_link_cntl_25gbase_kr1_* | 2 | Output | |
| stat_an_link_cntl_25gbase_cr1_* | | | |
| stat_an_link_cntl_50gbase_kr2_* | | | |
| stat_an_link_cntl_50gbase_cr2_* | | | |
| stat_an_fec_enable_* | 1 | Output | Used to enable the use of clause 74 FEC on the link. |
| stat_an_tx_pause_enable_* | 1 | Output | Used to enable station-to-station (global) pause packet generation in the transmit path to control data flow in the receive path. |
| stat_an_rx_pause_enable_* | 1 | Output | Used to enable station-to-station (global) pause packet interpretation in the receive path, in order to control data flow from the transmitter. |
| stat_an_autoneg_complete_* | 1 | Output | Indicates the auto-negotiation is complete and rx link status from the PCS has been received. |
| stat_an_parallel_detection_fault_* | 1 | Output | Indicates a parallel detection fault during auto-negotiation. |
| stat_an_start_tx_disable_* | 1 | Output | This signal is asserted during auto-negotiation for one AN_CLK period to signal the start of the TX_DISABLE phase at the very start of auto-negotiation. |

Table 5-2: Top Level Port List (Cont'd)

| Name | Size | Direction | Description |
|-------------------------------------|------|-----------|---|
| stat_an_start_an_good_check_* | 1 | Output | This signal is asserted during auto-negotiation for one AN_CLK period to signal the start of the AN_GOOD_CHECK phase, when the selected protocol has been enabled, and the circuit is awaiting rx_pcs_status. |
| stat_an_lp_ability_1000base_kx_* | 1 | Output | These signals indicate the advertised protocol from the link partner. They all become valid when the output signal stat_an_lp_ability_valid is asserted. A value of 1 indicates that the protocol is advertised as supported by the link partner. |
| stat_an_lp_ability_10gbase_kx4_* | 1 | Output | |
| stat_an_lp_ability_10gbase_kr_* | 1 | Output | |
| stat_an_lp_ability_40gbase_kr4_* | 1 | Output | |
| stat_an_lp_ability_40gbase_cr4_* | 1 | Output | |
| stat_an_lp_ability_100gbase_cr10_* | 1 | Output | |
| stat_an_lp_ability_100gbase_kp4_* | 1 | Output | |
| stat_an_lp_ability_100gbase_kr4_* | 1 | Output | |
| stat_an_lp_ability_100gbase_cr4_* | 1 | Output | |
| stat_an_lp_ability_25gbase_krcr_s_* | 1 | Output | |
| stat_an_lp_ability_25gbase_krcr_* | 1 | Output | |
| stat_an_lp_pause_* | 1 | Output | |
| stat_an_lp_asm_dir_* | 1 | Output | This signal indicates the advertised value of the ASMDIR bit, (C1), in the receive link codeword from the link partner. It becomes valid when the output signal stat_an_lp_ability_valid is asserted. |
| stat_an_lp_rf_* | 1 | Output | This bit indicates link partner remote fault. |
| stat_an_lp_fec_10g_ability_* | 1 | Output | This signal indicates the clause 74 FEC ability associated with 10Gb/s lane protocols that is being advertised by the link partner. It becomes valid when the output signal stat_an_lp_ability_valid is asserted. |
| stat_an_lp_fec_10g_request_* | 1 | Output | This signal indicates that the link partner is requesting that the clause 74 FEC be used on the 10Gb/s lane protocols. It becomes valid when the output signal stat_an_lp_ability_valid is asserted. |

Table 5-2: Top Level Port List (Cont'd)

| Name | Size | Direction | Description |
|------------------------------------|------|-----------|--|
| stat_an_lp_fec_25g_rs_request_* | 1 | Output | This signal indicates that the link partner is requesting the clause 91 (or 108) rs FEC be used for the 25gb/s lane protocols. It becomes valid when the output signal stat_an_lp_ability_valid is asserted. |
| stat_an_lp_fec_25g_baser_request_* | 1 | Output | This signal indicates that the link partner is requesting the clause 74 FEC be used for the 25Gb/s lane base-r protocols. It becomes valid when the output signal stat_an_lp_ability_valid is asserted. |
| stat_an_lp_autoneg_able_* | 1 | Output | This output signal indicates that the link partner is able to perform auto-negotiation. It becomes valid when the output signal stat_an_lp_ability_valid is asserted. |
| stat_an_lp_ability_valid_* | 1 | Output | This signal indicates when all of the link partner advertisements become valid. |
| stat_an_loc_np_ack_* | 1 | Output | This signal is used to indicate to the local host that the local next page data, presented at input pin loc_np_data, has been taken. This signal pulses High for 1 clock period when the AN IP samples the next page data on input pin oc_np_data. When the local host detects this signal High, it must replace the 48 bit next page codeword at input pin loc_np_data with the next 48-bit codeword to be sent. If the local host has no more next pages to send, it must clear the loc_np input. |
| stat_an_lp_np_* | 1 | Output | Link Partner Next Page. This signal is used to indicate that there is a valid 48-bit next page codeword from the remote link partner at output pin lp_np_data. This signal is driven Low when the lp_np_ack input signal is driven High, indicating that the local host has read the next page data. It remains Low until the next codeword becomes available on the lp_np_data output pin; then the lp_np output is driven High again. |
| stat_an_lp_ability_25gbase_kr1_* | 1 | Output | Indicates the advertised protocol from the link partner. Becomes valid when the output signal stat_an_lp_extended_ability_valid is asserted. A value of 1 indicates that the protocol is advertised as supported by the link partner. |
| stat_an_link_cntl_25gbase_cr1_* | 1 | Output | Indicates the advertised protocol from the link partner. Becomes valid when the output signal stat_an_lp_extended_ability_valid is asserted. A value of 1 indicates that the protocol is advertised as supported by the link partner. |

Table 5-2: Top Level Port List (Cont'd)

| Name | Size | Direction | Description |
|-------------------------------------|------|-----------|---|
| stat_an_lp_ability_50gbase_kr2_* | 1 | Output | Indicates the advertised protocol from the link partner. Becomes valid when the output signal stat_an_lp_extended_ability_valid is asserted. A value of 1 indicates that the protocol is advertised as supported by the link partner. |
| stat_an_lp_ability_50gbase_cr2_* | 1 | Output | Indicates the advertised protocol from the link partner. Becomes valid when the output signal stat_an_lp_extended_ability_valid is asserted. A value of 1 indicates that the protocol is advertised as supported by the link partner. |
| stat_an_lp_ability_extended_fec_* | 4 | Output | This output indicates the extended FEC abilities. |
| stat_an_rs_fec_enable_* | 1 | Output | Used to enable the use of clause 91 FEC on the link. |
| stat_an_lp_extended_ability_valid_* | 1 | Output | When this bit is 1, it indicates that the detected extended abilities are valid. |
| stat_lt_signal_detect_* | 4 | Output | This signal indicates when the respective link training state machine has entered the SEND_DATA state, in which normal PCS operation may resume. |
| stat_lt_training_* | 4 | Output | This signal indicates when the respective link training state machine is performing link training. |
| stat_lt_training_fail_* | 4 | Output | This signal is asserted during link training if the corresponding link training state machine detects a time-out during the training period. |
| stat_lt_rx_sof_* | 4 | Output | This output is High for 1 RX SerDes clock cycle to indicate the start of the link training frame. |
| stat_lt_frame_lock_* | 4 | Output | When link training has begun, these signals are asserted, for each PMD lane, when the corresponding link training receiver is able to establish a frame synchronization with the link partner. |
| stat_lt_preset_from_rx_* | 4 | Output | This signal reflects the value of the preset control bit received in the control block from the link partner. |
| stat_lt_initialize_from_rx_* | 4 | Output | This signal reflects the value of the initialize control bit received in the control block from the link partner. |
| stat_lt_k_p1_from_rx0_* | 2 | Output | This 2-bit field indicates the update control bits for the k+1 coefficient, as received from the link partner in the control block. |
| stat_lt_k0_from_rx0_* | 2 | Output | This 2-bit field indicates the update control bits for the k0 coefficient, as received from the link partner in the control block. |

Table 5-2: Top Level Port List (Cont'd)

| Name | Size | Direction | Description |
|----------------------------|------|-----------|---|
| stat_lt_k_m1_from_rx0_* | 2 | Output | This 2-bit field indicates the update control bits for the k-1 coefficient, as received from the link partner in the control block. |
| stat_lt_stat_p1_from_rx0_* | 2 | Output | This 2-bit field indicates the update status bits for the k+1 coefficient, as received from the link partner in the status block. |
| stat_lt_stat0_from_rx0_* | 2 | Output | This 2-bit fields indicates the update status bits for the k0 coefficient, as received from the link partner in the status block. |
| stat_lt_stat_m1_from_rx0_* | 2 | Output | This 2-bit field indicates the update status bits for the k-1 coefficient, as received from the link partner in the status block. |
| stat_lt_k_p1_from_rx1_* | 2 | Output | This 2-bit field indicates the update control bits for the k+1 coefficient, as received from the link partner in the control block. |
| stat_lt_k0_from_rx1_* | 2 | Output | This 2-bit field indicates the update control bits for the k0 coefficient, as received from the link partner in the control block. |
| stat_lt_k_m1_from_rx1_* | 2 | Output | This 2-bit field indicates the update control bits for the k-1 coefficient, as received from the link partner in the control block. |
| stat_lt_stat_p1_from_rx1_* | 2 | Output | This 2-bit field indicates the update status bits for the k+1 coefficient, as received from the link partner in the status block. |
| stat_lt_stat0_from_rx1_* | 2 | Output | This 2-bit fields indicates the update status bits for the k0 coefficient, as received from the link partner in the status block. |
| stat_lt_stat_m1_from_rx1_* | 2 | Output | This 2-bit field indicates the update status bits for the k-1 coefficient, as received from the link partner in the status block. |
| stat_lt_k_p1_from_rx2_* | 2 | Output | This 2-bit field indicates the update control bits for the k+1 coefficient, as received from the link partner in the control block. |
| stat_lt_k0_from_rx2_* | 2 | Output | This 2-bit field indicates the update control bits for the k0 coefficient, as received from the link partner in the control block. |
| stat_lt_k_m1_from_rx2_* | 2 | Output | This 2-bit field indicates the update control bits for the k-1 coefficient, as received from the link partner in the control block. |
| stat_lt_stat_p1_from_rx2_* | 2 | Output | This 2-bit field indicates the update status bits for the k+1 coefficient, as received from the link partner in the status block. |
| stat_lt_stat0_from_rx2_* | 2 | Output | This 2-bit fields indicates the update status bits for the k0 coefficient, as received from the link partner in the status block. |

Table 5-2: Top Level Port List (Cont'd)

| Name | Size | Direction | Description |
|--|------|-----------|--|
| stat_lt_stat_m1_from_rx2_* | 2 | Output | This 2-bit field indicates the update status bits for the k-1 coefficient, as received from the link partner in the status block. |
| stat_lt_k_p1_from_rx3_* | 2 | Output | This 2-bit field indicates the update control bits for the k+1 coefficient, as received from the link partner in the control block. |
| stat_lt_k0_from_rx3_* | 2 | Output | This 2-bit field indicates the update control bits for the k0 coefficient, as received from the link partner in the control block. |
| stat_lt_k_m1_from_rx3_* | 2 | Output | This 2-bit field indicates the update control bits for the k-1 coefficient, as received from the link partner in the control block. |
| stat_lt_stat_p1_from_rx3_* | 2 | Output | This 2-bit field indicates the update status bits for the k+1 coefficient, as received from the link partner in the status block. |
| stat_lt_stat0_from_rx3_* | 2 | Output | This 2-bit fields indicates the update status bits for the k0 coefficient, as received from the link partner in the status block. |
| stat_lt_stat_m1_from_rx3_* | 2 | Output | This 2-bit field indicates the update status bits for the k-1 coefficient, as received from the link partner in the status block. |
| Clause 74 FEC Interface Control / Status / Statistics Signals | | | |
| Ports under this section will be available when Clause 74 (BASE-KR FEC) is selected from the Configuration tab. | | | |
| ctl_fec_tx_enable_* | 1 | Input | Asserted to enable the clause 74 FEC encoding on the transmitted data. |
| ctl_fec_rx_enable_* | 1 | Input | Asserted to enable the clause 74 FEC decoding of the received data. |
| stat_fec_inc_correct_count_* | 4 | Output | This signal will be asserted roughly every 32 words, while the ctl_rx_fec_enable is asserted, if the FEC decoder detected and corrected a bit errors in the corresponding frame. |
| stat_fec_inc_cant_correct_count_* | 4 | Output | This signal will be asserted roughly every 32 words, while the ctl_rx_fec_enable is asserted, if the FEC decoder detected bit errors in the frame that it was unable to correct. |
| stat_fec_lock_error_* | 4 | Output | ctl_fec_rx_enable is asserted if the FEC decoder has been unable to detect the frame boundary after about 5 ms. It is cleared when the frame boundary is detected. |
| stat_fec_rx_lock_* | 4 | Output | This signal is asserted while the ctl_fec_rx_enable is asserted when the FEC decoder detects the frame boundary. |

Duplex Mode of Operation

In this mode of operation, both the core transmitter and receiver are active and loop back is provided at the GT output interface; that is, output is fed back as input. Packet generation and monitor modules are active in this mode. The generator module is responsible for generating the desired number of packets and transmitting to the core using the available data interface. Monitor module checks the packets from the receiver.

Figure 5-3 shows the duplex mode of operation.

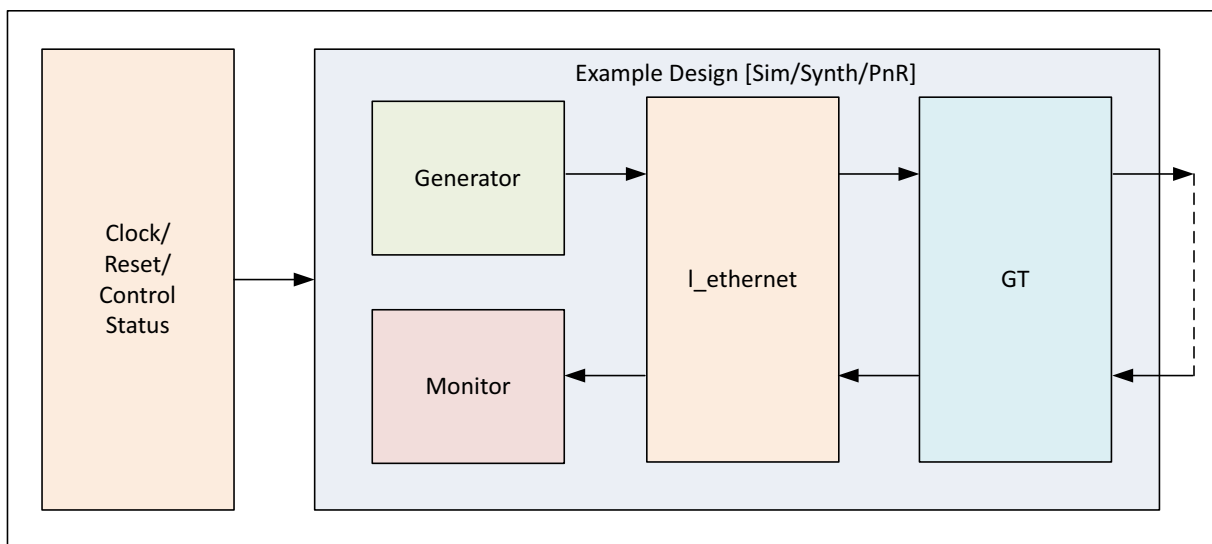


Figure 5-3: Duplex Mode of Operation

Shared Logic Implementation

Shared logic includes the GT common module that can be present as part of the core or in the example design.

By default, GT common is present inside the core. If you want to instantiate GT common in the example design, you must select the **Include Shared Logic in example design** option in the Shared Logic tab.

Figure 5-4 shows the implementation when shared logic is instantiated in the example design for a single core.

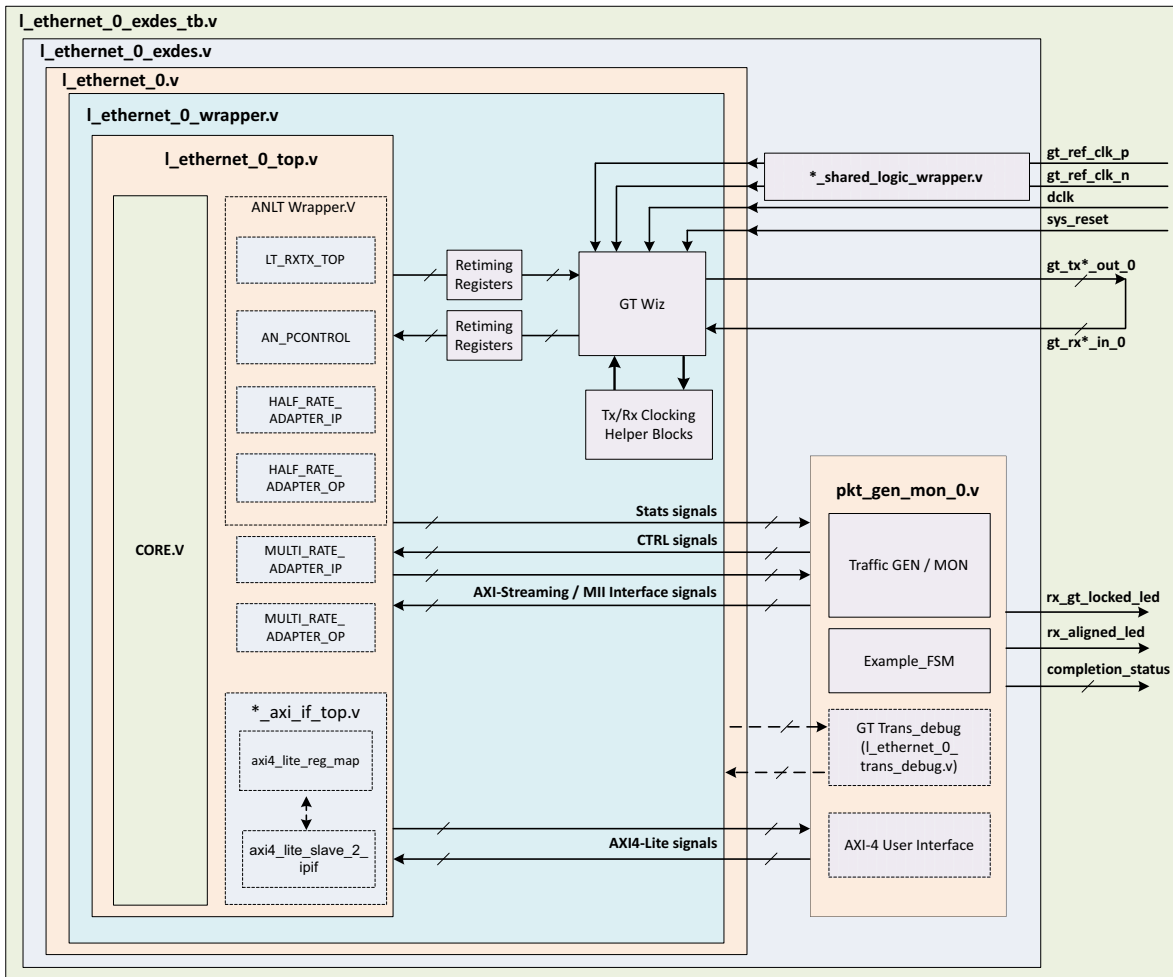


Figure 5-4: Single Core Example Design Hierarchy with Shared Logic Implementation

Figure 5-5 shows the implementation when shared logic is instantiated in the example design for multiple cores.

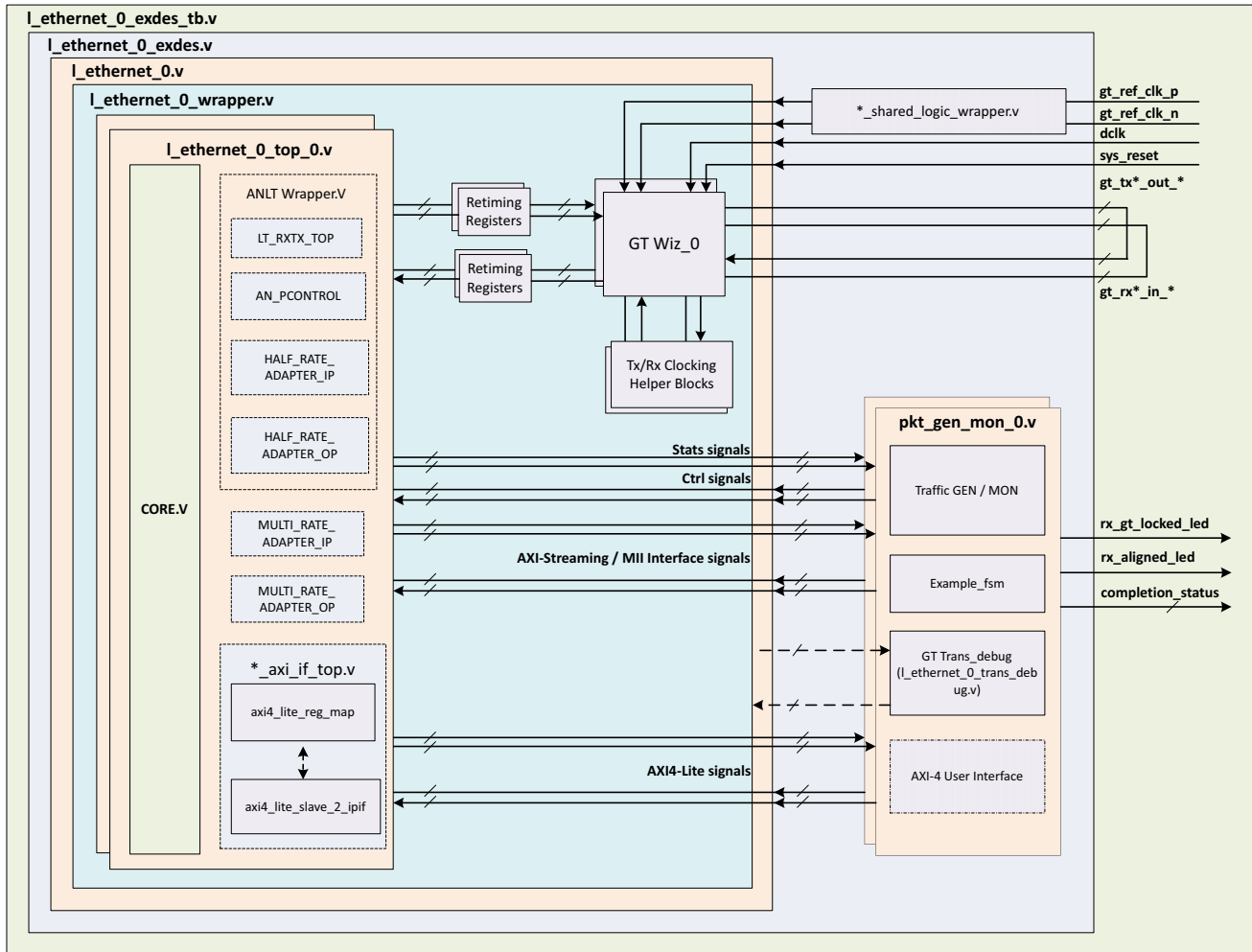


Figure 5-5: Multiple Core Example Design Hierarchy with Shared Logic Implementation

AXI4-Lite Interface Implementation

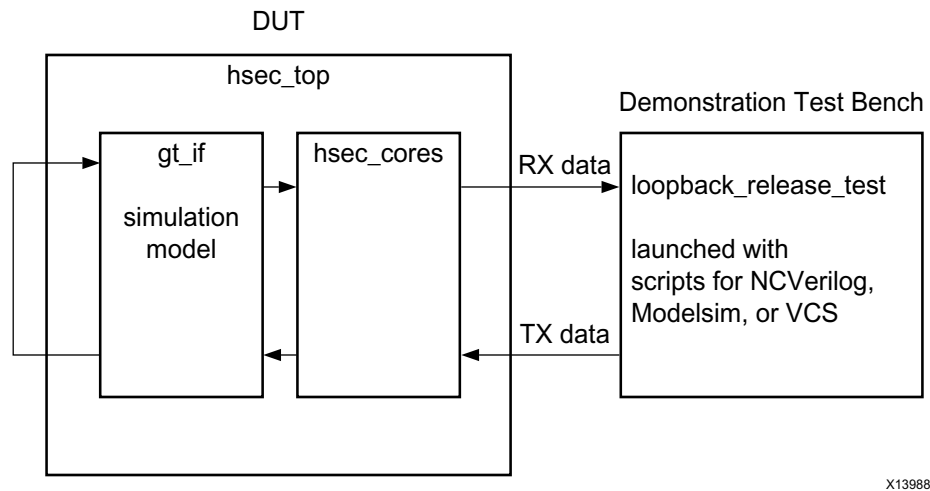
If you want to instantiate the AXI4-Lite interface to access the control and status registers of the `l_ethernet` core, you must enable the **Include AXI4-Lite** check box in the Configuration tab. It enables the `l_ethernet _0_axi_if_top` module (that contains `l_ethernet_0_pif_registers` with the `l_ethernet _0_slave_2_ipif` module). You can access the AXI4-lite interface logic registers (control, status and statistics) from the `l_ethernet _0_pkt_gen_mon` module.

This mode enables the following features:

- You can configure all the control (CTL) ports of the core through the AXI4-Lite interface. This operation is performed by writing to a set of address locations with the required data to the register map interface.
- You can access all the status and statistics registers from the core through the AXI4-Lite interface. This is performed by reading the address locations for the status and statistics registers through register map.

Test Bench

Each release includes a demonstration test bench that performs a loopback test on the complete 40G/50G High Speed Ethernet Subsystem. The test program exercises the datapath to check that the transmitted frames are received correctly. RTL simulation models for the 40G/50G High Speed Ethernet Subsystem are included. You must provide the correct path for the transceiver simulation model according to the latest simulation environment settings in your version of the Vivado® Design Suite.



X13988

Figure 6-1: Test Bench

Migrating and Upgrading

This is the first version of this core so this appendix is not applicable.

Upgrading in the Vivado Design Suite

Changes from v1.0 to v1.1

Ports Added

The following ports have been added.

- `ctl_gt_reset_all_*`
- `ctl_gt_reset_*`
- `ctl_gt_rx_reset_*`
- `gt_reset_all_in_*`
- `gt_tx_reset_in_*`
- `gt_rx_reset_in_*`
- `ctl_an_fec_25g_rs_request`
- `ctl_an_fec_25g_baser_request`
- `stat_an_lp_fec_25g_rs_request`
- `stat_an_lp_fec_25g_baser_request`
- `tx_ptp_rxtstamp_in`
- `rx_ptp_pcslane_out_*`
- `ctl_an_fec_25g_baser_request_*`

Ports Removed

- `stat_tx_underflow_err`

Port Changes

- rx_serdes_reset_done_in_* to rx_serdes_reset_*.
- SERDES_WIDTH with 64 for several I/O ports in Table 2-1.
- ctl_an_ability_25gbase_cr to ctl_an_ability_25gbase_krcr
- ctl_an_ability_25gbase_kr to ctl_an_ability_25gbase_krcr_s
- ctl_an_fec_request to ctl_an_fec_10g_request. Updated description.
- stat_rx_errored_block_increment_valid to stat_rx_bad_code_valid
- stat_rx_bad_sh_increment_0[3:0] to stat_rx_framing_err_0[3:0]
- stat_rx_bad_sh_increment_1[3:0] to stat_rx_framing_err_1[3:0]
- stat_rx_bad_sh_increment_2[3:0] to stat_rx_framing_err_2[3:0]
- stat_rx_bad_sh_increment_3[3:0] to stat_rx_framing_err_3[3:0]
- stat_rx_bad_sh_increment_valid_1 to stat_rx_valid_1
- stat_rx_bad_sh_increment_valid_2 to stat_rx_valid_2
- stat_rx_bad_sh_increment_valid_3 to stat_rx_valid_3
- rx_axis_tuser_0 to rx_axis_tuser_*

Hex Addresses Added

- CONFIGURATION_LT_SEED_REG1: 0114
- CONFIGURATION_LT_COEFFICIENT_REG1: 0134
- STAT_LT_COEFFICIENT1_REG: 0478
- STAT_RX_ERROR_LSB: 0668
- STAT_RX_ERROR_LSB: 066C

Signals Added to Register Definitions

- ctl_gt_rx_reset
- ctl_gt_tx_reset
- ctl_tx_ipg_value
- ctl_tx_custom_preamble_enable
- ctl_rx_custom_preamble_enable
- ctl_an_fec_25g_rs_request
- ctl_an_fec_25g_baser_request

- stat_an_lp_fec_25g_rs_request
- stat_an_lp_fec_25g_baser_request

Register Tables Added

- CONFIGURATION_LT_SEED_REG1: 0114
- CONFIGURATION_LT_COEFFICIENT_REG1: 0134
- STAT_LT_COEFFICIENT1_REG: 0478
- STAT_RX_ERROR_LSB: 0668
- STAT_RX_ERROR_MSB: 066C

Debugging

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

Finding Help on Xilinx.com

To help in the design and debug process when using the High Speed Ethernet IP core, the [Xilinx 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.

Documentation

This product guide is the main document associated with the High Speed Ethernet IP core. This guide, along with documentation related to all products that aid in the design process, can be found on the [Xilinx Support web page](#) or by using the Xilinx® Documentation Navigator.

Download the Xilinx 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 for this core can be located by using the Search Support box on the main [Xilinx support web page](#). To maximize your search results, use proper keywords such as

- Product name
- Tool message(s)
- Summary of the issue encountered

A filter search is available after results are returned to further target the results.

Master Answer Record for the 40G/50G High Speed Ethernet Core

AR: [54690](#)

Technical Support

Xilinx provides technical support at the [Xilinx Support web page](#) for this LogiCORE™ IP product when used as described in the product documentation. Xilinx 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 contact Xilinx Technical Support, navigate to the [Xilinx Support web page](#).

Debug Tools

There are many tools available to address High Speed Ethernet IP core debug issues. It is important to know which tools are useful for debugging various situations.

Example Design

The High Speed Ethernet IP core is delivered with an example design netlist complete with functional test benches. The design includes example transceivers and loopback tests for common simulator programs.

Vivado Design Suite Debug Feature

The 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 Integrated Design Environment is used for logic debugging and validation of a design running in Xilinx devices.

The Vivado logic analyzer is used with the logic debug IP cores, including:

- ILA 2.0 (and later versions)
- VIO 2.0 (and later versions)

See the *Vivado Design Suite User Guide: Programming and Debugging* (UG908) [\[Ref 10\]](#).

Reference Boards

Various Xilinx development boards support the High Speed Ethernet IP core. These boards can be used to prototype designs and establish that the core can communicate with the system. UltraScale™ devices are recommended for optimum performance. Ensure that the board transceivers support the required Ethernet bit rate. For example, the following board is suitable for many UltraScale implementations: UltraScale FPGA evaluation board, VCU108.

Simulation Debug

Each High Speed Ethernet IP core release includes a sample simulation test bench. This typically consists of a loopback from the TX side of the user interface, through the TX circuit, looping back to the RX circuit, and checking the received packets at the RX side of the user interface.

Each release usually includes a sample instantiation of a Xilinx transceiver corresponding to the device selected by the customer. The loopback simulation includes a path through the transceiver.

The simulation is run using provided scripts for several common industry-standard simulators.

If the simulation does not run properly from the scripts, the following items should be checked.

Simulator License Availability

If the simulator does not launch, you might not have a valid license. Ensure that the license is up to date. It is also possible that your organization has a license available for one of the other simulators, so try all the provided scripts.

Library File Location

Each simulation script calls up the required Xilinx library files. These are called by the corresponding `liblist*` file in the `bin` directory of each release.

There can be an error message indicating that the simulator is unable to find certain library files. In this case, the path to the library files might have to be modified. Check with your IT administrator to ensure that the paths are correct.

Version Compatibility

Each release has been tested according to the Xilinx tools version requested by customers. If the simulation does not complete successfully, you should first ensure that a properly up-to-date version of the Xilinx tools is used. The preferred version is indicated in the README file of the release, and is also indicated in the simulation sample log file included with the release.

Slow Simulation

Simulations can appear to run slowly under some circumstances. If a simulation is unacceptably slow, the following suggestions might improve the run time performance.

- Use a faster computer with more memory.
- Make use of a Platform Load Sharing Facility (LSF) if available in your organization.
- Bypass the Xilinx transceiver (this might require that you create your own test bench)
- Send fewer packets. This can be accomplished by modifying the appropriate parameter in the provided sample test bench.
- Specify a shorter time between alignment markers. This should result in a shorter lane alignment phase, at the expense of more overhead. However, when the High Speed Ethernet IP core is finally implemented in hardware, the distance between alignment markers should follow the specification requirements (after every 16,383 words). Contact Xilinx technical support for assistance if required.

Simulation Fails Before Completion

If the sample simulation fails or hangs before successfully completing, it is possible that a timeout has occurred. Ensure that the simulator timeouts are long enough to accommodate the waiting periods in the simulation, for example, during the lane alignment phase.

Simulation Completes But Fails

If the sample simulation completes with a failure, contact Xilinx technical support. Each release is tested prior to shipment and normally completes successfully. Consult the sample simulation log file for the expected behavior.

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 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 debugging tool for debugging the specific problems.

Many of these common issues can also be applied to debugging design simulations. Details are provided on:

- General Checks
- Transceiver Specific Checks
- Ethernet Specific Checks

General Checks

Ensure that all the timing constraints for the core were properly incorporated from the example design and that all constraints were met during implementation.

- 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 mixed-mode clock managers (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.

Transceiver Specific Checks

- Ensure that the polarities of the `txn/txp` and `rxn/rxp` lines are not reversed. If they are, these can be fixed by using the `TXPOLARITY` and `RXPOLARITY` ports of the transceiver.
- Check that the transceiver is not being held in reset or still being initialized. The `RESETDONE` outputs from the transceiver indicate when the transceiver is ready.
- Place the transceiver into parallel or serial near-end loopback.
- If correct operation is seen in the transceiver serial loopback, but not when loopback is performed through an optical cable, it might indicate a faulty optical module.
- If the core exhibits correct operation in the transceiver parallel loopback but not in serial loopback, this might indicate a transceiver issue.
- A mild form of bit error rate might be solved by adjusting the transmitter Pre-Emphasis and Differential Swing Control attributes of the transceiver.

Ethernet Specific Checks

Several issues can commonly occur during the first hardware test of an Ethernet IP core. These should be checked as indicated in the following subsections.

It is assumed that the Ethernet IP core has already passed all simulation testing which is being implemented in hardware. This is a prerequisite for any kind of hardware debug.

The usual sequence of debugging is to proceed in the following sequence:

1. Clean up signal integrity.
2. Ensure that each SerDes achieves clock data recovery (CDR) lock.
3. Check that each lane has achieved word alignment.
4. Check that lane alignment has been achieved.
5. Proceed to Interface and Protocol debug.

Signal Integrity

When bringing up a board for the first time and the High Speed Ethernet IP core does not seem to be achieving lane alignment, the most likely issue is related to signal integrity.



IMPORTANT: *Signal integrity issues must be addressed before any other debugging can take place.*

Even if lane alignment is achieved, if there are periodic bip-8 errors, signal integrity issues are indicated. Check the bip-8 signals to assist with debug.

Signal integrity should be debugged independently from the High Speed Ethernet IP core. The following procedures should be carried out.

Note: It assumed that the PCB itself has been designed and manufactured in accordance with the required trace impedances and trace lengths, including the requirements for skew set out in the *IEEE Standard for Ethernet* (IEEE Std 802.3-2012).

- Transceiver Settings
- Checking For Noise
- Bit Error Rate Testing

If assistance is required for transceiver and signal integrity debugging, contact Xilinx technical support.

Lane Swapping

In Ethernet, physical lanes can be swapped and the protocol aligns lanes correctly. Therefore, lane swapping should not cause any problems.

N/P Swapping

If the positive and negative signals of a differential pair are swapped, data is not received correctly on that lane. You should verify that each link has the correct polarity of each differential pair.

Clocking and Resets

See [Chapter 3, Designing with the Core](#) for these requirements.

Ensure that the clock frequencies for both the High Speed Ethernet IP core as well as the Xilinx transceiver reference clock match the configuration requested when the IP core was ordered. The core clock has a minimum frequency associated with it. The maximum core clock frequency is determined by timing constraints. The minimum core clock frequency is derived from the required Ethernet bandwidth plus the margin reserved for clock tolerance, wander and jitter.

The first thing to verify during debugging is to ensure that resets remain asserted until the clock is stable. It must be frequency-stable as well as free from glitches before the High Speed Ethernet IP core is taken out of reset. This applies to both the SerDes clock as well as the IP core clock.

If any subsequent instability is detected in a clock, the High Speed Ethernet IP core must be reset. One example of such instability is a loss of CDR lock. The user logic should determine all external conditions that would require a reset (for example, clock glitches, loss of CDR lock, power supply glitches, etc.).

Configuration changes cannot be made unless the IP core is reset. An example of a configuration change would be setting a different maximum packet length. Check the description for the particular signal on the port list to determine if this requirement applies to the parameter that is being changed.

Interface Debug

AXI4-Stream Interface

The High Speed Ethernet IP core user interface is called the AXI4-Stream. There are two versions used — regular AXI4-Stream and segmented AXI4-Stream. See the appropriate section in this guide for a detailed description of each.

TX Debug (Buffer Errors)

TX debug is assisted by using several diagnostic signals.

Data must be written to the TX AXI4-Stream such that there are no overflow or underflow conditions. AXI4-Stream bandwidth must always be greater than the Ethernet bandwidth to guarantee that data can be sent without interruption.

When writing data to the AXI4-Stream, the `tx_rdyout` signal must always be observed. This signal indicates whether the fill level of the TX buffer is within an acceptable range or not. If this signal is ever asserted, you must stop writing to the TX AXI4-Stream until the signal is deasserted.

Because the TX AXI4-Stream has greater bandwidth than the TX Ethernet interface, it is not unusual to see this signal being frequently asserted and this is not a cause for concern. You must ensure that TX writes are stopped when `tx_rdyout` is asserted.

The level at which `tx_rdyout` becomes asserted is determined by a pre-determined threshold.

When a packet data transaction has begun in the TX direction, it must continue until completion or there can be a buffer underflow as indicated by the signal `tx_unfout`. This must not be allowed to occur; data must be written on the TX AXI4-Stream without interruption. Ethernet packets must be present on the line from start to end with no gaps or idles. If `tx_unfout` is ever asserted, debugging must stop until the condition which caused the underflow has been addressed.

RX Debug (Receiver Errors)

Consult [Port Descriptions in Chapter 2](#) for a description of the diagnostic signals which are available to debug the RX.

If the Ethernet packets are being transmitted properly according to IEEE Std 802.3-2012, there should not be RX errors. However, the signal integrity of the received signals must be verified first.

The `stat_rx_bip_err` signals provide a per-lane indicator of signal quality. The `stat_rx_hi_ber` signal is asserted when the bit error rate is too high, according to IEEE Std 802.3-2012. The threshold is BER = 10⁻⁴.

To aid in debug, GT near-end PMA loopback can be performed with the signal `gt_loopback_in`. This connects the TX SerDes to the RX SerDes, effectively bypassing potential signal integrity problems. In this way, the received data can be checked against the transmitted packets to verify that the logic is operating properly.

Protocol Debug

To achieve error-free data transfers with the Ethernet IP core, the IEEE Std 802.3-2012 should be followed. Signal integrity should always be ensured before proceeding to the protocol debug.

Alignment Marker Spacing

According to IEEE Std 802.3-2012, the alignment marker spacing should be set to 16,383 for both the TX and RX. Check that both ends of the link are programmed to this value.

Diagnostic Signals

There are many error indicators available to check for protocol violations. Carefully read the description of each one to see if it is useful for a particular debugging problem.

The following is a suggested debug sequence.

1. Ensure that Word sync has been achieved.
2. Ensure that Lane sync has been achieved (this uses the lane marker alignment words which occur after every 16,383 words).
3. Verify that the bip8 indicators are clean.
4. Make sure there are no descrambler state errors.
5. Eliminate CRC32 errors, if any.
6. Make sure the AXI4-Stream protocol is being followed correctly.
7. Ensure that there are no overflow or underflow conditions when packets are sent.

Statistics Counters

When error-free communication has been achieved, the statistics indicators can be monitored to ensure that traffic characteristics meet expectations. Some signals are strobes only, which means that the counters are not part of the IP core. This is done so you can customize the counter size. The counters are optional.

Pause Processing Interface

The 40G/50G High Speed Ethernet Subsystem provides a comprehensive mechanism for pause packet termination and generation. The TX and RX have independent interfaces for processing pause information as described in this appendix.

TX Pause Generation

You can request a pause packet to be transmitted using the `ctl_tx_pause_req[8:0]` and `ctl_tx_pause_enable[8:0]` input buses. Bit [8] corresponds to global pause packets and bits [7:0] correspond to priority pause packets.



IMPORTANT: *Requesting both global and priority pause packets at the same time results in unpredictable behavior and must be avoided.*

The contents of the pause packet are determined using the following input pins.

Global pause packets:

- `ctl_tx_da_gpp[47:0]`
- `ctl_tx_sa_gpp[47:0]`
- `ctl_tx_ethertype_gpp[15:0]`
- `ctl_tx_opcode_gpp[15:0]`
- `ctl_tx_pause_quanta8[15:0]`

Priority pause packets:

- `ctl_tx_da_ppp[47:0]`
- `ctl_tx_sa_ppp[47:0]`
- `ctl_tx_ethertype_ppp[15:0]`
- `ctl_tx_opcode_ppp[15:0]`
- `ctl_tx_pause_quanta0[15:0]`
- `ctl_tx_pause_quanta1[15:0]`

- `ctl_tx_pause_quanta2[15:0]`
- `ctl_tx_pause_quanta3[15:0]`
- `ctl_tx_pause_quanta4[15:0]`
- `ctl_tx_pause_quanta5[15:0]`
- `ctl_tx_pause_quanta6[15:0]`
- `ctl_tx_pause_quanta7[15:0]`

The 40G/50G High Speed Ethernet Subsystem automatically calculates and adds the FCS to the packet. For priority pause packets the 40G/50G High Speed Ethernet Subsystem also automatically generates the enable vector based on the priorities that are requested.

To request a pause packet, you must set the corresponding bit of the `ctl_tx_pause_req[8:0]` and `ctl_tx_pause_enable[8:0]` bus to a 1 and keep it at 1 for the duration of the pause request (that is, if these inputs are set to 0, all pending pause packets are canceled). The 40G/50G High Speed Ethernet Subsystem transmits the pause packet immediately after the current packet in flight is completed.

To retransmit pause packets, the 40G/50G High Speed Ethernet Subsystem maintains a total of nine independent timers; one for each priority and one for global pause. These timers are loaded with the value of the corresponding input buses. After a pause packet is transmitted the corresponding timer is loaded with the corresponding value of `ctl_tx_pause_refresh_timer[8:0]` input bus. When a timer times out, another packet for that priority (or global) is transmitted as soon as the current packet in flight is completed. Additionally, you can manually force the timers to 0, and therefore force a retransmission, by setting the `ctl_tx_resend_pause` input to 1 for one clock cycle.

To reduce the number of pause packets for priority mode operation, a timer is considered "timed out" if any of the other timers time out. Additionally, while waiting for the current packet in flight to be completed, any new timer that times out or any new requests from you are merged into a single pause frame. For example, if two timers are counting down and you send a request for a third priority, the two timers are forced to be timed out and a pause packet for all three priorities is sent as soon as the current in-flight packet (if any) is transmitted.

Similarly, if one of the two timers times out without an additional request from you, both timers are forced to be timed out and a pause packet for both priorities is sent as soon as the current in-flight packet (if any) is transmitted.

You can stop pause packet generation by setting the appropriate bits of `ctl_tx_pause_req[8:0]` or `ctl_tx_pause_enable[8:0]` to 0.

RX Pause Termination

The 40G/50G High Speed Ethernet Subsystem terminates global and priority pause frames and provides a simple hand-shaking interface to allow user logic to respond to pause packets.

Determining Pause Packets

There are three steps in determining pause packets:

1. Checks are performed to see if a packet is a global or a priority control packet.

Note: Packets that pass step 1 are forwarded to you only if `ctl_rx_forward_control` is set to 1.

2. If step 1 passes, the packet is checked to determine if it is a global pause packet.
3. If step 2 fails, the packet is checked to determine if it is a priority pause packet.

For step 1, the following pseudo code shows the checking function:

```
assign da_match_gcp = (!ctl_rx_check_mcast_gcp && !ctl_rx_check_ucast_gcp) || ((DA
== ctl_rx_pause_da_ucast) && ctl_rx_check_ucast_gcp) || ((DA == 48'h0180c2000001) &&
ctl_rx_check_mcast_gcp);

assign sa_match_gcp = !ctl_rx_check_sa_gcp || (SA == ctl_rx_pause_sa);

assign etype_match_gcp = !ctl_rx_check_etype_gcp || (ETYPE == ctl_rx_etype_gcp);
assign opcode_match_gcp = !ctl_rx_check_opcode_gcp || ((OPCODE >=
ctl_rx_opcode_min_gcp) && (OPCODE <= ctl_rx_opcode_max_gcp));

assign global_control_packet = da_match_gcp && sa_match_gcp && etype_match_gcp &&
opcode_match_gcp && ctl_rx_enable_gcp;

assign da_match_pcp = (!ctl_rx_check_mcast_pcp && !ctl_rx_check_ucast_pcp) || ((DA
== ctl_rx_pause_da_ucast) && ctl_rx_check_ucast_pcp) || ((DA ==
ctl_rx_pause_da_mcast) && ctl_rx_check_mcast_pcp);

assign sa_match_pcp = !ctl_rx_check_sa_pcp || (SA == ctl_rx_pause_sa);

assign etype_match_pcp = !ctl_rx_check_etype_pcp || (ETYPE == ctl_rx_etype_pcp);

assign opcode_match_pcp = !ctl_rx_check_opcode_pcp || ((OPCODE >=
ctl_rx_opcode_min_pcp) && (OPCODE <= ctl_rx_opcode_max_pcp));

assign priority_control_packet = da_match_pcp && sa_match_pcp && etype_match_pcp &&
opcode_match_pcp && ctl_rx_enable_pcp;

assign control_packet = global_control_packet || priority_control_packet;
```

where DA is the destination address, SA is the source address, OPCODE is the opcode, and ETYPE is the ethertype/length field that are extracted from the incoming packet.

For step 2, the following pseudo code shows the checking function:

```
assign da_match_gpp = (!ctl_rx_check_mcast_gpp && !ctl_rx_check_ucast_gpp) || ((DA
== ctl_rx_pause_da_ucast) && ctl_rx_check_ucast_gpp) || ((DA == 48'h0180c2000001) &&
ctl_rx_check_mcast_gpp);

assign sa_match_gpp = !ctl_rx_check_sa_gpp || (SA == ctl_rx_pause_sa);
assign etype_match_gpp = !ctl_rx_check_etype_gpp || (ETYPE == ctl_rx_etype_gpp);

assign opcode_match_gpp = !ctl_rx_check_opcode_gpp || (OPCODE == ctl_rx_opcode_gpp);

assign global_pause_packet = da_match_gpp && sa_match_gpp && etype_match_gpp &&
opcode_match_gpp && ctl_rx_enable_gpp;
```

where DA is the destination address, SA is the source address, OPCODE is the opcode, and ETYPE is the ethertype/length field that are extracted from the incoming packet.

For step 3, the following pseudo code shows the checking function:

```
assign da_match_ppp = (!ctl_rx_check_mcast_ppp && !ctl_rx_check_ucast_ppp) && ((DA
== ctl_rx_pause_da_ucast) && ctl_rx_check_ucast_ppp) || ((DA ==
ctl_rx_pause_da_mcast) && ctl_rx_check_mcast_ppp);

assign sa_match_ppp = !ctl_rx_check_sa_ppp || (SA == ctl_rx_pause_sa);
assign etype_match_ppp = !ctl_rx_check_etype_ppp || (ETYPE == ctl_rx_etype_ppp);

assign opcode_match_ppp = !ctl_rx_check_opcode_ppp || (OPCODE == ctl_rx_opcode_ppp);
assign priority_pause_packet = da_match_ppp && sa_match_ppp && etype_match_ppp &&
opcode_match_ppp && ctl_rx_enable_ppp;
```

where DA is the destination address, SA is the source address, OPCODE is the opcode, and ETYPE is the ethertype/length field that are extracted from the incoming packet.

User Interface

A simple handshaking protocol is used to alert you of the reception of pause packets using the `ctl_rx_pause_enable[8:0]`, `stat_rx_pause_req[8:0]` and `ctl_rx_pause_ack[8:0]` buses. For both buses, Bit [8] corresponds to global pause packets and bits [7:0] correspond to priority pause packets.

The following steps occur when a pause packet is received:

1. If the corresponding bit of `ctl_rx_pause_enable[8:0]` is 0, the quanta is ignored and the 40G/50G High Speed Ethernet Subsystem stays in step 1. Otherwise, the corresponding bit of the `stat_rx_pause_req[8:0]` bus is set to 1, and the received quanta is loaded into a timer.

Note: If one of the bits of `ctl_rx_pause_enable[8:0]` is set to 0 (that is, disabled) when the pause processing is in step 2 or later, the 40G/50G High Speed Ethernet Subsystem completes the steps as normal until it comes back to step 1.

2. If `ctl_rx_check_ack` input is 1, the 40G/50G High Speed Ethernet Subsystem waits for you to set the appropriate bit of the `ctl_rx_pause_ack[8:0]` bus to 1.

3. After you set the proper bit of `ctl_rx_pause_ack[8:0]` to 1, or if `ctl_rx_check_ack` is 0, the 40G/50G High Speed Ethernet Subsystem starts counting down the timer.
4. When the timer times out, the 40G/50G High Speed Ethernet Subsystem sets the appropriate bit of `stat_rx_pause_req[8:0]` back to 0.
5. If `ctl_rx_check_ack` input is 1, the operation is complete when you set the appropriate bit of `ctl_rx_pause_ack[8:0]` back to 0.

If you do not set the appropriate bit of `ctl_rx_pause_ack[8:0]` back to 0, the 40G/50G High Speed Ethernet Subsystem deems the operation complete after 32 clock cycles.

The preceding steps are demonstrated in [Figure C-1](#) with each step shown on the wave form.

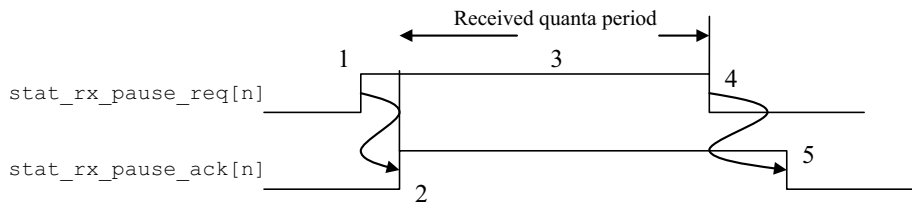


Figure C-1: RX Pause Interface Example

If at any time during step 2 to step 5 a new pause packet is received, the timer is loaded with the newly acquired quanta value and the process continues.

Additional Resources and Legal Notices

Xilinx Resources

For support resources such as Answers, Documentation, Downloads, and Forums, see [Xilinx Support](#).

References

1. *IEEE Standard for Ethernet* ([IEEE Std 802.3-2012](#))
2. Schedule 3; 25G Ethernet Consortium; issue 1.6
3. *UltraScale Architecture GTH Transceivers User Guide* ([UG576](#))
4. *UltraScale Architecture GTY Transceivers User Guide* ([UG578](#))
5. *Vivado Design Suite User Guide: Designing IP Subsystems using IP Integrator* ([UG994](#))
6. *Vivado Design Suite User Guide: Designing with IP* ([UG896](#))
7. *Vivado Design Suite User Guide: Getting Started* ([UG910](#))
8. *Vivado Design Suite User Guide: Logic Simulation* ([UG900](#))
9. *ISE® to Vivado Design Suite Migration Guide* ([UG911](#))
10. *Vivado Design Suite User Guide: Programming and Debugging* ([UG908](#))

Revision History

The following table shows the revision history for this document.

| Date | Version | Revision |
|------------|---------|--|
| 06/08/2016 | 1.1 | <ul style="list-style-type: none"> Updated Figures 2-9, 2-11, 2-12, 3-1, 3-2, 3-8, 4-1 through Figure 4-4. Added ports and modified port names. See Appendix A, Migrating and Upgrading. Changed "HSEC" to "40G/50G High Speed Ethernet Subsystem" Add the following features: supports custom preambles and Programmable Inter Packet Gap (IPG) Updated the introductory paragraph to Chapter 2, Product Specification. Removed Hierarchy section in Chapter 2. Added IMPORTANT note to Autoneg Complete section. Updated features in the PC Variant section. Changed LGMII to XL/LGMII throughout. Added MAC+PCS variant note for many of the Hex addresses in the tables in Chapter 2. Updated description in TX Debug (Buffer Errors) section |
| 04/06/2016 | 1.0 | Initial release |

Please Read: Important Legal Notices

The information disclosed to you hereunder (the "Materials") is provided solely for the selection and use of Xilinx products. To the maximum extent permitted by applicable law: (1) Materials are made available "AS IS" and with all faults, Xilinx hereby DISCLAIMS ALL WARRANTIES AND CONDITIONS, EXPRESS, IMPLIED, OR STATUTORY, INCLUDING BUT NOT LIMITED TO WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, OR FITNESS FOR ANY PARTICULAR PURPOSE; and (2) Xilinx shall not be liable (whether in contract or tort, including negligence, or under any other theory of liability) for any loss or damage of any kind or nature related to, arising under, or in connection with, the Materials (including your use of the Materials), including for any direct, indirect, special, incidental, or consequential loss or damage (including loss of data, profits, goodwill, or any type of loss or damage suffered as a result of any action brought by a third party) even if such damage or loss was reasonably foreseeable or Xilinx had been advised of the possibility of the same. Xilinx assumes no obligation to correct any errors contained in the Materials or to notify you of updates to the Materials or to product specifications. You may not reproduce, modify, distribute, or publicly display the Materials without prior written consent. Certain products are subject to the terms and conditions of Xilinx's limited warranty, please refer to Xilinx's Terms of Sale which can be viewed at <http://www.xilinx.com/legal.htm#tos>; IP cores may be subject to warranty and support terms contained in a license issued to you by Xilinx. Xilinx products are not designed or intended to be fail-safe or for use in any application requiring fail-safe performance; you assume sole risk and liability for use of Xilinx products in such critical applications, please refer to Xilinx's Terms of Sale which can be viewed at <http://www.xilinx.com/legal.htm#tos>.

© Copyright 2014–2016 Xilinx, Inc. Xilinx, the Xilinx logo, Artix, ISE, Kintex, Spartan, Virtex, Vivado, Zynq, and other designated brands included herein are trademarks of Xilinx in the United States and other countries. All other trademarks are the property of their respective owners.