10G/25G High Speed Ethernet Subsystem v2.0

Product Guide

Vivado Design Suite

PG210 November 30, 2016
Table of Contents

IP Facts

Chapter 1: Overview
- Feature Summary .......................................................... 5
- Applications .................................................................... 6
- Licensing and Ordering Information ................................. 7

Chapter 2: Product Specification
- Standards ....................................................................... 9
- Performance and Resource Utilization ......................... 9
- Latency ......................................................................... 10
- Port Descriptions .......................................................... 10
- Port Descriptions – PCS Variant ................................... 41
- Register Space ............................................................... 45

Chapter 3: Designing with the Subsystem
- Clocking ......................................................................... 98
- Resets ........................................................................... 104
- LogiCORE Example Design Clocking and Resets .......... 106
- Support for IEEE Standard 1588v2 ............................... 110
- RS-FEC Support ............................................................. 120
- Status/Control Interface ............................................... 123
- Pause Processing ........................................................... 125
- Auto-Negotiation ........................................................... 129
- Link Training ................................................................. 132

Chapter 4: Design Flow Steps
- Customizing and Generating the Core ......................... 136
- Constraining the Core ................................................... 143
- Simulation ..................................................................... 144
- Synthesis and Implementation ....................................... 144
<table>
<thead>
<tr>
<th>Chapter 5: Example Design</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Overview</td>
<td>145</td>
</tr>
<tr>
<td>Example Design Hierarchy (GT in Example Design)</td>
<td>149</td>
</tr>
<tr>
<td>User Interface</td>
<td>152</td>
</tr>
<tr>
<td>Core xci Top Level Port List</td>
<td>153</td>
</tr>
<tr>
<td>Duplex Mode of Operation</td>
<td>213</td>
</tr>
<tr>
<td>Runtime Switchable</td>
<td>213</td>
</tr>
<tr>
<td>Shared Logic Implementation</td>
<td>216</td>
</tr>
<tr>
<td>AXI4-Lite Interface Implementation</td>
<td>219</td>
</tr>
<tr>
<td>IEEE Clause 108 (RS-FEC) Integration</td>
<td>224</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Chapter 6: Batch Mode Test Bench</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Appendix A: Migrating and Upgrading</td>
<td></td>
</tr>
<tr>
<td>Changes from v2.0 (10/05/2016) to v2.0 (11/30/2016 version)</td>
<td>226</td>
</tr>
<tr>
<td>Migrating from the Legacy XGEMAC</td>
<td>230</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Appendix B: Debugging</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Finding Help on Xilinx.com</td>
<td>245</td>
</tr>
<tr>
<td>Debug Tools</td>
<td>247</td>
</tr>
<tr>
<td>Simulation Debug</td>
<td>247</td>
</tr>
<tr>
<td>Hardware Debug</td>
<td>250</td>
</tr>
<tr>
<td>Protocol Interface Debug</td>
<td>254</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Appendix C: Additional Resources and Legal Notices</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Xilinx Resources</td>
<td>255</td>
</tr>
<tr>
<td>References</td>
<td>255</td>
</tr>
<tr>
<td>Revision History</td>
<td>256</td>
</tr>
<tr>
<td>Please Read: Important Legal Notices</td>
<td>259</td>
</tr>
</tbody>
</table>
Introduction

The Xilinx® 10G/25G High Speed Ethernet subsystem implements the 25G Ethernet Media Access Controller (MAC) with a Physical Coding Sublayer (PCS) as specified by the 25G Ethernet Consortium. MAC and physical coding sublayer/physical medium attachment (PCS/PMA) or PCS/PMA alone are available. Legacy operation at 10 Gb/s is supported.

Features

- Designed to the Ethernet requirements for 10/25 Gb/s operation specified by IEEE 802.3 Clause 49, IEEE 802.3by, and the 25G Ethernet Consortium
- Includes complete Ethernet MAC and PCS/PMA functions or standalone PCS/PMA
- Simple packet-oriented user interface
- 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 Suite tools compile scripts
- BASE-R PCS sublayer operating at 10.3125 Gb/s or 25.78125 Gb/s
  - Optional clause 74 BASE-KR FEC sublayer
  - Optional Auto-Negotiation
  - Optional clause 108 25G Reed-Solomon Forward Error Correction (RS-FEC) sublayer
- Custom Preamble mode
- Optional IEEE 1588 1-step and 2-step timestamping
- Runtime switchable between 10G and 25G

Notes:
1. For a complete list of supported devices, see the Vivado IP catalog.
2. For the supported versions of the tools, see the Xilinx Design Tools: Release Notes Guide.

Note: To access the 25G specification, go to the 25G Ethernet Consortium website.

<table>
<thead>
<tr>
<th>Facts Table</th>
<th>Core Specifics</th>
</tr>
</thead>
<tbody>
<tr>
<td>Supported Device Family(1)</td>
<td>Zynq UltraScale+™ MPSoC, Virtex® UltraScale+, Kintex® UltraScale+ Vertex UltraScale™, Kintex UltraScale</td>
</tr>
<tr>
<td>Supported User Interfaces</td>
<td>AXI, XGMII, or XXVGMII</td>
</tr>
<tr>
<td>Resources</td>
<td>Performance and Resource Utilization web page</td>
</tr>
</tbody>
</table>

Provided with Core

<table>
<thead>
<tr>
<th>Design Files</th>
<th>Encrypted register transfer level (RTL)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Example Design</td>
<td>Verilog</td>
</tr>
<tr>
<td>Test Bench</td>
<td>Verilog</td>
</tr>
<tr>
<td>Constraints File</td>
<td>Xilinx Design Constraints (XDC)</td>
</tr>
<tr>
<td>Simulation Model</td>
<td>Verilog</td>
</tr>
<tr>
<td>Supported S/W Driver</td>
<td>N/A</td>
</tr>
</tbody>
</table>

Tested Design Flows(2)

<table>
<thead>
<tr>
<th>Design Entry</th>
<th>Vivado® Design Suite</th>
</tr>
</thead>
<tbody>
<tr>
<td>Simulation</td>
<td>For supported simulators, see the Xilinx Design Tools: Release Notes Guide.</td>
</tr>
<tr>
<td>Synthesis</td>
<td>Synopsis or Vivado Synthesis</td>
</tr>
</tbody>
</table>

Support

Provided by Xilinx at the Xilinx Support web page
Overview

This document details the features of the 10G/25G Ethernet Subsystem as defined by the 25G Ethernet Consortium [Ref 1]. PCS functionality is defined by IEEE Standard 802.3, 2015, Clause 49, Physical Coding Sublayer (PCS) for 64B/66B, type 10GBASE-R [Ref 2]. For 25G operation, clock frequencies are increased to provide a serial interface operating at 25.78125 Gb/s to leverage the latest high-speed serial transceivers. The low latency design is optimized for UltraScale™ architecture devices.

Feature Summary

25G Supported Features

- Complete Ethernet MAC and PCS functions
- Designed to Schedule 3 of the 25G Consortium
- Statistics and diagnostics
- 66-bit serializer/deserializer (SerDes) interface using Xilinx GTY transceiver operating with Asynchronous Gearbox enabled
- Pause Processing including IEEE std. 802.3 Annex 31D (Priority based Flow Control)
- Low latency
- Custom preamble and adjustable Inter Frame Gap
- Configurable for operation at 10.3125 Gb/s (Clause 49)

10G Supported Features

- Complete MAC and PCS functions
- Base-KR mode based on IEEE 802.3 Clause 49
- Statistics and diagnostics
- 66-bit SerDes interface
- Custom preamble and adjustable Inter Frame Gap
Optional Features

- Clause 73 Auto-Negotiation
- Clause 72.6.10 Link Training
- Clause 74 FEC - shortened cyclic code (2112, 2080)
- Clause 108 RS-FEC for 25G configuration (Reed-Solomon FEC)
- PCS only version with XGMII/XXVGMII interface (See the Port Descriptions – PCS Variant.)
- AXI4-Stream interface
- AXI4-Lite control and status interface

Applications

IEEE Std 802.3 enables several different Ethernet speeds for Local Area Network (LAN) applications, and 25 Gb/s is the latest addition to the standard. The capability to interconnect devices at 25 Gb/s Ethernet rates becomes especially relevant for next-generation data center networks where:

(i) To keep up with increasing CPU and storage bandwidth, rack or blade servers need to 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;

(ii) 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;

(iii) 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, twisted pair, or fiber.
Licensing and Ordering Information

License Checkers

If the IP requires a license key, the key must be verified. The Vivado® design tools have several license checkpoints for gating licensed IP through the flow. If the license check succeeds, the IP can continue generation. Otherwise, generation halts with error. License checkpoints are enforced by the following tools:

- Vivado Synthesis
- Vivado Implementation
- write_bitstream (Tcl Console command)

**IMPORTANT:** IP license level is ignored at checkpoints. The test confirms a valid license exists. It does not check IP license level.

License Type

**10G/25G Ethernet PCS/PMA (10G/25G BASE-R)**

This Xilinx IP module is provided at no additional cost with the Xilinx Vivado® Design Suite under the terms of the Xilinx End User License. Information about this and other Xilinx IP modules is available at the Xilinx Intellectual Property page. For information about pricing and availability of other Xilinx IP modules and tools, contact your local Xilinx sales representative.

For more information, visit the 10G/25G Ethernet Subsystem page.


These Xilinx IP module 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.

For more information, visit the 10G/25G Ethernet Subsystem page.
Figure 2-1 shows the block diagram of the 10G/25G Ethernet subsystem, not including the GTY transceiver.

Figure 2-1: Core Block Diagram
A PCS-only variant of the core is also available. The block diagram is shown in Figure 2-2.

Standards
The 10G/25G Ethernet core is designed to the standard specified in the 25G and 50G Ethernet Consortium [Ref 1] and the IEEE Std 802.3 [Ref 2] including IEEE 802.3by [Ref 3]

Performance and Resource Utilization
For full details about performance and resource utilization, visit the Performance and Resource Utilization web page.
Chapter 2: Product Specification

Latency

Table 2-1 provides the measured latency information for the 10G/25G IP core.

Table 2-1: Latency

<table>
<thead>
<tr>
<th>Core</th>
<th>Latency (ns)</th>
<th>User bus width (bits)</th>
<th>Core Clock Frequency (MHz)</th>
</tr>
</thead>
<tbody>
<tr>
<td>10G MAC + PCS</td>
<td>115.2</td>
<td>64</td>
<td>156.25</td>
</tr>
<tr>
<td>25G MAC + PCS</td>
<td>46.08</td>
<td>64</td>
<td>390.625</td>
</tr>
</tbody>
</table>

Port Descriptions

The following tables list the ports for the 10G/25G Ethernet subsystem with integrated MAC and PCS. These signals are usually found at the wrapper.v hierarchy.

When the AXI register interface is included, some of these ports are accessed by means of the registers instead of the broadside bus.

Transceiver Interface

Table 2-2 shows the transceiver I/O ports for the 10G/25G Ethernet subsystem. Refer to Clocking in Chapter 3 for details regarding each clock domain.

Table 2-2: Transceiver I/O

<table>
<thead>
<tr>
<th>Name</th>
<th>Direction</th>
<th>Description</th>
<th>Clock Domain</th>
</tr>
</thead>
<tbody>
<tr>
<td>gt_tx_reset</td>
<td>Input</td>
<td>Reset for the gigabit transceiver (GT) TX.</td>
<td>Asynch</td>
</tr>
<tr>
<td>gt_rx_reset</td>
<td>Input</td>
<td>GT RX reset.</td>
<td>Asynch</td>
</tr>
<tr>
<td>ctl_gt_reset_all</td>
<td>Input</td>
<td>Active-High asynchronous reset for the transceiver startup Finite State Machine (FSM). Note that this signal also initiates the reset sequence for the entire 10G/25G Ethernet subsystem.</td>
<td>Asynch</td>
</tr>
<tr>
<td>refclk_n0</td>
<td>Input</td>
<td>Differential reference clock input for the SerDes, negative phase.</td>
<td>Refer to Clocking.</td>
</tr>
<tr>
<td>refclk_p0</td>
<td>Input</td>
<td>Differential reference clock input for the SerDes, positive phase.</td>
<td>Refer to Clocking.</td>
</tr>
<tr>
<td>rx_serdes_data_n0</td>
<td>Input</td>
<td>Serial data from the line; negative phase of the differential signal</td>
<td>Refer to Clocking.</td>
</tr>
<tr>
<td>rx_serdes_data_p0</td>
<td>Input</td>
<td>Serial data from the line; positive phase of the differential signal</td>
<td>Refer to Clocking.</td>
</tr>
<tr>
<td>tx_serdes_data_n0</td>
<td>Output</td>
<td>Serial data to the line; negative phase of the differential signal.</td>
<td>Refer to Clocking.</td>
</tr>
</tbody>
</table>
Table 2-2: Transceiver I/O (Cont’d)

<table>
<thead>
<tr>
<th>Name</th>
<th>Direction</th>
<th>Description</th>
<th>Clock Domain</th>
</tr>
</thead>
<tbody>
<tr>
<td>tx_serdes_data_p0</td>
<td>Output</td>
<td>Serial data to the line; positive phase of the differential signal.</td>
<td>Refer to Clocking.</td>
</tr>
<tr>
<td>tx_serdes_clkout</td>
<td>Output</td>
<td>When present, same as tx_clk_out.</td>
<td>Refer to Clocking.</td>
</tr>
</tbody>
</table>

AXI4-Stream Clocks and Resets

Table 2-3: AXI4-Stream Interface–Clock/Reset Signals

<table>
<thead>
<tr>
<th>Name</th>
<th>Direction</th>
<th>Description</th>
<th>Clock Domain</th>
</tr>
</thead>
<tbody>
<tr>
<td>rx_clk_out</td>
<td>Output</td>
<td>Receive Local bus clock. All signals between the 10G/25G High Speed Ethernet Subsystem and the user-side logic are synchronized to the positive edge of this signal. The AXI4-Stream clock is 390.625 MHz. When the RX FIFO is included, the RX AXI4-Stream clock is an input and should be equal to or greater than tx_clk_out.</td>
<td>Refer to Clocking.</td>
</tr>
<tr>
<td>tx_clk_out</td>
<td>Output</td>
<td>Transmit Local bus clock. All signals between the 10G/25G High Speed Ethernet Subsystem and the user-side logic are synchronized to the positive edge of this signal. The AXI4-Stream clock is 390.625 MHz.</td>
<td>Refer to Clocking.</td>
</tr>
<tr>
<td>rx_reset</td>
<td>Input</td>
<td>Reset for the RX circuits. This signal is active-High (1 = reset) and must be held High until clk is stable. The core handles synchronizing the rx_reset input to the appropriate clock domains within the core.</td>
<td>Asynch</td>
</tr>
<tr>
<td>tx_reset</td>
<td>Input</td>
<td>Reset for the TX circuits. This signal is active-High (1 = reset) and must be held High until clk is stable. The core handles synchronizing the tx_reset input to the appropriate clock domains within the core.</td>
<td>Asynch</td>
</tr>
</tbody>
</table>

Transmit AXI4-Stream Interface

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

Table 2-4: AXI4-Stream Transmit Interface Signals

<table>
<thead>
<tr>
<th>Signal</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tx_axis_tdata[63:0]</td>
<td>In</td>
<td>AXI4-Stream data (64-bit interface)</td>
</tr>
<tr>
<td>tx_axis_tkeep[7:0]</td>
<td>In</td>
<td>AXI4-Stream Data Control (64-bit interface)</td>
</tr>
<tr>
<td>tx_axis_tvalid</td>
<td>In</td>
<td>AXI4-Stream Data Valid input</td>
</tr>
<tr>
<td>tx_axis_tuser</td>
<td>In</td>
<td>AXI4-Stream User Sideband interface. Equivalent to the tx_errin signal.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1 indicates a bad packet has been received.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0 indicates a good packet has been received.</td>
</tr>
</tbody>
</table>
Data Lane Mapping

For transmit data \( tx\_axis\_tdata[63:0] \), the port is divided into lane 0 to lane 7 (see Table 2-5).

\[ \text{Table 2-5: } \text{tx\_axis\_tdata Lanes} \]

<table>
<thead>
<tr>
<th>Lane/ tx_axis_tkeep</th>
<th>tx_axis_tdata[63:0] Bits</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>7:0</td>
</tr>
<tr>
<td>1</td>
<td>15:8</td>
</tr>
<tr>
<td>2</td>
<td>23:16</td>
</tr>
<tr>
<td>3</td>
<td>31:24</td>
</tr>
<tr>
<td>4</td>
<td>39:32</td>
</tr>
<tr>
<td>5</td>
<td>47:40</td>
</tr>
<tr>
<td>6</td>
<td>55:48</td>
</tr>
<tr>
<td>7</td>
<td>63:56</td>
</tr>
</tbody>
</table>

\[ \text{Table 2-4: } \text{AXI4-Stream Transmit Interface Signals (Cont'd)} \]

<table>
<thead>
<tr>
<th>Signal</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tx_axis_tlast</td>
<td>In</td>
<td>AXI4-Stream signal indicating End of Ethernet Packet.</td>
</tr>
<tr>
<td>tx_axis_tready</td>
<td>Out</td>
<td>AXI4-Stream acknowledge signal to indicate to start the Data transfer.</td>
</tr>
</tbody>
</table>
**Normal Transmission**

The timing of a normal frame transfer is shown in Figure 2-3. When the client wants to transmit a frame, it asserts the `tx_axis_tvalid` and places the data and control in `tx_axis_tdata` and `tx_axis_tkeep` in the same clock cycle. When this data is accepted by the core, indicated by `tx_axis_tready` being asserted, the client must provide the next cycle of data. If `tx_axis_tready` is not asserted by the core, the client must hold the current valid data value until it is. The end of packet is indicated to the core by `tx_axis_tlast` asserted for 1 cycle. The bits of `tx_axis_tkeep` are set appropriately to indicate the number of valid bytes in the final data transfer. `tx_axis_tuser` is also asserted to indicate a bad packet.

After `tx_axis_tlast` is deasserted, any data and control is deemed invalid until `tx_axis_tvalid` is next asserted.

![Figure 2-3: Normal Frame Transfer](image)

**Aborting a Transmission**

The aborted transfer of a packet on the client interface is called an underrun. This can happen if a FIFO in the AXI Transmit client interface empties before a frame is completed.

This is indicated to the core in one of two ways.

- An explicit error in which a frame transfer is aborted by deasserting `tx_axis_tuser` High while `tx_axis_tlast` is High (See Figure 2-5).
- An implicit underrun, in which a frame transfer is aborted by deasserting `tx_axis_tvalid` without asserting `tx_axis_tlast`. 
When either of the two scenarios occurs during a frame transmission, the core inserts error codes into the data stream to flag the current frame as an errored frame. It remains the responsibility of the client to re-queue the aborted frame for transmission, if necessary.

**Receive AXI4-Stream Interface**

Table 2-6 shows the AXI4-Stream receive interface signals.

<table>
<thead>
<tr>
<th>Signal</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>rx_axis_tdata[63:0]</td>
<td>Out</td>
<td>AXI4-Stream Data to upper layer</td>
</tr>
<tr>
<td>rx_axis_tkeep[7:0]</td>
<td>Out</td>
<td>AXI4-Stream Data Control to upper layer</td>
</tr>
<tr>
<td>rx_axis_tvalid</td>
<td>Out</td>
<td>AXI4-Stream Data Valid</td>
</tr>
<tr>
<td>rx_axis_tuser</td>
<td>Out</td>
<td>AXI4-Stream User Sideband interface. 1 indicates a bad packet has been received. 0 indicates a good packet has been received.</td>
</tr>
<tr>
<td>rx_axis_tlast</td>
<td>Out</td>
<td>AXI4-Stream signal indicating an end of packet.</td>
</tr>
</tbody>
</table>

**Data Lane Mapping**

For receive data rx_axis_tdata, the port is divided into lane 0 to lane 7 (see Table 2-7).

<table>
<thead>
<tr>
<th>Lane/ rx_axis_tkeep</th>
<th>rx_axis_tdata Bits</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>7:0</td>
</tr>
<tr>
<td>1</td>
<td>15:8</td>
</tr>
<tr>
<td>2</td>
<td>23:16</td>
</tr>
<tr>
<td>3</td>
<td>31:24</td>
</tr>
<tr>
<td>4</td>
<td>39:32</td>
</tr>
<tr>
<td>5</td>
<td>47:40</td>
</tr>
<tr>
<td>6</td>
<td>55:48</td>
</tr>
<tr>
<td>7</td>
<td>63:56</td>
</tr>
</tbody>
</table>

**Normal Frame Reception**

The timing of a normal inbound frame transfer is represented Figure 2-4. The client must be prepared to accept data at any time; there is no 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, \texttt{rx_axis_tvalid} is asserted to indicate that valid frame data is being transferred to the client on \texttt{rx_axis_tdata}. All bytes are always valid throughout the frame, as indicated by all \texttt{rx_axis_tkeep} bits being set to 1, except during the final transfer of the frame when \texttt{rx_axis_tlast} is asserted. During this final transfer of data for a frame, \texttt{rx_axis_tkeep} bits indicate the final valid bytes of the frame using the mapping from above. The valid bytes of the final transfer always lead out from \texttt{rx_axis_tdata[7:0]} (\texttt{rx_axis_tkeep[0]}) because Ethernet frame data is continuous and is received least significant byte first.

The \texttt{rx_axis_tlast} is asserted and \texttt{rx_axis_tuser} is deasserted, along with the final bytes of the transfer, only after all frame checks are completed. This is after the frame check sequence (FCS) field has been received. The core asserts the \texttt{rx_axis_tuser} signal to indicate that the frame was successfully received and that the frame should be analyzed by the client. This is also the end of packet signaled by \texttt{rx_axis_tlast} asserted for one cycle.

![Figure 2-4: Normal Frame Reception](image)

**Frame Reception with Errors**

The case of an unsuccessful frame reception (for example, a runt frame or a frame with an incorrect FCS) is shown in Figure 2-5. In this case, the bad frame is received and the signal \texttt{rx_axis_tuser} is asserted to the client at the end of the frame. It is then the responsibility of the client to drop the data already transferred for this frame.

The following conditions cause the assertion of \texttt{rx_axis_tlast} along with \texttt{rx_axis_tuser} = 1 signifying a bad_frame:

- FCS errors occur.
- Packets are shorter than 64 bytes (undersize or fragment frames).
- Frames of length greater than the maximum transmission unit (MTU) Size programmed are received, MTU Size Enable Frames are enabled.
• Any control frame that is received is not exactly the minimum frame length unless Control Frame Length Check Disable is set.

• The XGMII data stream contains error codes.

Figure 2-5: Frame Reception with Errors
# AXI4-Stream Control and Status Ports

## Table 2-8: AXI4-Stream Interface–TX Path Control/Status Signals

<table>
<thead>
<tr>
<th>Name</th>
<th>Direction</th>
<th>Description</th>
<th>Clock Domain</th>
</tr>
</thead>
<tbody>
<tr>
<td>ctl_tx_custom_preamble_enable</td>
<td>Input</td>
<td>When asserted, this signal enables the use of tx_preamblein as a custom preamble instead of inserting a standard preamble.</td>
<td>tx_clk_out</td>
</tr>
<tr>
<td>tx_preamblein [55:0]</td>
<td>Input</td>
<td>This is the custom preamble which is a separate input port rather than being in-line with the data. It should be valid during the start of packet.</td>
<td>tx_clk_out</td>
</tr>
<tr>
<td>ctl_tx_ipg_value[3:0]</td>
<td>Input</td>
<td>This signal can be optionally present. The ctl_tx_ipg_value defines the target average minimum Inter Packet Gap (IPG, in bytes) inserted between AXI4-Stream packets. Valid values are 8 to 12. The ctl_tx_ipg_value can be programmed to a value in the 0 to 7 range, but in that case, it is interpreted as 8 (the minimum valid value).</td>
<td>tx_clk_out</td>
</tr>
<tr>
<td>ctl_tx_enable</td>
<td>Input</td>
<td>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. 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 synchronized 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 then the core stops transmitting any more packets.</td>
<td>tx_clk_out</td>
</tr>
<tr>
<td>ctl_tx_send_rfi</td>
<td>Input</td>
<td>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 synchronized and is ready to accept data from the link partner.</td>
<td>tx_clk_out</td>
</tr>
<tr>
<td>ctl_tx_send_lfi</td>
<td>Input</td>
<td>Transmit Local Fault Indication (LFI) code word. Takes precedence over Remote Fault Indication (RFI).</td>
<td>tx_clk_out</td>
</tr>
<tr>
<td>ctl_tx_send_idle</td>
<td>Input</td>
<td>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 RFI code words.</td>
<td>tx_clk_out</td>
</tr>
<tr>
<td>ctl_tx_fcs_ins_enable</td>
<td>Input</td>
<td>Enable FCS insertion by the TX core. If this bit is set to 0, the core does not add FCS to packet. If this bit is set to 1, the core calculates and adds the FCS to the packet. This input cannot be changed dynamically between packets.</td>
<td>tx_clk_out</td>
</tr>
</tbody>
</table>
### Table 2-8: AXI4-Stream Interface—TX Path Control/Status Signals (Cont’d)

<table>
<thead>
<tr>
<th>Name</th>
<th>Direction</th>
<th>Description</th>
<th>Clock Domain</th>
</tr>
</thead>
<tbody>
<tr>
<td>ctl_tx_ignore_fcs</td>
<td>Input</td>
<td>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. <em>Note:</em> Statistics are reported as if there was no FCS error.</td>
<td>tx_clk_out</td>
</tr>
<tr>
<td>stat_tx_local_fault</td>
<td>Output</td>
<td>A value of 1 indicates the receive decoder state machine is in the TX_INIT state. This output is level sensitive.</td>
<td>tx_clk_out</td>
</tr>
</tbody>
</table>

### Table 2-9: AXI4-Stream Interface—RX Path Control/Status Signals

<table>
<thead>
<tr>
<th>Name</th>
<th>Direction</th>
<th>Description</th>
<th>Clock Domain</th>
</tr>
</thead>
<tbody>
<tr>
<td>rx_preambleout [55:0]</td>
<td>Output</td>
<td>This is the preamble, and now a separate output instead of inline with data as was done with previous releases.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>ctl_rx_enable</td>
<td>Input</td>
<td>RX Enable. For normal operation, this input must be set to 1. When this input is set the 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.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>ctl_rx_check_preamble</td>
<td>Input</td>
<td>When asserted, this input causes the MAC to check the preamble of the received frame.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>ctl_rx_check_sfd</td>
<td>Input</td>
<td>When asserted, this input causes the MAC to check the Start of Frame Delimiter of the received frame.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>ctl_rx_force_resync</td>
<td>Input</td>
<td>RX force resynchronization input. This signal is used to force the RX path to reset and re-synchronize. A value of 1 forces the reset operation. A value of 0 allows normal operation. Note that this input should normally be Low and should only be pulsed (1 cycle minimum pulse).</td>
<td>rx_clk_out</td>
</tr>
</tbody>
</table>
### Table 2-9: AXI4-Stream Interface–RX Path Control/Status Signals (Cont’d)

<table>
<thead>
<tr>
<th>Name</th>
<th>Direction</th>
<th>Description</th>
<th>Clock Domain</th>
</tr>
</thead>
<tbody>
<tr>
<td>ctl_rx_delete_fcs</td>
<td>Input</td>
<td>Enable FCS removal by the RX core. If this bit is set to 0, the core does not remove the FCS of the incoming packet. If this bit is set to 1, the core deletes the FCS to the received packet. Note that FCS is not deleted for packets that are less than or equal to 8 bytes long. This input should only be changed while the corresponding reset input is asserted.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>ctl_rx_ignore_fcs</td>
<td>Input</td>
<td>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 core does not flag an FCS error at the AXI4-Stream interface. <strong>Note:</strong> The statistics are reported as if the packet is good. The signal stat_rx_bad_fcs, however, reports the error.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>ctl_rx_max_packet_len[14:0]</td>
<td>Input</td>
<td>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. ctl_rx_max_packet_len[14] is reserved and must be set to 0.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>ctl_rx_min_packet_len[7:0]</td>
<td>Input</td>
<td>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.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>stat_rx_framing_err[2-1:0]</td>
<td>Output</td>
<td>The RX sync header bits framing error is a bus that indicates how many sync header errors were received. The value of the bus is only valid when stat_rx_framing_err_valid is a 1. The values can be updated at any time and are intended to be used as increment values for sync header error counters.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>stat_rx_framing_err_valid</td>
<td>Output</td>
<td>Valid indicator for stat_rx_framing_err. When sampled as a 1, the value on stat_rx_framing_err is valid.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>stat_rx_local_fault</td>
<td>Output</td>
<td>This output is High when stat_rx_internal_local_fault or stat_rx_received_local_fault is asserted. This output is level sensitive.</td>
<td>rx_clk_out</td>
</tr>
</tbody>
</table>
## Table 2-9: AXI4-Stream Interface–RX Path Control/Status Signals (Cont’d)

<table>
<thead>
<tr>
<th>Name</th>
<th>Direction</th>
<th>Description</th>
<th>Clock Domain</th>
</tr>
</thead>
<tbody>
<tr>
<td>stat_rx_block_lock[1-1:0]</td>
<td>Output</td>
<td>Block lock status. A value of 1 indicates that block lock is achieved as defined in Clause 49.2.14 and MDIO register 3.32.0. This output is level sensitive.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>stat_rx_remote_fault</td>
<td>Output</td>
<td>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, remote fault condition does not exist. This output is level sensitive.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>stat_rx_bad_fcs[2-1:0]</td>
<td>Output</td>
<td>Bad FCS indicator. The value on this bus indicates packets received with a bad FCS, but not a stomped FCS during a cycle. 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. Note that pulses can occur in back to back cycles.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>stat_rx_stomped_fcs[2-1:0]</td>
<td>Output</td>
<td>Stomped FCS indicator. The value on this bus indicates the packets 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. Note that pulses can occur in back to back cycles.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>stat_rx_truncated</td>
<td>Output</td>
<td>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. Note that pulses can occur in back to back cycles.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>stat_rx_internal_local_fault</td>
<td>Output</td>
<td>High when an internal local fault is generated due to any one of the following: test pattern generation or high bit error rate. Note that this signal remains High as long as the fault condition persists.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>stat_rx_received_local_fault</td>
<td>Output</td>
<td>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. Remains High as long as the fault condition persists.</td>
<td>rx_clk_out</td>
</tr>
</tbody>
</table>
Table 2-9: AXI4-Stream Interface—RX Path Control/Status Signals (Cont’d)

<table>
<thead>
<tr>
<th>Name</th>
<th>Direction</th>
<th>Description</th>
<th>Clock Domain</th>
</tr>
</thead>
<tbody>
<tr>
<td>stat rx hi_ber</td>
<td>Output</td>
<td>High Bit Error Rate (BER) indicator. When set to 1, the BER is too high as defined by IEEE Std. 802.3. Corresponds to MDIO register bit 3.32.1 as defined in Clause 49.2.14. This output is level sensitive.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>ctl rx custom preamble enable</td>
<td>Input</td>
<td>When asserted, this signal causes the side band of a packet presented on the AXI4-Stream to be the preamble as it appears on the line.</td>
<td>rx_clk_out</td>
</tr>
</tbody>
</table>

Miscellaneous Status/Control Signals

Table 2-10 shows the miscellaneous status and control I/O signals.

Table 2-10: Miscellaneous Status/Control Ports

<table>
<thead>
<tr>
<th>Name</th>
<th>Direction</th>
<th>Description</th>
<th>Clock Domain</th>
</tr>
</thead>
<tbody>
<tr>
<td>dclk</td>
<td>Input</td>
<td>Dynamic reconfiguration port (DRP) clock input. The required frequency is indicated on the readme file for the release. Refer to Clocking.</td>
<td></td>
</tr>
<tr>
<td>stat rx valid ctrl code</td>
<td>Output</td>
<td>Indicates that a PCS block with a valid control code was received.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>ctl local loopback</td>
<td>Input</td>
<td>Loopback enable. A value of 1 enables loopback as defined in Clause 49. Corresponds to management data input/output (MDIO) register bit 3.0.14 as defined in Clause 45. This input should only be changed while the corresponding reset input is asserted.</td>
<td>Asynch</td>
</tr>
<tr>
<td>stat rx got signal os</td>
<td>Output</td>
<td>Signal OS indication. If this bit is sampled as a 1, it indicates that a Signal OS word was received. Note that Signal OS should not be received in an Ethernet network.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>ctl rx process lfi</td>
<td>Input</td>
<td>When this input is set to 1, the RX core expects and processes LF control codes coming in from the transceiver. When set to 0, the RX core ignores LF control codes coming in from the transceiver.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>Name</td>
<td>Direction</td>
<td>Description</td>
<td>Clock Domain</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>-----------</td>
<td>-----------------------------------------------------------------------------</td>
<td>--------------</td>
</tr>
<tr>
<td>ctl_rx_test_pattern</td>
<td>Input</td>
<td>Test pattern checking enable for the RX core. A value of 1 enables test mode as defined in Clause 49. Corresponds to MDIO register bit 3.42.2 as defined in Clause 45. Checks for scrambled idle pattern.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>ctl_tx_test_pattern</td>
<td>Input</td>
<td>Test pattern generation enable for the TX core. A value of 1 enables test mode as defined in Clause 49. Corresponds to MDIO register bit 3.42.3 as defined in Clause 45. Generates a scrambled idle pattern.</td>
<td>tx_clk_out</td>
</tr>
<tr>
<td>stat_rx_test_pattern_mismatch[1:0]</td>
<td>Output</td>
<td>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 as defined in Clause 45. This output is pulsed for one clock cycle.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>ctl_rx_data_pattern_select</td>
<td>Input</td>
<td>Corresponds to MDIO register bit 3.42.0 as defined in Clause 45.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>ctl_rx_test_pattern_enable</td>
<td>Input</td>
<td>Test pattern enable for the RX core. A value of 1 enables test mode.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Corresponds to MDIO register bit 3.42.2 as defined in Clause 45. Takes second precedence.</td>
<td></td>
</tr>
<tr>
<td>ctl_tx_data_pattern_select</td>
<td>Input</td>
<td>Corresponds to MDIO register bit 3.42.0 as defined in Clause 45.</td>
<td>tx_clk_out</td>
</tr>
<tr>
<td>ctl_tx_test_pattern_enable</td>
<td>Input</td>
<td>Test pattern generation enable for the TX core. A value of 1 enables test mode.</td>
<td>tx_clk_out</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Corresponds to MDIO register bit 3.42.3 as defined in Clause 45. Takes second precedence.</td>
<td></td>
</tr>
<tr>
<td>ctl_tx_test_pattern_seed_a[57:0]</td>
<td>Input</td>
<td>Corresponds to MDIO registers 3.34 through to 3.37 as defined in Clause 45.</td>
<td>tx_clk_out</td>
</tr>
<tr>
<td>ctl_tx_test_pattern_seed_b[57:0]</td>
<td>Input</td>
<td>Corresponds to MDIO registers 3.38 through to 3.41 as defined in Clause 45.</td>
<td>tx_clk_out</td>
</tr>
<tr>
<td>ctl_tx_test_pattern_select</td>
<td>Input</td>
<td>Corresponds to MDIO register bit 3.42.1 as defined in Clause 45.</td>
<td>tx_clk_out</td>
</tr>
</tbody>
</table>
Statistics Interface Ports

Table 2-11 and Table 2-12 show the Statistics interface I/O ports.

Table 2-11: Statistics Interface - RX Path

<table>
<thead>
<tr>
<th>Name</th>
<th>Direction</th>
<th>Description</th>
<th>Clock Domain</th>
</tr>
</thead>
<tbody>
<tr>
<td>stat_rx_total_bytes[4-1:0]</td>
<td>Output</td>
<td>Increment for the total number of bytes received.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>stat_rx_total_packets[2-1:0]</td>
<td>Output</td>
<td>Increment for the total number of packets received.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>stat_rx_total_good_bytes[14-1:0]</td>
<td>Output</td>
<td>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.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>stat_rx_total_good_packets</td>
<td>Output</td>
<td>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.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>stat_rx_packet_bad_fcs</td>
<td>Output</td>
<td>Increment for packets between 64 and ctrl_rx_max_packet_len bytes that have Frame Check Sequence (FCS) errors.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>stat_rx_packet_64_bytes</td>
<td>Output</td>
<td>Increment for good and bad packets received that contain 64 bytes.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>stat_rx_packet_65_127_bytes</td>
<td>Output</td>
<td>Increment for good and bad packets received that contain 65 to 127 bytes.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>stat_rx_packet_128_255_bytes</td>
<td>Output</td>
<td>Increment for good and bad packets received that contain 128 to 255 bytes.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>stat_rx_packet_256_511_bytes</td>
<td>Output</td>
<td>Increment for good and bad packets received that contain 256 to 511 bytes.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>stat_rx_packet_512_1023_bytes</td>
<td>Output</td>
<td>Increment for good and bad packets received that contain 512 to 1,023 bytes.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>stat_rx_packet_1024_1518_bytes</td>
<td>Output</td>
<td>Increment for good and bad packets received that contain 1,024 to 1,518 bytes.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>stat_rx_packet_1519_1522_bytes</td>
<td>Output</td>
<td>Increment for good and bad packets received that contain 1,519 to 1,522 bytes.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>stat_rx_packet_1523_1548_bytes</td>
<td>Output</td>
<td>Increment for good and bad packets received that contain 1,523 to 1,548 bytes.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>stat_rx_packet_1549_2047_bytes</td>
<td>Output</td>
<td>Increment for good and bad packets received that contain 1,549 to 2,047 bytes.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>stat_rx_packet_2048_4095_bytes</td>
<td>Output</td>
<td>Increment for good and bad packets received that contain 2,048 to 4,095 bytes.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>stat_rx_packet_4096_8191_bytes</td>
<td>Output</td>
<td>Increment for good and bad packets received that contain 4,096 to 8,191 bytes.</td>
<td>rx_clk_out</td>
</tr>
</tbody>
</table>
### Table 2-11: Statistics Interface - RX Path (Cont’d)

<table>
<thead>
<tr>
<th>Name</th>
<th>Direction</th>
<th>Description</th>
<th>Clock Domain</th>
</tr>
</thead>
<tbody>
<tr>
<td>stat_rx_packet_8192_9215_bytes</td>
<td>Output</td>
<td>Increment for good and bad packets received that contain 8,192 to 9,215 bytes.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>stat_rx_packet_small</td>
<td>Output</td>
<td>Increment for all packets that are less than 64 bytes long. Packets that are less than 64 bytes are dropped.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>stat_rx_packet_large</td>
<td>Output</td>
<td>Increment for all packets that are more than 9,215 bytes long.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>stat_rx_unicast</td>
<td>Output</td>
<td>Increment for good unicast packets.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>stat_rx_multicast</td>
<td>Output</td>
<td>Increment for good multicast packets.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>stat_rx_broadcast</td>
<td>Output</td>
<td>Increment for good broadcast packets.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>stat_rx_oversize</td>
<td>Output</td>
<td>Increment for packets longer than ctl_rx_max_packet_len with good FCS.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>stat_rx_toolong</td>
<td>Output</td>
<td>Increment for packets longer than ctl_rx_max_packet_len with good and bad FCS.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>stat_rx_undersize</td>
<td>Output</td>
<td>Increment for packets shorter than stat_rx_min_packet_len with good FCS.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>stat_rx_fragment</td>
<td>Output</td>
<td>Indicates the increment for packets shorter than stat_rx_min_packet_len with bad FCS.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>stat_rx_vlan</td>
<td>Output</td>
<td>Increment for good 802.1Q tagged VLAN packets.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>stat_rx_inrangeerr</td>
<td>Output</td>
<td>Increment for packets with Length field error but with good FCS.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>stat_rx_jabber</td>
<td>Output</td>
<td>Increment for packets longer than ctl_rx_max_packet_len with bad FCS.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>stat_rx_pause</td>
<td>Output</td>
<td>Increment for 802.3x MAC Pause packet with good FCS.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>stat_rx_user_pause</td>
<td>Output</td>
<td>Increment for priority based pause packets with good FCS.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>stat_rx_bad_code[1-1:0]</td>
<td>Output</td>
<td>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 IEEE Std. 802.3. This output can be used to generate MDIO register as defined in Clause 45.</td>
<td>rx_clk_out</td>
</tr>
</tbody>
</table>
### Table 2-11: Statistics Interface - RX Path (Cont’d)

<table>
<thead>
<tr>
<th>Name</th>
<th>Direction</th>
<th>Description</th>
<th>Clock Domain</th>
</tr>
</thead>
<tbody>
<tr>
<td>stat_rx_bad_sfd</td>
<td>Output</td>
<td>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.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>stat_rx_bad_preamble</td>
<td>Output</td>
<td>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.</td>
<td>rx_clk_out</td>
</tr>
</tbody>
</table>

### Table 2-12: Statistics Interface - TX Path

<table>
<thead>
<tr>
<th>Name</th>
<th>Direction</th>
<th>Description</th>
<th>Clock Domain</th>
</tr>
</thead>
<tbody>
<tr>
<td>stat_tx_total_bytes[4-1:0]</td>
<td>Output</td>
<td>Increment for the total number of bytes transmitted.</td>
<td>tx_clk_out</td>
</tr>
<tr>
<td>stat_tx_total_packets</td>
<td>Output</td>
<td>Increment for the total number of packets transmitted.</td>
<td>tx_clk_out</td>
</tr>
<tr>
<td>stat_tx_total_good_bytes[14-1:0]</td>
<td>Output</td>
<td>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.</td>
<td>tx_clk_out</td>
</tr>
<tr>
<td>stat_tx_total_good_packets</td>
<td>Output</td>
<td>Increment for the total number of good packets transmitted.</td>
<td>tx_clk_out</td>
</tr>
<tr>
<td>stat_tx_bad_fcs</td>
<td>Output</td>
<td>Increment for packets greater than 64 bytes that have FCS errors.</td>
<td>tx_clk_out</td>
</tr>
<tr>
<td>stat_tx_packet_64_bytes</td>
<td>Output</td>
<td>Increment for good and bad packets transmitted that contain 64 bytes.</td>
<td>tx_clk_out</td>
</tr>
<tr>
<td>stat_tx_packet_65_127_bytes</td>
<td>Output</td>
<td>Increment for good and bad packets transmitted that contain 65 to 127 bytes.</td>
<td>tx_clk_out</td>
</tr>
<tr>
<td>stat_tx_packet_128_255_bytes</td>
<td>Output</td>
<td>Increment for good and bad packets transmitted that contain 128 to 255 bytes.</td>
<td>tx_clk_out</td>
</tr>
<tr>
<td>stat_tx_packet_256_511_bytes</td>
<td>Output</td>
<td>Increment for good and bad packets transmitted that contain 256 to 511 bytes.</td>
<td>tx_clk_out</td>
</tr>
<tr>
<td>stat_tx_packet_512_1023_bytes</td>
<td>Output</td>
<td>Increment for good and bad packets transmitted that contain 512 to 1,023 bytes.</td>
<td>tx_clk_out</td>
</tr>
<tr>
<td>stat_tx_packet_1024_1518_bytes</td>
<td>Output</td>
<td>Increment for good and bad packets transmitted that contain 1,024 to 1,518 bytes.</td>
<td>tx_clk_out</td>
</tr>
<tr>
<td>stat_tx_packet_1519_1522_bytes</td>
<td>Output</td>
<td>Increment for good and bad packets transmitted that contain 1,519 to 1,522 bytes.</td>
<td>tx_clk_out</td>
</tr>
</tbody>
</table>
### Table 2-12: Statistics Interface - TX Path (Cont’d)

<table>
<thead>
<tr>
<th>Name</th>
<th>Direction</th>
<th>Description</th>
<th>Clock Domain</th>
</tr>
</thead>
<tbody>
<tr>
<td>stat tx_packet_1523_1548_bytes</td>
<td>Output</td>
<td>Increment for good and bad packets transmitted that contain 1,523 to 1,548 bytes.</td>
<td>tx_clk_out</td>
</tr>
<tr>
<td>stat tx_packet_1549_2047_bytes</td>
<td>Output</td>
<td>Increment for good and bad packets transmitted that contain 1,549 to 2,047 bytes.</td>
<td>tx_clk_out</td>
</tr>
<tr>
<td>stat tx_packet_2048_4095_bytes</td>
<td>Output</td>
<td>Increment for good and bad packets transmitted that contain 2,048 to 4,095 bytes.</td>
<td>tx_clk_out</td>
</tr>
<tr>
<td>stat tx_packet_4096_8191_bytes</td>
<td>Output</td>
<td>Increment for good and bad packets transmitted that contain 4,096 to 8,191 bytes.</td>
<td>tx_clk_out</td>
</tr>
<tr>
<td>stat tx_packet_8192_9215_bytes</td>
<td>Output</td>
<td>Increment for good and bad packets transmitted that contain 8,192 to 9,215 bytes.</td>
<td>tx_clk_out</td>
</tr>
<tr>
<td>stat tx_packet_small</td>
<td>Output</td>
<td>Increment for all packets that are less than 64 bytes long.</td>
<td>tx_clk_out</td>
</tr>
<tr>
<td>stat tx_packet_large</td>
<td>Output</td>
<td>Increment for all packets that are more than 9,215 bytes long.</td>
<td>tx_clk_out</td>
</tr>
<tr>
<td>stat tx_unicast</td>
<td>Output</td>
<td>Increment for good unicast packets.</td>
<td>tx_clk_out</td>
</tr>
<tr>
<td>stat tx_multicast</td>
<td>Output</td>
<td>Increment for good multicast packets.</td>
<td>tx_clk_out</td>
</tr>
<tr>
<td>stat tx_broadcast</td>
<td>Output</td>
<td>Increment for good broadcast packets.</td>
<td>tx_clk_out</td>
</tr>
<tr>
<td>stat tx_vlan</td>
<td>Output</td>
<td>Increment for good 802.1Q tagged VLAN packets.</td>
<td>tx_clk_out</td>
</tr>
<tr>
<td>stat tx_pause</td>
<td>Output</td>
<td>Increment for 802.3x MAC Pause packet with good FCS.</td>
<td>tx_clk_out</td>
</tr>
<tr>
<td>stat tx_user_pause</td>
<td>Output</td>
<td>Increment for priority based pause packets with good FCS.</td>
<td>tx_clk_out</td>
</tr>
<tr>
<td>stat tx_frame_error</td>
<td>Output</td>
<td>Increment for packets with tx_errin set to indicate an End of Packet (EOP) abort.</td>
<td>tx_clk_out</td>
</tr>
</tbody>
</table>
Pause Interface

Table 2-13 through Table 2-15 show the Pause interface I/O ports.

Table 2-13: Pause Interface—Control Ports

<table>
<thead>
<tr>
<th>Name</th>
<th>Direction</th>
<th>Description</th>
<th>Clock Domain</th>
</tr>
</thead>
<tbody>
<tr>
<td>ctl_rx_pause_enable[9-1:0]</td>
<td>Input</td>
<td>RX pause enable signal. This input is used to enable the processing of the pause quanta for the corresponding priority. Note that this signal only affects the RX user interface, not the pause processing logic.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>ctl_tx_pause_enable[9-1:0]</td>
<td>Input</td>
<td>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.</td>
<td>tx_clk_out</td>
</tr>
</tbody>
</table>

Table 2-14: Pause Interface—RX Path

<table>
<thead>
<tr>
<th>Name</th>
<th>Direction</th>
<th>Description</th>
<th>Clock Domain</th>
</tr>
</thead>
<tbody>
<tr>
<td>ctl_rx_enable_gcp</td>
<td>Input</td>
<td>A value of 1 enables global control packet processing.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>ctl_rx_check_mcast_gcp</td>
<td>Input</td>
<td>A value of 1 enables global control multicast destination address processing.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>ctl_rx_check_uctcast_gcp</td>
<td>Input</td>
<td>A value of 1 enables global control unicast destination address processing.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>ctl_rx_pause_da_uctcast[47:0]</td>
<td>Input</td>
<td>Unicast destination address for pause processing.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>ctl_rx_check_sa_gcp</td>
<td>Input</td>
<td>A value of 1 enables global control source address processing.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>ctl_rx_pause_s[47:0]</td>
<td>Input</td>
<td>Source address for pause processing.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>ctl_rx_check_etype_gcp</td>
<td>Input</td>
<td>A value of 1 enables global control ethertype processing.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>ctl_rx_check_opcode_gcp</td>
<td>Input</td>
<td>A value of 1 enables global control opcode processing.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>ctl_rx_opcode_min_gcp[15:0]</td>
<td>Input</td>
<td>Minimum global control opcode value.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>ctl_rx_opcode_max_gcp[15:0]</td>
<td>Input</td>
<td>Maximum global control opcode value.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>ctl_rx_etype_gcp[15:0]</td>
<td>Input</td>
<td>Ethertype field for global control processing.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>ctl_rx_enable_pcp</td>
<td>Input</td>
<td>A value of 1 enables priority control packet processing.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>ctl_rx_check_mcast_pcp</td>
<td>Input</td>
<td>A value of 1 enables priority control multicast destination address processing.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>ctl_rx_check_uctcast_pcp</td>
<td>Input</td>
<td>A value of 1 enables priority control unicast destination address processing.</td>
<td>rx_clk_out</td>
</tr>
</tbody>
</table>
### Table 2-14: Pause Interface–RX Path (Cont’d)

<table>
<thead>
<tr>
<th>Name</th>
<th>Direction</th>
<th>Description</th>
<th>Clock Domain</th>
</tr>
</thead>
<tbody>
<tr>
<td>ctl_rx_pause_da_mcast[47:0]</td>
<td>Input</td>
<td>Multicast destination address for pause processing.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>ctl_rx_check_sa_pcp</td>
<td>Input</td>
<td>A value of 1 enables priority control source address processing.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>ctl_rx_check_etype_pcp</td>
<td>Input</td>
<td>A value of 1 enables priority control ethertype processing.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>ctl_rx_etype_pcp[15:0]</td>
<td>Input</td>
<td>Ethertype field for priority control processing.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>ctl_rx_check_opcode_pcp</td>
<td>Input</td>
<td>A value of 1 enables priority control opcode processing.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>ctl_rx_opcode_min_pcp[15:0]</td>
<td>Input</td>
<td>Minimum priority control opcode value.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>ctl_rx_opcode_max_pcp[15:0]</td>
<td>Input</td>
<td>Maximum priority control opcode value.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>ctl_rx_enable_gpp</td>
<td>Input</td>
<td>A value of 1 enables global pause packet processing.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>ctl_rx_check_mcast_gpp</td>
<td>Input</td>
<td>A value of 1 enables global pause multicast destination address processing.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>ctl_rx_check_ucast_gpp</td>
<td>Input</td>
<td>A value of 1 enables global pause unicast destination address processing.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>ctl_rx_check_sa_gpp</td>
<td>Input</td>
<td>A value of 1 enables global pause source address processing.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>ctl_rx_check_etype_gpp</td>
<td>Input</td>
<td>A value of 1 enables global pause ethertype processing.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>ctl_rx_etype_gpp[15:0]</td>
<td>Input</td>
<td>Ethertype field for global pause processing.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>ctl_rx_check_opcode_gpp</td>
<td>Input</td>
<td>A value of 1 enables global pause opcode processing.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>ctl_rx_opcode_gpp[15:0]</td>
<td>Input</td>
<td>Global pause opcode value.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>ctl_rx_enable_ppp[15:0]</td>
<td>Input</td>
<td>A value of 1 enables priority pause packet processing.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>ctl_rx_check_mcast_ppp</td>
<td>Input</td>
<td>A value of 1 enables priority pause multicast destination address processing.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>ctl_rx_check_ucast_ppp</td>
<td>Input</td>
<td>A value of 1 enables priority pause unicast destination address processing.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>ctl_rx_check_sa_ppp</td>
<td>Input</td>
<td>A value of 1 enables priority pause source address processing.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>ctl_rx_check_etype_ppp</td>
<td>Input</td>
<td>A value of 1 enables priority pause ethertype processing.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>ctl_rx_etype_ppp[15:0]</td>
<td>Input</td>
<td>Ethertype field for priority pause processing.</td>
<td>rx_clk_out</td>
</tr>
</tbody>
</table>
### Table 2-14: Pause Interface–RX Path (Cont’d)

<table>
<thead>
<tr>
<th>Name</th>
<th>Direction</th>
<th>Description</th>
<th>Clock Domain</th>
</tr>
</thead>
<tbody>
<tr>
<td>ctl_rx_check_opcode_ppp</td>
<td>Input</td>
<td>A value of 1 enables priority pause opcode processing.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>ctl_rx_opcode_ppp[15:0]</td>
<td>Input</td>
<td>Priority pause opcode value.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>stat_rx_pause_req[9-1:0]</td>
<td>Output</td>
<td>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.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>ctl_rx_pause_ack[9-1:0]</td>
<td>Input</td>
<td>Pause acknowledge signal. This bus is used to acknowledge the receipt of the pause frame from the user logic.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>ctl_rx_check_ack</td>
<td>Input</td>
<td>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.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>ctl_rx_forward_control</td>
<td>Input</td>
<td>A value of 1 indicates that the core forwards control packets. A value of 0 causes core to drop control packets.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>stat_rx_pause_valid[9-1:0]</td>
<td>Output</td>
<td>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 MAC Pause packet is received, bit[8] is set to 1.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>stat_rx_pause_quanta[8:0][15:0]</td>
<td>Output</td>
<td>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 MAC Pause packet is received, the quanta is placed in value [8].</td>
<td>rx_clk_out</td>
</tr>
</tbody>
</table>
### Table 2-15: Pause Interface–TX Path

<table>
<thead>
<tr>
<th>Name</th>
<th>Direction</th>
<th>Description</th>
<th>Clock Domain</th>
</tr>
</thead>
<tbody>
<tr>
<td>ctl tx_pause_req[9-1:0]</td>
<td>Input</td>
<td>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.</td>
<td>tx clk out</td>
</tr>
<tr>
<td>ctl tx_pause_quanta[8:0][15:0]</td>
<td>Input</td>
<td>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 stat tx_pause_quanta[8] is used for global pause operation. All other values are used for priority pause operation.</td>
<td>tx clk out</td>
</tr>
<tr>
<td>ctl tx_pause_refresh_timer[8:0][15:0]</td>
<td>Input</td>
<td>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 stat tx_pause_refresh_timer[8] is used for global pause operation. All other values are used for priority pause operation.</td>
<td>tx clk out</td>
</tr>
<tr>
<td>ctl tx_da_gpp[47:0]</td>
<td>Input</td>
<td>Destination address for transmitting global pause packets.</td>
<td>tx clk out</td>
</tr>
<tr>
<td>ctl tx_sa_gpp[47:0]</td>
<td>Input</td>
<td>Source address for transmitting global pause packets.</td>
<td>tx clk out</td>
</tr>
<tr>
<td>ctl tx_ethertype_gpp[15:0]</td>
<td>Input</td>
<td>Ethertype for transmitting global pause packets.</td>
<td>tx clk out</td>
</tr>
<tr>
<td>ctl tx_opcode_gpp[15:0]</td>
<td>Input</td>
<td>Opcode for transmitting global pause packets.</td>
<td>tx clk out</td>
</tr>
<tr>
<td>ctl tx_da_ppp[47:0]</td>
<td>Input</td>
<td>Destination address for transmitting priority pause packets.</td>
<td>tx clk out</td>
</tr>
<tr>
<td>ctl tx_sa_ppp[47:0]</td>
<td>Input</td>
<td>Source address for transmitting priority pause packets.</td>
<td>tx clk out</td>
</tr>
<tr>
<td>ctl tx_ethertype_ppp[15:0]</td>
<td>Input</td>
<td>Ethertype for transmitting priority pause packets.</td>
<td>tx clk out</td>
</tr>
<tr>
<td>ctl tx_opcode_ppp[15:0]</td>
<td>Input</td>
<td>Opcode for transmitting priority pause packets.</td>
<td>tx clk out</td>
</tr>
</tbody>
</table>
Table 2-16: Additional Ports for Auto-Negotiation

<table>
<thead>
<tr>
<th>Port Name</th>
<th>Direction</th>
<th>Description</th>
<th>Clock Domain</th>
</tr>
</thead>
<tbody>
<tr>
<td>an_clk</td>
<td>Input</td>
<td>Input Clock for the auto-negotiation circuit. The required frequency is indicated in the readme file for the release.</td>
<td>Refer to Clocking.</td>
</tr>
<tr>
<td>an_reset</td>
<td>Input</td>
<td>Asynchronous active-High reset.</td>
<td>Asynch</td>
</tr>
<tr>
<td>ctl_autoneg_enable</td>
<td>Input</td>
<td>Enable signal for autonegotiation.</td>
<td>an_clk</td>
</tr>
<tr>
<td>ctl_autoneg_bypass</td>
<td>Input</td>
<td>Input to disable autonegotiation and bypass the autonegotiation function. If this input is asserted, then autonegotiation is turned off, but the PCS is connected to the outputs to allow operation.</td>
<td>an_clk</td>
</tr>
<tr>
<td>ctl_an_nonce_seed[7:0]</td>
<td>Input</td>
<td>8-bit seed to initialize the nonce field Polynomial generator.</td>
<td>an_clk</td>
</tr>
<tr>
<td>ctl_an_pseudo_sel</td>
<td>Input</td>
<td>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 ).</td>
<td>an_clk</td>
</tr>
<tr>
<td>ctl_restart_negotiation</td>
<td>Input</td>
<td>This input is used to trigger a restart of the auto negotiation, regardless of what state the circuit is currently in.</td>
<td>an_clk</td>
</tr>
</tbody>
</table>
### Table 2-16: Additional Ports for Auto-Negotiation (Cont’d)

<table>
<thead>
<tr>
<th>Port Name</th>
<th>Direction</th>
<th>Description</th>
<th>Clock Domain</th>
</tr>
</thead>
<tbody>
<tr>
<td>ctl_an_local_fault</td>
<td>Input</td>
<td>This input signal is used to set the local_fault bit of the transmit link codeword.</td>
<td>an_clk</td>
</tr>
<tr>
<td><strong>Signals Used for Pause Ability Advertising</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ctl_an_pause</td>
<td>Input</td>
<td>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.</td>
<td>an_clk</td>
</tr>
<tr>
<td>ctl_an_asmdir</td>
<td>Input</td>
<td>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.</td>
<td>an_clk</td>
</tr>
<tr>
<td><strong>Ability Signal Inputs</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ctl_an_ability_1000base_kx</td>
<td>Input</td>
<td></td>
<td>an_clk</td>
</tr>
<tr>
<td>ctl_an_ability_100gbase_cr10</td>
<td>Input</td>
<td></td>
<td>an_clk</td>
</tr>
<tr>
<td>ctl_an_ability_100gbase_cr4</td>
<td>Input</td>
<td></td>
<td>an_clk</td>
</tr>
<tr>
<td>ctl_an_ability_100gbase_kp4</td>
<td>Input</td>
<td></td>
<td>an_clk</td>
</tr>
<tr>
<td>ctl_an_ability_100gbase_kr4</td>
<td>Input</td>
<td></td>
<td>an_clk</td>
</tr>
<tr>
<td>ctl_an_ability_10gbase_kr</td>
<td>Input</td>
<td>These inputs identify the Ethernet protocol abilities that is advertised in the transmit link codeword to the link partner. A value of 1 indicates that the interface advertises that it supports the protocol.</td>
<td>an_clk</td>
</tr>
<tr>
<td>ctl_an_ability_10gbase_kx4</td>
<td>Input</td>
<td></td>
<td>an_clk</td>
</tr>
<tr>
<td>ctl_an_ability_25gbase_cr</td>
<td>Input</td>
<td></td>
<td>an_clk</td>
</tr>
<tr>
<td>ctl_an_ability_25gbase_cr1</td>
<td>Input</td>
<td></td>
<td>an_clk</td>
</tr>
<tr>
<td>ctl_an_ability_25gbase_kr</td>
<td>Input</td>
<td></td>
<td>an_clk</td>
</tr>
<tr>
<td>ctl_an_ability_25gbase_kr1</td>
<td>Input</td>
<td></td>
<td>an_clk</td>
</tr>
<tr>
<td>ctl_an_ability_40gbase_cr4</td>
<td>Input</td>
<td></td>
<td>an_clk</td>
</tr>
<tr>
<td>ctl_an_ability_40gbase_kr4</td>
<td>Input</td>
<td></td>
<td>an_clk</td>
</tr>
<tr>
<td>ctl_an_ability_50gbase_cr2</td>
<td>Input</td>
<td></td>
<td>an_clk</td>
</tr>
<tr>
<td>ctl_an_ability_50gbase_kr2</td>
<td>Input</td>
<td></td>
<td>an_clk</td>
</tr>
<tr>
<td>ctl_an_fec_request</td>
<td>Input</td>
<td>Used to control the clause 74 FEC request bit in the transmit link codeword. This signal might not be present if the IP core does not support clause 74 FEC.</td>
<td>an_clk</td>
</tr>
</tbody>
</table>
### Table 2-16: Additional Ports for Auto-Negotiation (Cont’d)

<table>
<thead>
<tr>
<th>Port Name</th>
<th>Direction</th>
<th>Description</th>
<th>Clock Domain</th>
</tr>
</thead>
<tbody>
<tr>
<td>ctl_an_fec_ability_override</td>
<td>Input</td>
<td>Used to control the clause 74 FEC ability bit in the transmit link codeword. If this input is set, then 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.</td>
<td>an_clk</td>
</tr>
<tr>
<td>ctl_an_cl91_fec_ability</td>
<td>Input</td>
<td>This bit is used to indicate clause 91 FEC ability.</td>
<td>an_clk</td>
</tr>
<tr>
<td>ctl_an_cl91_fec_request</td>
<td>Input</td>
<td>This bit is used to request clause 91 FEC.</td>
<td>an_clk</td>
</tr>
<tr>
<td>stat_an_link_cntl_1000base_kx[1:0]</td>
<td>Output</td>
<td>Link Control outputs from the auto-negotiation controller for the various Ethernet protocols. Settings are as follows:</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 00: DISABLE; PCS is disconnected;</td>
<td>an_clk</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 01: SCAN_FOR_CARRIER; RX is connected to PCS;</td>
<td>an_clk</td>
</tr>
<tr>
<td>stat_an_link_cntl_100base_cr10[1:0]</td>
<td>Output</td>
<td>• 11: ENABLE; PCS is connected for mission mode operation.</td>
<td>an_clk</td>
</tr>
<tr>
<td>stat_an_link_cntl_100base_cr4[1:0]</td>
<td>Output</td>
<td>• 10: not used</td>
<td>an_clk</td>
</tr>
<tr>
<td>stat_an_link_cntl_100base_kp4[1:0]</td>
<td>Output</td>
<td></td>
<td>an_clk</td>
</tr>
<tr>
<td>stat_an_link_cntl_100base_kr4[1:0]</td>
<td>Output</td>
<td></td>
<td>an_clk</td>
</tr>
<tr>
<td>stat_an_link_cntl_10base_kr[1:0]</td>
<td>Output</td>
<td></td>
<td>an_clk</td>
</tr>
<tr>
<td>stat_an_link_cntl_10base_kx4[1:0]</td>
<td>Output</td>
<td></td>
<td>an_clk</td>
</tr>
<tr>
<td>stat_an_link_cntl_25base_cr[1:0]</td>
<td>Output</td>
<td></td>
<td>an_clk</td>
</tr>
<tr>
<td>stat_an_link_cntl_25base_cr1[1:0]</td>
<td>Output</td>
<td></td>
<td>an_clk</td>
</tr>
<tr>
<td>stat_an_link_cntl_25base_kr[1:0]</td>
<td>Output</td>
<td></td>
<td>an_clk</td>
</tr>
<tr>
<td>stat_an_link_cntl_25base_kr1[1:0]</td>
<td>Output</td>
<td></td>
<td>an_clk</td>
</tr>
<tr>
<td>stat_an_link_cntl_40base_cr4[1:0]</td>
<td>Output</td>
<td></td>
<td>an_clk</td>
</tr>
<tr>
<td>stat_an_link_cntl_40base_kr4[1:0]</td>
<td>Output</td>
<td></td>
<td>an_clk</td>
</tr>
<tr>
<td>stat_an_link_cntl_50base_cr2[1:0]</td>
<td>Output</td>
<td></td>
<td>an_clk</td>
</tr>
<tr>
<td>stat_an_link_cntl_50base_kr2[1:0]</td>
<td>Output</td>
<td></td>
<td>an_clk</td>
</tr>
<tr>
<td>stat_an_fec_enable</td>
<td>Output</td>
<td>Used to enable the use of clause 74 FEC on the link.</td>
<td>an_clk</td>
</tr>
<tr>
<td>stat_an_rs_fec_enable</td>
<td>Output</td>
<td>Used to enable the use of clause 91 FEC on the link.</td>
<td>an_clk</td>
</tr>
<tr>
<td>stat_an_tx_pause_enable</td>
<td>Output</td>
<td>Used to enable station-to-station (global) pause packet generation in the transmit path to control data flow in the receive path.</td>
<td>an_clk</td>
</tr>
<tr>
<td>stat_an_rx_pause_enable</td>
<td>Output</td>
<td>Used to enable station-to-station (global) pause packet interpretation in the receive path, to control data flow from the transmitter.</td>
<td>an_clk</td>
</tr>
</tbody>
</table>
### Table 2-16: Additional Ports for Auto-Negotiation (Cont’d)

<table>
<thead>
<tr>
<th>Port Name</th>
<th>Direction</th>
<th>Description</th>
<th>Clock Domain</th>
</tr>
</thead>
<tbody>
<tr>
<td>stat_an_autoneg_complete</td>
<td>Output</td>
<td>Indicates the auto-negotiation is complete and RX link status from the PCS has been received.</td>
<td>an_clk</td>
</tr>
<tr>
<td>stat_an_parallel_detection_fault</td>
<td>Output</td>
<td>Indicated a parallel detection fault during auto-negotiation.</td>
<td>an_clk</td>
</tr>
<tr>
<td>stat_an_lp_ability_1000base_kx</td>
<td>Output</td>
<td>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.</td>
<td>an_clk</td>
</tr>
<tr>
<td>stat_an_lp_ability_100gbase_cr10</td>
<td>Output</td>
<td></td>
<td>an_clk</td>
</tr>
<tr>
<td>stat_an_lp_ability_100gbase_cr4</td>
<td>Output</td>
<td></td>
<td>an_clk</td>
</tr>
<tr>
<td>stat_an_lp_ability_100gbase_kp4</td>
<td>Output</td>
<td></td>
<td>an_clk</td>
</tr>
<tr>
<td>stat_an_lp_ability_100gbase_kr4</td>
<td>Output</td>
<td></td>
<td>an_clk</td>
</tr>
<tr>
<td>stat_an_lp_ability_10gbase_kr</td>
<td>Output</td>
<td></td>
<td>an_clk</td>
</tr>
<tr>
<td>stat_an_lp_ability_10gbase_kx4</td>
<td>Output</td>
<td></td>
<td>an_clk</td>
</tr>
<tr>
<td>stat_an_lp_ability_25gbase_cr</td>
<td>Output</td>
<td>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.</td>
<td>an_clk</td>
</tr>
<tr>
<td>stat_an_lp_ability_25gbase_cr1</td>
<td>Output</td>
<td>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.</td>
<td>an_clk</td>
</tr>
<tr>
<td>stat_an_lp_ability_25gbase_kr1</td>
<td>Output</td>
<td>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.</td>
<td>an_clk</td>
</tr>
<tr>
<td>stat_an_lp_ability_40gbase_cr4</td>
<td>Output</td>
<td>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.</td>
<td>an_clk</td>
</tr>
</tbody>
</table>
### Table 2-16: Additional Ports for Auto-Negotiation (Cont’d)

<table>
<thead>
<tr>
<th>Port Name</th>
<th>Direction</th>
<th>Description</th>
<th>Clock Domain</th>
</tr>
</thead>
<tbody>
<tr>
<td>stat_an_lp_ability_50gbase_kr2</td>
<td>Output</td>
<td>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.</td>
<td>an_clk</td>
</tr>
<tr>
<td>stat_an_lp_pause</td>
<td>Output</td>
<td>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.</td>
<td>an_clk</td>
</tr>
<tr>
<td>stat_an_lp_asm_dir</td>
<td>Output</td>
<td>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.</td>
<td>an_clk</td>
</tr>
<tr>
<td>stat_an_lp_fec_ability</td>
<td>Output</td>
<td>This signal indicates the advertised value of the FEC ability bit in the receive link codeword from the link partner. It becomes valid when the output signal stat_AN_lp_Ability_Valid is asserted.</td>
<td>an_clk</td>
</tr>
<tr>
<td>stat_an_lp_fec_request</td>
<td>Output</td>
<td>This signal indicates the advertised value of the FEC Request bit in the receive link codeword from the link partner. It becomes valid when the output signal stat_AN_lp_Ability_Valid is asserted.</td>
<td>an_clk</td>
</tr>
<tr>
<td>stat_an_lp_autoneg_able</td>
<td>Output</td>
<td>This output signal indicates that the link partner is able to perform autonegotiation. It becomes valid when the output signal stat_AN_lp_Ability_Valid is asserted.</td>
<td>an_clk</td>
</tr>
<tr>
<td>stat_an_lp_ability_valid</td>
<td>Output</td>
<td>This signal indicates when all of the link partner advertisements become valid.</td>
<td>an_clk</td>
</tr>
</tbody>
</table>
### Table 2-16: Additional Ports for Auto-Negotiation (Cont’d)

<table>
<thead>
<tr>
<th>Port Name</th>
<th>Direction</th>
<th>Description</th>
<th>Clock Domain</th>
</tr>
</thead>
<tbody>
<tr>
<td>an_loc_np_data[47:0]</td>
<td>Input</td>
<td>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, and T, bit positions 15, 14, 12, and 11, are not transferred as part of the next page codeword. These bits are generated in the Auto-Negotiation Intellectual Property Core (ANIPC). However, the Message Protocol bit, MP, in bit position 13, is transferred.</td>
<td>an_clk</td>
</tr>
<tr>
<td>an_lp_np_data[47:0]</td>
<td>Output</td>
<td>Link Partner Next Page Data. This 48-bit word is driven by the ANIPC with the 48-bit next page codeword from the remote link partner.</td>
<td>an_clk</td>
</tr>
<tr>
<td>ctl_an_loc_np</td>
<td>Input</td>
<td>Local Next Page indicator. If this bit is 1, the ANIPC transfers the next page word at input loc_np_data to the remote link partner. If this bit is 0, the ANIPC does not initiate the next page protocol. If the link partner has next pages to send, and the loc_np bit is clear, the ANIPC transfers null message pages.</td>
<td>an_clk</td>
</tr>
<tr>
<td>ctl_an_lp_np_ack</td>
<td>Input</td>
<td>Link Partner Next Page Acknowledge. This is used to signal the ANIPC 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 ANIPC acknowledges reception of the next page codeword to the remote link partner and initiate transfer of the next codeword. During this time, the ANIPC will remove the lp_np signal until the new next page information is available.</td>
<td>an_clk</td>
</tr>
</tbody>
</table>
### Table 2-16: Additional Ports for Auto-Negotiation (Cont’d)

<table>
<thead>
<tr>
<th>Port Name</th>
<th>Direction</th>
<th>Description</th>
<th>Clock Domain</th>
</tr>
</thead>
<tbody>
<tr>
<td>stat_an_loc_np_ack</td>
<td>Output</td>
<td>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 ANIPC 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.</td>
<td>an_clk</td>
</tr>
<tr>
<td>stat_an_lp_np</td>
<td>Output</td>
<td>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, the lp_np output is driven High again.</td>
<td>an_clk</td>
</tr>
<tr>
<td>stat_an_lp_ability_extended_fec[1:0]</td>
<td>Output</td>
<td>This output indicates the extended FEC abilities as defined in Schedule 3.</td>
<td>an_clk</td>
</tr>
<tr>
<td>stat_an_lp_extended_ability_valid</td>
<td>Output</td>
<td>When this bit is 1, it indicates that the detected extended abilities are valid.</td>
<td>an_clk</td>
</tr>
<tr>
<td>stat_an_lp_rf</td>
<td>Output</td>
<td>This bit indicates link partner remote fault.</td>
<td>an_clk</td>
</tr>
</tbody>
</table>
**Table 2-16: Additional Ports for Auto-Negotiation (Cont’d)**

<table>
<thead>
<tr>
<th>Port Name</th>
<th>Direction</th>
<th>Description</th>
<th>Clock Domain</th>
</tr>
</thead>
<tbody>
<tr>
<td>stat_an_start_tx_disable</td>
<td>Output</td>
<td>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 state TX_DISABLE, this output will cycle High for 1 clock period. It effectively signals the start of auto-negotiation.</td>
<td>an_clk</td>
</tr>
<tr>
<td>stat_an_start_an_good_check</td>
<td>Output</td>
<td>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.</td>
<td>an_clk</td>
</tr>
</tbody>
</table>

**Link Training Ports**

Table 2-17 shows the Link Training ports.

**Table 2-17: Link Training Ports**

<table>
<thead>
<tr>
<th>Port Name</th>
<th>Direction</th>
<th>Description</th>
<th>Clock Domain</th>
</tr>
</thead>
<tbody>
<tr>
<td>ctl廖_training_enable</td>
<td>Input</td>
<td>Enables link training. When link training is disabled, all PCS lanes function in mission mode.</td>
<td>tx_serdes_clk</td>
</tr>
<tr>
<td>ctl廖_restart_training</td>
<td>Input</td>
<td>This signal triggers a restart of link training regardless of the current state.</td>
<td>tx_serdes_clk</td>
</tr>
<tr>
<td>ctl廖_rx_trained[1-1:0]</td>
<td>Input</td>
<td>This signal is asserted to indicate that the receiver finite impulse response (FIR) filter coefficients have all been set, and that the receiver portion of training is complete.</td>
<td>tx_serdes_clk</td>
</tr>
</tbody>
</table>
### Table 2-17: Link Training Ports (Cont’d)

<table>
<thead>
<tr>
<th>Port Name</th>
<th>Direction</th>
<th>Description</th>
<th>Clock Domain</th>
</tr>
</thead>
<tbody>
<tr>
<td>stat_lt_signal_detect[1-1:0]</td>
<td>Output</td>
<td>This signal indicates when the respective link training state machine has entered the SEND_DATA state, in which normal PCS operation can resume.</td>
<td>tx_serdes_clk</td>
</tr>
<tr>
<td>stat_lt_training[1-1:0]</td>
<td>Output</td>
<td>This signal indicates when the respective link training state machine is performing link training.</td>
<td>tx_serdes_clk</td>
</tr>
<tr>
<td>stat_lt_training_fail[1-1:0]</td>
<td>Output</td>
<td>This signal is asserted during link training if the corresponding link training state machine detects a time-out during the training period.</td>
<td>tx_serdes_clk</td>
</tr>
<tr>
<td>stat_lt_frame_lock[1-1:0]</td>
<td>Output</td>
<td>When link training has begun, these signals are asserted, for each physical medium dependent (PMD) lane, when the corresponding link training receiver is able to establish a frame synchronization with the link partner.</td>
<td>rx_serdes_clk</td>
</tr>
<tr>
<td>stat_lt_preset_from_rx[1-1:0]</td>
<td>Output</td>
<td>This signal reflects the value of the preset control bit received in the control block from the link partner.</td>
<td>rx_serdes_clk</td>
</tr>
<tr>
<td>stat_lt_initialize_from_rx[1-1:0]</td>
<td>Output</td>
<td>This signal reflects the value of the initialize control bit received in the control block from the link partner.</td>
<td>rx_serdes_clk</td>
</tr>
<tr>
<td>stat_lt_k_p1_from_rx0[1:0]</td>
<td>Output</td>
<td>This 2-bit field indicates the update control bits for the k+1 coefficient, as received from the link partner in the control block.</td>
<td>rx_serdes_clk</td>
</tr>
<tr>
<td>stat_lt_k0_from_rx0[1:0]</td>
<td>Output</td>
<td>This 2-bit field indicates the update control bits for the k0 coefficient, as received from the link partner in the control block.</td>
<td>rx_serdes_clk</td>
</tr>
<tr>
<td>stat_lt_k_m1_from_rx0[1:0]</td>
<td>Output</td>
<td>This 2-bit field indicates the update control bits for the k-1 coefficient, as received from the link partner in the control block.</td>
<td>rx_serdes_clk</td>
</tr>
<tr>
<td>stat_lt_stat_p1_from_rx0[1:0]</td>
<td>Output</td>
<td>This 2-bit field indicates the update status bits for the k+1 coefficient, as received from the link partner in the status block.</td>
<td>rx_serdes_clk</td>
</tr>
<tr>
<td>stat_lt_stat0_from_rx0[1:0]</td>
<td>Output</td>
<td>This 2-bit fields indicates the update status bits for the k0 coefficient, as received from the link partner in the status block.</td>
<td>rx_serdes_clk</td>
</tr>
<tr>
<td>Port Name</td>
<td>Direction</td>
<td>Description</td>
<td>Clock Domain</td>
</tr>
<tr>
<td>-----------</td>
<td>-----------</td>
<td>-------------</td>
<td>--------------</td>
</tr>
<tr>
<td>stat_lt_stat_m1_from_rx0[1:0]</td>
<td>Output</td>
<td>This 2-bit field indicates the update status bits for the k-1 coefficient, as received from the link partner in the status block</td>
<td>rx_serdes_clk</td>
</tr>
<tr>
<td>ctl_lt_pseudo_seed0[10:0]</td>
<td>Input</td>
<td>This 11-bit signal seeds the training pattern generator.</td>
<td>tx_serdes_clk</td>
</tr>
<tr>
<td>ctl_lt_preset_to_tx[1-1:0]</td>
<td>Input</td>
<td>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.</td>
<td>tx_serdes_clk</td>
</tr>
<tr>
<td>ctl_lt_initialize_to_tx[1-1:0]</td>
<td>Input</td>
<td>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.</td>
<td>tx_serdes_clk</td>
</tr>
<tr>
<td>ctl_lt_k_p1_to_tx0[1:0]</td>
<td>Input</td>
<td>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.</td>
<td>tx_serdes_clk</td>
</tr>
<tr>
<td>ctl_lt_k0_to_tx0[1:0]</td>
<td>Input</td>
<td>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.</td>
<td>tx_serdes_clk</td>
</tr>
<tr>
<td>ctl_lt_k_m1_to_tx0[1:0]</td>
<td>Input</td>
<td>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.</td>
<td>tx_serdes_clk</td>
</tr>
<tr>
<td>ctl_lt_stat_p1_to_tx0[1:0]</td>
<td>Input</td>
<td>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.</td>
<td>tx_serdes_clk</td>
</tr>
<tr>
<td>ctl_lt_stat0_to_tx0[1:0]</td>
<td>Input</td>
<td>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.</td>
<td>tx_serdes_clk</td>
</tr>
<tr>
<td>ctl_lt_stat_m1_to_tx0[1:0]</td>
<td>Input</td>
<td>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.</td>
<td>tx_serdes_clk</td>
</tr>
<tr>
<td>stat_lt_rx_sof[1-1:0]</td>
<td>Output</td>
<td>This output is High for 1 RX SerDes clock cycle to indicate the start of the link training frame.</td>
<td>rx_serdes_clk</td>
</tr>
</tbody>
</table>
Port Descriptions – PCS Variant

This section shows the 10G/25G PCS core ports. These are the ports when the PCS-only option is provided. There are no FCS functions. The PCS does not contain the Pause and Flow Control ports. The system interface is XXVMII. Table 2-18 shows the PCS variant I/O ports.

Table 2-18: PCS Variant I/O Ports

<table>
<thead>
<tr>
<th>Name</th>
<th>Direction</th>
<th>Description</th>
<th>Clock Domain</th>
</tr>
</thead>
<tbody>
<tr>
<td>stat_tx_local_fault</td>
<td>Output</td>
<td>A value of 1 indicates the transmit encoder state machine is in the TX_INIT state. This output is level sensitive.</td>
<td>tx_mii_clk</td>
</tr>
<tr>
<td>ctl_rx_prbs31_test_pattern_enable</td>
<td>Input</td>
<td>Corresponds to MDIO register bit 3.42.5 as defined in Clause 45. Takes first precedence.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>ctl_rx_test_pattern_enable</td>
<td>Input</td>
<td>Test pattern enable for the RX core. A value of 1 enables test mode. Corresponds to MDIO register bit 3.42.2 as defined in Clause 45. Takes second precedence.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>ctl_rx_data_pattern_select</td>
<td>Input</td>
<td>Corresponds to MDIO register bit 3.42.0 as defined in Clause 45.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>ctl_rx_test_pattern</td>
<td>Input</td>
<td>Test pattern enable for the RX core to receive scrambled idle pattern. Takes third precedence.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>ctl_tx_prbs31_test_pattern_enable</td>
<td>Input</td>
<td>Corresponds to MDIO register bit 3.42.4 as defined in Clause 45. Takes first precedence.</td>
<td>tx_mii_clk</td>
</tr>
<tr>
<td>ctl_tx_test_pattern_enable</td>
<td>Input</td>
<td>Test pattern generation enable for the TX core. A value of 1 enables test mode. Corresponds to MDIO register bit 3.42.3 as defined in Clause 45. Takes second precedence.</td>
<td>tx_mii_clk</td>
</tr>
<tr>
<td>ctl_tx_test_pattern_select</td>
<td>Input</td>
<td>Corresponds to MDIO register bit 3.42.1 as defined in Clause 45.</td>
<td>tx_mii_clk</td>
</tr>
<tr>
<td>ctl_tx_data_pattern_select</td>
<td>Input</td>
<td>Corresponds to MDIO register bit 3.42.0 as defined in Clause 45.</td>
<td>tx_mii_clk</td>
</tr>
<tr>
<td>ctl_tx_test_pattern_seed_a[57:0]</td>
<td>Input</td>
<td>Corresponds to MDIO registers 3.34 through to 3.37 as defined in Clause 45.</td>
<td>tx_mii_clk</td>
</tr>
</tbody>
</table>
### Table 2-18: PCS Variant I/O Ports (Cont’d)

<table>
<thead>
<tr>
<th>Name</th>
<th>Direction</th>
<th>Description</th>
<th>Clock Domain</th>
</tr>
</thead>
<tbody>
<tr>
<td>ctl_tx_test_pattern_seed_b[57:0]</td>
<td>Input</td>
<td>Corresponds to MDIO registers 3.38 through to 3.41 as defined in Clause 45.</td>
<td>tx_mii_clk</td>
</tr>
<tr>
<td>ctl_tx_test_pattern</td>
<td>Input</td>
<td>Scrambled idle Test pattern generation enable for the TX core. A value of 1 enables test mode. Takes third precedence.</td>
<td>tx_mii_clk</td>
</tr>
<tr>
<td>stat_tx_fifo_error</td>
<td>Output</td>
<td>Transmit clock compensation FIFO error indicator. A value of 1 indicates the clock compensation FIFO under or overflowed. If this output is sampled as a 1 in any clock cycle, the corresponding port must be reset to resume proper operation.</td>
<td>tx_mii_clk</td>
</tr>
<tr>
<td>stat_rx_fifo_error</td>
<td>Output</td>
<td>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.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>stat_rx_local_fault</td>
<td>Output</td>
<td>A value of 1 indicates the receive decoder state machine is in the RX_INIT state. This output is level sensitive.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>stat_rx_hi_ber</td>
<td>Output</td>
<td>High Bit Error Rate (BER) indicator. When set to 1, the BER is too high as defined by the 802.3. Corresponds to MDIO register bit 3.32.1 as defined in Clause 45. This output is level sensitive.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>stat_rx_block_lock[1-1:0]</td>
<td>Output</td>
<td>Block lock status for each PCS lane. A value of 1 indicates the corresponding lane has achieved a block lock as defined in Clause 49. Corresponds to MDIO register bit 3.50.7:0 and 3.51.11:0 as defined in Clause 45. This output is level sensitive.</td>
<td>rx_clk_out</td>
</tr>
</tbody>
</table>
### Table 2-18: PCS Variant I/O Ports (Cont’d)

<table>
<thead>
<tr>
<th>Name</th>
<th>Direction</th>
<th>Description</th>
<th>Clock Domain</th>
</tr>
</thead>
<tbody>
<tr>
<td>stat_rx_error</td>
<td>Output</td>
<td>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 45. This output is pulsed for one clock cycle.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>stat_rx_valid_ctrl_code</td>
<td>Output</td>
<td>Indicates that a PCS block with a valid control code was received.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>stat_rx_error_valid</td>
<td>Output</td>
<td>Increment valid indicator. If this signal is a 1 in any clock cycle, the value of stat_rx_error_valid[0:0] is valid.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>stat_rx_bad_code</td>
<td>Output</td>
<td>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 45.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>stat_rx_bad_code_valid</td>
<td>Output</td>
<td>Increment valid indicator. If this signal is a 1 in any clock cycle, the value of stat_rx_bad_code[0:0] is valid.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>stat_rx_framing_err</td>
<td>Output</td>
<td>Increment value for number of bad sync header bits detected. The value of this bus is only valid in the same cycle that the corresponding stat_rx_framing_err_valid is a 1.</td>
<td>rx_clk_out</td>
</tr>
<tr>
<td>stat_rx_framing_err_valid</td>
<td>Output</td>
<td>Increment valid indicator. If this signal is a 1 in any clock cycle, the value of stat_rx_framing_err[0:0] is valid.</td>
<td>rx_clk_out</td>
</tr>
</tbody>
</table>
Transceiver Interface Ports

Table 2-19 shows the transceiver I/O ports.

Table 2-19: Transceiver I/O

<table>
<thead>
<tr>
<th>Name</th>
<th>Direction</th>
<th>Description</th>
<th>Clock Domain</th>
</tr>
</thead>
<tbody>
<tr>
<td>GT_reset</td>
<td>Input</td>
<td>Active-High reset for the transceiver startup FSM. Note that this signal also initiates the reset sequence for the entire 10G/25G Ethernet IP core.</td>
<td>Asynch</td>
</tr>
<tr>
<td>refclk_n0</td>
<td>Input</td>
<td>Differential reference clock input for the SerDes, negative phase.</td>
<td>Refer to Clocking.</td>
</tr>
<tr>
<td>refclk_p0</td>
<td>Input</td>
<td>Differential reference clock input for the SerDes, negative phase.</td>
<td>Refer to Clocking.</td>
</tr>
<tr>
<td>rx_serdes_data_n0</td>
<td>Input</td>
<td>Serial data from the line; negative phase of the differential signal.</td>
<td>Refer to Clocking.</td>
</tr>
<tr>
<td>rx_serdes_data_p0</td>
<td>Input</td>
<td>Serial data from the line; positive phase of the differential signal.</td>
<td>Refer to Clocking.</td>
</tr>
<tr>
<td>tx_serdes_data_n0</td>
<td>Output</td>
<td>Serial data to the line; negative phase of the differential signal.</td>
<td>Refer to Clocking.</td>
</tr>
<tr>
<td>tx_serdes_data_p0</td>
<td>Output</td>
<td>Serial data to the line; positive phase of the differential signal.</td>
<td>Refer to Clocking.</td>
</tr>
<tr>
<td>tx_serdes_clkout</td>
<td>Output</td>
<td>When present, same as tx_clk_out.</td>
<td>Refer to Clocking.</td>
</tr>
</tbody>
</table>

XGMII/XXVGMII Interface Ports

Table 2-20 shows the XGMII/XXVGMII I/O ports.

Table 2-20: XGMII/XXVGMII Interface Ports

<table>
<thead>
<tr>
<th>Name</th>
<th>Direction</th>
<th>Description</th>
<th>Clock Domain</th>
</tr>
</thead>
<tbody>
<tr>
<td>rx_mii_d[64-1:0]</td>
<td>Output</td>
<td>Receive XGMII/XXVGMII Data bus.</td>
<td>rx_mii_clk</td>
</tr>
<tr>
<td>rx_mii_c[8-1:0]</td>
<td>Output</td>
<td>Receive XGMII/XXVGMII Control bus.</td>
<td>rx_mii_clk</td>
</tr>
<tr>
<td>rx_mii_clk</td>
<td>Input</td>
<td>Receive XGMII/XXVGMII Clock input.</td>
<td>Refer to Clocking.</td>
</tr>
<tr>
<td>tx_mii_d[64-1:0]</td>
<td>Input</td>
<td>Transmit XGMII/XXVGMII Data bus.</td>
<td>tx_mii_clk</td>
</tr>
<tr>
<td>tx_mii_c[8-1:0]</td>
<td>Input</td>
<td>Transmit XGMII/XXVGMII Control bus.</td>
<td>tx_mii_clk</td>
</tr>
<tr>
<td>rx_clk_out</td>
<td>Output</td>
<td>This is the reference clock for RX PCS stats.</td>
<td>Refer to Clocking.</td>
</tr>
<tr>
<td>tx_clk_out (or tx_mii_clk)</td>
<td>Output</td>
<td>This output is used to clock the TX mii bus. Data is clocked on the positive edge of this signal.</td>
<td>Refer to Clocking.</td>
</tr>
<tr>
<td>rx_mii_reset</td>
<td>Input</td>
<td>Reset input for the RX mii interface.</td>
<td>Asynch</td>
</tr>
<tr>
<td>tx_mii_reset</td>
<td>Input</td>
<td>Reset input for the TX mii interface.</td>
<td>Asynch</td>
</tr>
</tbody>
</table>
Chapter 2: Product Specification

Miscellaneous Status/Control Ports

Table 2-21 shows the miscellaneous status/control ports.

Table 2-21: Miscellaneous Status/Control Ports

<table>
<thead>
<tr>
<th>Name</th>
<th>Direction</th>
<th>Description</th>
<th>Clock Domain</th>
</tr>
</thead>
<tbody>
<tr>
<td>dclk</td>
<td>Input</td>
<td>DRP clock input. The required frequency is indicated on the readme file for the release.</td>
<td>Refer to Clocking.</td>
</tr>
<tr>
<td>ctl_local_loopback</td>
<td>Input</td>
<td>When High, this signal places the transceiver into the PMA loopback state.</td>
<td>Asynch</td>
</tr>
</tbody>
</table>

Register Space

The 10/25 G Ethernet can be optionally configured with AXI4-Lite registers to access the configuration and status signals.

AXI4-Lite Ports

Table 2-22 describes the port list for the AXI processor interface.

Table 2-22: AXI Ports

<table>
<thead>
<tr>
<th>Signal</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>s_axi_aclk</td>
<td>In</td>
<td>AXI4-Lite clock. Range between 10 MHz and 300 MHz</td>
</tr>
<tr>
<td>s_axi_aresten</td>
<td>In</td>
<td>Asynchronous active-Low reset</td>
</tr>
<tr>
<td>s_axi_awaddr[31:0]</td>
<td>In</td>
<td>Write address Bus</td>
</tr>
<tr>
<td>s_axi_awvalid</td>
<td>In</td>
<td>Write address valid</td>
</tr>
<tr>
<td>s_axi_awready</td>
<td>Out</td>
<td>Write address acknowledge</td>
</tr>
<tr>
<td>s_axi_wdata[31:0]</td>
<td>In</td>
<td>Write data bus</td>
</tr>
<tr>
<td>s_axi_wstrb[3:0]</td>
<td>In</td>
<td>Strobe signal for the data bus byte lane</td>
</tr>
<tr>
<td>s_axi_wvalid</td>
<td>Out</td>
<td>Write data valid</td>
</tr>
<tr>
<td>s_axi_wready</td>
<td>Out</td>
<td>Write data acknowledge</td>
</tr>
<tr>
<td>s_axi_bresp[1:0]</td>
<td>Out</td>
<td>Write transaction response</td>
</tr>
<tr>
<td>s_axi_bvalid</td>
<td>Out</td>
<td>Write response valid</td>
</tr>
<tr>
<td>s_axi_bready</td>
<td>In</td>
<td>Write response acknowledge</td>
</tr>
<tr>
<td>s_axi_araddr[31:0]</td>
<td>In</td>
<td>Read address bus</td>
</tr>
<tr>
<td>s_axi_arvalid</td>
<td>In</td>
<td>Read address valid</td>
</tr>
<tr>
<td>s_axi_arready</td>
<td>Out</td>
<td>Read address acknowledge</td>
</tr>
<tr>
<td>s_axi_rdata[31:0]</td>
<td>Out</td>
<td>Read data output</td>
</tr>
</tbody>
</table>
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 (i.e. `tick_reg_mode_sel`) should be set to 0. If `tick_reg_mode_sel` set to 1, `tick_reg` is used to read the statistics counters.

**Configuration Register Map**

The configuration space provides software with the ability to configure the IP core for various use cases. Certain features are optional and the assigned register might not exist in a particular variant, in which case the applicable registers are considered RESERVED.

In order for the programmed configurations to take effect, it is necessary to issue `s_axi_arresetn`, which is active-Low.
# Table 2-23: Configuration Register Map

<table>
<thead>
<tr>
<th>Hex Address</th>
<th>Register Name/Link to Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x0000</td>
<td>GT_RESET_REG: 0000</td>
<td></td>
</tr>
<tr>
<td>0x0004</td>
<td>RESET_REG: 0004</td>
<td></td>
</tr>
<tr>
<td>0x0008</td>
<td>MODE_REG: 0008</td>
<td></td>
</tr>
<tr>
<td>0x000C</td>
<td>CONFIGURATION_TX_REG1: 000C</td>
<td></td>
</tr>
<tr>
<td>0x0014</td>
<td>CONFIGURATION_RX_REG1: 0014</td>
<td></td>
</tr>
<tr>
<td>0x0018</td>
<td>CONFIGURATION_RX_MTU: 0018</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x001C</td>
<td>CONFIGURATION_VL_LENGTH_REG: 001C</td>
<td></td>
</tr>
<tr>
<td>0x0020</td>
<td>TICK_REG: 0020</td>
<td></td>
</tr>
<tr>
<td>0x0024</td>
<td>CONFIGURATION_REVISION_REG: 0024</td>
<td></td>
</tr>
<tr>
<td>0x0028</td>
<td>CONFIGURATION_TX_TEST_PAT_SEED_A_LSB: 0028</td>
<td></td>
</tr>
<tr>
<td>0x002C</td>
<td>CONFIGURATION_TX_TEST_PAT_SEED_A_MSB: 002C</td>
<td></td>
</tr>
<tr>
<td>0x0030</td>
<td>CONFIGURATION_TX_TEST_PAT_SEED_B_LSB: 0030</td>
<td></td>
</tr>
<tr>
<td>0x0034</td>
<td>CONFIGURATION_TX_TEST_PAT_SEED_B_MSB: 0034</td>
<td></td>
</tr>
<tr>
<td>0x0038</td>
<td>CONFIGURATION_1588_REG: 0038</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x0040</td>
<td>CONFIGURATION_TX_FLOW_CONTROL_REG1: 0040</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x0044</td>
<td>CONFIGURATION_TX_FLOW_CONTROL_REFRESH_REG1: 0044</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x0048</td>
<td>CONFIGURATION_TX_FLOW_CONTROL_REFRESH_REG2: 0048</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x004C</td>
<td>CONFIGURATION_TX_FLOW_CONTROL_REFRESH_REG3: 004C</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x0050</td>
<td>CONFIGURATION_TXFLOW_CONTROL_REFRESH_REG4: 0050</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x0054</td>
<td>CONFIGURATION_TXFLOW_CONTROL_REFRESH_REGS: 0054</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x0058</td>
<td>CONFIGURATION_TXFLOW_CONTROL_QUANTA_REG1: 0058</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x005C</td>
<td>CONFIGURATION_TXFLOW_CONTROL_QUANTA_REG2: 005C</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x0060</td>
<td>CONFIGURATION_TXFLOW_CONTROL_QUANTA_REG3: 0060</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x0064</td>
<td>CONFIGURATION_TXFLOW_CONTROL_QUANTA_REG4: 0064</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x0068</td>
<td>CONFIGURATION_TXFLOW_CONTROL_QUANTA_REGS: 0068</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x006C</td>
<td>CONFIGURATION_TXFLOW_CONTROL_PPPETYPE_OP_REG: 006C</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x0070</td>
<td>CONFIGURATION_TXFLOW_CONTROL_GPPETYPE_OP_REG: 0070</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x0074</td>
<td>CONFIGURATION_TXFLOW_CONTROL_GPP_DA_REG_LSB: 0074</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x0078</td>
<td>CONFIGURATION_TXFLOW_CONTROL_GPP_DA_REG_MSB: 0078</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x007C</td>
<td>CONFIGURATION_TXFLOW_CONTROL_GPP_SA_REG_LSB: 007C</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x0080</td>
<td>CONFIGURATION_TXFLOW_CONTROL_GPP_SA_REG_MSB: 0080</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x0084</td>
<td>CONFIGURATION_TXFLOW_CONTROL_PPP_DA_REG_LSB: 0084</td>
<td>Only in MAC+PCS variant</td>
</tr>
</tbody>
</table>
Table 2-23: Configuration Register Map (Cont’d)

<table>
<thead>
<tr>
<th>Hex Address</th>
<th>Register Name/Link to Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x0088</td>
<td>CONFIGURATION_TX_FLOW_CONTROL_PPP_DA_REG_MSB: 0088</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x008C</td>
<td>CONFIGURATION_TX_FLOW_CONTROL_PPP_SA_REG_LSB: 008C</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x0090</td>
<td>CONFIGURATION_TX_FLOW_CONTROL_PPP_SA_REG_MSB: 0090</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x0094</td>
<td>CONFIGURATION_RX_FLOW_CONTROL_REG1: 0094</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x0098</td>
<td>CONFIGURATION_RX_FLOW_CONTROL_REG2: 0098</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x009C</td>
<td>CONFIGURATION_RX_FLOW_CONTROL_PPP_ETYPE_OP_REG: 009C</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x00A0</td>
<td>CONFIGURATION_RX_FLOW_CONTROL_GPP_ETYPE_OP_REG: 00A0</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x00A4</td>
<td>CONFIGURATION_RX_FLOW_CONTROL_GCP_PCP_TYPE_REG: 00A4</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x00A8</td>
<td>CONFIGURATION_RX_FLOW_CONTROL_PCP_OP_REG: 00A8</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x00AC</td>
<td>CONFIGURATION_RX_FLOW_CONTROL_GCP_OP_REG: 00AC</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x00B0</td>
<td>CONFIGURATION_RX_FLOW_CONTROL_DA_REG1_LSB: 00B0</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x00B4</td>
<td>CONFIGURATION_RX_FLOW_CONTROL_DA_REG1_MSB: 00B4</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x00B8</td>
<td>CONFIGURATION_RX_FLOW_CONTROL_DA_REG2_LSB: 00B8</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x00BC</td>
<td>CONFIGURATION_RX_FLOW_CONTROL_DA_REG2_MSB: 00BC</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x00C0</td>
<td>CONFIGURATION_RX_FLOW_CONTROL_SA_REG1_LSB: 00C0</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x00C4</td>
<td>CONFIGURATION_RX_FLOW_CONTROL_SA_REG1_MSB: 00C4</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x00D0</td>
<td>CONFIGURATION_RSFEC_REG: 00D0</td>
<td></td>
</tr>
<tr>
<td>0x00D4</td>
<td>CONFIGURATION_FEC_REG: 00D4</td>
<td></td>
</tr>
<tr>
<td>0x00E0</td>
<td>CONFIGURATION_AN_CONTROL_REG1: 00E0</td>
<td></td>
</tr>
<tr>
<td>0x00E4</td>
<td>CONFIGURATION_AN_CONTROL_REG2: 00E4</td>
<td></td>
</tr>
<tr>
<td>0x00F8</td>
<td>CONFIGURATION_AN_ABILITY: 00F8</td>
<td></td>
</tr>
<tr>
<td>0x0100</td>
<td>CONFIGURATION_LT_CONTROL_REG1: 0100</td>
<td></td>
</tr>
<tr>
<td>0x0104</td>
<td>CONFIGURATION_LT_TRAINED_REG: 0104</td>
<td></td>
</tr>
<tr>
<td>0x0108</td>
<td>CONFIGURATION_LT_PRESET_REG: 0108</td>
<td></td>
</tr>
<tr>
<td>0x010C</td>
<td>CONFIGURATION_LT_INIT_REG: 010C</td>
<td></td>
</tr>
<tr>
<td>0x0110</td>
<td>CONFIGURATION_LT_SEED_REG0: 0110</td>
<td></td>
</tr>
<tr>
<td>0x0130</td>
<td>CONFIGURATION_LT_COEFFICIENT_REG0: 0130</td>
<td></td>
</tr>
<tr>
<td>0x0184</td>
<td>USER_REG_0: 0184</td>
<td></td>
</tr>
<tr>
<td>0x0188</td>
<td>USER_REG_0: 0188</td>
<td></td>
</tr>
</tbody>
</table>
Status Register Map

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

Status registers are cleared according to the following conditions.

- Applying `s_axi_aresetn` clears both TX and RX status registers
- When a particular status register is read (clear on read)
- Applying `rx_reset` clears the RX status registers only
- Applying `tx_reset` clears the TX status registers only

Table 2-24: Status Register Map

<table>
<thead>
<tr>
<th>Hex Address</th>
<th>Register Name/Link to Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x0180</td>
<td>CORE_SPEED_REG: 0180</td>
<td></td>
</tr>
<tr>
<td>0x0400</td>
<td>STAT_TX_STATUS_REG1: 0400</td>
<td></td>
</tr>
<tr>
<td>0x0404</td>
<td>STAT_RX_STATUS_REG1: 0404</td>
<td></td>
</tr>
<tr>
<td>0x0408</td>
<td>STAT_STATUS_REG1: 0408</td>
<td></td>
</tr>
<tr>
<td>0x040C</td>
<td>STAT_RX_BLOCK_LOCK_REG: 040C</td>
<td></td>
</tr>
<tr>
<td>0x043C</td>
<td>STAT_RX_RSFEC_STATUS_REG: 043C</td>
<td></td>
</tr>
<tr>
<td>0x0448</td>
<td>STAT_RX_FEC_STATUS_REG: 0448</td>
<td></td>
</tr>
<tr>
<td>0x044C</td>
<td>STAT_TX_RSFEC_STATUS_REG: 044C</td>
<td></td>
</tr>
<tr>
<td>0x0450</td>
<td>STAT_TX_FLOW_CONTROL_REG1: 0450</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x0454</td>
<td>STAT_RX_FLOW_CONTROL_REG1: 0454</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x0458</td>
<td>STAT_AN_STATUS: 0458</td>
<td></td>
</tr>
<tr>
<td>0x045C</td>
<td>STAT_AN_ABILITY: 045C</td>
<td></td>
</tr>
<tr>
<td>0x0460</td>
<td>STAT_AN_LINK_CTL: 0460</td>
<td></td>
</tr>
<tr>
<td>0x0464</td>
<td>STAT_LT_STATUS_REG1: 0464</td>
<td></td>
</tr>
<tr>
<td>0x0468</td>
<td>STAT_LT_STATUS_REG2: 0468</td>
<td></td>
</tr>
<tr>
<td>0x046C</td>
<td>STAT_LT_STATUS_REG3: 046C</td>
<td></td>
</tr>
<tr>
<td>0x0470</td>
<td>STAT_LT_STATUS_REG4: 0470</td>
<td></td>
</tr>
<tr>
<td>0x0474</td>
<td>STAT_LT_COEFFICIENT0_REG: 0474</td>
<td></td>
</tr>
<tr>
<td>0x0494</td>
<td>STAT_RX_VALID_CTRL_CODE: 0494</td>
<td></td>
</tr>
</tbody>
</table>

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]` (i.e. tick_reg_mode_sel). `pm_tick` will be used when `MODE_REG[30] = 0` and `TICK_REG` will be used when `MODE_REG[30] = 1` (1 = default).
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 snap-shot 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 writes. This allows for easy time-interval based statistics.

Statistic counter registers are cleared according to the following conditions:

- Applying s_axi_aresetn clears both TX and RX statistics counter registers
- Applying PM Tick clears both TX and RX statistics counter registers
- Applying rx_reset clears the RX statistics counter registers only
- Applying tx_reset clears the TX statistics counter registers only

**Table 2-25: Statistics Counters**

<table>
<thead>
<tr>
<th>Hex Address</th>
<th>Register Name/Link to Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x0500</td>
<td>STATUS_CYCLE_COUNT_LSB: 0500</td>
<td></td>
</tr>
<tr>
<td>0x0504</td>
<td>STATUS_CYCLE_COUNT_MSB: 0504</td>
<td></td>
</tr>
<tr>
<td>0x0648</td>
<td>STAT_RX_FRAMING_ERR_LSB: 0648</td>
<td></td>
</tr>
<tr>
<td>0x064C</td>
<td>STAT_RX_FRAMING_ERR_MSB: 064C</td>
<td></td>
</tr>
<tr>
<td>0x0660</td>
<td>STAT_RX_BAD_CODE_LSB: 0660</td>
<td></td>
</tr>
<tr>
<td>0x0664</td>
<td>STAT_RX_BAD_CODE_MSB: 0664</td>
<td></td>
</tr>
<tr>
<td>0x0668</td>
<td>STAT_RX_ERROR_LSB: 0668</td>
<td>Only in PCS variant</td>
</tr>
<tr>
<td>0x066C</td>
<td>STAT_RX_ERROR_MSB: 066C</td>
<td>Only in PCS variant</td>
</tr>
<tr>
<td>0x0670</td>
<td>STAT_RX_RSFEC_CORRECTED_CW_INC_LSB: 0670</td>
<td></td>
</tr>
<tr>
<td>0x0674</td>
<td>STAT_RX_RSFEC_CORRECTED_CW_INC_MSB: 0674</td>
<td></td>
</tr>
<tr>
<td>0x0678</td>
<td>STAT_RX_RSFEC_UNCORRECTED_CW_INC_LSB: 0678</td>
<td></td>
</tr>
<tr>
<td>0x067C</td>
<td>STAT_RX_RSFEC_UNCORRECTED_CW_INC_MSB: 067C</td>
<td></td>
</tr>
<tr>
<td>0x0680</td>
<td>STAT_RX_RSFEC_ERR_COUNT0_INC_LSB: 0680</td>
<td></td>
</tr>
<tr>
<td>0x0684</td>
<td>STAT_RX_RSFEC_ERR_COUNT0_INC_MSB: 0684</td>
<td></td>
</tr>
<tr>
<td>0x06A0</td>
<td>STAT_TX_FRAME_ERROR_LSB: 06A0</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x06A4</td>
<td>STAT_TX_FRAME_ERROR_MSB: 06A4</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x0700</td>
<td>STAT_TX_TOTAL_PACKETS_LSB: 0700</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x0704</td>
<td>STAT_TX_TOTAL_PACKETS_MSB: 0704</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x0708</td>
<td>STAT_TX_TOTAL_GOOD_PACKETS_LSB: 0708</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x070C</td>
<td>STAT_TX_TOTAL_GOOD_PACKETS_MSB: 070C</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x0710</td>
<td>STAT_TX_TOTAL_BYTES_LSB: 0710</td>
<td>Only in MAC+PCS variant</td>
</tr>
</tbody>
</table>
### Table 2-25:  Statistics Counters (Cont'd)

<table>
<thead>
<tr>
<th>Hex Address</th>
<th>Register Name/Link to Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x0714</td>
<td>STAT_TX_TOTAL_BYTES_MSB: 0714</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x0718</td>
<td>STAT_TX_TOTAL_GOOD_BYTES_LSB: 0718</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x071C</td>
<td>STAT_TX_TOTAL_GOOD_BYTES_MSB: 071C</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x0720</td>
<td>STAT_TX_PACKET_64_BYTES_MSB: 0720</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x0724</td>
<td>STAT_TX_PACKET_64_BYTES_MSB: 0724</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x0728</td>
<td>STAT_TX_PACKET_65_127_BYTES_LSB: 0728</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x072C</td>
<td>STAT_TX_PACKET_65_127_BYTES_MSB: 072C</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x0730</td>
<td>STAT_TX_PACKET_128_255_BYTES_LSB: 0730</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x0734</td>
<td>STAT_TX_PACKET_128_255_BYTES_MSB: 0734</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x0738</td>
<td>STAT_TX_PACKET_256_511_BYTES_LSB: 0738</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x073C</td>
<td>STAT_TX_PACKET_256_511_BYTES_MSB: 073C</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x0740</td>
<td>STAT_TX_PACKET_512_1023_BYTES_LSB: 0740</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x0744</td>
<td>STAT_TX_PACKET_512_1023_BYTES_MSB: 0744</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x0748</td>
<td>STAT_TX_PACKET_1024_1518_BYTES_LSB: 0748</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x074C</td>
<td>STAT_TX_PACKET_1024_1518_BYTES_MSB: 074C</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x0750</td>
<td>STAT_TX_PACKET_1519_1522_BYTES_LSB: 0750</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x0754</td>
<td>STAT_TX_PACKET_1519_1522_BYTES_MSB: 0754</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x0758</td>
<td>STAT_TX_PACKET_1523_1548_BYTES_LSB: 0758</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x075C</td>
<td>STAT_TX_PACKET_1523_1548_BYTES_MSB: 075C</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x0760</td>
<td>STAT_TX_PACKET_1549_2047_BYTES_LSB: 0760</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x0764</td>
<td>STAT_TX_PACKET_1549_2047_BYTES_MSB: 0764</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x0768</td>
<td>STAT_TX_PACKET_2048_4095_BYTES_LSB: 0768</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x076C</td>
<td>STAT_TX_PACKET_2048_4095_BYTES_MSB: 076C</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x0770</td>
<td>STAT_TX_PACKET_4096_8191_BYTES_LSB: 0770</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x0774</td>
<td>STAT_TX_PACKET_4096_8191_BYTES_MSB: 0774</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x0778</td>
<td>STAT_TX_PACKET_8192_9215_BYTES_LSB: 0778</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x077C</td>
<td>STAT_TX_PACKET_8192_9215_BYTES_MSB: 077C</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x0780</td>
<td>STAT_TX_PACKET_LARGE_LSB: 0780</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x0784</td>
<td>STAT_TX_PACKET_LARGE_MSB: 0784</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x0788</td>
<td>STAT_TX_PACKET_SMALL_LSB: 0788</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x078C</td>
<td>STAT_TX_PACKET_SMALL_MSB: 078C</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x07B8</td>
<td>STAT_TX_BAD_FCS_LSB: 07B8</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x07BC</td>
<td>STAT_TX_BAD_FCS_MSB: 07BC</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x07D0</td>
<td>STAT_TX_UNICAST_LSB: 07D0</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x07D4</td>
<td>STAT_TX_UNICAST_MSB: 07D4</td>
<td>Only in MAC+PCS variant</td>
</tr>
</tbody>
</table>
### Table 2-25: Statistics Counters (Cont’d)

<table>
<thead>
<tr>
<th>Hex Address</th>
<th>Register Name/Link to Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x07D8</td>
<td>STAT_TX_MULTICAST_LSB: 07D8</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x07DC</td>
<td>STAT_TX_MULTICAST_MSB: 07DC</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x07E0</td>
<td>STAT_TX_BROADCAST_LSB: 07E0</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x07E4</td>
<td>STAT_TX_BROADCAST_MSB: 07E4</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x07E8</td>
<td>STAT_TX_VLAN_LSB: 07E8</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x07EC</td>
<td>STAT_TX_VLAN_MSB: 07EC</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x07F0</td>
<td>STAT_TX_PAUSE_LSB: 07F0</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x07F4</td>
<td>STAT_TX_PAUSE_MSB: 07F4</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x07F8</td>
<td>STAT_TX_USER_PAUSE_LSB: 07F8</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x07FC</td>
<td>STAT_TX_USER_PAUSE_MSB: 07FC</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x0808</td>
<td>STAT_RX_TOTAL_PACKETS_LSB: 0808</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x080C</td>
<td>STAT_RX_TOTAL_PACKETS_MSB: 080C</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x0810</td>
<td>STAT_RX_TOTAL_GOOD_PACKETS_LSB: 0810</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x0814</td>
<td>STAT_RX_TOTAL_GOOD_PACKETS_MSB: 0814</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x0818</td>
<td>STAT_RX_TOTAL_BYTES_LSB: 0818</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x081C</td>
<td>STAT_RX_TOTAL_BYTES_MSB: 081C</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x0820</td>
<td>STAT_RX_TOTAL_GOOD_BYTES_LSB: 0820</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x0824</td>
<td>STAT_RX_TOTAL_GOOD_BYTES_MSB: 0824</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x0828</td>
<td>STAT_RX_PACKET_64_BYTES_LSB: 0828</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x082C</td>
<td>STAT_RX_PACKET_64_BYTES_MSB: 082C</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x0830</td>
<td>STAT_RX_PACKET_65_127_BYTES_LSB: 0830</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x0834</td>
<td>STAT_RX_PACKET_65_127_BYTES_MSB: 0834</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x0838</td>
<td>STAT_RX_PACKET_128_255_BYTES_LSB: 0838</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x083C</td>
<td>STAT_RX_PACKET_128_255_BYTES_MSB: 083C</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x0840</td>
<td>STAT_RX_PACKET_256_511_BYTES_LSB: 0840</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x0844</td>
<td>STAT_RX_PACKET_256_511_BYTES_MSB: 0844</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x0848</td>
<td>STAT_RX_PACKET_512_1023_BYTES_LSB: 0848</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x084C</td>
<td>STAT_RX_PACKET_512_1023_BYTES_MSB: 084C</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x0850</td>
<td>STAT_RX_PACKET_1024_1518_BYTES_LSB: 0850</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x0854</td>
<td>STAT_RX_PACKET_1024_1518_BYTES_MSB: 0854</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x0858</td>
<td>STAT_RX_PACKET_1519_1522_BYTES_LSB: 0858</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x085C</td>
<td>STAT_RX_PACKET_1519_1522_BYTES_MSB: 085C</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x0860</td>
<td>STAT_RX_PACKET_1523_1548_BYTES_LSB: 0860</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x0864</td>
<td>STAT_RX_PACKET_1523_1548_BYTES_MSB: 0864</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x0868</td>
<td>STAT_RX_PACKET_1549_2047_BYTES_LSB: 0868</td>
<td>Only in MAC+PCS variant</td>
</tr>
</tbody>
</table>
### Table 2-25: Statistics Counters (Cont’d)

<table>
<thead>
<tr>
<th>Hex Address</th>
<th>Register Name/Link to Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x086C</td>
<td>STAT_RX_PACKET_1549_2047_BYTES_MSB: 086C</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x0870</td>
<td>STAT_RX_PACKET_2048_4095_BYTES_LSB: 0870</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x0874</td>
<td>STAT_RX_PACKET_2048_4095_BYTES_MSB: 0874</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x0878</td>
<td>STAT_RX_PACKET_4096_8191_BYTES_LSB: 0878</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x087C</td>
<td>STAT_RX_PACKET_4096_8191_BYTES_MSB: 087C</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x0880</td>
<td>STAT_RX_PACKET_8192_9215_BYTES_LSB: 0880</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x0884</td>
<td>STAT_RX_PACKET_8192_9215_BYTES_MSB: 0884</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x0888</td>
<td>STAT_RX_PACKET_LARGE_LSB: 0888</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x088C</td>
<td>STAT_RX_PACKET_LARGE_MSB: 088C</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x0890</td>
<td>STAT_RX_PACKET_SMALL_LSB: 0890</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x0894</td>
<td>STAT_RX_PACKET_SMALL_MSB: 0894</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x0898</td>
<td>STAT_RX_UNDERSIZE_LSB: 0898</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x089C</td>
<td>STAT_RX_UNDERSIZE_MSB: 089C</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x08A0</td>
<td>STAT_RX_FRAGMENT_LSB: 08A0</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x08A4</td>
<td>STAT_RX_FRAGMENT_MSB: 08A4</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x08A8</td>
<td>STAT_RX_OVERSIZE_LSB: 08A8</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x08AC</td>
<td>STAT_RX_OVERSIZE_MSB: 08AC</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x08B0</td>
<td>STAT_RX_TOOLONG_LSB: 08B0</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x08B4</td>
<td>STAT_RX_TOOLONG_MSB: 08B4</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x08B8</td>
<td>STAT_RX_JABBER_LSB: 08B8</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x08BC</td>
<td>STAT_RX_JABBER_MSB: 08BC</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x08C0</td>
<td>STAT_RX_BAD_FCS_LSB: 08C0</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x08C4</td>
<td>STAT_RX_BAD_FCS_MSB: 08C4</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x08C8</td>
<td>STAT_RX_PACKET_BAD_FCS_LSB: 08C8</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x08CC</td>
<td>STAT_RX_PACKET_BAD_FCS_MSB: 08CC</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x08D0</td>
<td>STAT_RX_STOMPED_FCS_LSB: 08D0</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x08D4</td>
<td>STAT_RX_STOMPED_FCS_MSB: 08D4</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x08D8</td>
<td>STAT_RX_UNICAST_LSB: 08D8</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x08DC</td>
<td>STAT_RX_UNICAST_MSB: 08DC</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x08E0</td>
<td>STAT_RX_MULTICAST_LSB: 08E0</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x08E4</td>
<td>STAT_RX_MULTICAST_MSB: 08E4</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x08E8</td>
<td>STAT_RX_BROADCAST_LSB: 08E8</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x08EC</td>
<td>STAT_RX_BROADCAST_MSB: 08EC</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x08F0</td>
<td>STAT_RX_VLAN_LSB: 08F0</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x08F4</td>
<td>STAT_RX_VLAN_MSB: 08F4</td>
<td>Only in MAC+PCS variant</td>
</tr>
</tbody>
</table>
### Register Descriptions

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.

### Configuration Registers

The following tables define the bit assignments for the configuration registers.

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 a `s_axi_aresetn`.

A description of each signal is found in Port Descriptions.

#### GT_RESET_REG: 0000

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>RW</td>
<td>ctl_gt_reset_all</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>RW</td>
<td>ctl_gt_rx_reset</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>RW</td>
<td>ctl_gt_tx_reset</td>
</tr>
</tbody>
</table>

### Table 2-26: GT_RESET_REG: 0000

<table>
<thead>
<tr>
<th>Hex Address</th>
<th>Register Name/Link to Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x08F8</td>
<td>STAT_RX_PAUSE_LSB: 08F8</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x08FC</td>
<td>STAT_RX_PAUSE_MSB: 08FC</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x0900</td>
<td>STAT_RX_USER_PAUSE_LSB: 0900</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x0904</td>
<td>STAT_RX_USER_PAUSE_MSB: 0904</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x0908</td>
<td>STAT_RX_INRANGEERR_LSB: 0908</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x090C</td>
<td>STAT_RX_INRANGEERR_MSB: 090C</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x0910</td>
<td>STAT_RX_TRUNCATED_LSB: 0910</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x0914</td>
<td>STAT_RX_TRUNCATED_MSB: 0914</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x0918</td>
<td>STAT_RX_TEST_PATTERN_MISMATCH_LSB: 0918</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x091C</td>
<td>STAT_RX_TEST_PATTERN_MISMATCH_MSB: 091C</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x0920</td>
<td>STAT_FEC_INC_CORRECT_COUNT_LSB: 0920</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x0924</td>
<td>STAT_FEC_INC_CORRECT_COUNT_MSB: 0924</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x0928</td>
<td>STAT_FEC_INC_CANT_CORRECT_COUNT_LSB: 0928</td>
<td>Only in MAC+PCS variant</td>
</tr>
<tr>
<td>0x092C</td>
<td>STAT_FEC_INC_CANT_CORRECT_COUNT_MSB: 092C</td>
<td>Only in MAC+PCS variant</td>
</tr>
</tbody>
</table>
### RESET_REG: 0004

Table 2-27: RESET_REG: 0004

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>RW</td>
<td>rx_serdes_reset</td>
</tr>
<tr>
<td>29</td>
<td>0</td>
<td>RW</td>
<td>tx_serdes_reset</td>
</tr>
<tr>
<td>30</td>
<td>0</td>
<td>RW</td>
<td>rx_reset</td>
</tr>
<tr>
<td>31</td>
<td>0</td>
<td>RW</td>
<td>tx_reset</td>
</tr>
</tbody>
</table>

### MODE_REG: 0008

Table 2-28: MODE_REG: 0008

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>30</td>
<td>1</td>
<td>RW</td>
<td>tick_reg_mode_sel</td>
</tr>
<tr>
<td>31</td>
<td>0</td>
<td>RW</td>
<td>ctl_local_loopback</td>
</tr>
</tbody>
</table>

### CONFIGURATION_TX_REG1: 000C

Table 2-29: CONFIGURATION_TX_REG1: 000C

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>RW</td>
<td>ctl_tx_enable&lt;sup&gt;(1)&lt;/sup&gt;</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>RW</td>
<td>ctl_tx_fcs_ins_enable&lt;sup&gt;(1)&lt;/sup&gt;</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>RW</td>
<td>ctl_tx_ignore_fcs&lt;sup&gt;(1)&lt;/sup&gt;</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>RW</td>
<td>ctl_tx_send_lfi&lt;sup&gt;(1)&lt;/sup&gt;</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>RW</td>
<td>ctl_tx_send_rf&lt;sup&gt;(1)&lt;/sup&gt;</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>RW</td>
<td>ctl_tx_send_idle&lt;sup&gt;(1)&lt;/sup&gt;</td>
</tr>
<tr>
<td>13:10</td>
<td>12</td>
<td>RW</td>
<td>ctl_tx_ipg_value&lt;sup&gt;(1)&lt;/sup&gt;</td>
</tr>
<tr>
<td>14</td>
<td>0</td>
<td>RW</td>
<td>ctl_tx_test_pattern</td>
</tr>
<tr>
<td>15</td>
<td>0</td>
<td>RW</td>
<td>ctl_tx_test_pattern_enable</td>
</tr>
<tr>
<td>16</td>
<td>0</td>
<td>RW</td>
<td>ctl_tx_test_pattern_select</td>
</tr>
<tr>
<td>17</td>
<td>0</td>
<td>RW</td>
<td>ctl_tx_data_pattern_select</td>
</tr>
<tr>
<td>18</td>
<td>0</td>
<td>RW</td>
<td>ctl_tx_custom_preamble_enable&lt;sup&gt;(1)&lt;/sup&gt;</td>
</tr>
<tr>
<td>23</td>
<td>0</td>
<td>RW</td>
<td>ctl_tx_prbs31_test_pattern_enable&lt;sup&gt;(2)&lt;/sup&gt;</td>
</tr>
</tbody>
</table>

**Notes:**
1. Only in MAC+PCS variant
2. Only in PCS variant
### CONFIGURATION_RX_REG1: 0014

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>RW</td>
<td><code>ctl_rx_enable(1)</code></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>RW</td>
<td><code>ctl_rx_delete_fcs(1)</code></td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>RW</td>
<td><code>ctl_rx_ignore_fcs(1)</code></td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>RW</td>
<td><code>ctl_rx_process_lfi(1)</code></td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>RW</td>
<td><code>ctl_rx_check_sfd(1)</code></td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>RW</td>
<td><code>ctl_rx_check_preamble(1)</code></td>
</tr>
<tr>
<td>6</td>
<td>0</td>
<td>RW</td>
<td><code>ctl_rx_force_resync(1)</code></td>
</tr>
<tr>
<td>7</td>
<td>0</td>
<td>RW</td>
<td><code>ctl_rx_test_pattern</code></td>
</tr>
<tr>
<td>8</td>
<td>0</td>
<td>RW</td>
<td><code>ctl_rx_test_pattern_enable</code></td>
</tr>
<tr>
<td>9</td>
<td>0</td>
<td>RW</td>
<td><code>ctl_rx_data_pattern_select</code></td>
</tr>
<tr>
<td>10</td>
<td>-</td>
<td>-</td>
<td>Reserved</td>
</tr>
<tr>
<td>11</td>
<td>0</td>
<td>RW</td>
<td><code>ctl_rx_custom_preamble_enable</code></td>
</tr>
<tr>
<td>12</td>
<td>0</td>
<td>RW</td>
<td><code>ctl_rx_prbs31_test_pattern_enable(2)</code></td>
</tr>
</tbody>
</table>

Notes:
1. Only in MAC+PCS variant
2. Only in PCS variant

### CONFIGURATION_RX_MTU: 0018

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>7:0</td>
<td>64</td>
<td>RW</td>
<td><code>ctl_rx_min_packet_len</code></td>
</tr>
<tr>
<td>30:16</td>
<td>9,600</td>
<td>RW</td>
<td><code>ctl_rx_max_packet_len</code></td>
</tr>
</tbody>
</table>

### CONFIGURATION_VL_LENGTH_REG: 001C

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>15:0</td>
<td>20,479</td>
<td>RW</td>
<td><code>ctl_tx_vl_length_minus1</code></td>
</tr>
<tr>
<td>31:16</td>
<td>20,479</td>
<td>RW</td>
<td><code>ctl_rx_vl_length_minus1</code></td>
</tr>
</tbody>
</table>
TICK_REG: 0020

Table 2-33:  TICK_REG: 0020

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>WO</td>
<td>tick_reg</td>
</tr>
</tbody>
</table>

CONFIGURATION_REVISION_REG: 0024

Table 2-34:  CONFIGURATION_REVISION_REG: 0024

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>7:0</td>
<td>1</td>
<td>RO</td>
<td>major_rev</td>
</tr>
<tr>
<td>15:8</td>
<td>3</td>
<td>RO</td>
<td>minor_rev</td>
</tr>
<tr>
<td>31:24</td>
<td>0</td>
<td>RO</td>
<td>patch_rev</td>
</tr>
</tbody>
</table>

CONFIGURATION_TX_TEST_PAT_SEED_A_LSB: 0028

Table 2-35:  CONFIGURATION_TX_TEST_PAT_SEED_A_LSB: 0028

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:0</td>
<td>0</td>
<td>RW</td>
<td>ctl_tx_test_pattern_seed_a[31:0]</td>
</tr>
</tbody>
</table>

CONFIGURATION_TX_TEST_PAT_SEED_A_MSB: 002C

Table 2-36:  CONFIGURATION_TX_TEST_PAT_SEED_A_MSB: 002C

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>25:0</td>
<td>0</td>
<td>RW</td>
<td>ctl_tx_test_pattern_seed_a[57:32]</td>
</tr>
</tbody>
</table>

CONFIGURATION_TX_TEST_PAT_SEED_B_LSB: 0030

Table 2-37:  CONFIGURATION_TX_TEST_PAT_SEED_B_LSB: 0030

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:0</td>
<td>0</td>
<td>RW</td>
<td>ctl_tx_test_pattern_seed_b[31:0]</td>
</tr>
</tbody>
</table>

CONFIGURATION_TX_TEST_PAT_SEED_B_MSB: 0034

Table 2-38:  CONFIGURATION_TX_TEST_PAT_SEED_B_MSB: 0034

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>25:0</td>
<td>0</td>
<td>RW</td>
<td>ctl_tx_test_pattern_seed_b[57:32]</td>
</tr>
</tbody>
</table>
### CONFIGURATION_1588_REG: 0038

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>RW</td>
<td>ctl_tx_ptp_1step_enable</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>RW</td>
<td>ctl_ptp_transpclk_mode</td>
</tr>
<tr>
<td>26:16</td>
<td>0</td>
<td>RW</td>
<td>ctl_tx_ptp_latency_adjust</td>
</tr>
</tbody>
</table>

**Table 2-39: CONFIGURATION_1588_REG: 0038**

### CONFIGURATION_TX_FLOW_CONTROL_REG1: 0040

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>8:0</td>
<td>0</td>
<td>RW</td>
<td>ctl_tx_pause_enable</td>
</tr>
</tbody>
</table>

**Table 2-40: CONFIGURATION_TX_FLOW_CONTROL_REG1: 0040**

### CONFIGURATION_TX_FLOW_CONTROL_REFRESH_REG1: 0044

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>15:0</td>
<td>0</td>
<td>RW</td>
<td>ctl_tx_pause_refresh_timer0</td>
</tr>
<tr>
<td>31:16</td>
<td>0</td>
<td>RW</td>
<td>ctl_tx_pause_refresh_timer1</td>
</tr>
</tbody>
</table>

**Table 2-41: CONFIGURATION_TX_FLOW_CONTROL_REFRESH_REG1: 0044**

### CONFIGURATION_TX_FLOW_CONTROL_REFRESH_REG2: 0048

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>15:0</td>
<td>0</td>
<td>RW</td>
<td>ctl_tx_pause_refresh_timer2</td>
</tr>
<tr>
<td>31:16</td>
<td>0</td>
<td>RW</td>
<td>ctl_tx_pause_refresh_timer3</td>
</tr>
</tbody>
</table>

**Table 2-42: CONFIGURATION_TX_FLOW_CONTROL_REFRESH_REG2: 0048**

### CONFIGURATION_TX_FLOW_CONTROL_REFRESH_REG3: 004C

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>15:0</td>
<td>0</td>
<td>RW</td>
<td>ctl_tx_pause_refresh_timer4</td>
</tr>
<tr>
<td>31:16</td>
<td>0</td>
<td>RW</td>
<td>ctl_tx_pause_refresh_timer5</td>
</tr>
</tbody>
</table>

**Table 2-43: CONFIGURATION_TX_FLOW_CONTROL_REFRESH_REG3: 004C**
### Table 2-44: CONFIGURATION_TX_FLOW_CONTROL_REFRESH_REG4: 0050

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>15:0</td>
<td>0</td>
<td>RW</td>
<td>ctl_tx_pause_refresh_timer6</td>
</tr>
<tr>
<td>31:16</td>
<td>0</td>
<td>RW</td>
<td>ctl_tx_pause_refresh_timer7</td>
</tr>
</tbody>
</table>

### Table 2-45: CONFIGURATION_TX_FLOW_CONTROL_REFRESH_REG5: 0054

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>15:0</td>
<td>0</td>
<td>RW</td>
<td>ctl_tx_pause_refresh_timer8</td>
</tr>
</tbody>
</table>

### Table 2-46: CONFIGURATION_TX_FLOW_CONTROL_QUANTA_REG1: 0058

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>15:0</td>
<td>0</td>
<td>RW</td>
<td>ctl_tx_pause_quanta0</td>
</tr>
<tr>
<td>31:16</td>
<td>0</td>
<td>RW</td>
<td>ctl_tx_pause_quanta1</td>
</tr>
</tbody>
</table>

### Table 2-47: CONFIGURATION_TX_FLOW_CONTROL_QUANTA_REG2: 005C

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>15:0</td>
<td>0</td>
<td>RW</td>
<td>ctl_tx_pause_quanta2</td>
</tr>
<tr>
<td>31:16</td>
<td>0</td>
<td>RW</td>
<td>ctl_tx_pause_quanta3</td>
</tr>
</tbody>
</table>

### Table 2-48: CONFIGURATION_TX_FLOW_CONTROL_QUANTA_REG3: 0060

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>15:0</td>
<td>0</td>
<td>RW</td>
<td>ctl_tx_pause_quanta4</td>
</tr>
<tr>
<td>31:16</td>
<td>0</td>
<td>RW</td>
<td>ctl_tx_pause_quanta5</td>
</tr>
</tbody>
</table>
### Configuration TX Flow Control Quanta Register 4: 0064

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>15:0</td>
<td>0</td>
<td>RW</td>
<td>ctl_tx_pause_quanta6</td>
</tr>
<tr>
<td>31:16</td>
<td>0</td>
<td>RW</td>
<td>ctl_tx_pause_quanta7</td>
</tr>
</tbody>
</table>

### Configuration TX Flow Control Quanta Register 5: 0068

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>15:0</td>
<td>0</td>
<td>RW</td>
<td>ctl_tx_pause_quanta8</td>
</tr>
</tbody>
</table>

### Configuration TX Flow Control PPP EType Op Register: 006C

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>15:0</td>
<td>34824</td>
<td>RW</td>
<td>ctl_tx_ethertype_ppp</td>
</tr>
<tr>
<td>31:16</td>
<td>257</td>
<td>RW</td>
<td>ctl_tx_opcode_ppp</td>
</tr>
</tbody>
</table>

### Configuration TX Flow Control GPP EType Op Register: 0070

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>15:0</td>
<td>34824</td>
<td>RW</td>
<td>ctl_tx_ethertype_gpp</td>
</tr>
<tr>
<td>31:16</td>
<td>1</td>
<td>RW</td>
<td>ctl_tx_opcode_gpp</td>
</tr>
</tbody>
</table>

### Configuration TX Flow Control GPP DA Register LSB: 0074

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:0</td>
<td>0</td>
<td>RW</td>
<td>ctl_tx_da_gpp[31:0]</td>
</tr>
</tbody>
</table>

### Configuration TX Flow Control GPP DA Register MSB: 0078

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>15:0</td>
<td>0</td>
<td>RW</td>
<td>ctl_tx_da_gpp[47:32]</td>
</tr>
</tbody>
</table>
### Configuration_TX_Flow_Control_GPP_SA_REG_LSB: 007C

**Table 2-55: Configuration_TX_Flow Control_GPP_SA_REG_LSB: 007C**

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:0</td>
<td>0</td>
<td>RW</td>
<td>ctl_tx_sa_gpp[31:0]</td>
</tr>
</tbody>
</table>

### Configuration_TX_Flow_Control_GPP_SA_REG_MSB: 0080

**Table 2-56: Configuration_TX_Flow Control_GPP_SA_REG_MSB: 0080**

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:0</td>
<td>0</td>
<td>RW</td>
<td>ctl_tx_sa_gpp[47:32]</td>
</tr>
</tbody>
</table>

### Configuration_TX_Flow_Control_PPP_DA_REG_LSB: 0084

**Table 2-57: Configuration_TX_Flow Control_PPP_DA_REG_LSB: 0084**

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:0</td>
<td>0</td>
<td>RW</td>
<td>ctl_tx_da_ppp[31:0]</td>
</tr>
</tbody>
</table>

### Configuration_TX_Flow_Control_PPP_DA_REG_MSB: 0088

**Table 2-58: Configuration_TX_Flow Control_PPP_DA_REG_MSB: 0088**

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:0</td>
<td>0</td>
<td>RW</td>
<td>ctl_tx_da_ppp[47:32]</td>
</tr>
</tbody>
</table>

### Configuration_TX_Flow_Control_PPP_SA_REG_LSB: 008C

**Table 2-59: Configuration_TX_Flow Control_PPP_SA_REG_LSB: 008C**

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:0</td>
<td>0</td>
<td>RW</td>
<td>ctl_tx_sa_ppp[31:0]</td>
</tr>
</tbody>
</table>

### Configuration_TX_Flow_Control_PPP_SA_REG_MSB: 0090

**Table 2-60: Configuration_TX_Flow Control_PPP_SA_REG_MSB: 0090**

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:0</td>
<td>0</td>
<td>RW</td>
<td>ctl_tx_sa_ppp[47:32]</td>
</tr>
</tbody>
</table>
### CONFIGURATION_RX_FLOW_CONTROL_REG1: 0094

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>8:0</td>
<td>0</td>
<td>RW</td>
<td>ctl_rx_pause_enable</td>
</tr>
<tr>
<td>9</td>
<td>0</td>
<td>RW</td>
<td>ctl_rx_forward_control</td>
</tr>
<tr>
<td>10</td>
<td>0</td>
<td>RW</td>
<td>ctl_rx_enable_gcp</td>
</tr>
<tr>
<td>11</td>
<td>0</td>
<td>RW</td>
<td>ctl_rx_enable_pcp</td>
</tr>
<tr>
<td>12</td>
<td>0</td>
<td>RW</td>
<td>ctl_rx_enable_gpp</td>
</tr>
<tr>
<td>13</td>
<td>0</td>
<td>RW</td>
<td>ctl_rx_enable_ppp</td>
</tr>
<tr>
<td>14</td>
<td>0</td>
<td>RW</td>
<td>ctl_rx_check_ack</td>
</tr>
</tbody>
</table>

### CONFIGURATION_RX_FLOW_CONTROL_REG2: 0098

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>RW</td>
<td>ctl_rx_check_mcast_gcp</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>RW</td>
<td>ctl_rx_check_ucast_gcp</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>RW</td>
<td>ctl_rx_check_sa_gcp</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>RW</td>
<td>ctl_rx_check_etype_gcp</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>RW</td>
<td>ctl_rx_check_opcode_gcp</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>RW</td>
<td>ctl_rx_check_mcast_pcp</td>
</tr>
<tr>
<td>6</td>
<td>0</td>
<td>RW</td>
<td>ctl_rx_check_ucast_pcp</td>
</tr>
<tr>
<td>7</td>
<td>0</td>
<td>RW</td>
<td>ctl_rx_check_sa_pcp</td>
</tr>
<tr>
<td>8</td>
<td>0</td>
<td>RW</td>
<td>ctl_rx_check_etype_pcp</td>
</tr>
<tr>
<td>9</td>
<td>0</td>
<td>RW</td>
<td>ctl_rx_check_opcode_pcp</td>
</tr>
<tr>
<td>10</td>
<td>0</td>
<td>RW</td>
<td>ctl_rx_check_mcast_gpp</td>
</tr>
<tr>
<td>11</td>
<td>0</td>
<td>RW</td>
<td>ctl_rx_check_ucast_gpp</td>
</tr>
<tr>
<td>12</td>
<td>0</td>
<td>RW</td>
<td>ctl_rx_check_sa_gpp</td>
</tr>
<tr>
<td>13</td>
<td>0</td>
<td>RW</td>
<td>ctl_rx_check_etype_gpp</td>
</tr>
<tr>
<td>14</td>
<td>0</td>
<td>RW</td>
<td>ctl_rx_check_opcode_gpp</td>
</tr>
<tr>
<td>15</td>
<td>0</td>
<td>RW</td>
<td>ctl_rx_check_mcast_ppp</td>
</tr>
<tr>
<td>16</td>
<td>0</td>
<td>RW</td>
<td>ctl_rx_check_ucast_ppp</td>
</tr>
<tr>
<td>17</td>
<td>0</td>
<td>RW</td>
<td>ctl_rx_check_sa_ppp</td>
</tr>
<tr>
<td>18</td>
<td>0</td>
<td>RW</td>
<td>ctl_rx_check_etype_ppp</td>
</tr>
<tr>
<td>19</td>
<td>0</td>
<td>RW</td>
<td>ctl_rx_check_opcode_ppp</td>
</tr>
</tbody>
</table>
### Chapter 2: Product Specification

#### CONFIGURATION_RX_FLOW_CONTROL_PPP_ETYPE_OP_REG: 009C

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>15:0</td>
<td>34,824</td>
<td>RW</td>
<td>ctl_rx_etype_ppp</td>
</tr>
<tr>
<td>31:16</td>
<td>257</td>
<td>RW</td>
<td>ctl_rx_opcode_ppp</td>
</tr>
</tbody>
</table>

#### CONFIGURATION_RX_FLOW_CONTROL_GPP_ETYPE_OP_REG: 00A0

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>15:0</td>
<td>34,824</td>
<td>RW</td>
<td>ctl_rx_etype_gpp</td>
</tr>
<tr>
<td>31:16</td>
<td>1</td>
<td>RW</td>
<td>ctl_rx_opcode_gpp</td>
</tr>
</tbody>
</table>

#### CONFIGURATION_RX_FLOW_CONTROL_GCP_PCP_TYPE_REG: 00A4

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>15:0</td>
<td>34,824</td>
<td>RW</td>
<td>ctl_rx_etype_gcp</td>
</tr>
<tr>
<td>31:16</td>
<td>34,824</td>
<td>RW</td>
<td>ctl_rx_etype_pcp</td>
</tr>
</tbody>
</table>

#### CONFIGURATION_RX_FLOW_CONTROL_PCP_OP_REG: 00A8

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>15:0</td>
<td>257</td>
<td>RW</td>
<td>ctl_rx_opcode_min_pcp</td>
</tr>
<tr>
<td>31:16</td>
<td>257</td>
<td>RW</td>
<td>ctl_rx_opcode_max_pcp</td>
</tr>
</tbody>
</table>

#### CONFIGURATION_RX_FLOW_CONTROL_GCP_OP_REG: 00AC

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>15:0</td>
<td>1</td>
<td>RW</td>
<td>ctl_rx_opcode_min_gcp</td>
</tr>
<tr>
<td>31:16</td>
<td>6</td>
<td>RW</td>
<td>ctl_rx_opcode_max_gcp</td>
</tr>
</tbody>
</table>

#### CONFIGURATION_RX_FLOW_CONTROL_DA_REG1_LSB: 00B0

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:0</td>
<td>0</td>
<td>RW</td>
<td>ctl_pause_da_ucast[31:0]</td>
</tr>
</tbody>
</table>
### Chapter 2: Product Specification

#### CONFIGURATION_RX_FLOW_CONTROL_DA_REG1_MSB: 00B4

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>15:0</td>
<td>0</td>
<td>RW</td>
<td>ctl_rx_pau se_da_ ucast[47:32]</td>
</tr>
</tbody>
</table>

#### CONFIGURATION_RX_FLOW_CONTROL_DA_REG2_LSB: 00B8

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:0</td>
<td>0</td>
<td>RW</td>
<td>ctl_rx_pau se_da_mcast[31:0]</td>
</tr>
</tbody>
</table>

#### CONFIGURATION_RX_FLOW_CONTROL_DA_REG2_MSB: 00BC

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>15:0</td>
<td>0</td>
<td>RW</td>
<td>ctl_rx_pau se_da_mcast[47:32]</td>
</tr>
</tbody>
</table>

#### CONFIGURATION_RX_FLOW_CONTROL_SA_REG1_LSB: 00C0

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:0</td>
<td>0</td>
<td>RW</td>
<td>ctl_rx_pau se_sa[31:0]</td>
</tr>
</tbody>
</table>

#### CONFIGURATION_RX_FLOW_CONTROL_SA_REG1_MSB: 00C4

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>15:0</td>
<td>0</td>
<td>RW</td>
<td>ctl_rx_pau se_sa[47:32]</td>
</tr>
</tbody>
</table>
### CONFIGURATION_RSFEC_REG: 00D0

**Table 2-74: CONFIGURATION_RSFEC_REG: 00D0**

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>RW</td>
<td>ctl_rsfec_enable</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>RW</td>
<td>ctl_rsfec_consortium_25g</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>RW</td>
<td>ctl_rx_rsfec_enable_indication</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>RW</td>
<td>ctl_rx_rsfec_enable_correction</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>RW</td>
<td>ctl_rsfec_ieee_error_indication_mode</td>
</tr>
</tbody>
</table>

### CONFIGURATION_FEC_REG: 00D4

**Table 2-75: CONFIGURATION_FEC_REG: 00D4**

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>RW</td>
<td>ctl_fec_rx_enable</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>RW</td>
<td>ctl_fec_tx_enable</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>RW</td>
<td>ctl_fec_enable_error_to_pcs</td>
</tr>
</tbody>
</table>

### CONFIGURATION_AN_CONTROL_REG1: 00E0

**Table 2-76: CONFIGURATION_AN_CONTROL_REG1: 00E0**

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>RW</td>
<td>ctl_autoneg_enable</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>RW</td>
<td>ctl_autoneg_bypass</td>
</tr>
<tr>
<td>9:2</td>
<td>0</td>
<td>RW</td>
<td>ctl_an_nonce_seed</td>
</tr>
<tr>
<td>10</td>
<td>0</td>
<td>RW</td>
<td>ctl_an_pseudo_sel</td>
</tr>
<tr>
<td>11</td>
<td>0</td>
<td>RW</td>
<td>ctl_restart_negotiation</td>
</tr>
<tr>
<td>12</td>
<td>0</td>
<td>RW</td>
<td>ctl_an_local_fault</td>
</tr>
</tbody>
</table>

### CONFIGURATION_AN_CONTROL_REG2: 00E4

**Table 2-77: CONFIGURATION_AN_CONTROL_REG2: 00E4**

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>RW</td>
<td>ctl_an_pause</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>RW</td>
<td>ctl_an_asmdir</td>
</tr>
<tr>
<td>16</td>
<td>0</td>
<td>RW</td>
<td>ctl_an_fec_10g_request</td>
</tr>
<tr>
<td>17</td>
<td>0</td>
<td>RW</td>
<td>ctl_an_fec_ability_override</td>
</tr>
<tr>
<td>18</td>
<td>0</td>
<td>RW</td>
<td>ctl_an_cl91_fec_request</td>
</tr>
<tr>
<td>19</td>
<td>0</td>
<td>RW</td>
<td>ctl_an_cl91_fec_ability</td>
</tr>
</tbody>
</table>
### Table 2-77: **CONFIGURATION_AN_CONTROL_REG2: 00E4 (Cont’d)**

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>20</td>
<td>0</td>
<td>RW</td>
<td>ctl_an_fec_25g_rs_request</td>
</tr>
<tr>
<td>21</td>
<td>0</td>
<td>RW</td>
<td>ctl_an_fec_25g_baser_request</td>
</tr>
</tbody>
</table>

### CONFIGURATION_AN_ABILITY: 00F8

### Table 2-78: **CONFIGURATION_AN_ABILITY: 00F8**

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>RW</td>
<td>ctl_an_ability_1000base_kx</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>RW</td>
<td>ctl_an_ability_10gbase_kx4</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>RW</td>
<td>ctl_an_ability_10gbase_kr</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>RW</td>
<td>ctl_an_ability_40gbase_kr4</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>RW</td>
<td>ctl_an_ability_40gbase_cr4</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>RW</td>
<td>ctl_an_ability_100gbase_cr10</td>
</tr>
<tr>
<td>6</td>
<td>0</td>
<td>RW</td>
<td>ctl_an_ability_100gbase_kp4</td>
</tr>
<tr>
<td>7</td>
<td>0</td>
<td>RW</td>
<td>ctl_an_ability_100gbase_kr4</td>
</tr>
<tr>
<td>8</td>
<td>0</td>
<td>RW</td>
<td>ctl_an_ability_100gbase_cr4</td>
</tr>
<tr>
<td>9</td>
<td>0</td>
<td>RW</td>
<td>ctl_an_ability_25gbase_krcr_s</td>
</tr>
<tr>
<td>10</td>
<td>0</td>
<td>RW</td>
<td>ctl_an_ability_25gbase_krcr</td>
</tr>
<tr>
<td>11</td>
<td>0</td>
<td>RW</td>
<td>ctl_an_ability_25gbase_kr1</td>
</tr>
<tr>
<td>12</td>
<td>0</td>
<td>RW</td>
<td>ctl_an_ability_25gbase_cr1</td>
</tr>
<tr>
<td>13</td>
<td>0</td>
<td>RW</td>
<td>ctl_an_ability_50gbase_kr2</td>
</tr>
<tr>
<td>14</td>
<td>0</td>
<td>RW</td>
<td>ctl_an_ability_50gbase_cr2</td>
</tr>
</tbody>
</table>

### CONFIGURATION_LT_CONTROL_REG1: 0100

### Table 2-79: **CONFIGURATION_LT_CONTROL_REG1: 0100**

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>RW</td>
<td>ctl_lt_training_enable</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>RW</td>
<td>ctl_lt_restart_training</td>
</tr>
</tbody>
</table>

### CONFIGURATION_LT_TRAINED_REG: 0104

### Table 2-80: **CONFIGURATION_LT_TRAINED_REG: 0104**

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>RW</td>
<td>ctl_lt_rx_trained</td>
</tr>
</tbody>
</table>
### CONFIGURATION_LT_PRESET_REG: 0108

Table 2-81: **CONFIGURATION_LT_PRESET_REG: 0108**

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>RW</td>
<td><code>ctl_lt_preset_to_tx</code></td>
</tr>
</tbody>
</table>

### CONFIGURATION_LT_INIT_REG: 010C

Table 2-82: **CONFIGURATION_LT_INIT_REG: 010C**

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>RW</td>
<td><code>ctl_lt_initialize_to_tx</code></td>
</tr>
</tbody>
</table>

### CONFIGURATION_LT_SEED_REG0: 0110

Table 2-83: **CONFIGURATION_LT_SEED_REG0: 0110**

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>10:0</td>
<td>0</td>
<td>RW</td>
<td><code>ctl_lt_pseudo_seed0</code></td>
</tr>
</tbody>
</table>

### CONFIGURATION_LT_COEFFICIENT_REG0: 0130

Table 2-84: **CONFIGURATION_LT_COEFFICIENT_REG0: 0130**

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>1:0</td>
<td>0</td>
<td>RW</td>
<td><code>ctl_lt_k_p1_to_tx0</code></td>
</tr>
<tr>
<td>3:2</td>
<td>0</td>
<td>RW</td>
<td><code>ctl_lt_k0_to_tx0</code></td>
</tr>
<tr>
<td>5:4</td>
<td>0</td>
<td>RW</td>
<td><code>ctl_lt_k_m1_to_tx0</code></td>
</tr>
<tr>
<td>7:6</td>
<td>0</td>
<td>RW</td>
<td><code>ctl_lt_stat_p1_to_tx0</code></td>
</tr>
<tr>
<td>9:8</td>
<td>0</td>
<td>RW</td>
<td><code>ctl_lt_stat0_to_tx0</code></td>
</tr>
<tr>
<td>11:10</td>
<td>0</td>
<td>RW</td>
<td><code>ctl_lt_stat_m1_to_tx0</code></td>
</tr>
</tbody>
</table>

### USER_REG_0: 0184

Table 2-85: **USER_REG_0: 0184**

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:0</td>
<td>0</td>
<td>RW</td>
<td><code>user_reg0</code></td>
</tr>
</tbody>
</table>

### USER_REG_0: 0188

Table 2-86: **USER_REG_0: 0188**

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:0</td>
<td>0</td>
<td>RW</td>
<td><code>user_reg1</code></td>
</tr>
</tbody>
</table>
**Status Registers**

Table 2-87 to Table 2-105 define the bit assignments for the status registers.

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).

A description of each signal is found in Port Descriptions.

**CORE_SPEED_REG: 0180**

Table 2-87: CORE_SPEED_REG: 0180

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>RO</td>
<td>stat_core_speed</td>
</tr>
</tbody>
</table>

**STAT_TX_STATUS_REG1: 0400**

Table 2-88: STAT_TX_STATUS_REG1: 0400

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>RO LH</td>
<td>stat_tx_local_fault</td>
</tr>
</tbody>
</table>

**STAT_RX_STATUS_REG1: 0404**

Table 2-89: STAT_RX_STATUS_REG1: 0404

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>0</td>
<td>RO LH</td>
<td>stat_rx_hi_ber</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>RO LH</td>
<td>stat_rx_remote_fault(1)</td>
</tr>
<tr>
<td>6</td>
<td>0</td>
<td>RO LH</td>
<td>stat_rx_local_fault</td>
</tr>
<tr>
<td>7</td>
<td>0</td>
<td>RO LH</td>
<td>stat_rx_internal_local_fault(1)</td>
</tr>
<tr>
<td>8</td>
<td>0</td>
<td>RO LH</td>
<td>stat_rx_received_local_fault(1)</td>
</tr>
<tr>
<td>9</td>
<td>0</td>
<td>RO LH</td>
<td>stat_rx_bad_preamble(1)</td>
</tr>
<tr>
<td>10</td>
<td>0</td>
<td>RO LH</td>
<td>stat_rx_bad_sfd(1)</td>
</tr>
<tr>
<td>11</td>
<td>0</td>
<td>RO LH</td>
<td>stat_rx_got_signal_os(1)</td>
</tr>
</tbody>
</table>

**Notes:**
1. Only in MAC+PCS variant
STAT_STATUS_REG1: 0408

Table 2-90:  STAT_STATUS_REG1: 0408

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>RO LH</td>
<td>stat_tx_fifo_error(1)</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>RO LH</td>
<td>stat_tx_ptp_fifo_read_error</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>RO LH</td>
<td>stat_tx_ptp_fifo_write_error</td>
</tr>
<tr>
<td>16</td>
<td>0</td>
<td>RO LH</td>
<td>stat_rx_fifo_error(1)</td>
</tr>
</tbody>
</table>

Notes:
1. Only in PCS variant

STAT_RX_BLOCK_LOCK_REG: 040C

Table 2-91:  STAT_RX_BLOCK_LOCK_REG: 040C

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>RO LL</td>
<td>stat_rx_block_lock</td>
</tr>
</tbody>
</table>

STAT_RX_RSFEC_STATUS_REG: 043C

Table 2-92:  STAT_RX_RSFEC_STATUS_REG: 043C

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>RO LL</td>
<td>stat_rx_rsfec_lane_alignment_status</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>RO LL</td>
<td>stat_rx_rsfec_hi_ser</td>
</tr>
</tbody>
</table>

STAT_RX_FEC_STATUS_REG: 0448

Table 2-93:  STAT_RX_FEC_STATUS_REG: 0448

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>RO LL</td>
<td>stat_fec_rx_lock</td>
</tr>
<tr>
<td>16</td>
<td>1</td>
<td>RO LL</td>
<td>stat_fec_lock_error</td>
</tr>
</tbody>
</table>

STAT_TX_RSFEC_STATUS_REG: 044C

Table 2-94:  STAT_TX_RSFEC_STATUS_REG: 044C

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>RO LL</td>
<td>stat_tx_rsfec_lane_alignment_status</td>
</tr>
</tbody>
</table>
## STAT_TX_FLOW_CONTROL_REG1: 0450

Table 2-95: STAT_TX_FLOW_CONTROL_REG1: 0450

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>8:0</td>
<td>0</td>
<td>RO LH</td>
<td>stat_tx_pause_valid</td>
</tr>
</tbody>
</table>

## STAT_RX_FLOW_CONTROL_REG1: 0454

Table 2-96: STAT_RX_FLOW_CONTROL_REG1: 0454

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>8:0</td>
<td>0</td>
<td>RO LH</td>
<td>stat_rx_pause_req</td>
</tr>
<tr>
<td>17:9</td>
<td>0</td>
<td>RO LH</td>
<td>stat_rx_pause_valid</td>
</tr>
</tbody>
</table>

## STAT_AN_STATUS: 0458

Table 2-97: STAT_AN_STATUS: 0458

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>RO</td>
<td>stat_an_fec_enable</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>RO</td>
<td>stat_an_rs_fec_enable</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>RO</td>
<td>stat_an_autoneg_complete</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>RO</td>
<td>stat_an_parallel_detection_fault</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>RO</td>
<td>stat_an_tx_pause_enable</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>RO</td>
<td>stat_an_rx_pause_enable</td>
</tr>
<tr>
<td>6</td>
<td>0</td>
<td>RO LH</td>
<td>stat_an_lp_ability_valid</td>
</tr>
<tr>
<td>7</td>
<td>0</td>
<td>RO</td>
<td>stat_an_lp_autoneg_able</td>
</tr>
<tr>
<td>8</td>
<td>0</td>
<td>RO</td>
<td>stat_an_lp_pause</td>
</tr>
<tr>
<td>9</td>
<td>0</td>
<td>RO</td>
<td>stat_an_lp_asm_dir</td>
</tr>
<tr>
<td>10</td>
<td>0</td>
<td>RO</td>
<td>stat_an_lp_rf</td>
</tr>
<tr>
<td>11</td>
<td>0</td>
<td>RO</td>
<td>stat_an_lp_fec_10g_ability</td>
</tr>
<tr>
<td>12</td>
<td>0</td>
<td>RO</td>
<td>stat_an_lp_fec_10g_request</td>
</tr>
<tr>
<td>13</td>
<td>0</td>
<td>RO LH</td>
<td>stat_an_lp_extended_ability_valid</td>
</tr>
<tr>
<td>17:14</td>
<td>0</td>
<td>RO</td>
<td>stat_an_lp_ability_extended_fec</td>
</tr>
<tr>
<td>18</td>
<td>0</td>
<td>RO</td>
<td>stat_an_lp_fec_25g_rs_request</td>
</tr>
<tr>
<td>19</td>
<td>0</td>
<td>RO</td>
<td>stat_an_lp_fec_25g_baser_request</td>
</tr>
</tbody>
</table>
### STAT_AN_ABILITY: 045C

#### Table 2-98: STAT_AN_ABILITY: 045C

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>RO</td>
<td>stat_an_lp_ability_1000base_kx</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>RO</td>
<td>stat_an_lp_ability_10gbase_kx4</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>RO</td>
<td>stat_an_lp_ability_10gbase_kr</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>RO</td>
<td>stat_an_lp_ability_40gbase_kr4</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>RO</td>
<td>stat_an_lp_ability_40gbase_cr4</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>RO</td>
<td>stat_an_lp_ability_100gbase_cr10</td>
</tr>
<tr>
<td>6</td>
<td>0</td>
<td>RO</td>
<td>stat_an_lp_ability_100gbase_kp4</td>
</tr>
<tr>
<td>7</td>
<td>0</td>
<td>RO</td>
<td>stat_an_lp_ability_100gbase_kr4</td>
</tr>
<tr>
<td>8</td>
<td>0</td>
<td>RO</td>
<td>stat_an_lp_ability_100gbase_cr4</td>
</tr>
<tr>
<td>9</td>
<td>0</td>
<td>RO</td>
<td>stat_an_lp_ability_25gbase_krcr_s</td>
</tr>
<tr>
<td>10</td>
<td>0</td>
<td>RO</td>
<td>stat_an_lp_ability_25gbase_kcr</td>
</tr>
<tr>
<td>11</td>
<td>0</td>
<td>RO</td>
<td>stat_an_lp_ability_25gbase_kr1</td>
</tr>
<tr>
<td>12</td>
<td>0</td>
<td>RO</td>
<td>stat_an_lp_ability_25gbase_cr1</td>
</tr>
<tr>
<td>13</td>
<td>0</td>
<td>RO</td>
<td>stat_an_lp_ability_50gbase_kr2</td>
</tr>
<tr>
<td>14</td>
<td>0</td>
<td>RO</td>
<td>stat_an_lp_ability_50gbase_cr2</td>
</tr>
</tbody>
</table>
### STAT_AN_LINK_CTL: 0460

**Table 2-99: STAT_AN_LINK_CTL: 0460**

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>1:0</td>
<td>0</td>
<td>RO</td>
<td>stat_an_link_cntl_1000base_kx</td>
</tr>
<tr>
<td>3:2</td>
<td>0</td>
<td>RO</td>
<td>stat_an_link_cntl_10gbase_kx4</td>
</tr>
<tr>
<td>5:4</td>
<td>0</td>
<td>RO</td>
<td>stat_an_link_cntl_10gbase_kr</td>
</tr>
<tr>
<td>7:6</td>
<td>0</td>
<td>RO</td>
<td>stat_an_link_cntl_40gbase_kr4</td>
</tr>
<tr>
<td>9:8</td>
<td>0</td>
<td>RO</td>
<td>stat_an_link_cntl_40gbase_cr4</td>
</tr>
<tr>
<td>11:10</td>
<td>0</td>
<td>RO</td>
<td>stat_an_link_cntl_100gbase_cr10</td>
</tr>
<tr>
<td>13:12</td>
<td>0</td>
<td>RO</td>
<td>stat_an_link_cntl_100gbase_kp4</td>
</tr>
<tr>
<td>15:14</td>
<td>0</td>
<td>RO</td>
<td>stat_an_link_cntl_100gbase_kr4</td>
</tr>
<tr>
<td>17:16</td>
<td>0</td>
<td>RO</td>
<td>stat_an_link_cntl_100gbase_cr4</td>
</tr>
<tr>
<td>19:18</td>
<td>0</td>
<td>RO</td>
<td>stat_an_link_cntl_25gbase_kr1</td>
</tr>
<tr>
<td>21:20</td>
<td>0</td>
<td>RO</td>
<td>stat_an_link_cntl_25gbase_kr10</td>
</tr>
<tr>
<td>23:22</td>
<td>0</td>
<td>RO</td>
<td>stat_an_link_cntl_25gbase_kr</td>
</tr>
<tr>
<td>25:24</td>
<td>0</td>
<td>RO</td>
<td>stat_an_link_cntl_25gbase_cr1</td>
</tr>
<tr>
<td>27:26</td>
<td>0</td>
<td>RO</td>
<td>stat_an_link_cntl_50gbase_kr2</td>
</tr>
<tr>
<td>29:28</td>
<td>0</td>
<td>RO</td>
<td>stat_an_link_cntl_50gbase_cr2</td>
</tr>
</tbody>
</table>

### STAT_LT_STATUS_REG1: 0464

**Table 2-100: STAT_LT_STATUS_REG1: 0464**

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>RO</td>
<td>stat_lt_initialize_from_rx</td>
</tr>
<tr>
<td>16</td>
<td>0</td>
<td>RO</td>
<td>stat_lt_preset_from_rx</td>
</tr>
</tbody>
</table>

### STAT_LT_STATUS_REG2: 0468

**Table 2-101: STAT_LT_STATUS_REG2: 0468**

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>RO</td>
<td>stat_lt_training</td>
</tr>
<tr>
<td>16</td>
<td>0</td>
<td>RO</td>
<td>stat_lt_frame_lock</td>
</tr>
</tbody>
</table>
### STAT_LT_STATUS_REG3: 046C

**Table 2-102: STAT_LT_STATUS_REG3: 046C**

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>RO</td>
<td>stat_lt_signal_detect</td>
</tr>
<tr>
<td>16</td>
<td>0</td>
<td>RO</td>
<td>stat_lt_training_fail</td>
</tr>
</tbody>
</table>

### STAT_LT_STATUS_REG4: 0470

**Table 2-103: STAT_LT_STATUS_REG4: 0470**

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>RO LH</td>
<td>stat_lt_rx_sof</td>
</tr>
</tbody>
</table>

### STAT_LT_COEFFICIENT0_REG: 0474

**Table 2-104: STAT_LT_COEFFICIENT0_REG: 0474**

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>1:0</td>
<td>0</td>
<td>RO</td>
<td>stat_lt_k_p1_from_rx0</td>
</tr>
<tr>
<td>3:2</td>
<td>0</td>
<td>RO</td>
<td>stat_lt_k0_from_rx0</td>
</tr>
<tr>
<td>5:4</td>
<td>0</td>
<td>RO</td>
<td>stat_lt_k_m1_from_rx0</td>
</tr>
<tr>
<td>7:6</td>
<td>0</td>
<td>RO</td>
<td>stat_lt_stat_p1_from_rx0</td>
</tr>
<tr>
<td>9:8</td>
<td>0</td>
<td>RO</td>
<td>stat_lt_stat0_from_rx0</td>
</tr>
<tr>
<td>11:10</td>
<td>0</td>
<td>RO</td>
<td>stat_lt_stat_m1_from_rx0</td>
</tr>
</tbody>
</table>

### STAT_RX_VALID_CTRL_CODE: 0494

**Table 2-105: STAT_RX_VALID_CTRL_CODE: 0494**

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>RO LH</td>
<td>stat_rx_valid_ctrl_code</td>
</tr>
</tbody>
</table>

### Statistics Counters

**Table 2-106** through **Table 2-245** define the bit assignments for the statistics counters.

Counters are 48 bits and require two 32-bit address spaces containing the MSB and LSB as indicated. The default value of all counters is 0. Counters are cleared when read by `tick_reg` (or `pm_tick` if so selected) but the register containing the count retains its value. Each counter saturates at FFFFFFFF (hex).

A description of each signal is found in Port Descriptions.
## STATUS_CYCLE_COUNT_LSB: 0500

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:0</td>
<td>0</td>
<td>RO HIST</td>
<td>stat_cycle_count[31:0]</td>
</tr>
</tbody>
</table>

## STATUS_CYCLE_COUNT_MSB: 0504

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>15:0</td>
<td>0</td>
<td>RO HIST</td>
<td>stat_cycle_count[47:32]</td>
</tr>
</tbody>
</table>

## STAT_RX_FRAMING_ERR_LSB: 0648

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_rx_framing_err_count[31:0]</td>
</tr>
</tbody>
</table>

## STAT_RX_FRAMING_ERR_MSB: 064C

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>15:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_rx_framing_err_count[48-1:32]</td>
</tr>
</tbody>
</table>

## STAT_RX_BAD_CODE_LSB: 0660

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_rx_bad_code_count[31:0]</td>
</tr>
</tbody>
</table>

## STAT_RX_BAD_CODE_MSB: 0664

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>15:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_rx_bad_code_count[47:32]</td>
</tr>
</tbody>
</table>
STAT_RX_ERROR_LSB: 0668

Table 2-112: STAT_RX_ERROR_LSB: 0668

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_rx_error_count[31:0]</td>
</tr>
</tbody>
</table>

STAT_RX_ERROR_MSB: 066C

Table 2-113: STAT_RX_ERROR_MSB: 066C

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>15:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_rx_error_count[47:32]</td>
</tr>
</tbody>
</table>

STAT_RX_RSFEC_CORRECTED_CW_INC_LSB: 0670

Table 2-114: STAT_RX_RSFEC_CORRECTED_CW_INC_LSB: 0670

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_rx_rsfec_corrected_cw_inc_count[31:0]</td>
</tr>
</tbody>
</table>

STAT_RX_RSFEC_CORRECTED_CW_INC_MSB: 0674

Table 2-115: STAT_RX_RSFEC_CORRECTED_CW_INC_MSB: 0674

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>15:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_rx_rsfec_corrected_cw_inc_count[47:32]</td>
</tr>
</tbody>
</table>

STAT_RX_RSFEC_UNCORRECTED_CW_INC_LSB: 0678

Table 2-116: STAT_RX_RSFEC_UNCORRECTED_CW_INC_LSB: 0678

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_rx_rsfec_uncorrected_cw_inc_count[31:0]</td>
</tr>
</tbody>
</table>

STAT_RX_RSFEC_UNCORRECTED_CW_INC_MSB: 067C

Table 2-117: STAT_RX_RSFEC_UNCORRECTED_CW_INC_MSB: 067C

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>15:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_rx_rsfec_uncorrected_cw_inc_count[47:32]</td>
</tr>
</tbody>
</table>
### STAT_RX_RSFEC_ERR_COUNT0_INC_LSB: 0680

Table 2-118: STAT_RX_RSFEC_ERR_COUNT0_INC_LSB: 0680

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_rx_rsfec_err_count_inc_count[31:0]</td>
</tr>
</tbody>
</table>

### STAT_RX_RSFEC_ERR_COUNT0_INC_MSB: 0684

Table 2-119: STAT_RX_RSFEC_ERR_COUNT0_INC_MSB: 0684

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>15:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_rx_rsfec_err_count_inc_count[47:32]</td>
</tr>
</tbody>
</table>

### STAT_TX_FRAME_ERROR_LSB: 06A0

Table 2-120: STAT_TX_FRAME_ERROR_LSB: 06A0

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_tx_frame_error_count[31:0]</td>
</tr>
</tbody>
</table>

### STAT_TX_FRAME_ERROR_MSB: 06A4

Table 2-121: STAT_TX_FRAME_ERROR_MSB: 06A4

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>15:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_tx_frame_error_count[47:32]</td>
</tr>
</tbody>
</table>

### STAT_TX_TOTAL_PACKETS_LSB: 0700

Table 2-122: STAT_TX_TOTAL_PACKETS_LSB: 0700

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_tx_total_packets_count[31:0]</td>
</tr>
</tbody>
</table>

### STAT_TX_TOTAL_PACKETS_MSB: 0704

Table 2-123: STAT_TX_TOTAL_PACKETS_MSB: 0704

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>15:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_tx_total_packets_count[47:32]</td>
</tr>
</tbody>
</table>
### STAT_TX_TOTAL_GOOD_PACKETS_LSB: 0708

**Table 2-124:** \texttt{STAT_TX_TOTAL_GOOD_PACKETS_LSB: 0708}

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:0</td>
<td>0</td>
<td>HIST</td>
<td>stat tx total good packets count[31:0]</td>
</tr>
</tbody>
</table>

### STAT_TX_TOTAL_GOOD_PACKETS_MSB: 070C

**Table 2-125:** \texttt{STAT_TX_TOTAL_GOOD_PACKETS_MSB: 070C}

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>15:0</td>
<td>0</td>
<td>HIST</td>
<td>stat tx total good packets count[47:32]</td>
</tr>
</tbody>
</table>

### STAT_TX_TOTAL_BYTES_LSB: 0710

**Table 2-126:** \texttt{STAT_TX_TOTAL_BYTES_LSB: 0710}

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:0</td>
<td>0</td>
<td>HIST</td>
<td>stat tx total bytes count[31:0]</td>
</tr>
</tbody>
</table>

### STAT_TX_TOTAL_BYTES_MSB: 0714

**Table 2-127:** \texttt{STAT_TX_TOTAL_BYTES_MSB: 0714}

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>15:0</td>
<td>0</td>
<td>HIST</td>
<td>stat tx total bytes count[47:32]</td>
</tr>
</tbody>
</table>

### STAT_TX_TOTAL_GOOD_BYTES_LSB: 0718

**Table 2-128:** \texttt{STAT_TX_TOTAL_GOOD_BYTES_LSB: 0718}

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:0</td>
<td>0</td>
<td>HIST</td>
<td>stat tx total good bytes count[31:0]</td>
</tr>
</tbody>
</table>

### STAT_TX_TOTAL_GOOD_BYTES_MSB: 071C

**Table 2-129:** \texttt{STAT_TX_TOTAL_GOOD_BYTES_MSB: 071C}

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>15:0</td>
<td>0</td>
<td>HIST</td>
<td>stat tx total good bytes count[47:32]</td>
</tr>
</tbody>
</table>
### STAT_TX_PACKET_64_BYTES_LSB: 0720

**Table 2-130: STAT_TX_PACKET_64_BYTES_LSB: 0720**

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_tx_packet_64_bytes_count[31:0]</td>
</tr>
</tbody>
</table>

### STAT_TX_PACKET_64_BYTES_MSB: 0724

**Table 2-131: STAT_TX_PACKET_64_BYTES_MSB: 0724**

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>15:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_tx_packet_64_bytes_count[47:32]</td>
</tr>
</tbody>
</table>

### STAT_TX_PACKET_65_127_BYTES_LSB: 0728

**Table 2-132: STAT_TX_PACKET_65_127_BYTES_LSB: 0728**

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_tx_packet_65_127_bytes_count[31:0]</td>
</tr>
</tbody>
</table>

### STAT_TX_PACKET_65_127_BYTES_MSB: 072C

**Table 2-133: STAT_TX_PACKET_65_127_BYTES_MSB: 072C**

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>15:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_tx_packet_65_127_bytes_count[47:32]</td>
</tr>
</tbody>
</table>

### STAT_TX_PACKET_128_255_BYTES_LSB: 0730

**Table 2-134: STAT_TX_PACKET_128_255_BYTES_LSB: 0730**

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_tx_packet_128_255_bytes_count[31:0]</td>
</tr>
</tbody>
</table>

### STAT_TX_PACKET_128_255_BYTES_MSB: 0734

**Table 2-135: STAT_TX_PACKET_128_255_BYTES_MSB: 0734**

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>15:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_tx_packet_128_255_bytes_count[47:32]</td>
</tr>
</tbody>
</table>
### STAT_TX_PACKET_256_511_BYTES_LSB: 0738

**Table 2-136:** STAT_TX_PACKET_256_511_BYTES_LSB: 0738

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_tx_packet_256_511_bytes_count[31:0]</td>
</tr>
</tbody>
</table>

### STAT_TX_PACKET_256_511_BYTES_MSB: 073C

**Table 2-137:** STAT_TX_PACKET_256_511_BYTES_MSB: 073C

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>15:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_tx_packet_256_511_bytes_count[47:32]</td>
</tr>
</tbody>
</table>

### STAT_TX_PACKET_512_1023_BYTES_LSB: 0740

**Table 2-138:** STAT_TX_PACKET_512_1023_BYTES_LSB: 0740

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_tx_packet_512_1023_bytes_count[31:0]</td>
</tr>
</tbody>
</table>

### STAT_TX_PACKET_512_1023_BYTES_MSB: 0744

**Table 2-139:** STAT_TX_PACKET_512_1023_BYTES_MSB: 0744

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>15:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_tx_packet_512_1023_bytes_count[47:32]</td>
</tr>
</tbody>
</table>

### STAT_TX_PACKET_1024_1518_BYTES_LSB: 0748

**Table 2-140:** STAT_TX_PACKET_1024_1518_BYTES_LSB: 0748

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_tx_packet_1024_1518_bytes_count[31:0]</td>
</tr>
</tbody>
</table>

### STAT_TX_PACKET_1024_1518_BYTES_MSB: 074C

**Table 2-141:** STAT_TX_PACKET_1024_1518_BYTES_MSB: 074C

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>15:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_tx_packet_1024_1518_bytes_count[47:32]</td>
</tr>
</tbody>
</table>
### STAT_TX_PACKET_1519_1522_BYTES_LSB: 0750

**Table 2-142: STAT_TX_PACKET_1519_1522_BYTES_LSB: 0750**

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_tx_packet_1519_1522_bytes_count[31:0]</td>
</tr>
</tbody>
</table>

### STAT_TX_PACKET_1519_1522_BYTES_MSB: 0754

**Table 2-143: STAT_TX_PACKET_1519_1522_BYTES_MSB: 0754**

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>15:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_tx_packet_1519_1522_bytes_count[47:32]</td>
</tr>
</tbody>
</table>

### STAT_TX_PACKET_1523_1548_BYTES_LSB: 0758

**Table 2-144: STAT_TX_PACKET_1523_1548_BYTES_LSB: 0758**

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_tx_packet_1523_1548_bytes_count[31:0]</td>
</tr>
</tbody>
</table>

### STAT_TX_PACKET_1523_1548_BYTES_MSB: 075C

**Table 2-145: STAT_TX_PACKET_1523_1548_BYTES_MSB: 075C**

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>15:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_tx_packet_1523_1548_bytes_count[47:32]</td>
</tr>
</tbody>
</table>

### STAT_TX_PACKET_1549_2047_BYTES_LSB: 0760

**Table 2-146: STAT_TX_PACKET_1549_2047_BYTES_LSB: 0760**

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_tx_packet_1549_2047_bytes_count[31:0]</td>
</tr>
</tbody>
</table>

### STAT_TX_PACKET_1549_2047_BYTES_MSB: 0764

**Table 2-147: STAT_TX_PACKET_1549_2047_BYTES_MSB: 0764**

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>15:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_tx_packet_1549_2047_bytes_count[47:32]</td>
</tr>
</tbody>
</table>
### STAT_TX_PACKET_2048_4095_BYTES_LSB: 0768

Table 2-148:  **STAT_TX_PACKET_2048_4095_BYTES_LSB: 0768**

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_tx_packet_2048_4095_bytes_count[31:0]</td>
</tr>
</tbody>
</table>

### STAT_TX_PACKET_2048_4095_BYTES_MSB: 076C

Table 2-149:  **STAT_TX_PACKET_2048_4095_BYTES_MSB: 076C**

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>15:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_tx_packet_2048_4095_bytes_count[47:32]</td>
</tr>
</tbody>
</table>

### STAT_TX_PACKET_4096_8191_BYTES_LSB: 0770

Table 2-150:  **STAT_TX_PACKET_4096_8191_BYTES_LSB: 0770**

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_tx_packet_4096_8191_bytes_count[31:0]</td>
</tr>
</tbody>
</table>

### STAT_TX_PACKET_4096_8191_BYTES_MSB: 0774

Table 2-151:  **STAT_TX_PACKET_4096_8191_BYTES_MSB: 0774**

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>15:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_tx_packet_4096_8191_bytes_count[47:32]</td>
</tr>
</tbody>
</table>

### STAT_TX_PACKET_8192_9215_BYTES_LSB: 0778

Table 2-152:  **STAT_TX_PACKET_8192_9215_BYTES_LSB: 0778**

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_tx_packet_8192_9215_bytes_count[31:0]</td>
</tr>
</tbody>
</table>

### STAT_TX_PACKET_8192_9215_BYTES_MSB: 077C

Table 2-153:  **STAT_TX_PACKET_8192_9215_BYTES_MSB: 077C**

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>15:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_tx_packet_8192_9215_bytes_count[47:32]</td>
</tr>
</tbody>
</table>
**Chapter 2: Product Specification**

**STAT_TX_PACKET_LARGE_LSB: 0780**

Table 2-154: STAT_TX_PACKET_LARGE_LSB: 0780

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_tx_packet_large_count[31:0]</td>
</tr>
</tbody>
</table>

**STAT_TX_PACKET_LARGE_MSB: 0784**

Table 2-155: STAT_TX_PACKET_LARGE_MSB: 0784

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>15:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_tx_packet_large_count[47:32]</td>
</tr>
</tbody>
</table>

**STAT_TX_PACKET_SMALL_LSB: 0788**

Table 2-156: STAT_TX_PACKET_SMALL_LSB: 0788

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_tx_packet_small_count[31:0]</td>
</tr>
</tbody>
</table>

**STAT_TX_PACKET_SMALL_MSB: 078C**

Table 2-157: STAT_TX_PACKET_SMALL_MSB: 078C

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>15:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_tx_packet_small_count[47:32]</td>
</tr>
</tbody>
</table>

**STAT_TX_BAD_FCS_LSB: 07B8**

Table 2-158: STAT_TX_BAD_FCS_LSB: 07B8

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_tx_bad_fcs_count[31:0]</td>
</tr>
</tbody>
</table>

**STAT_TX_BAD_FCS_MSB: 07BC**

Table 2-159: STAT_TX_BAD_FCS_MSB: 07BC

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>15:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_tx_bad_fcs_count[47:32]</td>
</tr>
</tbody>
</table>
### STAT_TX_UNICAST_LSB: 07D0

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_tx_unicast_count[31:0]</td>
</tr>
</tbody>
</table>

### STAT_TX_UNICAST_MSB: 07D4

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>15:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_tx_unicast_count[47:32]</td>
</tr>
</tbody>
</table>

### STAT_TX_MULTICAST_LSB: 07D8

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_tx_multicast_count[31:0]</td>
</tr>
</tbody>
</table>

### STAT_TX_MULTICAST_MSB: 07DC

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>15:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_tx_multicast_count[47:32]</td>
</tr>
</tbody>
</table>

### STAT_TX_BROADCAST_LSB: 07E0

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_tx_broadcast_count[31:0]</td>
</tr>
</tbody>
</table>

### STAT_TX_BROADCAST_MSB: 07E4

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>15:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_tx_broadcast_count[47:32]</td>
</tr>
</tbody>
</table>
### STAT_TX_VLAN_LSB: 07E8

**Table 2-166:** STAT_TX_VLAN_LSB: 07E8

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_tx_vlan_count[31:0]</td>
</tr>
</tbody>
</table>

### STAT_TX_VLAN_MSB: 07EC

**Table 2-167:** STAT_TX_VLAN_MSB: 07EC

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>15:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_tx_vlan_count[47:32]</td>
</tr>
</tbody>
</table>

### STAT_TX_PAUSE_LSB: 07F0

**Table 2-168:** STAT_TX_PAUSE_LSB: 07F0

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_tx_pause_count[31:0]</td>
</tr>
</tbody>
</table>

### STAT_TX_PAUSE_MSB: 07F4

**Table 2-169:** STAT_TX_PAUSE_MSB: 07F4

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>15:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_tx_pause_count[47:32]</td>
</tr>
</tbody>
</table>

### STAT_TX_USER_PAUSE_LSB: 07F8

**Table 2-170:** STAT_TX_USER_PAUSE_LSB: 07F8

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_tx_user_pause_count[31:0]</td>
</tr>
</tbody>
</table>

### STAT_TX_USER_PAUSE_MSB: 07FC

**Table 2-171:** STAT_TX_USER_PAUSE_MSB: 07FC

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>15:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_tx_user_pause_count[47:32]</td>
</tr>
</tbody>
</table>

---

**10G/25G High Speed Ethernet v2.0**

PG210 November 30, 2016  
[www.xilinx.com](http://www.xilinx.com)
### STAT_RX_TOTAL_PACKETS_LSB: 0808

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_rx_total_packets_count[31:0]</td>
</tr>
</tbody>
</table>

### STAT_RX_TOTAL_PACKETS_MSB: 080C

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>15:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_rx_total_packets_count[47:32]</td>
</tr>
</tbody>
</table>

### STAT_RX_TOTAL_GOOD_PACKETS_LSB: 0810

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_rx_total_good_packets_count[31:0]</td>
</tr>
</tbody>
</table>

### STAT_RX_TOTAL_GOOD_PACKETS_MSB: 0814

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>15:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_rx_total_good_packets_count[47:32]</td>
</tr>
</tbody>
</table>

### STAT_RX_TOTAL_BYTES_LSB: 0818

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_rx_total_bytes_count[31:0]</td>
</tr>
</tbody>
</table>

### STAT_RX_TOTAL_BYTES_MSB: 081C

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>15:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_rx_total_bytes_count[47:32]</td>
</tr>
</tbody>
</table>

---
STAT_RX_TOTALGOODBYTES_LSB: 0820

Table 2-178:  STAT_RX_TOTALGOODBYTES_LSB: 0820

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_rx_total_good_bytes_count[31:0]</td>
</tr>
</tbody>
</table>

STAT_RX_TOTALGOODBYTES_MSB: 0824

Table 2-179:  STAT_RX_TOTALGOODBYTES_MSB: 0824

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>15:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_rx_total_good_bytes_count[47:32]</td>
</tr>
</tbody>
</table>

STAT_RX_PACKET64BYTES_LSB: 0828

Table 2-180:  STAT_RX_PACKET64BYTES_LSB: 0828

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_rx_packet_64_bytes_count[31:0]</td>
</tr>
</tbody>
</table>

STAT_RX_PACKET64BYTES_MSB: 082C

Table 2-181:  STAT_RX_PACKET64BYTES_MSB: 082C

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>15:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_rx_packet_64_bytes_count[47:32]</td>
</tr>
</tbody>
</table>

STAT_RX_PACKET65_127BYTES_LSB: 0830

Table 2-182:  STAT_RX_PACKET65_127BYTES_LSB: 0830

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_rx_packet_65_127_bytes_count[31:0]</td>
</tr>
</tbody>
</table>

STAT_RX_PACKET65_127BYTES_MSB: 0834

Table 2-183:  STAT_RX_PACKET65_127BYTES_MSB: 0834

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>15:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_rx_packet_65_127_bytes_count[47:32]</td>
</tr>
</tbody>
</table>
### STAT_RX_PACKET_128_255_BYTES_LSB: 0838

**Table 2-184: STAT_RX_PACKET_128_255_BYTES_LSB: 0838**

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_rx_packet_128_255_bytes_count[31:0]</td>
</tr>
</tbody>
</table>

### STAT_RX_PACKET_128_255_BYTES_MSB: 083C

**Table 2-185: STAT_RX_PACKET_128_255_BYTES_MSB: 083C**

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>15:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_rx_packet_128_255_bytes_count[47:32]</td>
</tr>
</tbody>
</table>

### STAT_RX_PACKET_256_511_BYTES_LSB: 0840

**Table 2-186: STAT_RX_PACKET_256_511_BYTES_LSB: 0840**

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_rx_packet_256_511_bytes_count[31:0]</td>
</tr>
</tbody>
</table>

### STAT_RX_PACKET_256_511_BYTES_MSB: 0844

**Table 2-187: STAT_RX_PACKET_256_511_BYTES_MSB: 0844**

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>15:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_rx_packet_256_511_bytes_count[47:32]</td>
</tr>
</tbody>
</table>

### STAT_RX_PACKET_512_1023_BYTES_LSB: 0848

**Table 2-188: STAT_RX_PACKET_512_1023_BYTES_LSB: 0848**

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_rx_packet_512_1023_bytes_count[31:0]</td>
</tr>
</tbody>
</table>

### STAT_RX_PACKET_512_1023_BYTES_MSB: 084C

**Table 2-189: STAT_RX_PACKET_512_1023_BYTES_MSB: 084C**

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>15:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_rx_packet_512_1023_bytes_count[47:32]</td>
</tr>
</tbody>
</table>
### STAT_RX_PACKET_1024_1518_BYTES_LSB: 0850

**Table 2-190: STAT_RX_PACKET_1024_1518_BYTES_LSB: 0850**

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_rx_packet_1024_1518_bytes_count[31:0]</td>
</tr>
</tbody>
</table>

### STAT_RX_PACKET_1024_1518_BYTES_MSB: 0854

**Table 2-191: STAT_RX_PACKET_1024_1518_BYTES_MSB: 0854**

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>15:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_rx_packet_1024_1518_bytes_count[47:32]</td>
</tr>
</tbody>
</table>

### STAT_RX_PACKET_1519_1522_BYTES_LSB: 0858

**Table 2-192: STAT_RX_PACKET_1519_1522_BYTES_LSB: 0858**

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_rx_packet_1519_1522_bytes_count[31:0]</td>
</tr>
</tbody>
</table>

### STAT_RX_PACKET_1519_1522_BYTES_MSB: 085C

**Table 2-193: STAT_RX_PACKET_1519_1522_BYTES_MSB: 085C**

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>15:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_rx_packet_1519_1522_bytes_count[47:32]</td>
</tr>
</tbody>
</table>

### STAT_RX_PACKET_1523_1548_BYTES_LSB: 0860

**Table 2-194: STAT_RX_PACKET_1523_1548_BYTES_LSB: 0860**

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_rx_packet_1523_1548_bytes_count[31:0]</td>
</tr>
</tbody>
</table>

### STAT_RX_PACKET_1523_1548_BYTES_MSB: 0864

**Table 2-195: STAT_RX_PACKET_1523_1548_BYTES_MSB: 0864**

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>15:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_rx_packet_1523_1548_bytes_count[47:32]</td>
</tr>
</tbody>
</table>
### STAT_RX_PACKET_1549_2047_BYTES_LSB: 0868

#### Table 2-196: STAT_RX_PACKET_1549_2047_BYTES_LSB: 0868

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_rx_packet_1549_2047_bytes_count[31:0]</td>
</tr>
</tbody>
</table>

### STAT_RX_PACKET_1549_2047_BYTES_MSB: 086C

#### Table 2-197: STAT_RX_PACKET_1549_2047_BYTES_MSB: 086C

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>15:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_rx_packet_1549_2047_bytes_count[47:32]</td>
</tr>
</tbody>
</table>

### STAT_RX_PACKET_2048_4095_BYTES_LSB: 0870

#### Table 2-198: STAT_RX_PACKET_2048_4095_BYTES_LSB: 0870

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_rx_packet_2048_4095_bytes_count[31:0]</td>
</tr>
</tbody>
</table>

### STAT_RX_PACKET_2048_4095_BYTES_MSB: 0874

#### Table 2-199: STAT_RX_PACKET_2048_4095_BYTES_MSB: 0874

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>15:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_rx_packet_2048_4095_bytes_count[47:32]</td>
</tr>
</tbody>
</table>

### STAT_RX_PACKET_4096_8191_BYTES_LSB: 0878

#### Table 2-200: STAT_RX_PACKET_4096_8191_BYTES_LSB: 0878

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_rx_packet_4096_8191_bytes_count[31:0]</td>
</tr>
</tbody>
</table>

### STAT_RX_PACKET_4096_8191_BYTES_MSB: 087C

#### Table 2-201: STAT_RX_PACKET_4096_8191_BYTES_MSB: 087C

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>15:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_rx_packet_4096_8191_bytes_count[47:32]</td>
</tr>
</tbody>
</table>

10G/25G High Speed Ethernet v2.0
PG210 November 30, 2016
www.xilinx.com
### STAT_RX_PACKET_8192_9215_BYTES_LSB: 0880

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_rx_packet_8192_9215_bytes_count[31:0]</td>
</tr>
</tbody>
</table>

### STAT_RX_PACKET_8192_9215_BYTES_MSB: 0884

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>15:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_rx_packet_8192_9215_bytes_count[47:32]</td>
</tr>
</tbody>
</table>

### STAT_RX_PACKET_LARGE_LSB: 0888

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_rx_packet_large_count[31:0]</td>
</tr>
</tbody>
</table>

### STAT_RX_PACKET_LARGE_MSB: 088C

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>15:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_rx_packet_large_count[47:32]</td>
</tr>
</tbody>
</table>

### STAT_RX_PACKET_SMALL_LSB: 0890

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_rx_packet_small_count[31:0]</td>
</tr>
</tbody>
</table>

### STAT_RX_PACKET_SMALL_MSB: 0894

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>15:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_rx_packet_small_count[47:32]</td>
</tr>
</tbody>
</table>
### STAT_RX_UNDERSIZE_LSB: 0898

**Table 2-208: STAT_RX_UNDERSIZE_LSB: 0898**

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_rx_undersize_count[31:0]</td>
</tr>
</tbody>
</table>

### STAT_RX_UNDERSIZE_MSB: 089C

**Table 2-209: STAT_RX_UNDERSIZE_MSB: 089C**

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>15:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_rx_undersize_count[47:32]</td>
</tr>
</tbody>
</table>

### STAT_RX_FRAGMENT_LSB: 08A0

**Table 2-210: STAT_RX_FRAGMENT_LSB: 08A0**

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_rx_fragment_count[31:0]</td>
</tr>
</tbody>
</table>

### STAT_RX_FRAGMENT_MSB: 08A4

**Table 2-211: STAT_RX_FRAGMENT_MSB: 08A4**

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>15:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_rx_fragment_count[47:32]</td>
</tr>
</tbody>
</table>

### STAT_RX_OVERSIZE_LSB: 08A8

**Table 2-212: STAT_RX_OVERSIZE_LSB: 08A8**

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_rx_oversize_count[31:0]</td>
</tr>
</tbody>
</table>

### STAT_RX_OVERSIZE_MSB: 08AC

**Table 2-213: STAT_RX_OVERSIZE_MSB: 08AC**

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>15:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_rx_oversize_count[47:32]</td>
</tr>
</tbody>
</table>
## STAT_RX_TOOLONG_LSB: 08B0

### Table 2-214: STAT_RX_TOOLONG_LSB: 08B0

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_rx_toolong_count[31:0]</td>
</tr>
</tbody>
</table>

## STAT_RX_TOOLONG_MSB: 08B4

### Table 2-215: STAT_RX_TOOLONG_MSB: 08B4

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>15:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_rx_toolong_count[47:32]</td>
</tr>
</tbody>
</table>

## STAT_RX_JABBER_LSB: 08B8

### Table 2-216: STAT_RX_JABBER_LSB: 08B8

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_rx_jabber_count[31:0]</td>
</tr>
</tbody>
</table>

## STAT_RX_JABBER_MSB: 08BC

### Table 2-217: STAT_RX_JABBER_MSB: 08BC

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>15:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_rx_jabber_count[47:32]</td>
</tr>
</tbody>
</table>

## STAT_RX_BAD_FCS_LSB: 08C0

### Table 2-218: STAT_RX_BAD_FCS_LSB: 08C0

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_rx_bad_fcs_count[31:0]</td>
</tr>
</tbody>
</table>

## STAT_RX_BAD_FCS_MSB: 08C4

### Table 2-219: STAT_RX_BAD_FCS_MSB: 08C4

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>15:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_rx_bad_fcs_count[47:32]</td>
</tr>
</tbody>
</table>

---
### STAT_RX_PACKET_BAD_FCS_LSB: 08C8

Table 2-220: STAT_RX_PACKET_BAD_FCS_LSB: 08C8

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_rx_packet_bad_fcs_count[31:0]</td>
</tr>
</tbody>
</table>

### STAT_RX_PACKET_BAD_FCS_MSB: 08CC

Table 2-221: STAT_RX_PACKET_BAD_FCS_MSB: 08CC

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>15:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_rx_packet_bad_fcs_count[47:32]</td>
</tr>
</tbody>
</table>

### STAT_RX_STOMPED_FCS_LSB: 08D0

Table 2-222: STAT_RX_STOMPED_FCS_LSB: 08D0

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_rx_stomped_fcs_count[31:0]</td>
</tr>
</tbody>
</table>

### STAT_RX_STOMPED_FCS_MSB: 08D4

Table 2-223: STAT_RX_STOMPED_FCS_MSB: 08D4

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>15:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_rx_stomped_fcs_count[47:32]</td>
</tr>
</tbody>
</table>

### STAT_RX_UNICAST_LSB: 08D8

Table 2-224: STAT_RX_UNICAST_LSB: 08D8

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_rx_unicast_count[31:0]</td>
</tr>
</tbody>
</table>

### STAT_RX_UNICAST_MSB: 08DC

Table 2-225: STAT_RX_UNICAST_MSB: 08DC

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>15:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_rx_unicast_count[47:32]</td>
</tr>
</tbody>
</table>
### STAT_RX_MULTICAST_LSB: 08E0

**Table 2-226:** STAT_RX_MULTICAST_LSB: 08E0

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_rx_multicast_count[31:0]</td>
</tr>
</tbody>
</table>

### STAT_RX_MULTICAST_MSB: 08E4

**Table 2-227:** STAT_RX_MULTICAST_MSB: 08E4

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>15:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_rx_multicast_count[47:32]</td>
</tr>
</tbody>
</table>

### STAT_RX_BROADCAST_LSB: 08E8

**Table 2-228:** STAT_RX_BROADCAST_LSB: 08E8

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_rx_broadcast_count[31:0]</td>
</tr>
</tbody>
</table>

### STAT_RX_BROADCAST_MSB: 08EC

**Table 2-229:** STAT_RX_BROADCAST_MSB: 08EC

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>15:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_rx_broadcast_count[47:32]</td>
</tr>
</tbody>
</table>

### STAT_RX_VLAN_LSB: 08F0

**Table 2-230:** STAT_RX_VLAN_LSB: 08F0

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_rx_vlan_count[31:0]</td>
</tr>
</tbody>
</table>

### STAT_RX_VLAN_MSB: 08F4

**Table 2-231:** STAT_RX_VLAN_MSB: 08F4

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>15:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_rx_vlan_count[47:32]</td>
</tr>
</tbody>
</table>
### STAT_RX_PAUSE_LSB: 08F8

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_rx_pause_count[31:0]</td>
</tr>
</tbody>
</table>

### STAT_RX_PAUSE_MSB: 08FC

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>15:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_rx_pause_count[47:32]</td>
</tr>
</tbody>
</table>

### STAT_RX_USER_PAUSE_LSB: 0900

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_rx_user_pause_count[31:0]</td>
</tr>
</tbody>
</table>

### STAT_RX_USER_PAUSE_MSB: 0904

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>15:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_rx_user_pause_count[47:32]</td>
</tr>
</tbody>
</table>

### STAT_RX_INRANGEERR_LSB: 0908

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_rx_inrangeerr_count[31:0]</td>
</tr>
</tbody>
</table>

### STAT_RX_INRANGEERR_MSB: 090C

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>15:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_rx_inrangeerr_count[47:32]</td>
</tr>
</tbody>
</table>
### STAT_RX_TRUNCATED_LSB: 0910

**Table 2-238:** STAT_RX_TRUNCATED_LSB: 0910

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_rx_truncated_count[31:0]</td>
</tr>
</tbody>
</table>

### STAT_RX_TRUNCATED_MSB: 0914

**Table 2-239:** STAT_RX_TRUNCATED_MSB: 0914

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>15:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_rx_truncated_count[47:32]</td>
</tr>
</tbody>
</table>

### STAT_RX_TEST_PATTERN_MISMATCH_LSB: 0918

**Table 2-240:** STAT_RX_TEST_PATTERN_MISMATCH_LSB: 0918

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_rx_test_pattern_mismatch_count[31:0]</td>
</tr>
</tbody>
</table>

### STAT_RX_TEST_PATTERN_MISMATCH_MSB: 091C

**Table 2-241:** STAT_RX_TEST_PATTERN_MISMATCH_MSB: 091C

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>15:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_rx_test_pattern_mismatch_count[47:32]</td>
</tr>
</tbody>
</table>

### STAT_FEC_INC_CORRECT_COUNT_LSB: 0920

**Table 2-242:** STAT_FEC_INC_CORRECT_COUNT_LSB: 0920

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_fec_inc_correct_count_count[31:0]</td>
</tr>
</tbody>
</table>

### STAT_FEC_INC_CORRECT_COUNT_MSB: 0924

**Table 2-243:** STAT_FEC_INC_CORRECT_COUNT_MSB: 0924

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>15:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_fec_inc_correct_count_count[47:32]</td>
</tr>
</tbody>
</table>
### STAT_FEC_INC_CANT_CORRECT_COUNT_LSB: 0928

Table 2-244: STAT_FEC_INC_CANT_CORRECT_COUNT_LSB: 0928

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_fec_inc_cant_correct_count_count[31:0]</td>
</tr>
</tbody>
</table>

### STAT_FEC_INC_CANT_CORRECT_COUNT_MSB: 092C

Table 2-245: STAT_FEC_INC_CANT_CORRECT_COUNT_MSB: 092C

<table>
<thead>
<tr>
<th>Bits</th>
<th>Default</th>
<th>Type</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>15:0</td>
<td>0</td>
<td>HIST</td>
<td>stat_fec_inc_cant_correct_count_count[47:32]</td>
</tr>
</tbody>
</table>
Chapter 3

Designing with the Subsystem

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

Clocking

This section describes the clocking for all the 10G/25G configurations at the component support wrapper layer. There are three fundamentally different clocking architectures depending on the functionality and options:

- PCS/PMA Only Clocking
- 10G/25G MAC with PCS/PMA Clocking
- Low Latency 10G/25G MAC with PCS/PMA Clocking

Also described is Auto-Negotiation and Link Training Clocking.
**PCS/PMA Only Clocking**

The clocking architecture for the 10G/25G PCS is illustrated below. There are three clock domains in the datapath, as illustrated by the dashed lines in Figure 3-1.

**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 390.625 MHz for 25G and 156.25 MHz for 10G.

**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 XGMII/XXVGMII data bus. This clock and the RX XGMII/XXVGMII bus must be within 100 ppm of the required frequency, which is 390.625 MHz for 25G and 156.25 MHz for 10G.

**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.

**10G/25G MAC with PCS/PMA Clocking**

The clocking architecture for the 10/25G 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.

![Figure 3-2: 10G/25G MAC with PCS/PMA Clocking](image-url)
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 390.625 MHz for 25G and 156.25 MHz for 10G.

tax_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 is the input clk for RX core. This rx_clk is available as rx_core_clk to you, which you must drive from the example design. This should be driven by tx_clk_out when FIFO is enabled. 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 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 10G/25G MAC with PCS/PMA Clocking

The clocking architecture for the Low Latency 10/25G 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.
Chapter 3: Designing with the Subsystem

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 390.625 MHz for 25G, and 156.25 MHz for 10G.

**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 BASE-KR 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.

**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.
**Chapter 3: Designing with the Subsystem**

**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 dclk 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.

---

**Resets**

*Figure 3-5* shows the reset structure for the 10G/25G Ethernet MAC with PCS/PMA as implemented at the component support wrapper layer. Clocks are not shown for clarity.

---

**Component Support Layer Resets**

In the example design, a single reset is used to reset the entire wrapper layer. Using the external stimulus fsm_reset, the example_fsm block issues the signal sys_reset which is connected to the three _wrapper resets. Therefore, the example design demonstrates that all three wrapper resets can be released simultaneously and correct operation follows.
Wrapper Resets

The _wrapper layer of the hierarchy is assumed to be what you instantiate in your own design. There are three resets to be handled as follows:

- GT_reset
- tx_reset
- rx_reset

You do not need to be concerned with timing the reset signals; this is taken care of by the reset_sequencer block.

**GT_reset**

The GT_reset is the asynchronous active-High reset input to the GT. You do not need to be concerned with the internal resets of the GT because this is taken care of by the GT helper blocks.

**tx_reset**

The tx_reset is the asynchronous active-High reset for the TX path logic of the 10G/25G Ethernet IP core. While it is connected to the GT reset in the example design, this reset can be asserted at any time to reset the TX path independently without disturbing the RX path.

**rx_reset**

The rx_reset is the asynchronous active-High reset for the RX path logic of the 10G/25G Ethernet IP core. While it is connected to the GT reset in the example design, this reset can be asserted at any time to reset the RX path independently without disturbing the TX path.
Figure 3-6: Detailed Diagram of Single Core - Synchronous Clock Mode
Figure 3-6 through Figure 3-9, illustrate the clocking and reset structure when you implement the Example Design using the Vivado® design tools.

Figure 3-7: Detailed Diagram of Single Core - Asynchronous Clock Mode
Figure 3-8: Detailed Diagram of Multiple Cores - Synchronous Clock Mode
Figure 3-9: Detailed Diagram of Multiple Cores - Asynchronous Clock Mode
Support for IEEE Standard 1588v2

Overview

This section details the packet timestamping function of the 10G/25G 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 provides one-step and two-step IEEE 1588v2 functionality.

If you select the IEEE PTP 1588v2 operation mode as "Two Step", then only the two-step related ports will be populated and the core will perform only the two-step time stamping functionality. If you select the IEEE PTP 1588v2 operation mode as "One Step", then all the ports related to one step and two step will be populated and both one step and two step core functionality will be available.

Ethernet frames are timestamped at both ingress and egress. The option can be used for implementing all kinds of IEEE 1588v2 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 section assumes that you are also implementing the IEEE 1588v2 Precision Time Protocol (PTP).

IEEE 1588v2 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, this is known as one-step operation. Alternatively, the timestamp value can be placed into a follow up packet; this is known as two-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 scope of this document. This document now describes the 1588 hardware timestamping features of the subsystem.

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.
Chapter 3: Designing with the Subsystem

- Correction Field format: IEEE 1588-2008 numerical format consisting of a 64-bit signed field representing nanoseconds multiplied by $2^{16}$ (see IEEE 1588 clause 13.3.2.7). This timer should count from 0 through the full range up to $2^{64}-1$ before wrapping around.

**Egress**

As seen in the preceding figure, timestamping logic exists in two locations depending on whether 1-step or 2-step operation is desired. 1-step operation requires the 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]`. 

**Figure 3-10: Egress**

Note: some core logic and lane logic blocks are not shown for clarity.
Based on rate and clock mode (Ordinary or Transparent), the suggested default values of the `ctl_tx_ptp_latency_adjust[10:0]` signal are as follows:

- 25G Ordinary Clock = 395
- 25G Transparent Clock = 471
- 10G Ordinary Clock = 970
- 10G Transparent Clock = 1177

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.
Chapter 3: Designing with the Subsystem

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 cyclic redundancy check (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, 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 since both the checksum and the correction field must both be fully contained in the MAC pipeline in order for the checksum to be correctly updated. This particular restriction does not apply to the ToD format because the original timestamp data is calculated as a zero value; consequently the checksum and timestamp position can be independently located within the frame.
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 do not require this function.

Timestamps are filtered after the PCS decoder to retain only those timestamps corresponding to an Start Of Packet (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

<table>
<thead>
<tr>
<th>Local Fault Indication</th>
<th>Direction</th>
<th>Description</th>
<th>Clock Domain</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td><strong>IEEE 1588 Interface – TX Path</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>ctl_tx_systemtimerin[80-1:0]</strong> Input</td>
<td>tx_serdes_clk</td>
</tr>
<tr>
<td></td>
<td></td>
<td>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, 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 SerDes clock domain.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Output</td>
<td><strong>tx_ptp_tstamp_valid_out</strong> This bit indicates that a valid timestamp is being presented on the TX system interface.</td>
<td>tx_clk_out</td>
</tr>
<tr>
<td></td>
<td>Output</td>
<td><strong>tx_ptp_tstamp_tag_out[15:0]</strong> Output Tag output corresponding to tx_ptp_tag_field_in[15:0]</td>
<td>tx_clk_out</td>
</tr>
<tr>
<td></td>
<td>Output</td>
<td><strong>tx_ptp_tstamp_out[80-1:0]</strong> Output Timestamp for the transmitted packet SOP corresponding to the time at which it passed the capture plane. Time format same as timer input.</td>
<td>tx_clk_out</td>
</tr>
<tr>
<td></td>
<td>Input</td>
<td><strong>tx_ptp_1588op_in[1:0]</strong> The signal should be valid on the first cycle of the packet. For PCS cores, the first cycle corresponds with the first data word 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.</td>
<td>tx_clk_out</td>
</tr>
<tr>
<td></td>
<td>Input</td>
<td><strong>ctl_tx_ptp_1step_enable</strong> When set to 1, this bit enables 1-step operation.</td>
<td>tx_clk_out</td>
</tr>
</tbody>
</table>
### Table 3-1: 1588v2 Port List and Descriptions (Cont’d)

<table>
<thead>
<tr>
<th>Local Fault Indication</th>
<th>Direction</th>
<th>Description</th>
<th>Clock Domain</th>
</tr>
</thead>
<tbody>
<tr>
<td>ctl_ptp_transpclk_mode</td>
<td>Input</td>
<td>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. The sign bit of the correction value is assumed to be 0 (positive time). It is expected that the corresponding incoming PTP packet correction field has already been adjusted with the proper RX timestamp.</td>
<td>tx_clk_out</td>
</tr>
</tbody>
</table>
| tx_ptp_tag_field_in[15:0]               | Input     | The usage of this field is dependent on the 1588 operation. The signal should be valid on the first cycle of the packet.  
  * 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_clk_out   |
| ctl_tx_ptp_latency_adjust[10:0]         | Input     | 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. | tx_clk_out   |
| stat_tx_ptp_fifo_write_error            | Output    | 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.                                                 | tx_clk_out   |
| stat_tx_ptp_fifo_read_error             | Output    | 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.                                                                 | tx_clk_out   |
### 1588v2 Port List and Descriptions (Cont'd)

<table>
<thead>
<tr>
<th>Local Fault Indication</th>
<th>Direction</th>
<th>Description</th>
<th>Clock Domain</th>
</tr>
</thead>
</table>
| tx_ptp_rxtstamp_in     | Input     | The usage of this field is dependent on the 1588 operation.  
• 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. For PCS cores, the first cycle corresponds to the first data word of the packet. | tx_clk_out   |

#### IEEE 1588 Interface – RX Path

<table>
<thead>
<tr>
<th>ctl_rx_systemtimerin[80-1:0]</th>
<th>Input</th>
<th>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.</th>
<th>rx_serdes_clk</th>
</tr>
</thead>
<tbody>
<tr>
<td>rx_ptp_tstamp_out[80-1:0]</td>
<td>Output</td>
<td>Timestamp for the received packet SOP corresponding to the time at which it passed the capture plane. Note that this signal will be valid on the first cycle of the packet.</td>
<td>rx_clk_out</td>
</tr>
</tbody>
</table>
IEEE 1588v2 PTP Functional Description

The IEEE 1588v2 feature of the 10G/25G High Speed Ethernet 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 above. However, it is required that this time source be in the same clock domain as the SerDes. This might require re-timing by an external circuit provided by the user.

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 during the same clock cycle as 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 10G/25G High Speed Ethernet subsystem. This is illustrated in the following schematic diagrams:

![Diagram](image-url)
### Performance

The following table shows the accuracy of the timestamp capture for a typical 10G/25G Ethernet subsystem. Refer to the preceding figures for the definition of timing parameters.

The following table presents typical test results for "t". The values include the delay through a SerDes. Contact Xilinx for more information for more information and the latest SerDes data.

**Table 3-2: Accuracy of the Timestamp Capture**

<table>
<thead>
<tr>
<th>Ethernet Port</th>
<th>clk (MHz)</th>
<th>t min (nsec)</th>
<th>t max (nsec)</th>
<th>SerDes phase FIFO variation (nsec)</th>
<th>time of day clock crossing jitter (nsec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>25 GE Ingress</td>
<td>390.625</td>
<td>30</td>
<td>31</td>
<td>+/- 1</td>
<td>+/- 2.56</td>
</tr>
<tr>
<td>25 GE Egress</td>
<td>390.625</td>
<td>30</td>
<td>31</td>
<td>+/- 1</td>
<td>+/- 2.56</td>
</tr>
</tbody>
</table>

Note that the SerDes clock is assumed to be 390.625 MHz.

In a typical application, the difference between the ingress and egress capture times is important for determining absolute time. The PTP algorithm can use asymmetry information to improve accuracy.
The 1588v2 feature requires that all clock frequencies be known in order to make internal calculations. The clock frequencies should be specified at the time the PTP IP core is ordered in order for the timestamp correction to work properly.

In a typical application, the PTP algorithm (or servo, not part of this IP) will remove jitter over the course of time (many packet samples). It is advantageous for the jitter to be as small as possible in order to minimize the convergence time as well as minimizing slave clock drift.

---

**RS-FEC Support**

**Overview**

This section describes the optional RS-FEC function of the 10G/25G Ethernet subsystem. The RS-FEC option must be specified at the time of generating the subsystem from the IP catalog or ordering the IP core asynchronously.

The RS-FEC block is positioned between the PCS and PMA as illustrated below.

With reference to the following diagram, the clocks and resets of the RS-FEC core are equivalent to the transceiver signals, with the transceiver resets being active-High.

![RS-FEC Block Diagram](image)
Chapter 3: Designing with the Subsystem

The internal details of the RS-FEC are beyond the scope of this document. Refer to IEEE 802.3 Clause 108 and Schedule 3 of the 25G Ethernet Consortium [Ref 1].

Further information can also be found in the 25G IEEE 802.3by Reed-Solomon Forward Error Correction LogiCORE IP Product Guide (PG217) [Ref 12].

Port Descriptions

Table 3-3: RS-FEC Port List and Descriptions

<table>
<thead>
<tr>
<th>Port</th>
<th>Direction</th>
<th>Description</th>
<th>Clock Domain</th>
</tr>
</thead>
</table>
| ctl_rsfec_ieee_error_indication_mode | Input     | The setting on this bit takes effect after rx_reseth has been asserted Low (~rx_serdes_reset). New value is sampled on first cycle on reset.  
• 1: Core conforms to the IEEE RS-FEC specification.  
• 0: If ctl_rx_rsfec_enable_correction and ctl_rx_rsfec_enable_indication are set to zero, the RS decoder is bypassed.  |              |
| ctl_rsfec_consortium_25g          | Input     | Switches between IEEE Clause 108 and 25G Ethernet Consortium mode.  
The setting on this bit takes effect after rx_reseth has been asserted Low (~rx_serdes_reset). New value is sampled on first cycle on reset.  
• 1 = 25G Consortium specification mode;  
• 0 = IEEE 802.3by mode  
Note that some variants of the 10G/25G Subsystem can have individual RX and TX consortium signals. |              |
| ctl_rsfec_enable                  | Input     | The setting on this bit takes effect after rx_reseth has been asserted Low (~rx_serdes_reset). New value is sampled on first cycle on reset.  
Enable RS-FEC function. Note that some variants of the 10G/25G Subsystem can have individual RX and TX enable signals. |              |
| ctl_rx_rsfec_enable_correction    | Input     | The setting on this bit takes effect after rx_reseth has been asserted Low (~rx_serdes_reset). New value is sampled on first cycle on reset.  
Equivalent to MDIO register 1.200.0  
• 0: Decoder performs error detection without error correction (see IEEE 802.3802.3by section 91.5.3.3).  
• 1: the decoder also performs error correction. |              |
Table 3-3: RS-FEC Port List and Descriptions (Cont’d)

<table>
<thead>
<tr>
<th>Port</th>
<th>Direction</th>
<th>Description</th>
<th>Clock Domain</th>
</tr>
</thead>
</table>
| ctl_rx_rsfec_enable_indication    | Input     | The setting on this bit takes effect after rx_resetn has been asserted Low (~rx_serdes_reset). New value sampled on first cycle on reset. Equivalent to MDIO register 1.200.1  
• 0: Bypass the error indication function (see IEEE Std 802.3by section 91.5.3.3).  
• 1: Decoder indicates errors to the PCS sublayer, |             |
| ctl_rx_vl_length_minus1[15:0]     | Input     | Normally set to 20,479 (4FFF hex). The normal value is equivalent to (16,383 x 5 -4) = 81,916. |             |
| ctl_rx_vl_marker_id0[63:0]        | Input     | Equivalent to the RX PCS lane 0 alignment marker defined in IEEE 802.3 Clause 82 for 40 G Ethernet. |             |
| ctl_rx_vl_marker_id1[63:0]        | Input     | Equivalent to the PCS lane 1 alignment marker. |             |
| ctl_rx_vl_marker_id2[63:0]        | Input     | Equivalent to the PCS lane 2 alignment marker. |             |
| ctl_rx_vl_marker_id3[63:0]        | Input     | Equivalent to the PCS lane 3 alignment marker. |             |
| ctl_tx_vl_length_minus1[15:0]     | Input     | Normally set to 20479 (decimal). The normal value is equivalent to (16,383 x 5 -4) = 81,916. |             |
| ctl_tx_vl_marker_id0[63:0]        | Input     | Equivalent to the TX PCS lane 0 alignment marker defined in IEEE 802.3 Clause 82 for 40 G Ethernet. |             |
| ctl_tx_vl_marker_id1[63:0]        | Input     | Equivalent to the PCS lane 1 alignment marker. |             |
| ctl_tx_vl Marker_id2[63:0]        | Input     | Equivalent to the PCS lane 2 alignment marker. |             |
| ctl_tx_vl Marker_id3[63:0]        | Input     | Equivalent to the PCS lane 3 alignment marker |             |

**RS-FEC Status Signals**

| Stat_Rx_rsfec_corrected_cW_inc    | Output    | Increment for corrected errors. |             |
| Stat_Rx_rsfec_uncorrected_cW_inc  | Output    | Increment for uncorrected errors. |             |
| Stat_Rx_rsfec_error_count_inc[2:0]| Output    | Increment for detected errors. |             |
| Stat_Rx_rsfec_hi_ser              | Output    | Set to one if the number of RS-FEC symbol errors in a window of 8192 codewords exceeds the threshold K = 417 and is set to zero otherwise. |             |
| Stat_Rx_rsfec_lane_alignment_status| Output    | A value of 1 indicates that the RX RS-FEC block has achieved alignment on the data from the transceiver. |             |
| Stat_Tx_rsfec_lane_alignment_status| Output    | A value of 1 indicates that the TX RS-FEC block has achieved alignment on the incoming PCS data. |             |


Chapter 3: Designing with the Subsystem

RS-FEC Functional Description

The RS-FEC feature of the 10G/25G subsystem provides error correction capability according to IEEE 802.3 Clause 108 or Schedule 3 of the 25G Ethernet Consortium.

The feature requires the insertion of PCS alignment markers as defined in IEEE 802.3 Table 82-2. Inputs are provided for the alignment markers and also for the value of words between alignment markers.

It is possible to bypass the RS-FEC function by means of the enable signals. This will bypass the RS-FEC function and connect the PCS directly to the transceiver, with the benefit of reduced latency. Refer to 25G IEEE 802.3 by Reed-Solomon Forward Error Correction LogiCORE IP Product Guide (PG217) [Ref 12] for the latest latency performance data in the various bypass modes, defined as follows:

- **FEC Bypass Correction:** The decoder performs error detection without correction, (see IEEE Std 802.3by section 108.5.3.2. The latency is reduced in this mode (see 25G IEEE 802.3by Reed-Solomon Forward Error Correction LogiCORE IP Product Guide (PG217) [Ref 12] for latency figures).

- **FEC Bypass Indication:** In this mode there is correction of the data but no error indication. An additional signal, rx_hi_ser, is generated in this mode to reduce the likelihood that errors in a packet are not detected. The RS decoder counts the number of symbol errors detected in consecutive non-overlapping blocks of 8192 codewords (see IEEE Std 802.3by section 108.5.3.2. The latency is reduced in this mode.

- **Decoder Bypass:** The RS decoder can be bypassed by setting the IEEE Error indication Low when the correction bypass and indication bypass are High.

Status/Control Interface

The Status/Control interface allows you to set up the 10G/25G Ethernet core configuration and to monitor its status. This sections describes in more detail some of the Status and Control signals.

**stat_rx_framing_err and stat_rx_framing_err_valid**

These signals are used to keep track of sync header errors. This set of buses is used to keep track of sync header errors. The stat_rx_framing_err output 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 is sampled as a 1.
stat_rx_block_lock

This bit indicates that the interface has achieved sync header lock as defined by IEEE Std. 802.3. A value of 1 indicates block lock is achieved.

stat_rx_local_fault

This output is High when stat_rx_internal_local_fault or stat_rx_received_local_fault is asserted. This is output is level sensitive.

RX Error Status

The core 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[1:0]

When this signal is positive, it indicates that the error detection logic has identified mismatches 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 for each CRC32 error detected.

stat_rx_bad_code

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

The 10G/25G 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 section.

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.

Each bit of this bus must be held at a steady state for a minimum of 16 cycles before the next transition.

**CAUTION!** 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 10G/25G Ethernet core automatically calculates and adds the FCS to the packet. For priority pause packets the 10G/25G Ethernet core 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 
\texttt{ctl\_tx\_pause\_req[8:0]} and \texttt{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 10G/25G Ethernet core transmits the pause packet immediately 
after the current packet in flight is completed.

\textbf{IMPORTANT: Each bit of this bus must be held at a steady state for a minimum of 16 cycles before the next transition.}

To retransmit pause packets, the 10G/25G Ethernet core 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 the 
\texttt{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 \texttt{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 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, 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 
\texttt{ctl\_tx\_pause\_req[8:0]} or \texttt{ctl\_tx\_pause\_enable[8:0]} to 0.
RX Pause Termination

The 10G/25G Ethernet core 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.

   Packets that pass step one are forwarded to you only if $ctl_{\text{rx}}_{\text{forward\_control}}$ is set to 1.

2. If step one passes, the packet is checked to determine if it is a global pause packet.

3. If step two 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:

```plaintext
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_pcp) && (OPCODE <= ctl_rx_opcode_max_pcp));
assign global_pause_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:

```plaintext
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) && (OPCODE <= ctl_rx_opcode_max_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:

```verbatim
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 these 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 hard CMAC 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.

   If one of the bits of `ctl_rx_pause_enable[8:0]` is set to 0 (disabled) when the pause processing is in step 2 or later, the core completes the steps as normal until it comes back to step 1.

2. If `ctl_rx_check_ack` input is 1, the core 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 core starts counting down the timer.

4. When the timer times out, the core 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 core deems the operation complete after 32 clock cycles.

These steps are demonstrated in Figure 3-15 with each step shown on the waveform.
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.

**Auto-Negotiation**

A block diagram of the 10G/25G Ethernet core with Auto-Negotiation (AN) and Link Training (LT) is shown in Figure 3-16.

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, ability is reported according to the available modes of operation.

Link Training is performed after auto-negotiation if the Link Training function is supported by both ends of the link. Link Training is typically required due to frequency-dependent losses which can occur as digital signals traverse the backplane. The primary function of the Link Training block included with this 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 transceiver). The other function of the Link Training block is to communicate training feedback from the receiver to the corresponding transmitter so that its equalizer circuit (part of the transceiver) can be adjusted as required. The decision-making algorithm is not part of this core.

When auto-negotiation and Link Training are complete, the datapath is switched to mission mode (the PCS), as shown in Figure 3-16.

**Overview**

Figure 3-17 shows the position of the auto-negotiation function in the OSI reference model.

![Auto-Negotiation Function in the OSI Model](image)

The Auto-Negotiation Intellectual Property Core (ANIPC) implements the requirements as specified in Clause 73, IEEE Std 802.3-2015, including those amendments specified in IEEE Std. P802.3ba and 802.3ap.

The functions of the ANIPC core are listed in clause 73, specifically Figure 73-11, Arbitration state diagram, in 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 transceiver 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 different to the mission mode operation. To accommodate this requirement, the ANIPC 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.

Functional Description

`.autoneg_enable`

When the `autoneg_enable` input signal is set to 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 non-zero value for every instance of the auto-negotiator. This is important to guarantee that no dead-locks 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 used in bit position 49 of the DME pages used during auto-negotiation. Any selection on this input is valid and does not result in any adverse behavior.

`Link Control`

When auto-negotiation begins, 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 monitored by the ANIPC hardware for an indication of the state of the various links that are 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 ANIPC 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, 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.
Chapter 3: Designing with the Subsystem

**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

Link Training is performed after Auto Negotiation 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 Link Training core is to provide register information and a training sequence over the backplane link which is then analyzed by a receiving circuit which is not part of the core.

The other function of the core is to communicate training feedback from the receiver to the corresponding transmitter so that its equalizer circuit (not part of the core) can be adjusted as required. The two circuits comprising the core are the receive Link Training block and the transmit Link Training block.

**IMPORTANT:** The logic responsible for adjusting the transmitter pre-emphasis taps must be supplied external to this IP core.

---

### Transmit

The Link Training transmit block constructs a 4,384-bit frame which contains a frame delimiter, control channel, and link training sequence. It is formatted as shown in Figure 3-18.

![Link Training Frame Structure](Figure 3-18: Link Training Frame Structure)

<table>
<thead>
<tr>
<th>Frame Marker</th>
<th>Coefficient Update</th>
<th>Status Update</th>
<th>Training Sequence</th>
</tr>
</thead>
<tbody>
<tr>
<td>32 bits</td>
<td>128 bits</td>
<td>128 bits</td>
<td>4096 bits</td>
</tr>
</tbody>
</table>

X15161-060716

548 octets 10.3125 Gb/s and 25.78125 Gb/s

Control Channel
Chapter 3: Designing with the Subsystem

Xilinx recommends 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 begins when \texttt{tx\_SOF} is asserted and ends at least 288 bit times later, or approximately 30 ns.

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 3-4 and Table 3-5 define these bits in the order in which they are transmitted starting with bit 15 and ending with bit 0.

<table>
<thead>
<tr>
<th>Bits</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>15:14</td>
<td>Reserved</td>
<td>Transmitted as 0, ignored on reception.</td>
</tr>
<tr>
<td>13</td>
<td>Preset</td>
<td>1 = Preset coefficients \</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0 = Normal operation</td>
</tr>
<tr>
<td>12</td>
<td>Initialize</td>
<td>1 = Initialize coefficients \</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0 = Normal operation</td>
</tr>
<tr>
<td>11:6</td>
<td>Reserved</td>
<td>Transmitted as 0, ignored on reception.</td>
</tr>
<tr>
<td>5:4</td>
<td>Coefficient (+1) update</td>
<td>1 1 = reserved \</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0 1 = increment \</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1 0 = decrease \</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0 0 = hold</td>
</tr>
<tr>
<td>3:2</td>
<td>Coefficient (0) update</td>
<td>1 1 = reserved \</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0 1 = increment \</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1 0 = decrease \</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0 0 = hold</td>
</tr>
<tr>
<td>1:0</td>
<td>Coefficient (-1) update</td>
<td>1 1 = reserved \</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0 1 = increment \</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1 0 = decrease \</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0 0 = hold</td>
</tr>
</tbody>
</table>
Chapter 3: Designing with the Subsystem

The functions of each bit are defined in IEEE Std. 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 Table 2-16.

**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. 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 which corresponds to the following polynomial:

\[ G(x) = 1 + x^9 + x^{11} \]

**Receive**

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

---

Table 3-5: Status Report Field Bit Definitions

<table>
<thead>
<tr>
<th>Bits</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>15</td>
<td>Receiver ready</td>
<td>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.</td>
</tr>
<tr>
<td>14:6</td>
<td>Reserved</td>
<td>Transmitted as 0, ignored on reception.</td>
</tr>
<tr>
<td>5:4</td>
<td>Coefficient (+1) update</td>
<td>0  1 = minimum  1  1 = maximum  1  0 = updated  0  0 = not_updated</td>
</tr>
<tr>
<td>3:2</td>
<td>Coefficient (0) update</td>
<td>1  1 = maximum  0  1 = minimum  1  0 = updated  0  0 = not_updated</td>
</tr>
<tr>
<td>1:0</td>
<td>Coefficient (-1) update</td>
<td>1  1 = maximum  0  1 = minimum  1  0 = updated  0  0 = not_updated</td>
</tr>
</tbody>
</table>
Chapter 3: Designing with the Subsystem

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 Std. 802.3, Clause 72, Fig. 72-4. When frame lock has been achieved, frame_lock is set to a value of TRUE.

Received Data

The receiver outputs the control channel with the bit definitions defined in Table 3-4 and Table 3-5 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.
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 core. More detailed information about the standard Vivado® design flows and the Vivado IP integrator can be found in the following Vivado Design Suite user guides:

- *Vivado Design Suite User Guide: Designing with IP* (UG896) [Ref 5]

Customizing and Generating the Core

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

If you are customizing and generating the core in the Vivado IP integrator, see the *Vivado Design Suite User Guide: Designing IP Subsystems using IP Integrator* (UG994) [Ref 4] for detailed information. IP integrator might auto-compute certain configuration values when validating or generating the design. To check whether the values do change, see the description of the parameter in this chapter. To view the parameter value, run the `validate_bd_design` command in the Tcl console.

You can customize the IP for use in your design by specifying values for the various parameters associated with the core using the following steps:

1. Select the IP from the IP catalog.
2. Double-click the selected IP or select the Customize IP command from the toolbar or right-click menu.

For details, see the *Vivado Design Suite User Guide: Designing with IP* (UG896) [Ref 5] and the *Vivado Design Suite User Guide: Getting Started* (UG910) [Ref 6].

*Note:* Figures in this chapter are illustrations of the Vivado IDE. The layout depicted here might vary from the current version.
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

<table>
<thead>
<tr>
<th>Option</th>
<th>Values</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>General</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Select Core</td>
<td>Ethernet MAC+PCS/PMA</td>
<td>Ethernet MAC+PCS/PMA</td>
</tr>
<tr>
<td></td>
<td>Ethernet PCS/PMA</td>
<td></td>
</tr>
<tr>
<td>Speed</td>
<td>25.7812G</td>
<td>25.7812G</td>
</tr>
<tr>
<td></td>
<td>10.3125G</td>
<td></td>
</tr>
<tr>
<td>Runtime Switchable Mode</td>
<td>0, 1</td>
<td>0</td>
</tr>
<tr>
<td>Num of Cores</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td></td>
</tr>
<tr>
<td></td>
<td>3</td>
<td></td>
</tr>
<tr>
<td></td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>Clocking</td>
<td>Synchronous</td>
<td>Asynchronous</td>
</tr>
<tr>
<td>Data Path Interface</td>
<td>AXI4-Stream</td>
<td>AXI4-Stream</td>
</tr>
<tr>
<td></td>
<td>Media Independent Interface (MII)</td>
<td></td>
</tr>
<tr>
<td><strong>PCS/PMA Options</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Base-R</td>
<td>Base-R</td>
<td>Base-KR</td>
</tr>
<tr>
<td>Base-KR</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Clause 74 (BASE-KR FEC)</td>
<td>0, 1</td>
<td>0</td>
</tr>
<tr>
<td>Clause 108 (RS-FEC)</td>
<td>0, 1</td>
<td>0</td>
</tr>
<tr>
<td>Auto Negotiation/Link Training Logic</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>Include AN/LT Logic</td>
<td></td>
</tr>
<tr>
<td>AN/LT Clock (5 MHz to 300 MHz)</td>
<td>5 MHz to 300 MHz</td>
<td>75 MHz</td>
</tr>
<tr>
<td><strong>Control and Statistics Interface</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Control and Statistics interface</td>
<td>Control and Status Vectors</td>
<td>Control and Status Vectors</td>
</tr>
<tr>
<td></td>
<td>Include AXI4-Lite</td>
<td></td>
</tr>
</tbody>
</table>

**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.
3. Clause 74 (BASE-KR FEC) logic is not supported for Base-R.
4. Clause 108 (RS-FEC) is not supported for Base-R, 10G speed and also for runtime switchable mode.
5. Clause 74 (BASE-KR FEC) and Clause 108 (RS-FEC) both can be selected in Vivado IDE but during functional operation only one can be enabled at a time using the respective control signals.
MAC Options Tab

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

![Figure 4-2: MAC Options Tab](image)

**Table 4-2: MAC Options**

<table>
<thead>
<tr>
<th>Option</th>
<th>Values</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>Optional Data Path Interface FIFO</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Include FIFO Logic</td>
<td>Checked, Unchecked</td>
<td>Checked</td>
</tr>
<tr>
<td>Flow Control</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Enable TX Flow Control Logic</td>
<td>Checked, Unchecked</td>
<td>Unchecked</td>
</tr>
<tr>
<td>Enable RX Flow Control Logic</td>
<td>Checked, Unchecked</td>
<td>Unchecked</td>
</tr>
<tr>
<td>IEEE PTP 1588v2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Enable Timestamping Logic</td>
<td>Checked, Unchecked</td>
<td>Unchecked</td>
</tr>
<tr>
<td>Operation Mode</td>
<td>One Step, Two Step</td>
<td>Two Step</td>
</tr>
</tbody>
</table>
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.

![GT Selection and Configuration Tab](image)

Figure 4-3: GT Selection and Configuration Tab
<table>
<thead>
<tr>
<th>Option</th>
<th>Values</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>GT Location</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Select whether the GT IP is included in the core or in the example design</td>
<td>Include GT subcore in core</td>
<td>Include GT subcore in core</td>
</tr>
<tr>
<td></td>
<td>Include GT subcore in example design</td>
<td></td>
</tr>
<tr>
<td><strong>GT Clocks</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GT RefClk (In MHz)</td>
<td>161.1328125</td>
<td>161.1328125</td>
</tr>
<tr>
<td></td>
<td>195.3125</td>
<td></td>
</tr>
<tr>
<td></td>
<td>201.4160156</td>
<td></td>
</tr>
<tr>
<td></td>
<td>257.8125</td>
<td></td>
</tr>
<tr>
<td></td>
<td>322.265625</td>
<td></td>
</tr>
<tr>
<td>GT DRP Clock (In MHz)</td>
<td>10 – 250 MHz</td>
<td>100.00</td>
</tr>
<tr>
<td><strong>Core to GT Association</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GT Type</td>
<td>GTY</td>
<td>GTY</td>
</tr>
<tr>
<td></td>
<td>GTH</td>
<td></td>
</tr>
<tr>
<td>GT Selection</td>
<td>Options based on device/package Quad groups. For example: Quad X0Y1 Quad X0Y2 Quad X0Y3 ...</td>
<td>Quad X0Y1</td>
</tr>
<tr>
<td>Lane-00 to Lane-03</td>
<td>Auto filled based on device/package. For example, if Num of Core = 4, and GT Selection = Quad X0Y1, four lanes are: X0Y4 X0Y5 X0Y6 X0Y7</td>
<td></td>
</tr>
<tr>
<td><strong>Others</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Enable Pipeline Registers</td>
<td>Checked, Unchecked</td>
<td>Unchecked</td>
</tr>
<tr>
<td>Enable Additional GT Control/Status and DRP Ports</td>
<td>Checked, Unchecked</td>
<td>Unchecked</td>
</tr>
</tbody>
</table>
Shared Logic Tab

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

Table 4-4:  Shared Logic Options

<table>
<thead>
<tr>
<th>Options</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>Include Shared Logic in core</td>
<td>Include Shared Logic in core</td>
</tr>
<tr>
<td>Include Shared Logic in example design</td>
<td></td>
</tr>
</tbody>
</table>

Output Generation

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

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

Required Constraints

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 7].

Simulation Speed Up

The example design contains wait timers. A `define SIM_SPEED_UP is available to improve simulation time by speeding up these wait times.

**VCS**

Use the vlogan option: +define+SIM_SPEED_UP

**ModelSim**

Use the vlog option: +define+SIM_SPEED_UP

**IES**

Use the ncvlog option: +define+SIM_SPEED_UP

**Vivado Simulator**

Use the xvlog option: -d SIM_SPEED_UP

---

Synthesis and Implementation

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

This chapter contains information about the example design provided in the Vivado® Design Suite when using the Vivado Integrated Design Environment (IDE).

Overview

Figure 5-1 shows the instantiation of various modules and their hierarchy for a single core configuration of xxv_ethernet_0 example design when the GT (serial transceiver) is inside the IP core.

Sync registers and pipeline registers are used for to synchronize the 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 datapath interface
  - AXI4-Lite for control and statistics interface

- **PCS configuration:**
  - MII for datapath interface
  - AXI4-Lite for control and statistics interface

The `xxv_ethernet_0_pkt_gen_mon` module is used to generate the data packets for sanity testing. The packet generation and checking is controlled by a FSM module.

The optional modules are described as follows:
- **xxv_ethernet_0_trans_debug**: This module is present in the example design when you enable the Additional GT Control and Status Ports check box from the GT Selection and Configuration Tab in the Vivado IDE or Include GT subcore in example design option in the GT Selection and Configuration tab or the Runtime Switchable mode option in the in the Configuration tab. This module brings out all the GT channel DRP ports, and some control and status ports of the transceiver module out of the xxv_ethernet core.

- **Retiming registers**: When you select the Enable Retiming Register option from the GT Selection and Configuration Tab, it includes a single stage pipeline register between core and the GT to ease timing, using the `gt_txusrclk2` and `gt_rxusrclk2` for TX and RX paths respectively. However, by default two-stage registering is done for the signals between GT and the core.

- **TX / RX Sync register**: The TX Sync register double synchronizes the data from the core to the GT with respect to the `tx_clk`. The RX Sync register double synchronizes the data from the GT to the core with respect to the `rx_serdes_clk`.

**Note**: For Runtime Switchable, if Auto Negotiation/Link training is selected in Vivado IDE, then AN operation will be performed only with the 10G data rate during switchings and LT will be performed in the mission mode.

**Note**: If Auto Negotiation/Link training is selected in Vivado IDE and the number of cores >= 3, then a PBLOCK constraint must be applied for the anlt_wrappers. The PBLOCK should be placed near to the selected transceivers (GT) and the size should be sufficient to fit the anlt_wrapper utilization. Refer to example_top.xdc for more information. Following is an example for a xcvu095-ffva2104-2-e device when four cores are selected and the transceivers are x0y4 to x0y7.

Example:

```plaintext
create_pblock pblock_ANLT
add_cells_to_pblock [get_pblocks pblock_ANLT] [get_cells -quiet [list DUT/inst/i_*_top_0/i_*_ANLT_WRAPPER DUT/inst/i_*_top_1/i_*_ANLT_WRAPPER DUT/inst/i_*_top_2/i_*_ANLT_WRAPPER DUT/inst/i_*_top_3/i_*_ANLT_WRAPPER]]
resize_pblock [get_pblocks pblock_ANLT] -add {SLICE_X0Y5:SLICE_X40Y180}
```
Chapter 5: Example Design

Figure 5-2 shows the instantiation of various modules and their hierarchy for the multiple core configuration of xxv_ethernet_0 example design.
Example Design Hierarchy (GT in Example Design)

**Figure 5-3:** Single Core with GT in Example Design Hierarchy

Figure 5-3 shows the instantiation of various modules and their hierarchy for a single core configuration of the xxv_ethernet_0 example design when the GT (serial transceiver) is outside the IP Core, that is, in the example design. This hierarchical example design is delivered when you select the **Include GT subcore in example design** option from the GT Selection and Configuration tab.

The xxv_ethernet_0_core_support.v is present in the hierarchy when you select the **Include GT subcore in example design** option from the GT Selection and Configuration tab or the **Include Shared Logic in example design** option from the Shared Logic tab. This instantiates the xxv_ethernet_0_sharedlogic_wrapper.v module and the xxv_ethernet_0.v module for the **Include Shared Logic in example design** option. The xxv_ethernet_0_gt_wrapper.v module will be present when you select the **GT subcore in example design** option.

The user interface available for MAC/PCS configuration and PCS configuration configurations is the same as mentioned in the **Overview**.
The `xxv_ethernet_0.v` module instantiates the necessary sync registers/retiming pipeline registers for the synchronization of data between the core and the GT.

The `xxv_ethernet_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.

Description of optional modules are as follows:

- `xxv_ethernet_0_sharedlogic_wrapper`

  This module is present in the example design when you select the **Include GT subcore in example design** option from the GT Selection and Configuration tab or **Include Shared Logic** in the Example Design from the Shared Logic tab. This module brings all modules that can be shared between multiple IP cores and designs outside the IP core.

- `xxv_ethernet_0_gt_wrapper`

  This module is present in the example design when you select the **Include GT subcore in example design** option from the GT Selection and Configuration tab. This module is having instantiations of the GT along with various helper blocks. The clocking helper blocks are used to generate the required clock frequency for the Core.

**Figure 5-4** shows the instantiation of various modules and their hierarchy for the multiple core configuration of the `xxv_ethernet_0` example design when the GT is in the example design.
**Chapter 5: Example Design**

**Figure 5-4:** Multiple Core with GT in Example Design Hierarchy
User Interface

General purpose I/Os (GPIOs) are provided to control the example design. The user input and user output ports are described in Table 5-1.

Table 5-1: User Input and User Output Ports

<table>
<thead>
<tr>
<th>Name</th>
<th>Size</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sys_reset</td>
<td>1</td>
<td>Input</td>
<td>Reset for xxv_ethernet core.</td>
</tr>
<tr>
<td>gt_ref_clk_p</td>
<td>1</td>
<td>Input</td>
<td>Differential input clk to GT.</td>
</tr>
<tr>
<td>gt_ref_clk_n</td>
<td>1</td>
<td>Input</td>
<td>Differential input clk to GT.</td>
</tr>
<tr>
<td>dclk</td>
<td>1</td>
<td>Input</td>
<td>Stable/free running input clk to GT.</td>
</tr>
<tr>
<td>rx_gt_locked_led_0</td>
<td>1</td>
<td>Output</td>
<td>Indicates that GT has been locked.</td>
</tr>
<tr>
<td>rx_block_lock_led_0</td>
<td>1</td>
<td>Output</td>
<td>Indicates RX block lock has been achieved.</td>
</tr>
<tr>
<td>restart_tx_rx_0</td>
<td>1</td>
<td>Input</td>
<td>This signal is used to restart the packet generation and reception for the data sanity test when the packet generator and the packet monitor are in idle state.</td>
</tr>
<tr>
<td>completion_status</td>
<td>5</td>
<td>Output</td>
<td>This signal represents the test status/result.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• 5'd0  Test did not run.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• 5'd1  PASSED 25GE/10GE CORE TEST SUCCESSFULLY COMPLETED</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• 5'd2  No block lock on any lanes.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• 5'd3  Not all lanes achieved block lock.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• 5'd4  Some lanes lost block lock after achieving block lock.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• 5'd5  No lane sync on any lanes.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• 5'd6  Not all lanes achieved sync.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• 5'd7  Some lanes lost sync after achieving sync.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• 5'd8  No alignment status or rx_status was achieved.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• 5'd9  Loss of alignment status or rx_status after both were achieved.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• 5'd10 TX timed out.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• 5'd11 No TX data was sent.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• 5'd12 Number of packets received did not equal the number of packets sent.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• 5'd13 Total number of bytes received did not equal the total number of bytes sent.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• 5'd14 A protocol error was detected.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• 5'd15 Bit errors were detected in the received packets.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• 5'd31 Test is stuck in reset.</td>
</tr>
</tbody>
</table>
Table 5-1: User Input and User Output Ports (Cont’d)

<table>
<thead>
<tr>
<th>Name</th>
<th>Size</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>mode_change_0</td>
<td>1</td>
<td>Input</td>
<td>This port is available only when Runtime Switchable is selected in Vivado IDE and this is used to switch the core speed.</td>
</tr>
<tr>
<td>core_speed_0</td>
<td>1</td>
<td>Input</td>
<td>This signal indicates the speed with which the core is working: 1'b1 = 10G and 1'b0 = 25G</td>
</tr>
</tbody>
</table>

Core xci Top Level Port List

The top level port list for the core xci with all features enabled is listed below:

In the following table an asterisk (*) represents CORE number, having value 0 to 3.

Example: port_name_*

- port_name_0: for first CORE
- port_name_1: for second CORE (will be present when user selects number of cores >=2)
- port_name_2: for third CORE (will be present when user selects number of cores >=3)
- port_name_3: for fourth CORE (will be present when user selects number of cores =4)

Table 5-2: Core xci Top Level Port List

<table>
<thead>
<tr>
<th>Name</th>
<th>Size</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
</table>
| sys_reset   | 1    | Input     | Reset for core.  
Note: This port is available when the Include GT subcore in core option is selected in the GT Selection and Configuration tab. |
| dclk        | 1    | Input     | Stable input clk to GT.  
Note: This port is available when the Include GT subcore in core option is selected in the GT Selection and Configuration tab. |
| gt_refclk_p | 1    | Input     | Differential input clk to GT.  
Note: This port is available when the Include GT subcore in core option is selected in the GT Selection and Configuration tab and the Include Shared Logic in core option is selected in the Shared Logic tab. |
### Table 5-2: Core xci Top Level Port List (Cont’d)

<table>
<thead>
<tr>
<th>Name</th>
<th>Size</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>gt_refclk_n</td>
<td>1</td>
<td>Input</td>
<td>Differential input clk to GT. Note: This port is available when the Include GT subcore in core option is selected in the GT Selection and Configuration tab and the Include Shared Logic in core option is selected in the Shared Logic tab.</td>
</tr>
<tr>
<td>qpll0clk_in</td>
<td>2/4</td>
<td>Input</td>
<td>QPLL0 clock input. Note: This port is available when the Include GT subcore in core option is selected in the GT Selection and Configuration tab and the Include Shared Logic in example design option is selected in the Shared Logic tab. Port width: 2-bit for 50G single core and 4-bit for 40G one core / 50G two core.</td>
</tr>
<tr>
<td>qpll0refclk_in</td>
<td>2/4</td>
<td>Input</td>
<td>QPLL0 ref clock input. Note: This port is available when the Include GT subcore in core option is selected in the GT Selection and Configuration tab and the Include Shared Logic in example design option is selected in the Shared Logic tab. Port width: 2-bit for 50G single core and 4-bit for 40G one core / 50G two core.</td>
</tr>
<tr>
<td>qpll1clk_in</td>
<td>2/4</td>
<td>Input</td>
<td>QPLL1 clock input. Note: This port is available when the Include GT subcore in core option is selected in the GT Selection and Configuration tab and the Include Shared Logic in example design option is selected in the Shared Logic tab. Port width: 2-bit for 50G single core and 4-bit for 40G one core / 50G two core.</td>
</tr>
<tr>
<td>qpll1refclk_in</td>
<td>2/4</td>
<td>Input</td>
<td>QPLL1 ref clock input. Note: This port is available when the Include GT subcore in core option is selected in the GT Selection and Configuration tab and the Include Shared Logic in example design option is selected in the Shared Logic tab. Port width: 2-bit for 50G single core and 4-bit for 40G one core / 50G two core.</td>
</tr>
<tr>
<td>gtwiz_reset_qpll0lock_in</td>
<td>1</td>
<td>Input</td>
<td>QPLL0 lock reset input to the GT. Note: This port is available when the Include GT subcore in core option is selected in the GT Selection and Configuration tab and the Include Shared Logic in example design option is selected in the Shared Logic tab.</td>
</tr>
</tbody>
</table>
### Table 5-2: Core xci Top Level Port List (Cont’d)

<table>
<thead>
<tr>
<th>Name</th>
<th>Size</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>gtwiz_reset_qpll0reset_out</td>
<td>1</td>
<td>Output</td>
<td>QPLL0 lock reset output from the GT. This port is available when the Include GT subcore in core option is selected in the GT Selection and Configuration tab and the Include Shared Logic in example design option is selected in the Shared Logic tab.</td>
</tr>
<tr>
<td>gtwiz_reset_qpll1lock_in</td>
<td>1</td>
<td>Input</td>
<td>QPLL1 lock reset input to the GT. Note: This port is available when the Include GT subcore in core option is selected in the GT Selection and Configuration tab and the Include Shared Logic in example design option is selected in the Shared Logic tab.</td>
</tr>
<tr>
<td>gtwiz_reset_qpll1reset_out</td>
<td>1</td>
<td>Output</td>
<td>QPLL1 lock reset output from the GT. Note: This port is available when the Include GT subcore in core option is selected in the GT Selection and Configuration tab and the Include Shared Logic in example design option is selected in the Shared Logic tab.</td>
</tr>
<tr>
<td>tx_clk_out_*</td>
<td>1</td>
<td>Output</td>
<td>TX user clock output from GT. Note: This port is available when the Select Core is Ethernet MAC+PCS/PMA and the Include GT subcore in core option is selected in the GT Selection and Configuration tab.</td>
</tr>
<tr>
<td>tx_mii_clk_*</td>
<td>1</td>
<td>Output</td>
<td>TX user clock output from GT. Note: This port is available when the Select Core is Ethernet PCS/PMA and the Include GT subcore in core option is selected in the GT Selection and Configuration tab.</td>
</tr>
<tr>
<td>rx_clk_out_*</td>
<td>1</td>
<td>Output</td>
<td>RX user clock output from GT. Note: This port is available when the Include GT subcore in core option is selected in the GT Selection and Configuration tab.</td>
</tr>
<tr>
<td>rx_serdes_clk_*</td>
<td>1</td>
<td>Input</td>
<td>RX serdes clock input to core Note: This port is available when the Include GT subcore in example design option is selected in the GT Selection and Configuration tab.</td>
</tr>
<tr>
<td>rx_serdes_reset_*</td>
<td>1</td>
<td>Input</td>
<td>RX serdes reset input to core Note: This port is available when the Include GT subcore in example design option is selected in the GT Selection and Configuration tab.</td>
</tr>
</tbody>
</table>
Table 5-2: Core xci Top Level Port List (Cont’d)

<table>
<thead>
<tr>
<th>Name</th>
<th>Size</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>rxreclklout_*</td>
<td>1</td>
<td>Output</td>
<td>RX recovered clock output from GT. [Note: This port is available when the Include GT subcore in core option is selected in the GT Selection and Configuration tab.]</td>
</tr>
<tr>
<td>tx_core_clk_*</td>
<td>1</td>
<td>Input</td>
<td>TX Core clock input from GT wrapper. [Note: This port is available when the Include GT subcore in example design option is selected in the GT Selection and Configuration tab.]</td>
</tr>
<tr>
<td>rx_core_clk_*</td>
<td>1</td>
<td>Input</td>
<td>RX Core clock input to the core.</td>
</tr>
<tr>
<td>tx_reset_*</td>
<td>1</td>
<td>Input</td>
<td>TX reset input to the core.</td>
</tr>
<tr>
<td>user_tx_reset_*</td>
<td>1</td>
<td>Output</td>
<td>TX reset output for the user logic. [Note: This port is available when the Include GT subcore in core option is selected in the GT Selection and Configuration tab and the Include Shared Logic in core option is selected in the Shared Logic tab.]</td>
</tr>
<tr>
<td>gt_reset_tx_done_out_*</td>
<td>1</td>
<td>Output</td>
<td>TX reset done signal from the GT. [Note: This port is available when the Include GT subcore in core option is selected in the GT Selection and Configuration tab and the Include Shared Logic in example design option is selected in the Shared Logic tab.]</td>
</tr>
<tr>
<td>rx_reset_*</td>
<td>1</td>
<td>Input</td>
<td>RX reset input to the core.</td>
</tr>
<tr>
<td>user_rx_reset_*</td>
<td>1</td>
<td>Output</td>
<td>RX reset output for the user logic. [Note: This port is available when the Include GT subcore in core option is selected in the GT Selection and Configuration tab and the Include Shared Logic in core option is selected in the Shared Logic tab.]</td>
</tr>
<tr>
<td>gt_reset_rx_done_out_*</td>
<td>1</td>
<td>Output</td>
<td>RX reset done signal from the GT. [Note: This port is available when the Include GT subcore in core option is selected in the GT Selection and Configuration tab and the Include Shared Logic in example design option is selected in the Shared Logic tab.]</td>
</tr>
<tr>
<td>ctl_gt_reset_all_*</td>
<td>1</td>
<td>Output</td>
<td>gt_reset_all signal from the AXI4-Lite register map. [Note: This port is available when the 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.]</td>
</tr>
</tbody>
</table>
### Table 5-2: Core xci Top Level Port List (Cont’d)

<table>
<thead>
<tr>
<th>Name</th>
<th>Size</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ctl_gt_tx_reset_*</td>
<td>1</td>
<td>Output</td>
<td>gt_tx_reset signal from the AXI4-Lite register map.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Note: This port is available when the Include AXI4-Lite option is selected from the Configuration tab and the Include Shared Logic in example design option is selected in the Shared Logic tab.</td>
</tr>
<tr>
<td>ctl_gt_rx_reset_*</td>
<td>1</td>
<td>Output</td>
<td>gt_rx_reset signal from the AXI4-Lite register map.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Note: This port is available when the 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.</td>
</tr>
<tr>
<td>gt_reset_all_in_+</td>
<td>1</td>
<td>Input</td>
<td>gt_reset_all signal from the reset_wrapper of shared logic wrapper.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Note: This port is available when the Include GT subcore in core option is selected in the GT Selection and Configuration tab and the Include Shared Logic in example design option is selected in the Shared Logic tab.</td>
</tr>
<tr>
<td>gt_tx_reset_in_+</td>
<td>1</td>
<td>Input</td>
<td>gt_tx_reset_in signal from reset_wrapper of shared logic wrapper.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Note: This port is available when the Include GT subcore in core option is selected in the GT Selection and Configuration tab and the Include Shared Logic in example design option is selected in the Shared Logic tab.</td>
</tr>
<tr>
<td>gt_rx_reset_in_+</td>
<td>1</td>
<td>Input</td>
<td>gt_rx_reset_in signal from reset_wrapper of shared logic wrapper.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Note: This port is available when the Include GT subcore in core option is selected in the GT Selection and Configuration tab and the Include Shared Logic in example design option is selected in the Shared Logic tab.</td>
</tr>
<tr>
<td>gt_refclk_out</td>
<td>1</td>
<td>Output</td>
<td>Indicates the GT_refclk output</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>This port is available when the Include GT subcore in core option is selected in the GT Selection and Configuration tab and the Include Shared Logic in example design option is selected in the Shared Logic tab.</td>
</tr>
</tbody>
</table>
### Table 5-2: Core xci Top Level Port List (Cont’d)

<table>
<thead>
<tr>
<th>Name</th>
<th>Size</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Common Transceiver Interface ports</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>gt_loopback_in_*</td>
<td>3</td>
<td>Input</td>
<td>GT loopback input signal. Refer to the GT user guide. <strong>Note:</strong> This port is available when the Include GT subcore in core option is selected in the GT Selection and Configuration tab.</td>
</tr>
<tr>
<td>gt_loopback_out_*</td>
<td>1</td>
<td>Output</td>
<td>GT loopback output signal from AXI4-Lite register map. Refer to the GT user guide. <strong>Note:</strong> This port is available when Include AXI4-Lite is selected from the Configuration tab and the Include GT subcore in example design option is selected in the GT Selection and Configuration tab.</td>
</tr>
<tr>
<td>gt_txp_out</td>
<td>1</td>
<td>Output</td>
<td>Differential serial GT TX output. <strong>Note:</strong> This port is available when the Include GT subcore in core option and Board support is selected in the GT Selection and Configuration tab.</td>
</tr>
<tr>
<td>gt_txn_out</td>
<td></td>
<td>Output</td>
<td>Differential serial GT TX output. <strong>Note:</strong> This port is available when the Include GT subcore in core option and Board support is selected in the GT Selection and Configuration tab.</td>
</tr>
<tr>
<td>gt_rxn_in</td>
<td></td>
<td>Input</td>
<td>Differential serial GT RX input. <strong>Note:</strong> This port is available when the Include GT subcore in core option and Board support is selected in the GT Selection and Configuration tab.</td>
</tr>
<tr>
<td>gt_rxp_in</td>
<td></td>
<td>Input</td>
<td>Differential serial GT RX input. <strong>Note:</strong> This port is available when the Include GT subcore in core option and Board support is selected.</td>
</tr>
<tr>
<td>gt_rxp_in_0</td>
<td>1</td>
<td>Input</td>
<td>Differential serial GT RX input for lane 0. <strong>Note:</strong> This port is available when the Include GT subcore in core option is selected in the GT Selection and Configuration tab.</td>
</tr>
<tr>
<td>gt_rxn_in_0</td>
<td>1</td>
<td>Input</td>
<td>Differential serial GT RX input for lane 0. <strong>Note:</strong> This port is available when the Include GT subcore in core option is selected in the GT Selection and Configuration tab.</td>
</tr>
</tbody>
</table>
**Table 5-2: Core xci Top Level Port List (Cont’d)**

<table>
<thead>
<tr>
<th>Name</th>
<th>Size</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
</table>
| gt_rxp_in_1 | 1    | Input     | Differential serial GT RX input for lane 1.  
**Note:** This port is available when **Num of Cores** is >1 and the **Include GT subcore in core** option is selected in the GT Selection and Configuration tab. |
| gt_rxn_in_1 | 1    | Input     | Differential serial GT RX input for lane 1.  
**Note:** This port is available when **Num of Cores** is >1 and the **Include GT subcore in core** option is selected in the GT Selection and Configuration tab. |
| gt_rxp_in_2 | 1    | Input     | Differential serial GT RX input for lane 2.  
**Note:** This port is available when **Num of Cores** is >2 and the **Include GT subcore in core** option is selected in the GT Selection and Configuration tab. |
| gt_rxn_in_2 | 1    | Input     | Differential serial GT RX input for lane 2.  
**Note:** This port is available when **Num of Cores** is >2 and the **Include GT subcore in core** option is selected in the GT Selection and Configuration tab. |
| gt_rxp_in_3 | 1    | Input     | Differential serial GT RX input for lane 3.  
**Note:** This port is available when **Num of Cores** is >3 and the **Include GT subcore in core** option is selected in the GT Selection and Configuration tab. |
| gt_rxn_in_3 | 1    | Input     | Differential serial GT RX input for lane 3.  
**Note:** This port is available when **Num of Cores** is >3 and the **Include GT subcore in core** option is selected in the GT Selection and Configuration tab. |
| gt_txp_out_0 | 1    | Output    | Differential serial GT TX output for lane 0.  
**Note:** This port is available when the **Include GT subcore in core** option is selected in the GT Selection and Configuration tab. |
| gttxn_out_0  | 1    | Output    | Differential serial GT TX output for lane 0.  
**Note:** This port is available when the **Include GT subcore in core** option is selected in the GT Selection and Configuration tab. |
| gt_txp_out_1 | 1    | Output    | Differential serial GT TX output for lane 1.  
**Note:** This port is available when **Num of Cores** is >1 and the **Include GT subcore in core** option is selected in the GT Selection and Configuration tab. |
Table 5-2: Core xci Top Level Port List (Cont’d)

<table>
<thead>
<tr>
<th>Name</th>
<th>Size</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>gt_txn_out_1</td>
<td>1</td>
<td>Output</td>
<td>Differential serial GT TX output for lane 1. Note: This port is available when Num of Cores is &gt;1 and the Include GT subcore in core option is selected in the GT Selection and Configuration tab.</td>
</tr>
<tr>
<td>gt_xp_out_2</td>
<td>1</td>
<td>Output</td>
<td>Differential serial GT TX output for lane 2. Note: This port is available when Num of Cores is &gt;2 and the Include GT subcore in core option is selected in the GT Selection and Configuration tab.</td>
</tr>
<tr>
<td>gt_txn_out_2</td>
<td>1</td>
<td>Output</td>
<td>Differential serial GT TX output for lane 2. Note: This port is available when Num of Cores is &gt;2 and the Include GT subcore in core option is selected in the GT Selection and Configuration tab.</td>
</tr>
<tr>
<td>gt_xp_out_3</td>
<td>1</td>
<td>Output</td>
<td>Differential serial GT TX output for lane 3. Note: This port is available when Num of Cores is &gt;3 and the Include GT subcore in core option is selected in the GT Selection and Configuration tab.</td>
</tr>
<tr>
<td>gt_txn_out_3</td>
<td>1</td>
<td>Output</td>
<td>Differential serial GT TX output for lane 3. Note: This port is available when Num of Cores is &gt;3 and the Include GT subcore in core option is selected in the GT Selection and Configuration tab.</td>
</tr>
<tr>
<td>rxgearboxslip_in_*</td>
<td>1</td>
<td>Output</td>
<td>Rxgearboxslip signal from core to GT. Note: This port is available when the Include GT subcore in example design option is selected in the GT Selection and Configuration tab.</td>
</tr>
<tr>
<td>rxdatavalid_out_*</td>
<td>2</td>
<td>Input</td>
<td>Rx data valid signal from GT to core. Note: This port is available when the Include GT subcore in example design option is selected in the GT Selection and Configuration tab.</td>
</tr>
<tr>
<td>rx_serdes_data_out_*</td>
<td>64/128</td>
<td>Input</td>
<td>Rx data signal from GT to core. Note: This port is available when the Include GT subcore in example design option is selected in the GT Selection and Configuration tab. The data width is 64 bits for 10G configuration and 128 bits for 25G configuration.</td>
</tr>
</tbody>
</table>
### Table 5-2: Core xci Top Level Port List (Cont’d)

<table>
<thead>
<tr>
<th>Name</th>
<th>Size</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>rxheader_out_*</td>
<td>6</td>
<td>Input</td>
<td>rx header signal from GT to core.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>Note:</strong> This port is available when the Include GT subcore in example design option is selected in the GT Selection and Configuration tab.</td>
</tr>
<tr>
<td>rxheadervalid_out_*</td>
<td>2</td>
<td>Input</td>
<td>rx header valid signal from GT to core.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>Note:</strong> This port is available when the Include GT subcore in example design option is selected in the GT Selection and Configuration tab.</td>
</tr>
<tr>
<td>tx_serdes_data_in_*</td>
<td>64/128</td>
<td>Output</td>
<td>tx data signal from core to GT.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>Note:</strong> This port is available when the Include GT subcore in example design option is selected in the GT Selection and Configuration tab.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>The data width is 64 bits for 10G configuration and 128 bits for 25G configuration.</td>
</tr>
<tr>
<td>txheader_in_*</td>
<td>6</td>
<td>Output</td>
<td>tx header signal from core to GT.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>Note:</strong> This port is available when the Include GT subcore in example design option is selected in the GT Selection and Configuration tab.</td>
</tr>
</tbody>
</table>

#### Transceiver control and status debug ports

Ports under this section will be available when the Include GT subcore in core option is selected in the GT Selection and Configuration tab or Enable Additional GT Control/Status and DRP Ports is selected from the GT Selection and Configuration tab.

<table>
<thead>
<tr>
<th>Name</th>
<th>Size</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>gt_dmonitorout_*</td>
<td>16</td>
<td>Output</td>
<td>Refer to the GT user guide for the port description.</td>
</tr>
<tr>
<td>gt_eyescandataerror_*</td>
<td>1</td>
<td>Output</td>
<td>Refer to the GT user guide for the port description.</td>
</tr>
<tr>
<td>gt_eyescanreset_*</td>
<td>1</td>
<td>Input</td>
<td>Refer to the GT user guide for the port description.</td>
</tr>
<tr>
<td>gt_eyescantrigger_*</td>
<td>1</td>
<td>Input</td>
<td>Refer to the GT user guide for the port description.</td>
</tr>
<tr>
<td>gt_pcsrsvdin_*</td>
<td>16</td>
<td>Input</td>
<td>Refer to the GT user guide for the port description.</td>
</tr>
<tr>
<td>gt_rxbufreset_*</td>
<td>1</td>
<td>Input</td>
<td>Refer to the GT user guide for the port description.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>Note:</strong> Port width: 2-bit for 50G single core and 4-bits for 40G.</td>
</tr>
</tbody>
</table>
### Table 5-2: Core xci Top Level Port List (Cont’d)

<table>
<thead>
<tr>
<th>Name</th>
<th>Size</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>gt_rxbufstatus_*</td>
<td>3</td>
<td>Output</td>
<td>Refer to the GT user guide for the port description.</td>
</tr>
<tr>
<td>gt_rxcdrhold_*</td>
<td>1</td>
<td>Input</td>
<td>Refer to the GT user guide for the port description.</td>
</tr>
<tr>
<td>gt_rxcommadeten_*</td>
<td>1</td>
<td>Input</td>
<td>Refer to the GT user guide for the port description.</td>
</tr>
<tr>
<td>gt_rxdfegchold_*</td>
<td>1</td>
<td>Input</td>
<td>Refer to the GT user guide for the port description.</td>
</tr>
<tr>
<td>gt_rxfelpreset_*</td>
<td>1</td>
<td>Input</td>
<td>Refer to the GT user guide for the port description.</td>
</tr>
<tr>
<td>gt_rxlatclk_*</td>
<td>1</td>
<td>Input</td>
<td>Refer to the GT user guide for the port description.</td>
</tr>
<tr>
<td>gt_rxlpmen_*</td>
<td>1</td>
<td>Input</td>
<td>Refer to the GT user guide for the port description.</td>
</tr>
<tr>
<td>gt_rxpcsreset_*</td>
<td>1</td>
<td>Input</td>
<td>Refer to the GT user guide for the port description.</td>
</tr>
<tr>
<td>gt_rxpmareset_*</td>
<td>1</td>
<td>Input</td>
<td>Refer to the GT user guide for the port description.</td>
</tr>
<tr>
<td>gt_rxpolarity_*</td>
<td>1</td>
<td>Input</td>
<td>Refer to the GT user guide for the port description.</td>
</tr>
<tr>
<td>gt_rxrbscntreset_*</td>
<td>1</td>
<td>Input</td>
<td>Refer to the GT user guide for the port description.</td>
</tr>
<tr>
<td>gt_rxrbsserr_*</td>
<td>1</td>
<td>Input</td>
<td>Refer to the GT user guide for the port description.</td>
</tr>
<tr>
<td>gt_rxrbsssel_*</td>
<td>4</td>
<td>Input</td>
<td>Refer to the GT user guide for the port description.</td>
</tr>
<tr>
<td>gt_rxrate_*</td>
<td>3</td>
<td>Input</td>
<td>Refer to the GT user guide for the port description.</td>
</tr>
<tr>
<td>gt_rxslide_*</td>
<td>1</td>
<td>Input</td>
<td>Refer to the GT user guide for the port description.</td>
</tr>
<tr>
<td>gt_rxstartofseq_*</td>
<td>2</td>
<td>Output</td>
<td>Refer to the GT user guide for the port description.</td>
</tr>
<tr>
<td>gt_txbufstatus_*</td>
<td>2</td>
<td>Output</td>
<td>Refer to the GT user guide for the port description.</td>
</tr>
<tr>
<td>gt_txdiffctrl_*</td>
<td>5</td>
<td>Input</td>
<td>Refer to the GT user guide for the port description.</td>
</tr>
<tr>
<td>gt_txinhibit_*</td>
<td>1</td>
<td>Input</td>
<td>Refer to the GT user guide for the port description.</td>
</tr>
<tr>
<td>gt_txlatclk_*</td>
<td>1</td>
<td>Input</td>
<td>Refer to the GT user guide for the port description.</td>
</tr>
<tr>
<td>gt_txmaincursor_*</td>
<td>7</td>
<td>Input</td>
<td>Refer to the GT user guide for the port description.</td>
</tr>
</tbody>
</table>
### Table 5-2: Core xci Top Level Port List (Cont’d)

<table>
<thead>
<tr>
<th>Name</th>
<th>Size</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>gt_txpcsreset_*</td>
<td>1</td>
<td>Input</td>
<td>Refer to the GT user guide for the port description.</td>
</tr>
<tr>
<td>gt_txpmareset_*</td>
<td>1</td>
<td>Input</td>
<td>Refer to the GT user guide for the port description.</td>
</tr>
<tr>
<td>gt_tpxpolarity_*</td>
<td>1</td>
<td>Input</td>
<td>Refer to the GT user guide for the port description.</td>
</tr>
<tr>
<td>gt_txpostcursor_*</td>
<td>5</td>
<td>Input</td>
<td>Refer to the GT user guide for the port description.</td>
</tr>
<tr>
<td>gt_txprbsforceerr_*</td>
<td>1</td>
<td>Input</td>
<td>Refer to the GT user guide for the port description.</td>
</tr>
<tr>
<td>gt_txprbsssel_*</td>
<td>4</td>
<td>Input</td>
<td>Refer to the GT user guide for the port description.</td>
</tr>
<tr>
<td>gt_txprecursor_*</td>
<td>5</td>
<td>Input</td>
<td>Refer to the GT user guide for the port description.</td>
</tr>
<tr>
<td>gtwiz_reset_tx_datapath_*</td>
<td>1</td>
<td>Input</td>
<td>Refer to the GT user guide for the port description.</td>
</tr>
<tr>
<td>gtwiz_reset_rx_datapath_*</td>
<td>1</td>
<td>Input</td>
<td>Refer to the GT user guide for the port description.</td>
</tr>
<tr>
<td>gt_ch_drpclk_*</td>
<td>1</td>
<td>Input</td>
<td>Refer to the GT user guide for the port description.</td>
</tr>
<tr>
<td>gt_ch_drpdo_*</td>
<td>16</td>
<td>Output</td>
<td>Refer to the GT user guide for the port description.</td>
</tr>
<tr>
<td>gt_ch_drprdy_*</td>
<td>1</td>
<td>Output</td>
<td>Refer to the GT user guide for the port description.</td>
</tr>
<tr>
<td>gt_ch_drpen_*</td>
<td>1</td>
<td>Input</td>
<td>Refer to the GT user guide for the port description.</td>
</tr>
<tr>
<td>gt_ch_drpwe_*</td>
<td>1</td>
<td>Input</td>
<td>Refer to the GT user guide for the port description.</td>
</tr>
<tr>
<td>gt_ch_drpaddr_*</td>
<td>10</td>
<td>Input</td>
<td>Refer to the GT user guide for the port description.</td>
</tr>
<tr>
<td>gt_ch_drpdi_*</td>
<td>16</td>
<td>Input</td>
<td>Refer to the GT user guide for the port description.</td>
</tr>
</tbody>
</table>

**AXI4-Lite Interface Ports**

Ports under this section will be available when the **Include AXI4-Lite** is selected from the Configuration tab.

<table>
<thead>
<tr>
<th>Name</th>
<th>Size</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>s_axi_aclk_*</td>
<td>1</td>
<td>Input</td>
<td>AXI clock signal</td>
</tr>
<tr>
<td>s_axi_aresetn_*</td>
<td>1</td>
<td>Input</td>
<td>AXI reset signal</td>
</tr>
<tr>
<td>pm_tick_*</td>
<td>1</td>
<td>Input</td>
<td>PM tick user input</td>
</tr>
<tr>
<td>s_axi_awaddr_*</td>
<td>32</td>
<td>Input</td>
<td>AXI write address</td>
</tr>
</tbody>
</table>
## Table 5-2: Core xci Top Level Port List (Cont’d)

<table>
<thead>
<tr>
<th>Name</th>
<th>Size</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>s_axi_awvalid_*</td>
<td>1</td>
<td>Input</td>
<td>AXI write address valid</td>
</tr>
<tr>
<td>s_axi_awready_*</td>
<td>1</td>
<td>Output</td>
<td>AXI write address ready</td>
</tr>
<tr>
<td>s_axi_wdata_*</td>
<td>32</td>
<td>Input</td>
<td>AXI write data</td>
</tr>
<tr>
<td>s_axi_wstrb_*</td>
<td>4</td>
<td>Input</td>
<td>AXI write strobe. This signal indicates which byte lanes hold valid data.</td>
</tr>
<tr>
<td>s_axi_wvalid_*</td>
<td>1</td>
<td>Input</td>
<td>AXI write data valid. This signal indicates that valid write data and strobes are available.</td>
</tr>
<tr>
<td>s_axi_wready_*</td>
<td>1</td>
<td>Output</td>
<td>AXI write data ready</td>
</tr>
</tbody>
</table>
| 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                                               |
| 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 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.

<table>
<thead>
<tr>
<th>Name</th>
<th>Size</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tx_unfout_*</td>
<td>1</td>
<td>Output</td>
<td>Underflow signal for TX data path from core. If tx_unfout_* is sampled as 1, a violation has occurred meaning the current packet is corrupted. Error control blocks are transmitted as long as the underflow condition persists. It is up to the user logic to ensure a complete packet is input to the core without under-running the TX data path interface.</td>
</tr>
<tr>
<td>tx_axis_tready_*</td>
<td>1</td>
<td>Output</td>
<td>TX path ready signal from core.</td>
</tr>
<tr>
<td>tx_axis_tvalid_*</td>
<td>1</td>
<td>Input</td>
<td>Transmit AXI4-Stream Data valid.</td>
</tr>
<tr>
<td>tx_axis_tdata_*</td>
<td>64</td>
<td>Input</td>
<td>Transmit AXI4-Stream Data bus.</td>
</tr>
<tr>
<td>tx_axis_tlast_*</td>
<td>1</td>
<td>Input</td>
<td>Transmit AXI4-Stream tlast.</td>
</tr>
<tr>
<td>tx_axis_tkeep_*</td>
<td>8</td>
<td>Input</td>
<td>Transmit AXI4-Stream tkeep.</td>
</tr>
<tr>
<td>tx_axis_tuser_*</td>
<td>1</td>
<td>Input</td>
<td>Transmit AXI4-Stream tuser.</td>
</tr>
<tr>
<td>tx_preamblein_*</td>
<td>56</td>
<td>Input</td>
<td>Transmit AXI4-Stream preamble.</td>
</tr>
<tr>
<td>rx_axis_tvalid_*</td>
<td>1</td>
<td>Output</td>
<td>Receive AXI4-Stream Data valid.</td>
</tr>
<tr>
<td>rx_axis_tdata_*</td>
<td>64</td>
<td>Output</td>
<td>Receive AXI4-Stream Data bus.</td>
</tr>
<tr>
<td>rx_axis_tlast_*</td>
<td>1</td>
<td>Input</td>
<td>Receive AXI4-Stream tlast.</td>
</tr>
<tr>
<td>rx_axis_tkeep_*</td>
<td>8</td>
<td>Input</td>
<td>Receive AXI4-Stream tkeep.</td>
</tr>
<tr>
<td>rx_axis_tuser_*</td>
<td>1</td>
<td>Input</td>
<td>Receive AXI4-Stream tuser.</td>
</tr>
<tr>
<td>rx_preamblein_*</td>
<td>56</td>
<td>Input</td>
<td>Receive AXI4-Stream preamble.</td>
</tr>
</tbody>
</table>
### MII User Interface Signals
Ports under this section will be available when Ethernet PCS/PMA is selected from the Configuration tab.

<table>
<thead>
<tr>
<th>Name</th>
<th>Size</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tx_mii_d_*</td>
<td>64</td>
<td>Input</td>
<td>Transmit LGMII Data bus.</td>
</tr>
<tr>
<td>tx_mii_c_*</td>
<td>8</td>
<td>Input</td>
<td>LGMII Control bus.</td>
</tr>
<tr>
<td>rx_mii_d_*</td>
<td>64</td>
<td>Output</td>
<td>Receive LGMII Data bus.</td>
</tr>
<tr>
<td>rx_mii_c_*</td>
<td>8</td>
<td>Output</td>
<td>Receive LGMII Control bus.</td>
</tr>
</tbody>
</table>
### TX Path Control / Status / Statistics Signals

<table>
<thead>
<tr>
<th>Name</th>
<th>Size</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ctl_tx_enable_*</td>
<td>1</td>
<td>Input</td>
<td>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. 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 then the core stops transmitting any more packets. <strong>Note:</strong> This port is available when Include AXI4-Lite is not selected and Select Core is Ethernet MAC+PCS/PMA in the Configuration tab.</td>
</tr>
<tr>
<td>ctl_tx_send_rfi_*</td>
<td>1</td>
<td>Input</td>
<td>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. <strong>Note:</strong> This port is available when Select Core is Ethernet MAC+PCS/PMA in the Configuration tab.</td>
</tr>
<tr>
<td>ctl_tx_send_lfi_*</td>
<td>1</td>
<td>Input</td>
<td>Transmit Local Fault Indication (LFI) code word. Takes precedence over RFI. <strong>Note:</strong> This port is available when Select Core is Ethernet MAC+PCS/PMA in the Configuration tab.</td>
</tr>
<tr>
<td>ctl_tx_send_idle_*</td>
<td>1</td>
<td>Input</td>
<td>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. <strong>Note:</strong> This port is available when Select Core is Ethernet MAC+PCS/PMA in the Configuration tab.</td>
</tr>
</tbody>
</table>
**Table 5-2: Core xci Top Level Port List (Cont’d)**

<table>
<thead>
<tr>
<th>Name</th>
<th>Size</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ctl_tx_fcs_ins_enable_*</td>
<td>1</td>
<td>Input</td>
<td>Enable FCS insertion by the TX core. If this bit is set to 0, the core does not add FCS to the packet. If this bit is set to 1, the HSEC core calculates and adds the FCS to the packet. This input cannot be changed dynamically between packets.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>Note:</strong> This port is available when Include AXI4-Lite is not selected and Select Core is Ethernet MAC+PCS/PMA in the Configuration tab.</td>
</tr>
<tr>
<td>ctl_tx_ignore_fcs_*</td>
<td>1</td>
<td>Input</td>
<td>Enable FCS error checking at the LBUS 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.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>Note:</strong> Statistics are reported as if there was no FCS error.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>Note:</strong> This port is available when Include AXI4-Lite is not selected and Select Core is Ethernet MAC+PCS/PMA in the Configuration tab.</td>
</tr>
<tr>
<td>ctl_tx_test_pattern_*</td>
<td>1</td>
<td>Input</td>
<td>Test pattern generation enable for the TX core. A value of 1 enables test mode as defined in Clause 49. Corresponds to MDIO register bit 3.42.3 as defined in Clause 45. Generates a scrambled idle pattern.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>Note:</strong> This port is available when Include AXI4-Lite is not selected in the Configuration tab.</td>
</tr>
<tr>
<td>ctl_tx_test_pattern_enable_*</td>
<td>1</td>
<td>Input</td>
<td>Test pattern enable for the RX core. A value of 1 enables test mode. Corresponds to MDIO register bit 3.42.2 as defined in Clause 45. Takes second precedence.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>Note:</strong> This port is available when Include AXI4-Lite is not selected in the Configuration tab.</td>
</tr>
<tr>
<td>ctl_tx_test_pattern_select_*</td>
<td>1</td>
<td>Input</td>
<td>Corresponds to MDIO register bit 3.42.1 as defined in Clause 45.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>Note:</strong> This port is available when Include AXI4-Lite is not selected in the Configuration tab.</td>
</tr>
<tr>
<td>Name</td>
<td>Size</td>
<td>Direction</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>------</td>
<td>-----------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>ctl_tx_data_pattern_select_*</td>
<td>1</td>
<td>Input</td>
<td>Corresponds to MDIO register bit 3.42.0 as defined in Clause 45.</td>
</tr>
<tr>
<td>Note:</td>
<td></td>
<td></td>
<td>This port is available when Include AXI4-Lite is not selected in the Configuration tab.</td>
</tr>
<tr>
<td>ctl_tx_test_pattern_seed_a_*</td>
<td>58</td>
<td>Input</td>
<td>Corresponds to MDIO registers 3.34 through to 3.37 as defined in Clause 45.</td>
</tr>
<tr>
<td>Note:</td>
<td></td>
<td></td>
<td>This port is available when Include AXI4-Lite is not selected in the Configuration tab.</td>
</tr>
<tr>
<td>ctl_tx_test_pattern_seed_b_*</td>
<td>58</td>
<td>Input</td>
<td>Corresponds to MDIO registers 3.38 through to 3.41 as defined in Clause 45.</td>
</tr>
<tr>
<td>Note:</td>
<td></td>
<td></td>
<td>This port is available when Include AXI4-Lite is not selected in the Configuration tab.</td>
</tr>
<tr>
<td>ctl_tx_prbs31_test_pattern_enable_*</td>
<td>1</td>
<td>Input</td>
<td>Corresponds to MDIO register bit 3.42.4 as defined in Clause 45. Takes first precedence.</td>
</tr>
<tr>
<td>Note:</td>
<td></td>
<td></td>
<td>This port is available when Include AXI4-Lite is not selected in the GT Selection and Configuration tab and Select Core is PCS/ in the Configuration tab.</td>
</tr>
<tr>
<td>ctl_tx_ipg_value_*</td>
<td>4</td>
<td>Input</td>
<td>Corresponds to MDIO register bit 3.42.4 as defined in Clause 45. Takes first precedence.</td>
</tr>
<tr>
<td>Note:</td>
<td></td>
<td></td>
<td>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.</td>
</tr>
<tr>
<td>Note:</td>
<td></td>
<td></td>
<td>This port is available when Include AXI4-Lite is not selected in the Configuration tab and Select Core is PCS/ in the Configuration tab.</td>
</tr>
<tr>
<td>ctl_tx_custom_preamble_enable_*</td>
<td>1</td>
<td>Input</td>
<td>When asserted, this signal enables the use of tx_preamblein as a custom preamble instead of inserting a standard preamble.</td>
</tr>
<tr>
<td>Note:</td>
<td></td>
<td></td>
<td>This port is available when Include AXI4-Lite is not selected in the Configuration tab and Select Core is Ethernet MAC+PCS/ PMA and Include FIFO Logic is disabled in the MAC Options tab.</td>
</tr>
</tbody>
</table>
### Chapter 5: Example Design

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

<table>
<thead>
<tr>
<th>Name</th>
<th>Size</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>stat_tx_local_fault_*</td>
<td>1</td>
<td>Output</td>
<td>A value of 1 indicates the receive decoder state machine is in the TX_INIT state. This output is level sensitive.</td>
</tr>
<tr>
<td>stat_tx_fifo_error_*</td>
<td>1</td>
<td>Output</td>
<td>Indicates when TX FIFO goes into an underflow or overflow condition. Note: This port is available when Select Core is Ethernet PCS/PMA in the Configuration tab.</td>
</tr>
<tr>
<td>stat_tx_total_bytes_*</td>
<td>5</td>
<td>Output</td>
<td>Increment for the total number of bytes transmitted. Note: This port is available when Select Core is Ethernet MAC+PCS/PMA in the Configuration tab.</td>
</tr>
<tr>
<td>stat_tx_total_packets_*</td>
<td>1</td>
<td>Output</td>
<td>Increment for the total number of packets transmitted. Note: This port is available when Select Core is Ethernet MAC+PCS/PMA in the Configuration tab.</td>
</tr>
<tr>
<td>stat_tx_total_good_bytes_*</td>
<td>14</td>
<td>Output</td>
<td>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 Select Core is Ethernet MAC+PCS/PMA in the Configuration tab.</td>
</tr>
<tr>
<td>stat_tx_total_good_packets_*</td>
<td>1</td>
<td>Output</td>
<td>Increment for the total number of good packets transmitted. Note: This port is available when Select Core is Ethernet MAC+PCS/PMA in the Configuration tab.</td>
</tr>
<tr>
<td>stat_tx_bad_fcs_*</td>
<td>1</td>
<td>Output</td>
<td>Increment for packets greater than 64 bytes that have FCS errors. Note: This port is available when Select Core is Ethernet MAC+PCS/PMA in the Configuration tab.</td>
</tr>
<tr>
<td>stat_tx_packet_64_bytes_*</td>
<td>1</td>
<td>Output</td>
<td>Increment for good and bad packets transmitted that contain 64 bytes. Note: This port is available when Select Core is Ethernet MAC+PCS/PMA in the Configuration tab.</td>
</tr>
<tr>
<td>stat_tx_packet_65_127_bytes_*</td>
<td>1</td>
<td>Output</td>
<td>Increment for good and bad packets transmitted that contain 65 to 127 bytes. Note: This port is available when Select Core is Ethernet MAC+PCS/PMA in the Configuration tab.</td>
</tr>
</tbody>
</table>
### Table 5-2: Core XCI Top Level Port List (Cont’d)

<table>
<thead>
<tr>
<th>Name</th>
<th>Size</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>stat_tx_packet_128_255_bytes_*</td>
<td>1</td>
<td>Output</td>
<td>Increment for good and bad packets transmitted that contain 128 to 255 bytes.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>Note:</strong> This port is available when Select Core is Ethernet MAC+PCS/PMA in the Configuration tab.</td>
</tr>
<tr>
<td>stat_tx_packet_256_511_bytes_*</td>
<td>1</td>
<td>Output</td>
<td>Increment for good and bad packets transmitted that contain 256 to 511 bytes.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>Note:</strong> This port is available when Select Core is Ethernet MAC+PCS/PMA in the Configuration tab.</td>
</tr>
<tr>
<td>stat_tx_packet_512_1023_bytes_*</td>
<td>1</td>
<td>Output</td>
<td>Increment for good and bad packets transmitted that contain 512 to 1,023 bytes.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>Note:</strong> This port is available when Select Core is Ethernet MAC+PCS/PMA in the Configuration tab.</td>
</tr>
<tr>
<td>stat_tx_packet_1024_1518_bytes_*</td>
<td>1</td>
<td>Output</td>
<td>Increment for good and bad packets transmitted that contain 1,024 to 1,518 bytes.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>Note:</strong> This port is available when Select Core is Ethernet MAC+PCS/PMA in the Configuration tab.</td>
</tr>
<tr>
<td>stat_tx_packet_1519_1522_bytes_*</td>
<td>1</td>
<td>Output</td>
<td>Increment for good and bad packets transmitted that contain 1,519 to 1,522 bytes.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>Note:</strong> This port is available when Select Core is Ethernet MAC+PCS/PMA in the Configuration tab.</td>
</tr>
<tr>
<td>stat_tx_packet_1523_1548_bytes_*</td>
<td>1</td>
<td>Output</td>
<td>Increment for good and bad packets transmitted that contain 1,523 to 1,548 bytes.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>Note:</strong> This port is available when Select Core is Ethernet MAC+PCS/PMA in the Configuration tab.</td>
</tr>
<tr>
<td>stat_tx_packet_1549_2047_bytes_*</td>
<td>1</td>
<td>Output</td>
<td>Increment for good and bad packets transmitted that contain 1,549 to 2,047 bytes.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>Note:</strong> This port is available when Select Core is Ethernet MAC+PCS/PMA in the Configuration tab.</td>
</tr>
</tbody>
</table>
Table 5-2: Core xci Top Level Port List (Cont’d)

<table>
<thead>
<tr>
<th>Name</th>
<th>Size</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
</table>
| 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 Select Core is Ethernet MAC+PCS/PMA in the Configuration tab. |
| 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 Select Core is Ethernet MAC+PCS/PMA in the Configuration tab. |
| 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 Select Core is Ethernet MAC+PCS/PMA in the Configuration tab. |
| 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 Select Core is Ethernet MAC+PCS/PMA in the Configuration tab. |
| stat_tx_packet_large_*                    | 1    | Output    | Increment for all packets that are more than 9,215 bytes long.  
**Note:** This port is available when Select Core is Ethernet MAC+PCS/PMA in the Configuration tab. |
| stat_tx_frame_error_*                     | 1    | Output    | Increment for packets with tx_errin set to indicate an EOP abort.  
**Note:** This port is available when Select Core is Ethernet MAC+PCS/PMA in the Configuration tab. |
### RX Path Control / Status / Statistics Signals

<table>
<thead>
<tr>
<th>Name</th>
<th>Size</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ctl_rx_enable_*</td>
<td>1</td>
<td>Input</td>
<td>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. <em>Note:</em> This port is available when <strong>Include AXI4-Lite</strong> is not selected in the Configuration tab and <strong>Select Core</strong> is Ethernet MAC+PCS/PMA.</td>
</tr>
<tr>
<td>ctl_rx_check_preamble_*</td>
<td>1</td>
<td>Input</td>
<td>When asserted, this input causes the Ethernet MAC to check the preamble of the received frame. <em>Note:</em> This port is available when <strong>Include AXI4-Lite</strong> is not selected in the Configuration tab and <strong>Select Core</strong> is Ethernet MAC+PCS/PMA.</td>
</tr>
<tr>
<td>ctl_rx_check_sfd_*</td>
<td>1</td>
<td>Input</td>
<td>When asserted, this input causes the Ethernet MAC to check the start of frame Delimiter of the received frame. <em>Note:</em> This port is available when <strong>Include AXI4-Lite</strong> is not selected in the Configuration tab and <strong>Select Core</strong> is Ethernet MAC+PCS/PMA.</td>
</tr>
<tr>
<td>ctl_rx_force_resync_*</td>
<td>1</td>
<td>Input</td>
<td>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. <em>Note:</em> This input should normally be Low and should only be pulsed (1 cycle minimum pulse) to force realignment. <em>Note:</em> This port is available when <strong>Include AXI4-Lite</strong> is not selected in the Configuration tab and <strong>Select Core</strong> is Ethernet MAC+PCS/PMA.</td>
</tr>
</tbody>
</table>
### Table 5-2: Core xci Top Level Port List (Cont’d)

<table>
<thead>
<tr>
<th>Name</th>
<th>Size</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ctl_rx_delete_fcs_*</td>
<td>1</td>
<td>Input</td>
<td>Enable FCS removal by the RX core. If this bit is set to 0, the HSEC core does not remove the FCS of the incoming packet. If this bit is set to 1, the HSEC core deletes the FCS to the received packet. FCS is not deleted for packets that are ( \leq 8 ) bytes long. This input should only be changed while the corresponding reset input is asserted. Note: This port is available when Include AXI4-Lite is not selected in the Configuration tab and Select Core is Ethernet MAC+PCS/PMA.</td>
</tr>
<tr>
<td>ctl_rx_ignore_fcs_*</td>
<td>1</td>
<td>Input</td>
<td>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 HSEC core does not flag an FCS error at the user interface. Note: The statistics are reported as if the packet is good. The stat_rx_bad_fcs signal, however, reports the error. Note: This port is available when Include AXI4-Lite is not selected in the Configuration tab and Select Core is Ethernet MAC+PCS/PMA.</td>
</tr>
<tr>
<td>ctl_rx_max_packet_len_*</td>
<td>15</td>
<td>Input</td>
<td>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. Note: This port is available when Include AXI4-Lite is not selected in the Configuration tab and Select Core is Ethernet MAC+PCS/PMA.</td>
</tr>
</tbody>
</table>
### Table 5-2: Core xci Top Level Port List (Cont’d)

<table>
<thead>
<tr>
<th>Name</th>
<th>Size</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
</table>
| ctl_rx_min_packet_len_*             | 8    | Input     | 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.  
**Note:** This port is available when Include AXI4-Lite is not selected in the Configuration tab and Select Core 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 transceiver. When set to 0, the RX core ignores LF control codes coming in from the transceiver.  
**Note:** This port is available when Include AXI4-Lite is not selected in the Configuration tab and Select Core is Ethernet MAC+PCS/PMA. |
| ctl_rx_test_pattern_*               | 1    | Input     | Test pattern checking enable for the RX core. A value of 1 enables test mode as defined in Clause 49. Corresponds to MDIO register bit 3.42.2 as defined in Clause 45. Checks for scrambled idle pattern.  
**Note:** This port is available when Include AXI4-Lite is not selected in the Configuration tab. |
| ctl_rx_data_pattern_select_*        |      |           | Corresponds to MDIO register bit 3.42.0 as defined in Clause 45.  
**Note:** This port is available when Include AXI4-Lite is not selected in the Configuration tab and Select Core is Ethernet MAC+PCS/PMA and the Include FIFO Logic is disabled. |
| ctl_rx_test_pattern_enable_*        |      |           | Test pattern enable for the RX core. A value of 1 enables test mode. Corresponds to MDIO register bit 3.42.2 as defined in Clause 45. Takes second precedence.  
**Note:** This port is available when Include AXI4-Lite is not selected in the Configuration tab and Select Core is Ethernet MAC+PCS/PMA and the Include FIFO Logic is disabled. |
### Table 5-2: Core xci Top Level Port List (Cont’d)

<table>
<thead>
<tr>
<th>Name</th>
<th>Size</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
</table>
| ctl_rx_rate_10g_25gn_*                    |      | Input     | This port indicates the lane rate of the mission mode. 0 corresponds to 25G and 1 corresponds to 10G.  
**Note:** This port is available when Runtime Switching Mode is selected in the Configuration tab. |
| ctl_tx_prbs31_test_pattern_enable_*       | 1    | Input     | Corresponds to MDIO register bit 3.42.1 as defined in Clause 45.  
**Note:** This port is available when Include AXI4-Lite is not selected in the Configuration tab and Select Core is PCS/PMA. |
| 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 Include AXI4-Lite is not selected in the Configuration tab and Select Core is Ethernet MAC+PCS/PMA and the Include FIFO Logic is disabled. |
<p>| 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_*               | 1    | Output    | Valid indicator for stat_rx_framing_err. When 1 stat_rx_framing_err_0 is valid. |
| stat_rx_framing_err_*                     | 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 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_hi_ber_*                          | 1    | Output    | High Bit Error Rate (BER) indicator. When set to 1, the BER is too high as defined by IEEE Std 802.3-2015. Corresponds to MDIO register bit 3.32.1 as defined in Clause 82.3. This output is level sensitive. |</p>
<table>
<thead>
<tr>
<th>Name</th>
<th>Size</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>stat_rx_bad_code_*</td>
<td>2</td>
<td>Output</td>
<td>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 802.3-2015. This output can be used to generate MDIO register 3.33:7:0 as defined in Clause 82.3.</td>
</tr>
<tr>
<td>stat_rx_bad_code_valid_*</td>
<td>1</td>
<td>Output</td>
<td>Indicates when stat_rx_bad_code is valid.</td>
</tr>
<tr>
<td>stat_rx_error_valid_*</td>
<td>1</td>
<td>Output</td>
<td>Indicates when stat_rx_error is valid.</td>
</tr>
<tr>
<td>stat_rx_error_*</td>
<td>8</td>
<td>Output</td>
<td>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.</td>
</tr>
<tr>
<td>stat_rx_fifo_error_*</td>
<td>1</td>
<td>Output</td>
<td>Indicates when RX FIFO goes into an underflow or overflow condition.</td>
</tr>
<tr>
<td>stat_rx_total_packets_*</td>
<td>2</td>
<td>Output</td>
<td>Increment for the total number of packets received.</td>
</tr>
<tr>
<td>stat_rx_total_good_packets_*</td>
<td>1</td>
<td>Output</td>
<td>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.</td>
</tr>
<tr>
<td>stat_rx_total_bytes_*</td>
<td>6</td>
<td>Output</td>
<td>Increment for the total number of bytes received.</td>
</tr>
<tr>
<td>Name</td>
<td>Size</td>
<td>Direction</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>------</td>
<td>-----------</td>
<td>------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| 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 Select Core is Ethernet MAC+PCS/PMA in the Configuration tab. |
| 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 Select Core is Ethernet MAC+PCS/PMA in the Configuration tab. |
| stat_rx_jabber_*                 | 1    | Output    | Increment for packets longer than ctl_rx_max_packet_len with bad FCS. 
**Note:** This port is available when Select Core is Ethernet MAC+PCS/PMA in the Configuration tab. |
| stat_rx_packet_large_*           | 1    | Output    | Increment for all packets that are more than 9,215 bytes long. 
**Note:** This port is available when Select Core is Ethernet MAC+PCS/PMA in the Configuration tab. |
| stat_rx_oversize_*               | 1    | Output    | Increment for packets longer than ctl_rx_max_packet_len with good FCS. 
**Note:** This port is available when Select Core is Ethernet MAC+PCS/PMA in the Configuration tab. |
| stat_rx_undersize_*              | 2    | Output    | Increment for packets shorter than stat_rx_min_packet_len with good FCS. 
**Note:** This port is available when Select Core is Ethernet MAC+PCS/PMA in the Configuration tab. |
| stat_rx_toolong_*                | 1    | Output    | Increment for packets longer than ctl_rx_max_packet_len with good and bad FCS. 
**Note:** This port is available when Select Core is Ethernet MAC+PCS/PMA in the Configuration tab. |
| stat_rx_fragment_*               | 2    | Output    | Increment for packets shorter than stat_rx_min_packet_len with bad FCS. 
**Note:** This port is available when Select Core is Ethernet MAC+PCS/PMA in the Configuration tab. |
<table>
<thead>
<tr>
<th>Name</th>
<th>Size</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
</table>
| stat_rx_packet_64_bytes_* | 1    | Output    | Increment for good and bad packets received that contain 64 bytes.  
**Note**: This port is available when *Select Core* is *Ethernet MAC+PCS/PMA* in the Configuration tab. |
| 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 *Select Core* is *Ethernet MAC+PCS/PMA* in the Configuration tab. |
| 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 *Select Core* is *Ethernet MAC+PCS/PMA* in the Configuration tab. |
| 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 *Select Core* is *Ethernet MAC+PCS/PMA* in the Configuration tab. |
| 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 *Select Core* is *Ethernet MAC+PCS/PMA* in the Configuration tab. |
| 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 *Select Core* is *Ethernet MAC+PCS/PMA* in the Configuration tab. |
| 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 *Select Core* is *Ethernet MAC+PCS/PMA* in the Configuration tab. |
| 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 *Select Core* is *Ethernet MAC+PCS/PMA* in the Configuration tab. |
Table 5-2: Core xci Top Level Port List (Cont’d)

<table>
<thead>
<tr>
<th>Name</th>
<th>Size</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>stat_rx_packet_1549_2047_bytes_*</td>
<td>1</td>
<td>Output</td>
<td>Increment for good and bad packets received that contain 1,549 to 2,047 bytes. Note: This port is available when Select Core is Ethernet MAC+PCS/PMA in the Configuration tab.</td>
</tr>
<tr>
<td>stat_rx_packet_2048_4095_bytes_*</td>
<td>1</td>
<td>Output</td>
<td>Increment for good and bad packets received that contain 2,048 to 4,095 bytes. Note: This port is available when Select Core is Ethernet MAC+PCS/PMA in the Configuration tab.</td>
</tr>
<tr>
<td>stat_rx_packet_4096_8191_bytes_*</td>
<td>1</td>
<td>Output</td>
<td>Increment for good and bad packets received that contain 4,096 to 8,191 bytes. Note: This port is available when Select Core is Ethernet MAC+PCS/PMA in the Configuration tab.</td>
</tr>
<tr>
<td>stat_rx_packet_8192_9215_bytes_*</td>
<td>1</td>
<td>Output</td>
<td>Increment for good and bad packets received that contain 8,192 to 9,215 bytes. Note: This port is available when Select Core is Ethernet MAC+PCS/PMA in the Configuration tab.</td>
</tr>
<tr>
<td>stat_rx_bad_fcs_*</td>
<td>2</td>
<td>Output</td>
<td>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 Select Core is Ethernet MAC+PCS/PMA in the Configuration tab.</td>
</tr>
<tr>
<td>stat_rx_packet_bad_fcs_*</td>
<td>1</td>
<td>Output</td>
<td>Increment for packets between 64 and ctl_rx_max_packet_len bytes that have FCS errors. Note: This port is available when Select Core is Ethernet MAC+PCS/PMA in the Configuration tab.</td>
</tr>
</tbody>
</table>
### Table 5-2: Core xci Top Level Port List (Cont’d)

<table>
<thead>
<tr>
<th>Name</th>
<th>Size</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>stat_rx_stomped_fcs_*</td>
<td>2</td>
<td>Output</td>
<td>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.</td>
</tr>
<tr>
<td>stat_rx_bad_preamble_*</td>
<td>1</td>
<td>Output</td>
<td>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.</td>
</tr>
<tr>
<td>stat_rx_bad_sfd_*</td>
<td>1</td>
<td>Output</td>
<td>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.</td>
</tr>
<tr>
<td>stat_rx_got_signal_os_*</td>
<td>1</td>
<td>Output</td>
<td>Signal OS indication. If this bit is sampled as a 1, it indicates that a Signal OS word was received.</td>
</tr>
<tr>
<td>stat_rx_test_pattern_mismatch_*</td>
<td>2</td>
<td>Output</td>
<td>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.</td>
</tr>
</tbody>
</table>

*Note:* This port is available when Select Core is Ethernet MAC+PCS/PMA in the Configuration tab.
Table 5-2:  Core xci Top Level Port List (Cont’d)

<table>
<thead>
<tr>
<th>Name</th>
<th>Size</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
</table>
| 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 Select Core is Ethernet MAC+PCS/PMA in the Configuration tab. |
| 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 Select Core is Ethernet MAC+PCS/PMA in the Configuration tab. |
| 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 Select Core is Ethernet MAC+PCS/PMA in the Configuration tab. |
| 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 Select Core is Ethernet MAC+PCS/PMA in the Configuration tab. |
| stat_rx_valid_ctrl_code_* | 1    | Output    | Indicates that a PCS block with a valid control code was received.                                                                                                                                      |  |
### Table 5-2: Core xci Top Level Port List (Cont’d)

<table>
<thead>
<tr>
<th>Name</th>
<th>Size</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>TX Pause Interface Control / Status / Statistics Signals</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ctrl_tx_pause_req_*</td>
<td>9</td>
<td>Input</td>
<td>If a bit of this bus is set to 1, the core transmits a pause packet using the associated quanta value on the ctrl_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.</td>
</tr>
<tr>
<td>ctrl_tx_pause_enable_*</td>
<td>9</td>
<td>Input</td>
<td>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.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>Note:</strong> This port is available when Include AXI4-Lite is not selected in the Configuration tab.</td>
</tr>
<tr>
<td>ctrl_tx_resend_pause_*</td>
<td>1</td>
<td>Input</td>
<td>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.</td>
</tr>
<tr>
<td>ctrl_tx_pause_quanta0_*</td>
<td>16</td>
<td>Input</td>
<td>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 ctrl_tx_pause_quanta[8] is used for global pause operation. All other values are used for priority pause operation.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>Note:</strong> This port is available when Include AXI4-Lite is not selected in the Configuration tab.</td>
</tr>
<tr>
<td>ctrl_tx_pause_quanta1_*</td>
<td>16</td>
<td>Input</td>
<td>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 ctrl_tx_pause_quanta[8] is used for global pause operation. All other values are used for priority pause operation.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>Note:</strong> This port is available when Include AXI4-Lite is not selected in the Configuration tab.</td>
</tr>
<tr>
<td>Name</td>
<td>Size</td>
<td>Direction</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------</td>
<td>------</td>
<td>-----------</td>
<td>--------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>ctl_tx_pause_quanta2_*</td>
<td>16</td>
<td>Input</td>
<td>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. <em>Note:</em> This port is available when Include AXI4-Lite is not selected in the Configuration tab.</td>
</tr>
<tr>
<td>ctl_tx_pause_quanta3_*</td>
<td>16</td>
<td>Input</td>
<td>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. <em>Note:</em> This port is available when Include AXI4-Lite is not selected in the Configuration tab.</td>
</tr>
<tr>
<td>ctl_tx_pause_quanta4_*</td>
<td>16</td>
<td>Input</td>
<td>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. <em>Note:</em> This port is available when Include AXI4-Lite is not selected in the Configuration tab.</td>
</tr>
<tr>
<td>ctl_tx_pause_quanta5_*</td>
<td>16</td>
<td>Input</td>
<td>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. <em>Note:</em> This port is available when Include AXI4-Lite is not selected in the Configuration tab.</td>
</tr>
</tbody>
</table>
Table 5-2: Core xci Top Level Port List (Cont’d)

<table>
<thead>
<tr>
<th>Name</th>
<th>Size</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
</table>
| ctl_tx_pause_quanta6_*       | 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 Include AXI4-Lite is not selected in the Configuration tab.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| ctl_tx_pause_quanta7_*       | 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 Include AXI4-Lite is not selected in the Configuration tab.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| ctl_tx_pause_quanta8_*       | 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 Include AXI4-Lite is not selected in the Configuration tab.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| ctl_tx_pause_refresh_timer0_*| 16   | Input     | 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 `ctl_tx_pause_refresh_timer[8]` is used for global pause operation. All other values are used for priority pause operation.  
**Note:** This port is available when Include AXI4-Lite is not selected in the Configuration tab.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
Table 5-2: Core xci Top Level Port List (Cont’d)

<table>
<thead>
<tr>
<th>Name</th>
<th>Size</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ctl_tx_pause_refresh_timer1_*</td>
<td>16</td>
<td>Input</td>
<td>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 ctl_tx_pause_refresh_timer[8] is used for global pause operation. All other values are used for priority pause operation. <strong>Note:</strong> This port is available when Include AXI4-Lite is not selected in the Configuration tab.</td>
</tr>
<tr>
<td>ctl_tx_pause_refresh_timer2_*</td>
<td>16</td>
<td>Input</td>
<td>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 ctl_tx_pause_refresh_timer[8] is used for global pause operation. All other values are used for priority pause operation. <strong>Note:</strong> This port is available when Include AXI4-Lite is not selected in the Configuration tab.</td>
</tr>
<tr>
<td>ctl_tx_pause_refresh_timer3_*</td>
<td>16</td>
<td>Input</td>
<td>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 ctl_tx_pause_refresh_timer[8] is used for global pause operation. All other values are used for priority pause operation. <strong>Note:</strong> This port is available when Include AXI4-Lite is not selected in the Configuration tab.</td>
</tr>
<tr>
<td>ctl_tx_pause_refresh_timer4_*</td>
<td>16</td>
<td>Input</td>
<td>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 ctl_tx_pause_refresh_timer[8] is used for global pause operation. All other values are used for priority pause operation. <strong>Note:</strong> This port is available when Include AXI4-Lite is not selected in the Configuration tab.</td>
</tr>
</tbody>
</table>
### Table 5-2: Core xci Top Level Port List (Cont’d)

<table>
<thead>
<tr>
<th>Name</th>
<th>Size</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ctl_tx_pause_refresh_timer5_*</td>
<td>16</td>
<td>Input</td>
<td>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 ctl_tx_pause_refresh_timer[8] is used for global pause operation. All other values are used for priority pause operation. Note: This port is available when Include AXI4-Lite is not selected in the Configuration tab.</td>
</tr>
<tr>
<td>ctl_tx_pause_refresh_timer6_*</td>
<td>16</td>
<td>Input</td>
<td>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 ctl_tx_pause_refresh_timer[8] is used for global pause operation. All other values are used for priority pause operation. Note: This port is available when Include AXI4-Lite is not selected in the Configuration tab.</td>
</tr>
<tr>
<td>ctl_tx_pause_refresh_timer7_*</td>
<td>16</td>
<td>Input</td>
<td>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 ctl_tx_pause_refresh_timer[8] is used for global pause operation. All other values are used for priority pause operation. Note: This port is available when Include AXI4-Lite is not selected in the Configuration tab.</td>
</tr>
<tr>
<td>ctl_tx_pause_refresh_timer8_*</td>
<td>16</td>
<td>Input</td>
<td>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 ctl_tx_pause_refresh_timer[8] is used for global pause operation. All other values are used for priority pause operation. Note: This port is available when Include AXI4-Lite is not selected in the Configuration tab.</td>
</tr>
<tr>
<td>ctl_tx_da_gpp_*</td>
<td>48</td>
<td>Input</td>
<td>Destination address for transmitting global pause packets. Note: This port is available when Include AXI4-Lite is not selected in the Configuration tab.</td>
</tr>
</tbody>
</table>

**ctl_tx_da_gpp_***

- **Size**: 48
- **Direction**: Input
- **Description**: Destination address for transmitting global pause packets.
- **Note**: This port is available when Include AXI4-Lite is not selected in the Configuration tab.
### Table 5-2: Core xci Top Level Port List (Cont’d)

<table>
<thead>
<tr>
<th>Name</th>
<th>Size</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ctl_tx_sa_gpp_*</td>
<td>48</td>
<td>Input</td>
<td>Source address for transmitting global pause packets. <strong>Note:</strong> This port is available when Include AXI4-Lite is not selected in the Configuration tab.</td>
</tr>
<tr>
<td>ctl_tx_ethertype_gpp_*</td>
<td>16</td>
<td>Input</td>
<td>Ethertype for transmitting global pause packets. <strong>Note:</strong> This port is available when Include AXI4-Lite is not selected in the Configuration tab.</td>
</tr>
<tr>
<td>ctl_tx_opcode_gpp_*</td>
<td>16</td>
<td>Input</td>
<td>Opcode for transmitting global pause packets. <strong>Note:</strong> This port is available when Include AXI4-Lite is not selected in the Configuration tab.</td>
</tr>
<tr>
<td>ctl_tx_da_ppp_*</td>
<td>48</td>
<td>Input</td>
<td>Destination address for transmitting priority pause packets. <strong>Note:</strong> This port is available when Include AXI4-Lite is not selected in the Configuration tab.</td>
</tr>
<tr>
<td>ctl_tx_sa_ppp_*</td>
<td>48</td>
<td>Input</td>
<td>Source address for transmitting priority pause packets. <strong>Note:</strong> This port is available when Include AXI4-Lite is not selected in the Configuration tab.</td>
</tr>
<tr>
<td>ctl_tx_ethertype_ppp_*</td>
<td>16</td>
<td>Input</td>
<td>Ethertype for transmitting priority pause packets. <strong>Note:</strong> This port is available when Include AXI4-Lite is not selected in the Configuration tab.</td>
</tr>
<tr>
<td>ctl_tx_opcode_ppp_*</td>
<td>16</td>
<td>Input</td>
<td>Opcode for transmitting priority pause packets. <strong>Note:</strong> This port is available when Include AXI4-Lite is not selected in the Configuration tab.</td>
</tr>
<tr>
<td>stat_tx_pause_valid_*</td>
<td>9</td>
<td>Output</td>
<td>If a bit of this bus is set to 1, the HSEC core 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.</td>
</tr>
<tr>
<td>stat_tx_unicast_*</td>
<td>1</td>
<td>Output</td>
<td>Increment for good unicast packets.</td>
</tr>
<tr>
<td>stat_tx_multicast_*</td>
<td>1</td>
<td>Output</td>
<td>Increment for good multicast packets.</td>
</tr>
<tr>
<td>stat_tx_broadcast_*</td>
<td>1</td>
<td>Output</td>
<td>Increment for good broadcast packets.</td>
</tr>
</tbody>
</table>
Table 5-2: Core cxi Top Level Port List (Cont’d)

<table>
<thead>
<tr>
<th>Name</th>
<th>Size</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>stat_tx_vlan_*</td>
<td>1</td>
<td>Output</td>
<td>Increment for good 802.1Q tagged VLAN packets.</td>
</tr>
<tr>
<td>stat_tx_pause_*</td>
<td>1</td>
<td>Output</td>
<td>Increment for 802.3x Ethernet MAC Pause packet with good FCS.</td>
</tr>
<tr>
<td>stat_tx_user_pause_*</td>
<td>1</td>
<td>Output</td>
<td>Increment for priority-based pause packets with good FCS.</td>
</tr>
</tbody>
</table>

**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 Select Core is Ethernet MAC+PCS/PMA.

<table>
<thead>
<tr>
<th>Name</th>
<th>Size</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
</table>
| ctl_rx_forward_control_*      | 1    | Input     | A value of 1 indicates that the CORE forwards control packets to you. A value of 0 causes CORE to drop control packets.  
*Note:* This port is available when Include AXI4-Lite is not selected in the Configuration tab. |
| 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 Include AXI4-Lite is not selected in the Configuration tab. |
| 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 Include AXI4-Lite is not selected in the Configuration tab. |
| ctl_rx_enable_gcp_*           | 1    | Input     | A value of 1 enables global control packet processing.  
*Note:* This port is available when Include AXI4-Lite is not selected in the Configuration tab. |
| ctl_rx_check_mcast_gcp_*      | 1    | Input     | A value of 1 enables global control multicast destination address processing.  
*Note:* This port is available when Include AXI4-Lite is not selected in the Configuration tab. |
## Table 5-2: Core xci Top Level Port List (Cont’d)

<table>
<thead>
<tr>
<th>Name</th>
<th>Size</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ctl_rx_check_ucaст_gcp_*</td>
<td>1</td>
<td>Input</td>
<td>A value of 1 enables global control unicast destination address processing.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><em>Note:</em> This port is available when Include AXI4-Lite is not selected in the Configuration tab.</td>
</tr>
<tr>
<td>ctl_rx_pause_da_ucaст_*</td>
<td>48</td>
<td>Input</td>
<td>Unicast destination address for pause processing.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><em>Note:</em> This port is available when Include AXI4-Lite is not selected in the Configuration tab.</td>
</tr>
<tr>
<td>ctl_rx_check_sа_gcp_*</td>
<td>1</td>
<td>Input</td>
<td>A value of 1 enables global control source address processing.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><em>Note:</em> This port is available when Include AXI4-Lite is not selected in the Configuration tab.</td>
</tr>
<tr>
<td>ctl_rx_pause_sа_*</td>
<td>48</td>
<td>Input</td>
<td>Source address for pause processing.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><em>Note:</em> This port is available when Include AXI4-Lite is not selected in the Configuration tab.</td>
</tr>
<tr>
<td>ctl_rx_check_etype_gcp_*</td>
<td>1</td>
<td>Input</td>
<td>A value of 1 enables global control ethertype processing.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><em>Note:</em> This port is available when Include AXI4-Lite is not selected in the Configuration tab.</td>
</tr>
<tr>
<td>ctl_rx_etype_gcp_*</td>
<td>16</td>
<td>Input</td>
<td>Ethertype field for global control processing.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><em>Note:</em> This port is available when Include AXI4-Lite is not selected in the Configuration tab.</td>
</tr>
<tr>
<td>ctl_rx_check_opcode_gcp_*</td>
<td>1</td>
<td>Input</td>
<td>A value of 1 enables global control opcode processing.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><em>Note:</em> This port is available when Include AXI4-Lite is not selected in the Configuration tab.</td>
</tr>
<tr>
<td>ctl_rx_opcode_min_gcp_*</td>
<td>16</td>
<td>Input</td>
<td>Minimum global control opcode value.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><em>Note:</em> This port is available when Include AXI4-Lite is not selected in the Configuration tab.</td>
</tr>
<tr>
<td>ctl_rx_opcode_max_gcp_*</td>
<td>16</td>
<td>Input</td>
<td>Maximum global control opcode value.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><em>Note:</em> This port is available when Include AXI4-Lite is not selected in the Configuration tab.</td>
</tr>
</tbody>
</table>
### Table 5-2: Core xci Top Level Port List (Cont’d)

<table>
<thead>
<tr>
<th>Name</th>
<th>Size</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ctl_rx_enable_pcp_*</td>
<td>1</td>
<td>Input</td>
<td>A value of 1 enables priority control packet processing.</td>
</tr>
<tr>
<td>Note: This port is available when Include AXI4-Lite is not selected in the Configuration tab.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ctl_rx_check_mcast_pcp_*</td>
<td>1</td>
<td>Input</td>
<td>A value of 1 enables priority control multicast destination address processing.</td>
</tr>
<tr>
<td>Note: This port is available when Include AXI4-Lite is not selected in the Configuration tab.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ctl_rx_check_ucast_pcp_*</td>
<td>1</td>
<td>Input</td>
<td>A value of 1 enables priority control unicast destination address processing.</td>
</tr>
<tr>
<td>Note: This port is available when Include AXI4-Lite is not selected in the Configuration tab.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ctl_rx_pause_da_mcast_*</td>
<td>48</td>
<td>Input</td>
<td>Multicast destination address for pause processing.</td>
</tr>
<tr>
<td>Note: This port is available when Include AXI4-Lite is not selected in the Configuration tab.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ctl_rx_check_sa_pcp_*</td>
<td>1</td>
<td>Input</td>
<td>A value of 1 enables priority control source address processing.</td>
</tr>
<tr>
<td>Note: This port is available when Include AXI4-Lite is not selected in the Configuration tab.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ctl_rx_check_etype_pcp_*</td>
<td>1</td>
<td>Input</td>
<td>A value of 1 enables priority control ethertype processing.</td>
</tr>
<tr>
<td>Note: This port is available when Include AXI4-Lite is not selected in the Configuration tab.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ctl_rx_etype_pcp_*</td>
<td>16</td>
<td>Input</td>
<td>Ethertype field for priority control processing.</td>
</tr>
<tr>
<td>Note: This port is available when Include AXI4-Lite is not selected in the Configuration tab.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ctl_rx_check_opcode_pcp_*</td>
<td>1</td>
<td>Input</td>
<td>A value of 1 enables priority control opcode processing.</td>
</tr>
<tr>
<td>Note: This port is available when Include AXI4-Lite is not selected in the Configuration tab.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ctl_rx_opcode_min_pcp_*</td>
<td>16</td>
<td>Input</td>
<td>Minimum priority control opcode value.</td>
</tr>
<tr>
<td>Note: This port is available when Include AXI4-Lite is not selected in the Configuration tab.</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
**Table 5-2: Core xci Top Level Port List (Cont’d)**

<table>
<thead>
<tr>
<th>Name</th>
<th>Size</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
</table>
| ctl_rx_opcode_max_pcp_*      | 16   | Input     | Maximum priority control opcode value.  
*Note:* This port is available when Include AXI4-Lite is not selected in the Configuration tab. |
| ctl_rx_enable_gpp_*          | 1    | Input     | A value of 1 enables global pause packet processing.  
*Note:* This port is available when Include AXI4-Lite is not selected in the Configuration tab. |
| ctl_rx_check_mcast_gpp_*     | 1    | Input     | A value of 1 enables global pause multicast destination address processing.  
*Note:* This port is available when Include AXI4-Lite is not selected in the Configuration tab. |
| ctl_rx_check_ucast_gpp_*     | 1    | Input     | A value of 1 enables global pause unicast destination address processing.  
*Note:* This port is available when Include AXI4-Lite is not selected in the Configuration tab. |
| ctl_rx_check_sa_gpp_*        | 1    | Input     | A value of 1 enables global pause source address processing.  
*Note:* This port is available when Include AXI4-Lite is not selected in the Configuration tab. |
| ctl_rx_check_etype_gpp_*     | 1    | Input     | A value of 1 enables global pause ethertype processing.  
*Note:* This port is available when Include AXI4-Lite is not selected in the Configuration tab. |
| ctl_rx_etype_gpp_*           | 16   | Input     | Ethertype field for global pause processing.  
*Note:* This port is available when Include AXI4-Lite is not selected in the Configuration tab. |
| ctl_rx_check_opcode_gpp_*    | 1    | Input     | A value of 1 enables global pause opcode processing.  
*Note:* This port is available when Include AXI4-Lite is not selected in the Configuration tab. |
| ctl_rx_opcode_gpp_*          | 16   | Input     | Global pause opcode value.  
*Note:* This port is available when Include AXI4-Lite is not selected in the Configuration tab. |
### Chapter 5: Example Design

#### Table 5-2: Core xci Top Level Port List (Cont’d)

<table>
<thead>
<tr>
<th>Name</th>
<th>Size</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ctl_rx_enable_ppp_*</td>
<td>1</td>
<td>Input</td>
<td>A value of 1 enables priority pause packet processing.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><em>Note:</em> This port is available when Include AXI4-Lite is not selected.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>in the Configuration tab.</td>
</tr>
<tr>
<td>ctl_rx_check_mcast_ppp_*</td>
<td>1</td>
<td>Input</td>
<td>A value of 1 enables priority pause multicast destination address processing.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><em>Note:</em> This port is available when Include AXI4-Lite is not selected.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>in the Configuration tab.</td>
</tr>
<tr>
<td>ctl_rx_check_ucast_ppp_*</td>
<td>1</td>
<td>Input</td>
<td>A value of 1 enables priority pause unicast destination address processing.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><em>Note:</em> This port is available when Include AXI4-Lite is not selected.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>in the Configuration tab.</td>
</tr>
<tr>
<td>ctl_rx_check_sa_ppp_*</td>
<td>1</td>
<td>Input</td>
<td>A value of 1 enables priority pause source address processing.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><em>Note:</em> This port is available when Include AXI4-Lite is not selected.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>in the Configuration tab.</td>
</tr>
<tr>
<td>ctl_rx_check_etype_ppp_*</td>
<td>1</td>
<td>Input</td>
<td>A value of 1 enables priority pause ethertype processing.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><em>Note:</em> This port is available when Include AXI4-Lite is not selected.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>in the Configuration tab.</td>
</tr>
<tr>
<td>ctl_rx_etype_ppp_*</td>
<td>16</td>
<td>Input</td>
<td>Ethertype field for priority pause processing.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><em>Note:</em> This port is available when Include AXI4-Lite is not selected.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>in the Configuration tab.</td>
</tr>
<tr>
<td>ctl_rx_check_opcode_ppp_*</td>
<td>1</td>
<td>Input</td>
<td>A value of 1 enables priority pause opcode processing.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><em>Note:</em> This port is available when Include AXI4-Lite is not selected.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>in the Configuration tab.</td>
</tr>
<tr>
<td>ctl_rx_opcode_ppp_*</td>
<td>16</td>
<td>Input</td>
<td>Priority pause opcode value.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><em>Note:</em> This port is available when Include AXI4-Lite is not selected.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>in the Configuration tab.</td>
</tr>
<tr>
<td>stat_rx_unicast_*</td>
<td>1</td>
<td>Output</td>
<td>Increment for good unicast packets.</td>
</tr>
<tr>
<td>stat_rx_multicast_*</td>
<td>1</td>
<td>Output</td>
<td>Increment for good multicast packets.</td>
</tr>
<tr>
<td>stat_rx_broadcast_*</td>
<td>1</td>
<td>Output</td>
<td>Increment for good broadcast packets.</td>
</tr>
</tbody>
</table>
### Table 5-2: Core xci Top Level Port List (Cont’d)

<table>
<thead>
<tr>
<th>Name</th>
<th>Size</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>stat_rx_vlan_*</td>
<td>1</td>
<td>Output</td>
<td>Increment for good 802.1Q tagged VLAN packets.</td>
</tr>
<tr>
<td>stat_rx_pause_*</td>
<td>1</td>
<td>Output</td>
<td>Increment for 802.3x Ethernet MAC Pause packet with good FCS.</td>
</tr>
<tr>
<td>stat_rx_user_pause_*</td>
<td>1</td>
<td>Output</td>
<td>Increment for priority-based pause packets with good FCS.</td>
</tr>
<tr>
<td>stat_rx_inrangeerr_*</td>
<td>1</td>
<td>Output</td>
<td>Increment for packets with Length field error but with good FCS.</td>
</tr>
<tr>
<td>stat_rx_pause_valid_*</td>
<td>9</td>
<td>Output</td>
<td>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.</td>
</tr>
<tr>
<td>stat_rx_pause_quanta0_*</td>
<td>16</td>
<td>Output</td>
<td>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].</td>
</tr>
<tr>
<td>stat_rx_pause_quanta1_*</td>
<td>16</td>
<td>Output</td>
<td>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].</td>
</tr>
<tr>
<td>stat_rx_pause_quanta2_*</td>
<td>16</td>
<td>Output</td>
<td>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].</td>
</tr>
<tr>
<td>stat_rx_pause_quanta3_*</td>
<td>16</td>
<td>Output</td>
<td>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].</td>
</tr>
<tr>
<td>stat_rx_pause_quanta4_*</td>
<td>16</td>
<td>Output</td>
<td>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].</td>
</tr>
</tbody>
</table>
Table 5-2: Core xci Top Level Port List (Cont’d)

<table>
<thead>
<tr>
<th>Name</th>
<th>Size</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>stat_rx_pause_quanta5_*</td>
<td>16</td>
<td>Output</td>
<td>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].</td>
</tr>
<tr>
<td>stat_rx_pause_quanta6_*</td>
<td>16</td>
<td>Output</td>
<td>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].</td>
</tr>
<tr>
<td>stat_rx_pause_quanta7_*</td>
<td>16</td>
<td>Output</td>
<td>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].</td>
</tr>
<tr>
<td>stat_rx_pause_quanta8_*</td>
<td>16</td>
<td>Output</td>
<td>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].</td>
</tr>
<tr>
<td>stat_rx_pause_req_*</td>
<td>9</td>
<td>Output</td>
<td>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.</td>
</tr>
</tbody>
</table>

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.

<table>
<thead>
<tr>
<th>Name</th>
<th>Size</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ctl_tx_systemtimerin_*</td>
<td>80</td>
<td>Input</td>
<td>System timer input for the TX.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>In normal clock mode, the time format is according to the IEEE 1588 format, with 48 bits for seconds and 32 bits for nanoseconds.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>In transparent clock mode, bit 63 is expected to be zero, bits 62:16 carry nanoseconds, and bits 15:0 carry fractional nanoseconds.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Refer to IEEE 1588v2 for the representational definitions.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>This input must be in the TX clock domain.</td>
</tr>
</tbody>
</table>
### Core xci Top Level Port List (Cont’d)

<table>
<thead>
<tr>
<th>Name</th>
<th>Size</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ctl_rx_systemtimerin_*</td>
<td>80</td>
<td>Input</td>
<td>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.</td>
</tr>
<tr>
<td>ctl_tx_ptp_1step_enable_*</td>
<td>1</td>
<td>Input</td>
<td>When set to 1, this bit enables 1-step operation.</td>
</tr>
<tr>
<td>ctl_tx_ptp_latency_adjust_*</td>
<td>11</td>
<td>Input</td>
<td>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.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>Note:</strong> This port is available when Include AXI4-Lite is not selected in the Configuration tab.</td>
</tr>
<tr>
<td>ctl_ptp_transpclk_mode_*</td>
<td>1</td>
<td>Input</td>
<td>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. The sign bit of the correction value is assumed to be 0 (positive time). It is expected that the corresponding incoming PTP packet correction field has already been adjusted with the proper RX timestamp.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>Note:</strong> This port is available when Include AXI4-Lite is not selected in the Configuration tab.</td>
</tr>
<tr>
<td>stat_tx_ptp_fifo_read_error_*</td>
<td>1</td>
<td>Output</td>
<td>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.</td>
</tr>
<tr>
<td>stat_tx_ptp_fifo_write_error_*</td>
<td>1</td>
<td>Output</td>
<td>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.</td>
</tr>
</tbody>
</table>
### Table 5-2: Core xci Top Level Port List (Cont’d)

<table>
<thead>
<tr>
<th>Name</th>
<th>Size</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
</table>
| 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  
|                       |      |           | • 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_valid_out_* | 1    | Output    | This bit indicates that a valid timestamp is being presented on the rx.  
|                       |      |           | **Note:** This will be present only when core is Ethernet Mac+PCS/PMA.  
| 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.                                                                 |
### Table 5-2: Core xci Top Level Port List (Cont’d)

<table>
<thead>
<tr>
<th>Name</th>
<th>Size</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tx_ptp_upd_chksum_in_*</td>
<td>1</td>
<td>Input</td>
<td>TX UPD checksum value.</td>
</tr>
<tr>
<td>Note: This port is available when PTP Operation mode is selected as one step in MAC options tab.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>tx_ptp_tstamp_offset_in_*</td>
<td>16</td>
<td>Input</td>
<td>TX PTP timestamp offset.</td>
</tr>
<tr>
<td>Note: This port is available when PTP Operation mode is selected as one step in MAC options tab.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>tx_ptp_chksum_offset_in_*</td>
<td>16</td>
<td>Input</td>
<td>TX PTP check sum offset.</td>
</tr>
<tr>
<td>Note: This port is available when PTP Operation mode is selected as one step in MAC options tab.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>tx_ptp_rxtstamp_in_*</td>
<td>64</td>
<td>Input</td>
<td>TX PTP rx timestamp.</td>
</tr>
<tr>
<td>Note: This port is available when PTP Operation mode is selected as one step in MAC options tab.</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### AN and LT Interface Control / Status / Statistics Signals

Ports under this section will be available when the Include AN/LT Logic is selected from the Configuration tab.

<table>
<thead>
<tr>
<th>Name</th>
<th>Size</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>an_clk_*</td>
<td>1</td>
<td>Input</td>
<td>Input Clock for the Auto-Negotiation circuit.</td>
</tr>
<tr>
<td>an_reset_*</td>
<td>1</td>
<td>Input</td>
<td>Asynchronous active-High reset corresponding to an_clk domain.</td>
</tr>
<tr>
<td>an_loc_np_data_*</td>
<td>48</td>
<td>Input</td>
<td>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, &amp; T, bit positions 15, 14, 12, &amp; 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.</td>
</tr>
<tr>
<td>an_lp_np_data_*</td>
<td>48</td>
<td>Output</td>
<td>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.</td>
</tr>
<tr>
<td>lt_tx_sof_*</td>
<td>4</td>
<td>Output</td>
<td>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.</td>
</tr>
</tbody>
</table>
### Table 5-2: Core xci Top Level Port List (Cont’d)

<table>
<thead>
<tr>
<th>Name</th>
<th>Size</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ctl_autoneg_enable_*</td>
<td>1</td>
<td>Input</td>
<td>Enable signal for auto negotiation.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>Note:</strong> This port is available when Include AXI4-Lite is not selected in the Configuration tab.</td>
</tr>
<tr>
<td>ctl_autoneg_bypass_*</td>
<td>1</td>
<td>Input</td>
<td>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.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>Note:</strong> This port is available when Include AXI4-Lite is not selected in the Configuration tab.</td>
</tr>
<tr>
<td>ctl_an_nonce_seed_*</td>
<td>8</td>
<td>Input</td>
<td>8-bit seed to initialize the nonce field polynomial generator.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>Note:</strong> This port is available when Include AXI4-Lite is not selected in the Configuration tab.</td>
</tr>
<tr>
<td>ctl_an_pseudo_sel_*</td>
<td>1</td>
<td>Input</td>
<td>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).</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>Note:</strong> This port is available when Include AXI4-Lite is not selected in the Configuration tab.</td>
</tr>
<tr>
<td>ctl_restart_negotiation_*</td>
<td>1</td>
<td>Input</td>
<td>This input is used to trigger a restart of the auto negotiation, regardless of what state the circuit is currently in.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>Note:</strong> This port is available when Include AXI4-Lite is not selected in the Configuration tab.</td>
</tr>
<tr>
<td>ctl_an_local_fault_*</td>
<td>1</td>
<td>Input</td>
<td>This input signal is used to set the local_fault bit of the transmit link codeword.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>Note:</strong> This port is available when Include AXI4-Lite is not selected in the Configuration tab.</td>
</tr>
<tr>
<td>ctl_an_pause_*</td>
<td>1</td>
<td>Input</td>
<td>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.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>Note:</strong> This port is available when Include AXI4-Lite is not selected in the Configuration tab.</td>
</tr>
</tbody>
</table>
### Table 5-2: Core xci Top Level Port List (Cont’d)

<table>
<thead>
<tr>
<th>Name</th>
<th>Size</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ctl_an_asmdir_*</td>
<td>1</td>
<td>Input</td>
<td>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 Include AXI4-Lite is not selected in the Configuration tab.]</td>
</tr>
<tr>
<td>ctl_an_fec_10g_request_*</td>
<td>1</td>
<td>Input</td>
<td>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 Include AXI4-Lite is not selected in the Configuration tab.]</td>
</tr>
<tr>
<td>ctl_an_fec_25g_rs_request_*</td>
<td>1</td>
<td>Input</td>
<td>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 Include AXI4-Lite is not selected in the Configuration tab.]</td>
</tr>
<tr>
<td>ctl_an_fec_25g_baser_request_*</td>
<td>1</td>
<td>Input</td>
<td>Indicates the baser FEC request.  [Note: Note: This port is available when Include AXI4-Lite is not selected in the Configuration tab.]</td>
</tr>
<tr>
<td>ctl_an_fec_ability_override_*</td>
<td>1</td>
<td>Input</td>
<td>Used to set the clause 74 FEC ability bit in the transmit link codeword. If this input is set, then 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 Include AXI4-Lite is not selected in the Configuration tab.]</td>
</tr>
<tr>
<td>ctl_an_loc_np_*</td>
<td>1</td>
<td>Input</td>
<td>Local Next Page indicator. If this bit is 1, then the AN IP transfers the next page word at input loc_np_data to the remote link partner. If this bit is 0, then 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, then the AN IP transfers null message pages.</td>
</tr>
</tbody>
</table>
### Table 5-2: Core xci Top Level Port List (Cont’d)

<table>
<thead>
<tr>
<th>Name</th>
<th>Size</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ctl_an_lp_np_ack_*</td>
<td>1</td>
<td>Input</td>
<td>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.</td>
</tr>
<tr>
<td>ctl_an_cl91_fec_request_*</td>
<td>1</td>
<td>Input</td>
<td>This bit is used to request clause 91 FEC. <strong>Note:</strong> This port is available when Include AXI4-Lite is not selected in the Configuration tab.</td>
</tr>
<tr>
<td>ctl_an_cl91_fec_ability_*</td>
<td>1</td>
<td>Input</td>
<td>This bit is used to set clause 91 FEC ability. <strong>Note:</strong> This port is available when Include AXI4-Lite is not selected in the Configuration tab.</td>
</tr>
</tbody>
</table>
### Table 5-2: Core xci Top Level Port List (Cont’d)

<table>
<thead>
<tr>
<th>Name</th>
<th>Size</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ctl_an_ability_1000base_kx_*</td>
<td>1</td>
<td>Input</td>
<td>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.</td>
</tr>
<tr>
<td>ctl_an_ability_10gbase_kx4_*</td>
<td>1</td>
<td>Input</td>
<td></td>
</tr>
<tr>
<td>ctl_an_ability_10gbase_kr_*</td>
<td>1</td>
<td>Input</td>
<td></td>
</tr>
<tr>
<td>ctl_an_ability_40gbase_kr4_*</td>
<td>1</td>
<td>Input</td>
<td></td>
</tr>
<tr>
<td>ctl_an_ability_40gbase_cr4_*</td>
<td>1</td>
<td>Input</td>
<td></td>
</tr>
<tr>
<td>ctl_an_ability_100gbase_cr10_*</td>
<td>1</td>
<td>Input</td>
<td></td>
</tr>
<tr>
<td>ctl_an_ability_100gbase_kp4_*</td>
<td>1</td>
<td>Input</td>
<td></td>
</tr>
<tr>
<td>ctl_an_ability_100gbase_kr4_*</td>
<td>1</td>
<td>Input</td>
<td></td>
</tr>
<tr>
<td>ctl_an_ability_100gbase_cr4_*</td>
<td>1</td>
<td>Input</td>
<td></td>
</tr>
<tr>
<td>ctl_an_ability_25gbase_krcr_s_*</td>
<td>1</td>
<td>Input</td>
<td></td>
</tr>
<tr>
<td>ctl_an_ability_25gbase_krcr_*</td>
<td>1</td>
<td>Input</td>
<td></td>
</tr>
<tr>
<td>ctl_an_ability_25gbase_kr1_*</td>
<td>1</td>
<td>Input</td>
<td></td>
</tr>
<tr>
<td>ctl_an_ability_25gbase_cr1_*</td>
<td>1</td>
<td>Input</td>
<td></td>
</tr>
<tr>
<td>ctl_an_ability_50gbase_kr2_*</td>
<td>1</td>
<td>Input</td>
<td></td>
</tr>
<tr>
<td>ctl_an_ability_50gbase_cr2_*</td>
<td>1</td>
<td>Input</td>
<td></td>
</tr>
<tr>
<td>ctl_lt_training_enable_*</td>
<td>1</td>
<td>Input</td>
<td>Enables link training. When link training is disabled, all PCS lanes function in mission mode.</td>
</tr>
<tr>
<td>Note: This port is available when include AXI4-Lite is not selected in the Configuration tab.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ctl_lt_restart_training_*</td>
<td>1</td>
<td>Input</td>
<td>This signal triggers a restart of link training regardless of the current state.</td>
</tr>
<tr>
<td>Note: This port is available when include AXI4-Lite is not selected in the Configuration tab.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ctl_lt_rx_trained_*</td>
<td>4</td>
<td>Input</td>
<td>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.</td>
</tr>
<tr>
<td>Note: This port is available when include AXI4-Lite is not selected in the Configuration tab.</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Table 5-2: Core xci Top Level Port List (Cont’d)

<table>
<thead>
<tr>
<th>Name</th>
<th>Size</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ctl_lt_preset_to_tx_*</td>
<td>4</td>
<td>Input</td>
<td>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.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>Note:</strong> This port is available when Include AXI4-Lite is not selected in the Configuration tab.</td>
</tr>
<tr>
<td>ctl_lt_initialize_to_tx_*</td>
<td>4</td>
<td>Input</td>
<td>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.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>Note:</strong> This port is available when Include AXI4-Lite is not selected in the Configuration tab.</td>
</tr>
<tr>
<td>ctl_lt_pseudo_seed0_*</td>
<td>11</td>
<td>Input</td>
<td>This 11-bit signal seeds the training pattern generator.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>Note:</strong> This port is available when Include AXI4-Lite is not selected in the Configuration tab.</td>
</tr>
<tr>
<td>ctl_lt_k_p1_to_tx0_*</td>
<td>2</td>
<td>Input</td>
<td>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.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>Note:</strong> This port is available when Include AXI4-Lite is not selected in the Configuration tab.</td>
</tr>
<tr>
<td>ctl_lt_k0_to_tx0_*</td>
<td>2</td>
<td>Input</td>
<td>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.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>Note:</strong> This port is available when Include AXI4-Lite is not selected in the Configuration tab.</td>
</tr>
<tr>
<td>ctl_lt_k_m1_to_tx0_*</td>
<td>2</td>
<td>Input</td>
<td>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.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>Note:</strong> This port is available when Include AXI4-Lite is not selected in the Configuration tab.</td>
</tr>
<tr>
<td>ctl_lt_stat_p1_to_tx0_*</td>
<td>2</td>
<td>Input</td>
<td>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.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>Note:</strong> This port is available when Include AXI4-Lite is not selected in the Configuration tab.</td>
</tr>
</tbody>
</table>
### Table 5-2: Core xci Top Level Port List (Cont’d)

<table>
<thead>
<tr>
<th>Name</th>
<th>Size</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ctl_lt_stat0_to_tx0_*</td>
<td>2</td>
<td>Input</td>
<td>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. <strong>Note:</strong> This port is available when <em>Include AXI4-Lite</em> is not selected in the Configuration tab.</td>
</tr>
<tr>
<td>ctl.lt.stat_m1_to_tx0_*</td>
<td>2</td>
<td>Input</td>
<td>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. <strong>Note:</strong> This port is available when <em>Include AXI4-Lite</em> is not selected in the Configuration tab.</td>
</tr>
<tr>
<td>stat_an_link_cntl_1000base_kx_*</td>
<td>2</td>
<td>Output</td>
<td>Link Control outputs from the auto negotiation controller for the various Ethernet protocols. Settings are as follows: 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</td>
</tr>
<tr>
<td>stat_an_link_cntl_10gbase_kx4_*</td>
<td>2</td>
<td>Output</td>
<td></td>
</tr>
<tr>
<td>stat_an_link_cntl_10gbase_kr_*</td>
<td>2</td>
<td>Output</td>
<td></td>
</tr>
<tr>
<td>stat_an_link_cntl_40gbase_kr4_*</td>
<td>2</td>
<td>Output</td>
<td></td>
</tr>
<tr>
<td>stat_an_link_cntl_40gbase_cr4_*</td>
<td>2</td>
<td>Output</td>
<td></td>
</tr>
<tr>
<td>stat_an_link_cntl_1000base_cr10_*</td>
<td>2</td>
<td>Output</td>
<td></td>
</tr>
<tr>
<td>stat_an_link_cntl_1000base_kp4_*</td>
<td>2</td>
<td>Output</td>
<td></td>
</tr>
<tr>
<td>stat_an_link_cntl_1000base_kr4_*</td>
<td>2</td>
<td>Output</td>
<td></td>
</tr>
<tr>
<td>stat_an_link_cntl_1000base_cr4_*</td>
<td>2</td>
<td>Output</td>
<td></td>
</tr>
<tr>
<td>stat_an_link_cntl_25gbase_krcr_s_*</td>
<td>2</td>
<td>Output</td>
<td></td>
</tr>
<tr>
<td>stat_an_link_cntl_25gbase_krcr_*</td>
<td>2</td>
<td>Output</td>
<td></td>
</tr>
<tr>
<td>stat_an_link_cntl_25gbase_kr1_*</td>
<td>2</td>
<td>Output</td>
<td></td>
</tr>
<tr>
<td>stat_an_link_cntl_50gbase_kr2_*</td>
<td>2</td>
<td>Output</td>
<td></td>
</tr>
<tr>
<td>stat_an_link_cntl_50gbase_cr2_*</td>
<td>2</td>
<td>Output</td>
<td></td>
</tr>
<tr>
<td>stat_an_fec_enable_*</td>
<td>1</td>
<td>Output</td>
<td>Used to enable the use of clause 74 FEC on the link.</td>
</tr>
</tbody>
</table>
### Table 5-2: Core xci Top Level Port List (Cont’d)

<table>
<thead>
<tr>
<th>Name</th>
<th>Size</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>stat_an_tx_pause_enable_*</td>
<td>1</td>
<td>Output</td>
<td>Used to enable station-to-station (global) pause packet generation in the transmit path to control data flow in the receive path.</td>
</tr>
<tr>
<td>stat_an_rx_pause_enable_*</td>
<td>1</td>
<td>Output</td>
<td>Used to enable station-to-station (global) pause packet interpretation in the receive path, in order to control data flow from the transmitter.</td>
</tr>
<tr>
<td>stat_an_autoneg_complete_*</td>
<td>1</td>
<td>Output</td>
<td>Indicates the auto-negotiation is complete and rx link status from the PCS has been received.</td>
</tr>
<tr>
<td>stat_an_parallel_detection_fault_*</td>
<td>1</td>
<td>Output</td>
<td>Indicated a parallel detection fault during auto-negotiation.</td>
</tr>
<tr>
<td>stat_an_lp_ability_1000base_kx_*</td>
<td>1</td>
<td>Output</td>
<td>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.</td>
</tr>
<tr>
<td>stat_an_lp_ability_10gbase_kx4_*</td>
<td>1</td>
<td>Output</td>
<td></td>
</tr>
<tr>
<td>stat_an_lp_ability_10gbase_kr_*</td>
<td>1</td>
<td>Output</td>
<td></td>
</tr>
<tr>
<td>stat_an_lp_ability_40gbase_kr4_*</td>
<td>1</td>
<td>Output</td>
<td></td>
</tr>
<tr>
<td>stat_an_lp_ability_40gbase_cr4_*</td>
<td>1</td>
<td>Output</td>
<td></td>
</tr>
<tr>
<td>stat_an_lp_ability_100gbase_cr10_*</td>
<td>1</td>
<td>Output</td>
<td></td>
</tr>
<tr>
<td>stat_an_lp_ability_100gbase_kp4_*</td>
<td>1</td>
<td>Output</td>
<td></td>
</tr>
<tr>
<td>stat_an_lp_ability_100gbase_kr4_*</td>
<td>1</td>
<td>Output</td>
<td></td>
</tr>
<tr>
<td>stat_an_lp_ability_100gbase_cr4_*</td>
<td>1</td>
<td>Output</td>
<td></td>
</tr>
<tr>
<td>stat_an_lp_ability_25gbase_krcr_s_*</td>
<td>1</td>
<td>Output</td>
<td></td>
</tr>
<tr>
<td>stat_an_lp_ability_25gbase_krcr_*</td>
<td>1</td>
<td>Output</td>
<td></td>
</tr>
<tr>
<td>stat_an_lp_ability_25gbase_cr1_*</td>
<td>1</td>
<td>Output</td>
<td>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.</td>
</tr>
<tr>
<td>stat_an_rxcdrhold_*</td>
<td>1</td>
<td>Output</td>
<td>Indicates the rx cdr hold signal.</td>
</tr>
</tbody>
</table>
### Table 5-2: Core xci Top Level Port List (Cont’d)

<table>
<thead>
<tr>
<th>Name</th>
<th>Size</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>stat_an_lp_pause_*</td>
<td>1</td>
<td>Output</td>
<td>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.</td>
</tr>
<tr>
<td>stat_an_lp_asm_dir_*</td>
<td>1</td>
<td>Output</td>
<td>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.</td>
</tr>
<tr>
<td>stat_an_lp_rf_*</td>
<td>1</td>
<td>Output</td>
<td>This bit indicates link partner remote fault.</td>
</tr>
<tr>
<td>stat_an_lp_fec_10g_ability_*</td>
<td>1</td>
<td>Output</td>
<td>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.</td>
</tr>
<tr>
<td>stat_an_lp_fec_10g_request_*</td>
<td>1</td>
<td>Output</td>
<td>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.</td>
</tr>
<tr>
<td>stat_an_lp_fec_25g_rs_request_*</td>
<td>1</td>
<td>Output</td>
<td>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.</td>
</tr>
<tr>
<td>stat_an_lp_fec_25g_baser_request_*</td>
<td>1</td>
<td>Output</td>
<td>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.</td>
</tr>
<tr>
<td>stat_an_lp_autoneg_able_*</td>
<td>1</td>
<td>Output</td>
<td>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.</td>
</tr>
<tr>
<td>stat_an_lp_ability_valid_*</td>
<td>1</td>
<td>Output</td>
<td>This signal indicates when all of the link partner advertisements become valid.</td>
</tr>
</tbody>
</table>
### Table 5-2: Core xci Top Level Port List (Cont’d)

<table>
<thead>
<tr>
<th>Name</th>
<th>Size</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>stat_an_loc_np_ack_*</td>
<td>1</td>
<td>Output</td>
<td>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, then it must clear the loc_np input.</td>
</tr>
<tr>
<td>stat_an_lp_np_*</td>
<td>1</td>
<td>Output</td>
<td>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.</td>
</tr>
<tr>
<td>stat_an_lp_ability_25gbase_kr1_*</td>
<td>1</td>
<td>Output</td>
<td>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.</td>
</tr>
<tr>
<td>stat_an_link_cntl_25gbase_cr1_*</td>
<td>1</td>
<td>Output</td>
<td>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.</td>
</tr>
<tr>
<td>stat_an_lp_ability_50gbase_kr2_*</td>
<td>1</td>
<td>Output</td>
<td>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.</td>
</tr>
<tr>
<td>Name</td>
<td>Size</td>
<td>Direction</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------------------------</td>
<td>------</td>
<td>-----------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>stat_an_lp_ability_50gbase_cr2_*</td>
<td>1</td>
<td>Output</td>
<td>Indicates the advertised protocol from the link partner. Becomes valid when the output signal <code>stat_an_lp_extended_ability_valid</code> is asserted. A value of 1 indicates that the protocol is advertised as supported by the link partner.</td>
</tr>
<tr>
<td>stat_an_lp_ability_extended_fec_*</td>
<td>4</td>
<td>Output</td>
<td>This output indicates the extended FEC abilities.</td>
</tr>
<tr>
<td>stat_an_rs_fec_enable_*</td>
<td>1</td>
<td>Output</td>
<td>Used to enable the use of clause 91 FEC on the link.</td>
</tr>
<tr>
<td>stat_an_lp_extended_ability_valid_*</td>
<td>1</td>
<td>Output</td>
<td>When this bit is 1, it indicates that the detected extended abilities are valid.</td>
</tr>
<tr>
<td>stat_lt_signal_detect_*</td>
<td>4</td>
<td>Output</td>
<td>This signal indicates when the respective link training state machine has entered the SEND_DATA state, in which normal PCS operation may resume.</td>
</tr>
<tr>
<td>stat_lt_training_*</td>
<td>4</td>
<td>Output</td>
<td>This signal indicates when the respective link training state machine is performing link training.</td>
</tr>
<tr>
<td>stat_lt_training_fail_*</td>
<td>4</td>
<td>Output</td>
<td>This signal is asserted during link training if the corresponding link training state machine detects a time-out during the training period.</td>
</tr>
<tr>
<td>stat_lt_rx_sof_*</td>
<td>4</td>
<td>Output</td>
<td>This output is High for 1 RX SerDes clock cycle to indicate the start of the link training frame.</td>
</tr>
<tr>
<td>stat_lt_frame_lock_*</td>
<td>4</td>
<td>Output</td>
<td>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.</td>
</tr>
<tr>
<td>stat_lt_preset_from_rx_*</td>
<td>4</td>
<td>Output</td>
<td>This signal reflects the value of the preset control bit received in the control block from the link partner.</td>
</tr>
<tr>
<td>stat_lt_initialize_from_rx_*</td>
<td>4</td>
<td>Output</td>
<td>This signal reflects the value of the initialize control bit received in the control block from the link partner.</td>
</tr>
<tr>
<td>stat_lt_k1_from_rx0_*</td>
<td>2</td>
<td>Output</td>
<td>This 2-bit field indicates the update control bits for the ‘k+1’ coefficient, as received from the link partner in the control block.</td>
</tr>
<tr>
<td>stat_lt_k0_from_rx0_*</td>
<td>2</td>
<td>Output</td>
<td>This 2-bit field indicates the update control bits for the ‘k0’ coefficient, as received from the link partner in the control block.</td>
</tr>
</tbody>
</table>
### 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.

<table>
<thead>
<tr>
<th>Name</th>
<th>Size</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>stat_lt_k_m1_from_rx0_*</td>
<td>2</td>
<td>Output</td>
<td>This 2-bit field indicates the update control bits for the ‘k-1’ coefficient, as received from the link partner in the control block.</td>
</tr>
<tr>
<td>stat_lt_stat_p1_from_rx0_*</td>
<td>2</td>
<td>Output</td>
<td>This 2-bit field indicates the update status bits for the ‘k+1’ coefficient, as received from the link partner in the status block.</td>
</tr>
<tr>
<td>stat_lt_stat0_from_rx0_*</td>
<td>2</td>
<td>Output</td>
<td>This 2-bit fields indicates the update status bits for the ‘k0’ coefficient, as received from the link partner in the status block.</td>
</tr>
<tr>
<td>stat_lt_stat_m1_from_rx0_*</td>
<td>2</td>
<td>Output</td>
<td>This 2-bit field indicates the update status bits for the ‘k-1’ coefficient, as received from the link partner in the status block.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name</th>
<th>Size</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>stat_fec_inc_correct_count_*</td>
<td>4</td>
<td>Output</td>
<td>This signal will be asserted roughly every 32 words, while the \texttt{ctl_rx_fec_enable} is asserted, if the FEC decoder detected and corrected a bit errors in the corresponding frame.</td>
</tr>
<tr>
<td>stat_fec_inc_cant_correct_count_*</td>
<td>4</td>
<td>Output</td>
<td>This signal will be asserted roughly every 32 words, while the \texttt{ctl_rx_fec_enable} is asserted, if the FEC decoder detected bit correction error.</td>
</tr>
<tr>
<td>stat_fec_lock_error_*</td>
<td>4</td>
<td>Output</td>
<td>\texttt{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.</td>
</tr>
<tr>
<td>stat_fec_rx_lock_*</td>
<td>4</td>
<td>Output</td>
<td>This signal is asserted while the \texttt{ctl_fec_rx_enable} is asserted when the FEC decoder detects the frame boundary.</td>
</tr>
<tr>
<td>Name</td>
<td>Size</td>
<td>Direction</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------------</td>
<td>------</td>
<td>-----------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>IEEE Clause 108 (RS-FEC) Control / Status /Statistics Signals</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Port that will be available when IEEE Clause 108 (RS-FEC) is selected from Configuration tab</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ctl_rx_rsfec_enable_correction_*</td>
<td>1</td>
<td>Input</td>
<td>The setting on this bit takes effect after rx_resetn has been asserted Low (~rx_serdes_reset). New value is sampled on first cycle on reset.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Equivalent to MDIO register 1.200.0</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• 0: Decoder performs error detection without error correction (see IEEE 802.3802.3by section 91.5.3.3).</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• 1: the decoder also performs error correction.</td>
</tr>
<tr>
<td>ctl_rx_rsfec_enable_indication_*</td>
<td>1</td>
<td>Input</td>
<td>The setting on this bit takes effect after rx_resetn has been asserted Low (~rx_serdes_reset). New value sampled on the first cycle on reset.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Equivalent to MDIO register 1.200.1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• 0: Bypass the error indication function (see IEEE Std 802.3by section 91.5.3.3).</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• 1: Decoder indicates errors to the PCS sublayer.</td>
</tr>
<tr>
<td>ctl_rsfec_enable_*</td>
<td>1</td>
<td>Input</td>
<td>The setting on this bit takes effect after rx_resetn has been asserted Low (~rx_serdes_reset). New value sampled on the first cycle on reset.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Enable RS-FEC function. Note that some variants of the 10G/25G Subsystem can have individual RX and TX enable signals.</td>
</tr>
<tr>
<td>ctl_rsfec_ieee_error_indication_mode_*</td>
<td>1</td>
<td>Input</td>
<td>The setting on this bit takes effect after rx_resetn has been asserted Low (~rx_serdes_reset). New value sampled on the first cycle on reset.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• 1: Core conforms to the IEEE RS-FEC specification.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• 0: If ctl_rx_rsfec_enable_correction and ctl_rx_rsfec_enable_indication are set to zero, the RS decoder is bypassed.</td>
</tr>
</tbody>
</table>
### Table 5-2: Core xci Top Level Port List (Cont’d)

<table>
<thead>
<tr>
<th>Name</th>
<th>Size</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
</table>
| ctl_rsfec_consortium_25g_*                  | 1    | Input     | Switches between IEEE Clause 108 and 25G Ethernet Consortium mode. The setting on this bit takes effect after \( \text{rx\_resetn} \) has been asserted Low (\( \sim \text{rx\_serdes\_reset} \)). New value is sampled on the first cycle on reset.  
  - 1 = 25G Consortium specification mode;  
  - 0 = IEEE 802.3by mode  
  Note that some variants of the 10G/25G Subsystem can have individual RX and TX consortium signals. |
| stat_rx_rsfec_hi_ser_*                      | 1    | Output    | Set to one if the number of RS-FEC symbol errors in a window of 8192 codewords exceeds the threshold \( K = 417 \) and is set to zero otherwise. |
| stat_rx_rsfec_lane_alignment_status_*       | 1    | Output    | A value of 1 indicates that the RX RS-FEC block has achieved alignment on the data from the transceiver.                                      |
| stat_rx_rsfec_corrected_cw_inc_*           | 1    | Output    | Increment for corrected errors.                                                                                                             |
| stat_rx_rsfec_uncorrected_cw_inc_*         | 1    | Output    | Increment for uncorrected errors.                                                                                                           |
| stat_rx_rsfec_err_count0_inc_*             | 3    | Output    | A value of 1 indicates that the detected errors.                                                                                             |
| stat_tx_rsfec_lane_alignment_status_*       | 1    | Output    | A value of 1 indicates that the TX RS-FEC block has achieved alignment on the incoming PCS data.                                              |

**PLL and SYS clock select lines when RUNTIME SWITCH feature selected.**

<table>
<thead>
<tr>
<th>Name</th>
<th>Size</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
</table>
| gt_drp_done_*                       | 1    | Input     | Indicates the completion of the GT DRP operation. Used to reset the GT module.  
  **Note:** This port is available when Runtime Switchable Mode is selected in the Configuration tab. |
| txpllclksel_in_*                    | 1    | Input     | TX PLL clock select lines  
  **Note:** This port is available when Runtime Switchable Mode is selected in the Configuration tab. |
| rxpllclksel_in_*                    | 1    | Input     | RX PLL clock select lines  
  **Note:** This port is available when Runtime Switchable Mode is selected in the Configuration tab. |
<table>
<thead>
<tr>
<th>Name</th>
<th>Size</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
</table>
| txsysclksel_in_*     | 1    | Input     | Select the TX PLL clock source to drive txoutclk  
*Note:* This port is available when Runtime Switchable Mode is selected in the Configuration tab.                                                                                             |
| rxsysclksel_in_*     | 1    | Input     | Select the RX PLL clock source to drive rxoutclk  
*Note:* This port is available when Runtime Switchable Mode is selected in the Configuration tab.                                                                                   |
| rxafecfoken_*        | 1    | Input     | Only for UltraScale+™ devices                                                                                                                                                                              |
| rxdfecfokfnum_*      | 1    | Input     | Only for UltraScale+ devices                                                                                                                                                                               |
| speed_*              | 1    | Input     | This signal indicates the speed with which the core is working:  
*1'b1 = 10G and 1'b0 = 25G*  
*Note:* This port is available when Runtime Switchable Mode is selected in the Configuration tab.                                                                                   |
| anlt_done_*          | 1    | Output    | Indicates the completion of Auto negotiation and Link Training  
*Note:* This port is available when Runtime Switchable Mode is selected in the Configuration tab.                                                                                     |
Duplex Mode of Operation

In this mode of operation, both the transmitter and receiver of the core are active and loopback 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 transmit to the core using the available data interface. The monitor module checks the packets from the receiver.

Figure 5-5 shows the duplex mode of operation.

![Figure 5-5: Duplex Mode of Operation](image)

Runtime Switchable

This configuration gives the flexibility to switch the line rate between 10G to 25G and vice-versa any time. To activate this feature, select the check box **Runtime Switchable mode** option in the Configuration tab.

When this option is selected the 
*_trans_debug* module will be present inside the
*_pkt_gen_mon.v* module of the example design. This *
*_trans_debug* module is responsible for performing all the GT DRP write operations to switch the transceiver mode, that is, 10G to 25G or 25G to 10G. When you set the *mode_change_* * input signal High for two clock cycles and then make it Low, it starts the DRP write operation to the GT channel for the specific core and resets the specific core.

The DRP writes are done only for the channel. The QPLL0 of the common is fixed for the line rate 25G and the QPLL1 is fixed for the line rate 10G.
The state transition occurred during this process is shown in Figure 5-6:

![State Transition Diagram for Runtime Switchable DRP Operation without AN/LT](image-url)
Chapter 5: Example Design

Figure 5-7: State Transition Diagram for Runtime Switchable DRP Operation with AN/LT
Shared Logic Implementation

Shared logic includes all the shareable modules that can be present as part of the core or in the Example Design.

By default GT common, reset logic and clocking modules are present inside the IP core. In case of the following conditions, these modules will be placed outside the core so that they can be shared with other designs.

- When you select the **Include GT subcore in example design** option in the GT Selection and Configuration tab.
- When you select the **Include Shared Logic in Example Design** option in the Shared Logic tab.

When the shared logic in the example design is selected, a new `xxv_ethernet_*_core_support.v` module will be instantiated between the `xxv_ethernet_*_exdes.v` and DUT (that is, `xxv_ethernet_*_.v`). This module will have all the sub modules that can be shared between multiple designs.

*Figure 5-8* shows the implementation when shared logic is instantiated in the example design for single core.
Figure 5-8: Single Core Example Design Hierarchy With Shared Logic Implementation
**Figure 5-9** shows the implementation when shared logic is instantiated in the example design for multiple cores.

The following modules are the part of shared logic wrapper.

- *clocking_wrapper*
  
  This module contains all the clk resources that can be shared with other designs.

- *common_wrapper*
  
  This module contains the GT common that can be shared with other designs.

- *reset_wrapper*
  
  This module contains all the reset logics for the specific selected Vivado IDE configuration.
AXI4-Lite Interface Implementation

In order to instantiate the AXI4-Lite interface to access the control and status registers of the xxv_ethernet core, enable the **Include AXI4-Lite** check box in the Configuration Tab of the Vivado IDE. This option enables the xxv_ethernet _0_axi_if_top module (which contains xxv_ethernet_0_pif_registers with the xxv_ethernet _0_slave_2_ipif module). You can access the AXI4-Lite interface logic registers (control, status and statistics) from the xxv_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 operation is performed by reading the address locations for the status and statistics registers through register map.

AXI4 Interface User Logic

The following sections provide the AXI4-Lite interface state machine control and ports.

**User State Machine**

The read and write through the AXI4-Lite slave module interface is controlled by a state machine as shown below:

![User State Machine for AXI4-Lite Interface](image)

*Figure 5-10: User State Machine for AXI4-Lite Interface*
A functional description of each state is described as below:

- **IDLE_STATE**: By default the FSM will be in IDLE_STATE. When the user_read_req signal becomes High, then it moves to the READ_STATE else if the user_write_req signal is High, it moves to WRITE_STATE else it remains in IDLE_STATE.

- **WRITE_STATE**: You provide S_AXI_AWVALID, S_AXI_AWADDR, S_AXI_WVALID, S_AXI_WDATA and S_AXI_WSTRB in this state to write to the register map through AXI. When S_AXI_BVALID and S_AXI_BREADY from the AXI slave are High then it moves to ACK_STATE. If any write operation happens in any illegal addresses, the S_AXI_BRESP[1:0] indicates 2'b10 that asserts the write error signal.

- **READ_STATE**: You provide S_AXI_ARVALID and S_AXI_ARADDR in this state to read from the register map through AXI. When S_AXI_RVALID and S_AXI_RREADY are High then it moves to ACK_STATE. If any read operation happens from any illegal addresses, the S_AXI_RRESP[1:0] indicates 2'b10 that asserts the read error signal.

- **ACK_STATE**: The state moves to IDLE_STATE.

### AXI User Interface Ports

**Table 5-3: AXI User Interface Ports**

<table>
<thead>
<tr>
<th>Name</th>
<th>Size</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>S_AXI_ACLK</td>
<td>1</td>
<td>Input</td>
<td>AXI clock signal</td>
</tr>
<tr>
<td>S_AXI_ARSETN</td>
<td>1</td>
<td>Input</td>
<td>AXI active-Low synchronous reset</td>
</tr>
<tr>
<td>S_AXI_PM_TICK</td>
<td>1</td>
<td>Input</td>
<td>PM tick user input</td>
</tr>
<tr>
<td>S_AXI_AWADDR</td>
<td>32</td>
<td>Input</td>
<td>AXI write address</td>
</tr>
<tr>
<td>S_AXI_AWVALID</td>
<td>1</td>
<td>Input</td>
<td>AXI write address valid</td>
</tr>
<tr>
<td>S_AXI_AWREADY</td>
<td>1</td>
<td>Output</td>
<td>AXI write address ready</td>
</tr>
<tr>
<td>S_AXI_WDATA</td>
<td>32</td>
<td>Input</td>
<td>AXI write data</td>
</tr>
<tr>
<td>S_AXI_WSTRB</td>
<td>4</td>
<td>Input</td>
<td>AXI write strobe. This signal indicates which byte lanes hold valid data.</td>
</tr>
<tr>
<td>S_AXI_WVALID</td>
<td>1</td>
<td>Input</td>
<td>AXI write data valid. This signal indicates that valid write data and strobes are available.</td>
</tr>
<tr>
<td>S_AXI_WREADY</td>
<td>1</td>
<td>Output</td>
<td>AXI write data ready</td>
</tr>
</tbody>
</table>
### Table 5-3: AXI User Interface Ports (Cont’d)

<table>
<thead>
<tr>
<th>Name</th>
<th>Size</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>S_AXI_BRESP</td>
<td>2</td>
<td>Output</td>
<td>AXI write response. This signal indicates the status of the write transaction.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>’b00 = OKAY</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>’b01 = EXOKAY</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>’b10 = SLVERR</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>’b11 = DECERR</td>
</tr>
<tr>
<td>S_AXI_BVALID</td>
<td>1</td>
<td>Output</td>
<td>AXI write response valid. This signal indicates that the channel is signaling a valid write response.</td>
</tr>
<tr>
<td>S_AXI_BREADY</td>
<td>1</td>
<td>Input</td>
<td>AXI write response ready.</td>
</tr>
<tr>
<td>S_AXI_ARADDR</td>
<td>32</td>
<td>Input</td>
<td>AXI read address</td>
</tr>
<tr>
<td>S_AXI_ARVALID</td>
<td>1</td>
<td>Input</td>
<td>AXI read address valid</td>
</tr>
<tr>
<td>S_AXI_ARREADY</td>
<td>1</td>
<td>Output</td>
<td>AXI read address ready</td>
</tr>
<tr>
<td>S_AXI_RDATA</td>
<td>32</td>
<td>Output</td>
<td>AXI read data issued by slave</td>
</tr>
<tr>
<td>S_AXI_RRESP</td>
<td>2</td>
<td>Output</td>
<td>AXI read response. This signal indicates the status of the read transfer.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>’b00 = OKAY</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>’b01 = EXOKAY</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>’b10 = SLVERR</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>’b11 = DECERR</td>
</tr>
<tr>
<td>S_AXI_RVALID</td>
<td>1</td>
<td>Output</td>
<td>AXI read data valid</td>
</tr>
<tr>
<td>S_AXI_RREADY</td>
<td>1</td>
<td>Input</td>
<td>AXI read ready. This signal indicates the user/master can accept the read data and response information.</td>
</tr>
</tbody>
</table>
Chapter 5: Example Design

Valid Write Transactions

Invalid Write Transactions

Figure 5-11: AXI4-Lite User Side Write Transaction

Figure 5-12: AXI4-Lite User Side Write Transaction with Invalid Write Address
Valid Read Transactions

Invalid Read Transactions

Figure 5-13: AXI4-Lite User Side Read Transaction

Figure 5-14: AXI4-Lite User Side Read Transaction with Invalid Read Address
IEEE Clause 108 (RS-FEC) Integration

If you want to include IEEE clause 108 RS-FEC soft IP (for error correction) in between 25G Ethernet IP and the GT, you must select the **Include Clause 108 (RS-FEC)** check box in the Configuration tab. This option is available only for 25G speed only.

This feature enables the IEEE Clause108 RS-FEC soft IP component instantiated in between the 25G core and the GT. The TX SerDes lines from the 25G core will be input to the RS-FEC soft IP for forward error correction encoding. The output from the RS-FEC module is then fed to GT. Similarly, the RX SerDes lines from the GT will be fed to the RS-FEC module for error correction decoding and then to the 25G core.

Refer to the *25G IEEE 802.3by Reed-Solomon Forward Error Correction LogiCORE IP Product Guide* (PG217) [Ref 12] for IEEE clause 108 Reed-Solomon Forward Error Correction for LogiCORE™ IP core and its functionality.

*Figure 5-15: RS-FEC Integration in between 25G and GT*
Batch Mode Test Bench

Each batch mode release of the 10G/25G Ethernet subsystem includes a demonstration test bench that performs a loopback test on the complete subsystem. For your convenience, scripts are provided to launch the test bench from several industry-standard simulators. The test program exercises the datapath to check that the transmitted frames are received correctly. Register Transfer Level (RTL) simulation models for the 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.

![Diagram of the test bench](image.png)

Figure 6-1: Test Bench
Appendix A

Migrating and Upgrading

This appendix contains information about upgrading to a more recent version of the IP core in the Vivado® Design Suite.

A script is provided for upgrading the design to a more recent version of the Vivado Design Suite. The script is found in the `/compile/xilinx/upgrade_IP` directory. Run this script for the following upgrades:

- To upgrade the transceiver wrapper to the latest version.
- To use the latest transceiver simulation model, which if not upgraded can result in simulation errors.

**TIP:** Before running the script, save a copy of the original design in the event that you need to revert to the previous version.

---

Changes from v2.0 (10/05/2016) to v2.0 (11/30/2016 version)

*Ports Added To Table 5-2, Core xci Top Level Port List*

- `gtwiz_reset_qpll1lock_in`
- `gtwiz_reset_qpll1reset_out`
- `gt_refclk_out`
- `tx_unfout_*`
- `stat_rx_valid_ctrl_code_*`
- `ctl_tx_ptp_1step_enable_*`
- `ctl_tx_ptp_latency_adjust_*`
- `ctl_ptp_transpclk_mode_*`
- `tx_ptp_upd_chksum_in_*`
- `tx_ptp_tstamp_offset_in_*`
Appendix A: Migrating and Upgrading

• tx_ptp_chksum_offset_in_*
• tx_ptp_rxtstamp_in_*
• ctl_an_fec_25g_baser_request_*
• stat_an_lp_ability_25gbase_cr1_*
• stat_an_rxcdrhold_*
• ctl_fec_enable_error_to_pcs_*
• ctl_rx_rsfec_enable_correction_*
• ctl_rx_rsfec_enable_indication_*
• ctl_rsfec_enable_*
• ctl_rsfec_ieee_error_indication_mode_*
• ctl_rsfec_consortium_25g_*
• stat_rx_rsfec_hi_ser_*
• stat_rx_rsfec_lane_alignment_status_*
• stat_rx_rsfec_corrected_cw_inc_*
• stat_rx_rsfec_uncorrected_cw_inc_*
• stat_rx_rsfec_err_count0_inc_*
• stat_tx_rsfec_lane_alignment_status_*
• gt_drp_done_*
• speed_*
• anlt_done_*

Ports Removed from Table 5-2, Core xci Top Level Port List

• stat_rx_framing_err_valid_1_*
• stat_rx_framing_err_1_*
• stat_rx_framing_err_valid_2_*
• stat_rx_framing_err_2_*
• stat_rx_framing_err_valid_3_*
• stat_rx_framing_err_3_*
• stat_rx_vl_demuxed_*
• stat_rx_vl_number_0_*
• stat_rx_vl_number_1_*
• stat_rx_vl_number_2_*
• stat_rx_vl_number_3_*
• stat_rx_synced_*
• stat_rx_synced_err_*
• stat_rx_mf_len_err_*
• stat_rx_mf_repeat_err_*
• stat_rx_mf_err_*
• stat_rx_misaligned_*
• stat_rx_aligned_err_*
• stat_rx_bip_err_0_*
• stat_rx_bip_err_1_*
• stat_rx_bip_err_2_*
• stat_rx_bip_err_3_*
• stat_rx_aligned_*
• stat_rx_status_*
• tx_ptp_pcslane_out_*
• rx_lane_aligner_fill_0_*
• rx_lane_aligner_fill_1_*
• rx_lane_aligner_fill_2_*
• rx_lane_aligner_fill_3_*
• ctl_lt_pseudo_seed1_*
• ctl_lt_k_p1_to_tx1_*
• ctl_lt_k0_to_tx1_*
• ctl_lt_stat_p1_to_tx1_*
• ctl_lt_stat0_to_tx1_*
• ctl_lt_stat_m1_to_tx1_*
• ctl_lt_pseudo_seed2_*
• ctl_lt_k_p1_to_tx2_*
• ctl_lt_k0_to_tx2_*
• ctl_lt_k_m1_to_tx2_*
• ctl_lt_k_m1_to_tx2_*
Appendix A: Migrating and Upgrading

- ctl_lt_stat0_to_tx2_*
- ctl_lt_stat_m1_to_tx2_*
- ctl_lt_pseudo_seed3_*
- ctl_lt_k_p1_to_tx3_*
- ctl_lt_k0_to_tx3_*
- ctl_lt_k_m1_to_tx3_*
- ctl_lt_stat_p1_to_tx3_*
- Ctl_lt_stat0_to_tx3_*
- ctl_lt_stat_m1_to_tx3_*
- stat_an_start_tx_disable_*
- stat_an_start_an_good_check_*
- stat_lt_k_p1_from_rx1_*
- stat_lt_k0_from_rx1_*
- stat_lt_k_m1_from_rx1_*
- stat_lt_stat_p1_from_rx1_*
- stat_lt_stat0_from_rx1_*
- stat_lt_stat_m1_from_rx1_*
- stat_lt_k_p1_from_rx2_*
- Stat_lt_k0_from_rx2_*
- stat_lt_k_m1_from_rx2_*
- stat_lt_stat_p1_from_rx2_*
- stat_lt_stat0_from_rx2_*
- stat_lt_stat_m1_from_rx2_*
- stat_lt_k_p1_from_rx3_*
- stat_lt_k0_from_rx3_*
- stat_lt_k_m1_from_rx3_*
- stat_lt_stat_p1_from_rx3_*
- stat_lt_stat0_from_rx3_*
- stat_lt_stat_m1_from_rx3_*
Migrating from the Legacy XGEMAC

This section contains information about upgrading from the legacy version of the Xilinx 10G EMAC IP to the new 10G/25G High Speed Ethernet IP core.

Note: For all new designs, Xilinx recommends that you use the most recent version of the 10G/25G High Speed Ethernet IP Core.

The Xilinx 10G Ethernet MAC and the 10G/25G High Speed Ethernet core are both designed to the specifications of the Ethernet IEEE 802.3 standard. There are significant differences in how some features are designed and/or handled. There are also differences in signal and parameter names and the corresponding AXI registers. Note that this section only outlines the features in the legacy 10G EMAC IP and compares them with the new 10G/25G High Speed Ethernet IP core. For a list of new features or features exclusive to the new IP, refer to Chapter 2, Product Specification.

Feature Comparison

The following features are different in the new 10/25G High Speed Ethernet IP.

- **Padding**
  The Pad field is not added by the 10/25G High Speed Ethernet IP. You must present a packet that meets the minimum length of 64 bytes to the IP core.

- **IFG extension**
  The inter-frame gap (IFG/IPG) can be extended up to 12B using the parameter ctl_tx_ipg_value[2:0].

- **Deficit Idle Count (DIC)**
  This feature is not supported.

- **Management Data Input/Output (MDIO) master**
  The 10G/25G High Speed Ethernet IP does not provide an MDIO master. The contents of the appropriate MDIO registers are available in the status signals.

- **Fault Handling**
  The user logic must designed differently for TX faults. Legacy 10G EMAC TX transmits RF or idles and drops packets by default if LF/RF is received. An option to disable fault transmission is provided. 10/25G High Speed Ethernet IP requires you to control if LF/RF are transmitted. You must provide the fault status signals as well.

- **VLAN**
  The new 10G/25G IP provides no VLAN specific features. However you can set the ctl_rx_max_packet_len attribute appropriately to allow the standard VLAN frame (1522 B) and also design the user logic to handle any number of stacked VLAN tags.
• Enabling Link Training without Auto-negotiation
  On the legacy 10G IP, it appears that link training was always enabled (see below) whereas the 10G/25G High Speed Ethernet IP performs link training after auto-negotiation and thus both features have to be enabled.

• Link Training Translation
  The legacy 10-BaseKR subsystem included logic that allowed it to be trained by a far-end device without user interaction. This feature is not available in the 10/25G High Speed Ethernet IP and the user logic must be designed to support this feature.

• Standalone MAC with 64-bit internal XGMII interface for connecting to XAUI/RXAUI
  The 10/25G High Speed Ethernet IP does not provide the 64-bit internal XGMII interface required to connect to the XAUI/RXAUI interfaces. Contact Xilinx Technical Support if more assistance is required on this feature.

• External XGMII DDR interface to external PHY
  This feature is not available in the 10/25G High Speed Ethernet IP because a standalone MAC without PCS is not an option.

• Pause Interface
  The 10G/25G High Speed Ethernet IP does not support the legacy PFC. However up to 8 Priority pause frames and global frames can be generated and terminated.

**Port/Signal Comparison**

There are significant differences in the names and functions of the signals and parameters. Because it is difficult to draw a one-to-one comparison, the following tables are based on features and their presentation in the two IP cores.

**Parameter/Attribute Comparison**

**Transmitter configuration**

The legacy XGEMAC used the `mac_tx_configuration_vector[368:0]` for all TX configuration whereas the 10G/25G High Speed Ethernet IP deploys various signals for the same purpose. Table A-1 draws a comparison.

<table>
<thead>
<tr>
<th>Feature</th>
<th>Legacy XGEMAC</th>
<th>10/25G High Speed Ethernet IP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Legacy Pause Refresh Value</td>
<td><code>mac_tx_configuration_vector[367:352]</code></td>
<td><code>ctl_tx_pause_refresh_timer7[15:0]</code></td>
</tr>
<tr>
<td>TX Priority 7 Pause Quanta refresh value</td>
<td><code>mac_tx_configuration_vector[351:336]</code></td>
<td><code>ctl_tx_pause_quanta7[15:0]</code></td>
</tr>
<tr>
<td>TX Priority 7 Pause Quanta</td>
<td><code>mac_tx_configuration_vector[335:320]</code></td>
<td><code>ctl_tx_pause_quanta7[15:0]</code></td>
</tr>
<tr>
<td>TX Priority 6 Pause Quanta refresh value</td>
<td><code>mac_tx_configuration_vector[319:304]</code></td>
<td><code>ctl_tx_pause_refresh_timer6[15:0]</code></td>
</tr>
<tr>
<td>TX Priority 6 Pause Quanta</td>
<td><code>mac_tx_configuration_vector[303:288]</code></td>
<td><code>ctl_tx_pause_quanta6[15:0]</code></td>
</tr>
</tbody>
</table>
### Table A-1: Transmitter Configuration Comparison (Cont’d)

<table>
<thead>
<tr>
<th>Feature</th>
<th>Legacy XGEMAC</th>
<th>10/25G High Speed Ethernet IP</th>
</tr>
</thead>
<tbody>
<tr>
<td>TX Priority 5 Pause Quanta</td>
<td>mac_tx_configuration_vector[287:272]</td>
<td>ctl_tx_pause_refresh_timer5[15:0]</td>
</tr>
<tr>
<td>refresh value</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TX Priority 5 Pause Quanta</td>
<td>mac_tx_configuration_vector[271:256]</td>
<td>ctl_tx_pause_quanta5[15:0]</td>
</tr>
<tr>
<td>TX Priority 4 Pause Quanta</td>
<td>mac_tx_configuration_vector[255:240]</td>
<td>ctl_tx_pause_refresh_timer4[15:0]</td>
</tr>
<tr>
<td>refresh value</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TX Priority 4 Pause Quanta</td>
<td>mac_tx_configuration_vector[239:224]</td>
<td>ctl_tx_pause_quanta4[15:0]</td>
</tr>
<tr>
<td>TX Priority 3 Pause Quanta</td>
<td>mac_tx_configuration_vector[223:208]</td>
<td>ctl_tx_pause_refresh_timer3[15:0]</td>
</tr>
<tr>
<td>refresh value</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TX Priority 3 Pause Quanta</td>
<td>mac_tx_configuration_vector[207:192]</td>
<td>ctl_tx_pause_quanta3[15:0]</td>
</tr>
<tr>
<td>TX Priority 2 Pause Quanta</td>
<td>mac_tx_configuration_vector[191:176]</td>
<td>ctl_tx_pause_refresh_timer2[15:0]</td>
</tr>
<tr>
<td>refresh value</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TX Priority 2 Pause Quanta</td>
<td>mac_tx_configuration_vector[175:160]</td>
<td>ctl_tx_pause_quanta2[15:0]</td>
</tr>
<tr>
<td>TX Priority 1 Pause Quanta</td>
<td>mac_tx_configuration_vector[159:144]</td>
<td>ctl_tx_pause_refresh_timer1[15:0]</td>
</tr>
<tr>
<td>refresh value</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TX Priority 1 Pause Quanta</td>
<td>mac_tx_configuration_vector[143:128]</td>
<td>ctl_tx_pause_quanta1[15:0]</td>
</tr>
<tr>
<td>TX Priority 0 Pause Quanta</td>
<td>mac_tx_configuration_vector[127:112]</td>
<td>ctl_tx_pause_refresh_timer0[15:0]</td>
</tr>
<tr>
<td>refresh value</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TX Priority 0 Pause Quanta</td>
<td>mac_tx_configuration_vector[111:96]</td>
<td>ctl_tx_pause_quanta0[15:0]</td>
</tr>
<tr>
<td>TX Priority 7 Flow Control Enable</td>
<td>mac_tx_configuration_vector[95]</td>
<td>ctl_tx_pause_enable[7]</td>
</tr>
<tr>
<td>TX Priority 6 Flow Control Enable</td>
<td>mac_tx_configuration_vector[94]</td>
<td>ctl_tx_pause_enable[6]</td>
</tr>
<tr>
<td>TX Priority 5 Flow Control Enable</td>
<td>mac_tx_configuration_vector[93]</td>
<td>ctl_tx_pause_enable[5]</td>
</tr>
<tr>
<td>TX Priority 3 Flow Control Enable</td>
<td>mac_tx_configuration_vector[91]</td>
<td>ctl_tx_pause_enable[3]</td>
</tr>
<tr>
<td>TX Priority 2 Flow Control Enable</td>
<td>mac_tx_configuration_vector[90]</td>
<td>ctl_tx_pause_enable[2]</td>
</tr>
<tr>
<td>TX Priority 1 Flow Control Enable</td>
<td>mac_tx_configuration_vector[89]</td>
<td>ctl_tx_pause_enable[1]</td>
</tr>
<tr>
<td>TX Priority 0 Flow Control Enable</td>
<td>mac_tx_configuration_vector[88]</td>
<td>ctl_tx_pause_enable[0]</td>
</tr>
<tr>
<td>Auto XON enable</td>
<td>mac_tx_configuration_vector[81]</td>
<td></td>
</tr>
<tr>
<td>Priority flow control enable</td>
<td>mac_tx_configuration_vector[80]</td>
<td></td>
</tr>
<tr>
<td>TX Pause Frame Source Address</td>
<td>mac_tx_configuration_vector[79:32]</td>
<td>ctl_tx_pause_da[47:0]</td>
</tr>
</tbody>
</table>
Appendix A: Migrating and Upgrading

Table A-1: Transmitter Configuration Comparison (Cont’d)

<table>
<thead>
<tr>
<th>Feature</th>
<th>Legacy XGEMAC</th>
<th>10/25G High Speed Ethernet IP</th>
</tr>
</thead>
<tbody>
<tr>
<td>TX MTU Size</td>
<td>mac_tx_configuration_vector[30:16]</td>
<td>Not required because the TX can handle any size frame presented to the IP.</td>
</tr>
<tr>
<td>TX MTU enable</td>
<td>mac_tx_configuration_vector[14]</td>
<td>Not applicable because there is no need to set the TX. MTU does not have to be set.</td>
</tr>
<tr>
<td>Deficit Idle Count Enable</td>
<td>mac_tx_configuration_vector[10]</td>
<td>Not supported</td>
</tr>
<tr>
<td>TX LAN/WAN mode</td>
<td>mac_tx_configuration_vector[9]</td>
<td>This feature is not available as-is. The 10G/25G High Speed Ethernet IP always maintains the average IPG per the IEEE 802.3 spec. However you can design the user logic to insert idles using ctl_send_idle and increase the IPG value using ctl_tx_ipg_value[3:0] to meet the OC-192 SONET requirements.</td>
</tr>
<tr>
<td>TX IFG Adjust enable</td>
<td>mac_tx_configuration_vector[8]</td>
<td>ctl_tx_ipg_value sets the appropriate value of the custom IPG/IFG</td>
</tr>
<tr>
<td>TX Preserve Preamble Enable</td>
<td>mac_tx_configuration_vector[7]</td>
<td>ctl_tx_custom_preamble_enable enables the use of a custom preamble. Tx_preamblein presents the custom preamble and rx_preambleout has the preamble field from the received frame.</td>
</tr>
<tr>
<td>TX Flow control Enable</td>
<td>mac_tx_configuration_vector[5]</td>
<td>ctl_tx_pause_enable[8:0]</td>
</tr>
<tr>
<td>TX Jumbo Frame Enable</td>
<td>mac_tx_configuration_vector[4]</td>
<td>Not required</td>
</tr>
<tr>
<td>TX In-band FCS enable</td>
<td>mac_tx_configuration_vector[3]</td>
<td>ctl_tx_fcs_ins_enable</td>
</tr>
<tr>
<td>TX VLAN enable</td>
<td>mac_tx_configuration_vector[2]</td>
<td>Not required</td>
</tr>
<tr>
<td>TX Enable</td>
<td>mac_tx_configuration_vector[1]</td>
<td>ctl_tx_enable</td>
</tr>
<tr>
<td>TX Reset</td>
<td>mac_tx_configuration_vector[0]</td>
<td>tx_reset</td>
</tr>
</tbody>
</table>

Receiver Configuration

The legacy XGEMAC used the rx_configuration_vector[95:0] for all RX configuration whereas the 10G/25G High Speed Ethernet IP core deploys various signals for the same purpose. Table A-2 draws a comparison.

Table A-2: Receiver Configuration Comparison

<table>
<thead>
<tr>
<th>Feature</th>
<th>Legacy XGEMAC</th>
<th>10/25G High Speed Ethernet IP</th>
</tr>
</thead>
<tbody>
<tr>
<td>RX Priority 7-0 Flow control enable</td>
<td>rx_configuration_vector[95:88]</td>
<td>ctl_rx_pause_enable[8:0]</td>
</tr>
<tr>
<td>Priority Flow Control Enable</td>
<td>rx_configuration_vector[80]</td>
<td></td>
</tr>
<tr>
<td>RX Pause Frame SA</td>
<td>rx_configuration_vector[79:32]</td>
<td>ctl_rx_pause_sa[47:0]</td>
</tr>
</tbody>
</table>
#### Table A-2: Receiver Configuration Comparison (Cont’d)

<table>
<thead>
<tr>
<th>Feature</th>
<th>Legacy XGEMAC</th>
<th>10/25G High Speed Ethernet IP</th>
</tr>
</thead>
<tbody>
<tr>
<td>RX MTU size</td>
<td>rx_configuration_vector[30:16]</td>
<td>Set using ctl_rx_max_packet_len and ctl_rx_min_packet_len signals.</td>
</tr>
<tr>
<td>RX MTU Enable</td>
<td>rx_configuration_vector[14]</td>
<td>Not required.</td>
</tr>
<tr>
<td>Reconciliation Sublayer Fault Inhibit</td>
<td>rx_configuration_vector[10]</td>
<td>Design user logic to set ctl_tx_send_idle when RFI is received.</td>
</tr>
<tr>
<td>Control Frame Length check Disable</td>
<td>rx_configuration_vector[9]</td>
<td>Not available</td>
</tr>
<tr>
<td>RX Length/Type Error disable</td>
<td>rx_configuration_vector[8]</td>
<td>The length/type error cannot be disabled on the 10/25G High Speed Ethernet IP core.</td>
</tr>
<tr>
<td>RX preserve preamble enable</td>
<td>rx_configuration_vector[7]</td>
<td>ctl_rx_custom_preamble_enable</td>
</tr>
<tr>
<td>RX Flow control enable</td>
<td>rx_configuration_vector[5]</td>
<td>ctl_rx_pause_enable[8:0]</td>
</tr>
<tr>
<td>RX in-band FCS enable</td>
<td>rx_configuration_vector[3]</td>
<td>ctl_rx_delete_fcs</td>
</tr>
<tr>
<td>RX VLAN enable</td>
<td>rx_configuration_vector[2]</td>
<td>Up to the user logic to implement this functionality.</td>
</tr>
<tr>
<td>RX enable</td>
<td>rx_configuration_vector[1]</td>
<td>ctl_rx_enable</td>
</tr>
<tr>
<td>RX reset</td>
<td>rx_configuration_vector[0]</td>
<td>rx_reset</td>
</tr>
</tbody>
</table>

#### Status Vector

#### Table A-3: Status Vector Comparison

<table>
<thead>
<tr>
<th>Feature</th>
<th>Legacy XGEMAC</th>
<th>10/25G High Speed Ethernet IP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Remote Fault RX</td>
<td>status_vector[1]</td>
<td>This feature is not available.</td>
</tr>
<tr>
<td>Local Fault RX</td>
<td>status_vector[0]</td>
<td>This feature is not available.</td>
</tr>
</tbody>
</table>

#### TX Statistics

The legacy XGEMAC used the `tx_statistics_vector[25:0]` for all TX statistics whereas the 10G/25G High Speed Ethernet IP core deploys various signals for the same purpose. **Table A-4** draws a comparison.

#### Table A-4: TX Statistics Comparison

<table>
<thead>
<tr>
<th>Feature</th>
<th>Legacy XGEMAC</th>
<th>10/25G High Speed Ethernet MAC</th>
</tr>
</thead>
<tbody>
<tr>
<td>PFC Frame Transmitted</td>
<td>tx_statistics_vector[26]</td>
<td>stat_tx_user_pause</td>
</tr>
<tr>
<td>Pause Frame Transmitted</td>
<td>tx_statistics_vector[25]</td>
<td>stat_tx_pause</td>
</tr>
<tr>
<td>Bytes Valid</td>
<td>tx_statistics_vector[24:21]</td>
<td>stat_tx_total_good_bytes</td>
</tr>
<tr>
<td>VLAN Frame</td>
<td>tx_statistics_vector[20]</td>
<td>stat_tx_vlan</td>
</tr>
<tr>
<td>Frame length count</td>
<td>tx_statistics_vector[19:5]</td>
<td>stat_tx_packet_* signals that can also be used for the packet histogram.</td>
</tr>
</tbody>
</table>
Appendix A: Migrating and Upgrading

RX Statistics

The legacy XGEMAC used the `rx_statistics_vector[30:0]` for all RX statistics whereas the 10G/25G High Speed Ethernet IP core deploys various signals for the same purpose. Table A-5 draws a comparison.

<table>
<thead>
<tr>
<th>Feature</th>
<th>Legacy XGEMAC</th>
<th>10/25G High Speed Ethernet MAC</th>
</tr>
</thead>
<tbody>
<tr>
<td>PFC Frame</td>
<td><code>rx_statistics_vector[30]</code></td>
<td><code>stat_rx_user_pause</code></td>
</tr>
<tr>
<td>Length/Type out of range</td>
<td><code>rx_statistics_vector[29]</code></td>
<td><code>stat_rx_inrange_err</code></td>
</tr>
<tr>
<td>Bad Opcode</td>
<td><code>rx_statistics_vector[28]</code></td>
<td></td>
</tr>
<tr>
<td>Flow Control Frame</td>
<td><code>rx_statistics_vector[27]</code></td>
<td><code>stat_rx_pause</code></td>
</tr>
<tr>
<td>VLAN Frame</td>
<td><code>rx_statistics_vector[22]</code></td>
<td><code>stat_rx_vlan</code></td>
</tr>
<tr>
<td>Out of Bounds</td>
<td><code>rx_statistics_vector[21]</code></td>
<td><code>stat_rx_toolong</code></td>
</tr>
<tr>
<td>Control Frame</td>
<td><code>rx_statistics_vector[20]</code></td>
<td></td>
</tr>
<tr>
<td>Broadcast frame</td>
<td><code>rx_statistics_vector [3]</code></td>
<td><code>stat_rx_broadcast</code></td>
</tr>
<tr>
<td>Bad frame</td>
<td><code>rx_statistics_vector [1]</code></td>
<td><code>stat_rx_total_good_packets sampled as 0</code></td>
</tr>
<tr>
<td>Good Frame</td>
<td><code>rx_statistics_vector [0]</code></td>
<td><code>stat_rx_total_good_packets sampled as 1</code></td>
</tr>
<tr>
<td>RX statistics Valid</td>
<td><code>rx_statistics_valid</code></td>
<td>Not required</td>
</tr>
</tbody>
</table>

Register Space

Both the legacy XGEMAC and the 10/25G High Speed Ethernet IP core provide an AXI User Interface. While much of the configuration parameters are similar, the registers and the memory map of the registers are different between the two IP cores. The following sections discuss the comparison between the configuration, statistics and other registers.
### TX Configuration Registers

<table>
<thead>
<tr>
<th>Feature</th>
<th>Legacy XGEMAC</th>
<th>10/25G HSEC</th>
</tr>
</thead>
<tbody>
<tr>
<td>TX MTU Size</td>
<td>@0x418: Transmitter MTU Configuration Word [14:0]</td>
<td>Not required because the TX can handle any size frame presented to the IP.</td>
</tr>
<tr>
<td>TX MTU enable</td>
<td>@0x418: Transmitter MTU Configuration Word [16]</td>
<td>Not applicable because there is no need to set the TX MTU.</td>
</tr>
<tr>
<td>Deficit Idle Count Enable</td>
<td>@0x408: Transmitter Configuration Word[24]</td>
<td>Not supported</td>
</tr>
<tr>
<td>TX LAN/WAN mode</td>
<td>@0x408: Transmitter Configuration Word[26]</td>
<td>This feature is not available as-is. The 10G/25G High Speed Ethernet IP subsystem always maintains the average IPG per the IEEE 802.3 spec. However, you can design the user logic to insert idles using @0x000C: CONFIGURATION_TX_REG1[5] ctl_send_idle and increase the IPG value using @0x000C: CONFIGURATION_TX_REG1[13:10] ctl_tx_ipg_value[3:0] to meet the OC-192 SONET requirements</td>
</tr>
<tr>
<td>TX IFG Adjust enable</td>
<td>@0x408: Transmitter Configuration Word[25]</td>
<td>@0x000C: CONFIGURATION_TX_REG1[13:10] ctl_tx_ipg_value sets the appropriate value of the custom IPG/IFG</td>
</tr>
<tr>
<td>TX Preserve Preamble Enable</td>
<td>@0x408: Transmitter Configuration Word[23]</td>
<td>@0x000C: CONFIGURATION_TX_REG1[18] ctl_tx_custom_preamble_enable enables the use of custom preamble. Tx_preamblein presents the custom preamble and rx_preambleout has the preamble field from the received frame.</td>
</tr>
<tr>
<td>TX Flow control Enable</td>
<td>@0x40C: Flow control Configuration register [30]</td>
<td>ctl_tx_pause_enable[8:0]</td>
</tr>
<tr>
<td>TX Jumbo Frame Enable</td>
<td>@0x408: Transmitter Configuration Word[30]</td>
<td>Not required.</td>
</tr>
<tr>
<td>TX In-band FCS enable</td>
<td>@0x408: Transmitter Configuration Word[29]</td>
<td>@0x000C: CONFIGURATION_TX_REG1[1] ctl_tx_fcs_ins_enable</td>
</tr>
<tr>
<td>TX VLAN enable</td>
<td>@0x408: Transmitter Configuration Word[27]</td>
<td>Not required.</td>
</tr>
</tbody>
</table>
### Table A-6: TX Configuration Registers Comparison (Cont’d)

<table>
<thead>
<tr>
<th>Feature</th>
<th>Legacy XGEMAC</th>
<th>10/25G HSEC</th>
</tr>
</thead>
<tbody>
<tr>
<td>TX Enable</td>
<td>@0x408: Transmitter Configuration Word [28]</td>
<td>@0x000C: CONFIGURATION_TX_REG[0] ctl_tx_enable</td>
</tr>
<tr>
<td>TX Reset</td>
<td>@0x408: Transmitter Configuration Word [31]</td>
<td>@0x0004: RESET_REG[31] tx_reset</td>
</tr>
</tbody>
</table>
## RX Configuration

### Table A-7: RX Configuration Comparison

<table>
<thead>
<tr>
<th>Feature</th>
<th>Legacy XGEMAC</th>
<th>10/25G High Speed Ethernet IP Core</th>
</tr>
</thead>
<tbody>
<tr>
<td>RX Priority 7-0 Flow control enable</td>
<td>Bit 95:88</td>
<td>@0x0094: CONFIGURATION_RX_FLOW_CONTROL_REG1 [7:0]</td>
</tr>
<tr>
<td>Priority Flow Control Enable</td>
<td>Bit 80</td>
<td>Does not support legacy PFC</td>
</tr>
<tr>
<td>RX Pause Frame SA</td>
<td>@0x404: Receiver Configuration Word 1[47:32], @0x400: Receiver Configuration Word 0</td>
<td>@0x00C4: CONFIGURATION_RX_FLOW_CONTROL_SA_REG1_MSB[15:0], @0x00C0: CONFIGURATION_RX_FLOW_CONTROL_SA_REG1_LSB</td>
</tr>
<tr>
<td>RX MTU size</td>
<td>@0x414: Receiver MTU Configuration Word [14:0]</td>
<td>Set using @0x0018: CONFIGURATION_RX_MTU[30:16] ctrl_rx_max_packet_len and @0x0018: CONFIGURATION_RX_MTU[7:0] ctrl_rx_min_packet_len signals</td>
</tr>
<tr>
<td>RX MTU Enable</td>
<td>@0x414: Receiver MTU Configuration Word [16]</td>
<td>Not required.</td>
</tr>
<tr>
<td>Reconciliation Sublayer Fault Inhibit</td>
<td>@0x410: Reconciliation Sublayer Configuration Word [27]</td>
<td>Design user logic to set @0x000C: CONFIGURATION_TX_REG1[5] ctrl_tx_send_idle when RFI is received</td>
</tr>
<tr>
<td>Control Frame Length check Disable</td>
<td>@0x404: Receiver Configuration Word 1[24]</td>
<td>Not available</td>
</tr>
<tr>
<td>RX Length/Type Error disable</td>
<td>@0x404: Receiver Configuration Word 1[25]</td>
<td>The length/type error cannot be disabled on the 10/25G High Speed Ethernet subsystem.</td>
</tr>
<tr>
<td>RX preserve preamble enable</td>
<td>@0x404: Receiver Configuration Word 1[26]</td>
<td>@0x0014: CONFIGURATION_RX_REG1[11] ctrl_rx_custom_preamble_enable</td>
</tr>
<tr>
<td>RX Flow control enable</td>
<td>@0x40C: Flow Control Configuration Register [29]</td>
<td>@0x0094: CONFIGURATION_RX_FLOW_CONTROL[8:0] ctrl_rx_pause_enable[8:0]</td>
</tr>
<tr>
<td>RX Jumbo Frame Enable</td>
<td>@0x404: Receiver Configuration Word 1[30]</td>
<td>Set using ctrl_rx_max_packet_len signal</td>
</tr>
<tr>
<td>RX in-band FCS enable</td>
<td>@0x404: Receiver Configuration Word 1[29]</td>
<td>@0x0014: CONFIGURATION_RX_REG1[1] ctrl_rx_delete_fcs</td>
</tr>
<tr>
<td>RX VLAN enable</td>
<td>@0x404: Receiver Configuration Word 1[27]</td>
<td>Up to the user logic to implement this functionality</td>
</tr>
</tbody>
</table>
### Table A-7: RX Configuration Comparison (Cont’d)

<table>
<thead>
<tr>
<th>Feature</th>
<th>Legacy XGEMAC</th>
<th>10/25G High Speed Ethernet IP Core</th>
</tr>
</thead>
<tbody>
<tr>
<td>RX enable</td>
<td>@0x404: Receiver Configuration Word 1[28]</td>
<td>@0x0014: CONFIGURATION_RX_REG1[0] ctl_rx_enable</td>
</tr>
<tr>
<td>RX reset</td>
<td>@0x404: Receiver Configuration Word 1[31]</td>
<td>@0x0004: RESET_REG[30] rx_reset</td>
</tr>
</tbody>
</table>

### Status Vector

#### Table A-8: Status Vector Comparison

<table>
<thead>
<tr>
<th>Feature</th>
<th>Legacy XGEMAC</th>
<th>10/25G High Speed Ethernet IP Core</th>
</tr>
</thead>
<tbody>
<tr>
<td>Remote Fault RX</td>
<td>@0x410: Reconciliation Sublayer Configuration Word[29]</td>
<td>This feature is not available.</td>
</tr>
<tr>
<td>Local Fault RX</td>
<td>@0x410: Reconciliation Sublayer Configuration Word[28]</td>
<td>This feature is not available.</td>
</tr>
</tbody>
</table>

### Statistics Counters

Table A-9 lists the different statistics counters and their addresses.

#### Table A-9: Statistics Counters

<table>
<thead>
<tr>
<th>Address (Hex)</th>
<th>Register</th>
<th>Register</th>
<th>Address (Hex)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x200</td>
<td>Received Bytes (LSW)</td>
<td>STAT_RX_TOTAL_BYTES_LSB</td>
<td>0x0808</td>
</tr>
<tr>
<td>0x204</td>
<td>Received Bytes (MSW)</td>
<td>STAT_RX_TOTAL_BYTES_MSB</td>
<td>0x080C</td>
</tr>
<tr>
<td>0x208</td>
<td>Transmitted Bytes (LSW)</td>
<td>STAT_TX_TOTAL_BYTES_LSB</td>
<td>0x0710</td>
</tr>
<tr>
<td>0x20C</td>
<td>Transmitted Bytes (MSW)</td>
<td>STAT_TX_TOTAL_BYTES_MSB</td>
<td>0x0714</td>
</tr>
<tr>
<td>0x210</td>
<td>Undersize Frames Received (LSW)</td>
<td>STAT_RX_UNDERSIZE_LSB</td>
<td>0x0898</td>
</tr>
<tr>
<td>0x214</td>
<td>Undersize Frames Received (MSW)</td>
<td>STAT_RX_UNDERSIZE_MSB</td>
<td>0x089C</td>
</tr>
<tr>
<td>0x218</td>
<td>Fragment Frames Received (LSW)</td>
<td>STAT_RX_FRAGMENT_LSB</td>
<td>0x08A0</td>
</tr>
<tr>
<td>0x21C</td>
<td>Fragment Frames Received (MSW)</td>
<td>STAT_RX_FRAGMENT_MSB</td>
<td>0x08A4</td>
</tr>
<tr>
<td>0x220</td>
<td>64-byte frames received OK (LSW)</td>
<td>STAT_RX_PACKET_64_BYTES_LSB</td>
<td>0x0828</td>
</tr>
<tr>
<td>0x224</td>
<td>64-byte frames received OK (MSW)</td>
<td>STAT_RX_PACKET_64_BYTES_MSB</td>
<td>0x082C</td>
</tr>
<tr>
<td>0x228</td>
<td>65-127 byte frames received OK (LSW)</td>
<td>STAT_RX_PACKET_65_127_BYTES_LSB</td>
<td>0x0830</td>
</tr>
<tr>
<td>0x22C</td>
<td>65-127 byte frames received OK (MSW)</td>
<td>STAT_RX_PACKET_65_127_BYTES_MSB</td>
<td>0x0834</td>
</tr>
<tr>
<td>0x230</td>
<td>128-255 byte frames received OK (LSW)</td>
<td>STAT_RX_PACKET_128_255_BYTES_LSB</td>
<td>0x0838</td>
</tr>
<tr>
<td>0x234</td>
<td>128-255 byte frames received OK (MSW)</td>
<td>STAT_RX_PACKET_128_255_BYTES_MSB</td>
<td>0x083C</td>
</tr>
</tbody>
</table>
### Table A-9: Statistics Counters (Cont’d)

<table>
<thead>
<tr>
<th>Address (Hex)</th>
<th>Register Description</th>
<th>Register Address (Hex)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x238</td>
<td>256-511 byte frames received OK (LSW)</td>
<td>STAT_RX_PACKET_256_511_BYTES_LSB 0x0840</td>
</tr>
<tr>
<td>0x23C</td>
<td>256-511 byte frames received OK (MSW)</td>
<td>STAT_RX_PACKET_256_511_BYTES_MSB 0x0844</td>
</tr>
<tr>
<td>0x240</td>
<td>512-1023 byte frames received OK (LSW)</td>
<td>STAT_RX_PACKET_512_1023_BYTES_LSB 0x0848</td>
</tr>
<tr>
<td>0x244</td>
<td>512-1023 byte frames received OK (MSW)</td>
<td>STAT_RX_PACKET_512_1023_BYTES_MSB 0x084C</td>
</tr>
<tr>
<td>0x248</td>
<td>1024 – MaxFrameSize byte frames received OK (LSW)</td>
<td>STAT_RX_PACKET_1024_1518_BYTES_LSB 0x0850, STAT_RX_PACKET_1519_1522.Bytes_LSB 0x0858, STAT_RX_PACKET_1523_1548_BYTES_LSB 0x0860, STAT_RX_PACKET_1549_2047_BYTES_LSB 0x0868, STAT_RX_PACKET_2048_4095_BYTES_LSB 0x0870, STAT_RX_PACKET_4096_8191_BYTES_LSB 0x0878, STAT_RX_PACKET_8192_9215_BYTES_LSB 0x0880, STAT_RX_PACKET_LARGE_LSB 0x0888</td>
</tr>
<tr>
<td>0x24C</td>
<td>1024 – MaxFrameSize byte frames received OK (MSW)</td>
<td>STAT_RX_PACKET_1024_1518_BYTES_MSB 0x0854, STAT_RX_PACKET_1519_1522_BYTES_MSB 0x0858, STAT_RX_PACKET_1523_1548_BYTES_MSB 0x0860, STAT_RX_PACKET_1549_2047_BYTES_MSB 0x0868, STAT_RX_PACKET_2048_4095_BYTES_MSB 0x0870, STAT_RX_PACKET_4096_8191_BYTES_MSB 0x0878, STAT_RX_PACKET_8192_9215_BYTES_MSB 0x0880, STAT_RX_PACKET_LARGE_MSB 0x088C</td>
</tr>
<tr>
<td>0x250</td>
<td>Oversize frames received OK (LSW)</td>
<td>STAT_RX_OVERSIZE_LSB 0x08A8</td>
</tr>
<tr>
<td>0x254</td>
<td>Oversize frames received OK (MSW)</td>
<td>STAT_RX_OVERSIZE_MSB 0x08AC</td>
</tr>
<tr>
<td>0x258</td>
<td>64-byte frames transmitted OK (LSW)</td>
<td>STAT_TX_PACKET_64_BYTES_LSB 0x0720</td>
</tr>
<tr>
<td>0x25C</td>
<td>64-byte frames transmitted OK (MSW)</td>
<td>STAT_TX_PACKET_64_BYTES_MSB 0x0724</td>
</tr>
<tr>
<td>0x260</td>
<td>65-127 byte frames transmitted OK (LSW)</td>
<td>STAT_TX_PACKET_65_127_BYTES_LSB 0x0728</td>
</tr>
<tr>
<td>0x264</td>
<td>65-127 byte frames transmitted OK (MSW)</td>
<td>STAT_TX_PACKET_65_127_BYTES_MSB 0x072C</td>
</tr>
<tr>
<td>0x268</td>
<td>128-255 byte frames transmitted OK (LSW)</td>
<td>STAT_TX_PACKET_128_255_BYTES_LSB 0x0730</td>
</tr>
</tbody>
</table>
### Table A-9: Statistics Counters (Cont’d)

<table>
<thead>
<tr>
<th>Address (Hex)</th>
<th>Register</th>
<th>Register</th>
<th>Address (Hex)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x26C</td>
<td>128-255 byte frames transmitted OK (MSW)</td>
<td>STAT_TX_PACKET_128_255_BYTES_MSB</td>
<td>0x0734</td>
</tr>
<tr>
<td>0x270</td>
<td>256-511 byte frames transmitted OK (LSW)</td>
<td>STAT_TX_PACKET_256_511_BYTES_LSB</td>
<td>0x0738</td>
</tr>
<tr>
<td>0x274</td>
<td>256-511 byte frames transmitted OK (MSW)</td>
<td>STAT_TX_PACKET_256_511_BYTES_MSB</td>
<td>0x073C</td>
</tr>
<tr>
<td>0x278</td>
<td>512-1023 byte frames transmitted OK (LSW)</td>
<td>STAT_TX_PACKET_512_1023_BYTES_LSB</td>
<td>0x0740</td>
</tr>
<tr>
<td>0x27C</td>
<td>512-1023 byte frames transmitted OK (MSW)</td>
<td>STAT_TX_PACKET_512_1023_BYTES_MSB</td>
<td>0x0744</td>
</tr>
<tr>
<td>0x280</td>
<td>1024 – MaxFrameSize byte frames transmitted OK (LSW)</td>
<td>STAT_TX_PACKET_1024_1518_BYTES_LSB</td>
<td>0x0748</td>
</tr>
<tr>
<td></td>
<td></td>
<td>STAT_TX_PACKET_1519_1522_BYTES_LSB</td>
<td>0x0750</td>
</tr>
<tr>
<td></td>
<td></td>
<td>STAT_TX_PACKET_1523_1548BYTES_LSB</td>
<td>0x0758</td>
</tr>
<tr>
<td></td>
<td></td>
<td>STAT_TX_PACKET_1549_2047_BYTES_LSB</td>
<td>0x0760</td>
</tr>
<tr>
<td></td>
<td></td>
<td>STAT_TX_PACKET_2048_4095_BYTES_LSB</td>
<td>0x0768</td>
</tr>
<tr>
<td></td>
<td></td>
<td>STAT_TX_PACKET_4096_8191_BYTES_LSB</td>
<td>0x0770</td>
</tr>
<tr>
<td></td>
<td></td>
<td>STAT_TX_PACKET_8192_9215_BYTES_LSB</td>
<td>0x0778</td>
</tr>
<tr>
<td></td>
<td></td>
<td>STAT_TX_PACKET_LARGE_LSB</td>
<td>0x0780</td>
</tr>
<tr>
<td>0x284</td>
<td>1024 – MaxFrameSize byte frames transmitted OK (MSW)</td>
<td>STAT_TX_PACKET_1024_1518_BYTES_MSB</td>
<td>0x074C</td>
</tr>
<tr>
<td></td>
<td></td>
<td>STAT_TX_PACKET_1519_1522_BYTES_MSB</td>
<td>0x0754</td>
</tr>
<tr>
<td></td>
<td></td>
<td>STAT_TX_PACKET_1523_1548BYTES_MSB</td>
<td>0x075C</td>
</tr>
<tr>
<td></td>
<td></td>
<td>STAT_TX_PACKET_1549_2047_BYTES_LSB</td>
<td>0x0764</td>
</tr>
<tr>
<td></td>
<td></td>
<td>STAT_TX_PACKET_2048_4095_BYTES_LSB</td>
<td>0x076C</td>
</tr>
<tr>
<td></td>
<td></td>
<td>STAT_TX_PACKET_4096_8191_BYTES_LSB</td>
<td>0x0774</td>
</tr>
<tr>
<td></td>
<td></td>
<td>STAT_TX_PACKET_8192_9215_BYTES_MSB</td>
<td>0x077C</td>
</tr>
<tr>
<td></td>
<td></td>
<td>STAT_TX_PACKET_LARGE_MSB</td>
<td>0x0784</td>
</tr>
<tr>
<td>0x288</td>
<td>Oversize frames transmitted OK (LSW)</td>
<td>N/A</td>
<td></td>
</tr>
<tr>
<td>0x28C</td>
<td>Oversize frames transmitted OK (MSW)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0x290</td>
<td>Frames received OK (LSW)</td>
<td>STAT_RX_TOTAL_GOOD_PACKETS_LSB</td>
<td>0x0810</td>
</tr>
<tr>
<td>0x294</td>
<td>Frames received OK (MSW)</td>
<td>STAT_RX_TOTAL_GOOD_PACKETS_MSB</td>
<td>0x0814</td>
</tr>
<tr>
<td>0x298</td>
<td>Frame Check Sequence Error (LSW)</td>
<td>STAT_RX_PACKET_BAD_FCS_LSB</td>
<td>0x08C8</td>
</tr>
<tr>
<td>0x29C</td>
<td>Frame Check Sequence Error (MSW)</td>
<td>STAT_RX_PACKET_BAD_FCS_MSB</td>
<td>0x08CC</td>
</tr>
<tr>
<td>0x2A0</td>
<td>Broadcast Frames received OK (LSW)</td>
<td>STAT_RX_BROADCAST_LSB</td>
<td>0x08E8</td>
</tr>
</tbody>
</table>
### Table A-9: Statistics Counters (Cont’d)

<table>
<thead>
<tr>
<th>Address (Hex)</th>
<th>Register</th>
<th>Register</th>
<th>Address (Hex)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x2A4</td>
<td>Broadcast Frames received OK (MSW)</td>
<td>STAT_RX_BROADCAST_MSB</td>
<td>0x8EC</td>
</tr>
<tr>
<td>0x2A8</td>
<td>Multicast Frames received OK (LSW)</td>
<td>STAT_RX_MULTICAST_LSB</td>
<td>0x08E0</td>
</tr>
<tr>
<td>0x2AC</td>
<td>Multicast Frames received OK (MSW)</td>
<td>STAT_RX_MULTICAST_MSB</td>
<td>0x08E4</td>
</tr>
<tr>
<td>0x2B0</td>
<td>Control Frames received OK (LSW)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0x2B4</td>
<td>Control Frames received OK (MSW)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0x2B8</td>
<td>Length/Type out of range (LSW)</td>
<td>STAT_RX_INRANGEERR_LSB</td>
<td>0x0908</td>
</tr>
<tr>
<td>0x2BC</td>
<td>Length/Type out of range (MSW)</td>
<td>STAT_RX_INRANGEERR_MSB</td>
<td>0x090C</td>
</tr>
<tr>
<td>0x2C0</td>
<td>VLAN tagged frames received OK (LSW)</td>
<td>STAT_RX_VLAN_LSB</td>
<td>0x08F0</td>
</tr>
<tr>
<td>0x2C4</td>
<td>VLAN tagged frames received OK (MSW)</td>
<td>STAT_RX_VLAN_MSB</td>
<td>0x08F4</td>
</tr>
<tr>
<td>0x2C8</td>
<td>PAUSE frames received OK (LSW)</td>
<td>STAT_RX_PAUSE_LSB</td>
<td>0x08F8</td>
</tr>
<tr>
<td>0x2CC</td>
<td>PAUSE frames received OK (MSW)</td>
<td>STAT_RX_PAUSE_MSB</td>
<td>0x08FC</td>
</tr>
<tr>
<td>0x2D0</td>
<td>Control frames received with unsupported opcode (LSW)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0x2D4</td>
<td>Control frames received with unsupported opcode (MSW)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0x2D8</td>
<td>Frames transmitted OK (LSW)</td>
<td>STAT_TX_TOTAL_GOOD_PACKETS_LSB</td>
<td>0x0708</td>
</tr>
<tr>
<td>0x2DC</td>
<td>Frames transmitted OK (MSW)</td>
<td>STAT_TX_TOTAL_GOOD_PACKETS_MSB</td>
<td>0x070C</td>
</tr>
<tr>
<td>0x2E0</td>
<td>Broadcast Frames transmitted OK (LSW)</td>
<td>STAT_TX_BROADCAST_LSB</td>
<td>0x07E0</td>
</tr>
<tr>
<td>0x2E4</td>
<td>Broadcast Frames transmitted OK (MSW)</td>
<td>STAT_TX_BROADCAST_MSB</td>
<td>0x07E4</td>
</tr>
<tr>
<td>0x2E8</td>
<td>Multicast Frames transmitted OK (LSW)</td>
<td>STAT_TX_MULTICAST_LSB</td>
<td>0x07D8</td>
</tr>
<tr>
<td>0x2EC</td>
<td>Multicast Frames transmitted OK (MSW)</td>
<td>STAT_TX_MULTICAST_MSB</td>
<td>0x07DC</td>
</tr>
<tr>
<td>0x2F0</td>
<td>Underrun errors (LSW)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0x2F4</td>
<td>Underrun errors (MSW)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0x2F8</td>
<td>Control Frames transmitted OK (LSW)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0x2FC</td>
<td>Control Frames transmitted OK (MSW)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0x300</td>
<td>VLAN tagged frames transmitted OK (LSW)</td>
<td>STAT_TX_VLAN_LSB</td>
<td>0x07E8</td>
</tr>
</tbody>
</table>
### Appendix A: Migrating and Upgrading

#### Table A-9: Statistics Counters (Cont’d)

<table>
<thead>
<tr>
<th>Address (Hex)</th>
<th>Register</th>
<th>Register</th>
<th>Address (Hex)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x304</td>
<td>VLAN tagged frames transmitted OK (MSW)</td>
<td>STAT_TX_VLAN_MSB</td>
<td>0x07EC</td>
</tr>
<tr>
<td>0x308</td>
<td>PAUSE frames transmitted OK (LSW)</td>
<td>STAT_TX_PAUSE_LSB</td>
<td>0x07F0</td>
</tr>
<tr>
<td>0x30C</td>
<td>PAUSE frames transmitted OK (MSW)</td>
<td>STAT_TX_PAUSE_MSB</td>
<td>0x07F4</td>
</tr>
</tbody>
</table>

#### Pause Processing

The following section outlines the configuration for priority based flow control.

#### Table A-10: Pause Processing

<table>
<thead>
<tr>
<th>Address (Hex)</th>
<th>Register</th>
<th>Register</th>
<th>Address (Hex)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x480</td>
<td>Priority 0 Quanta Register</td>
<td>CONFIGURATION_TX_FLOW_CONTROL_QUANTA_REG1 [15:0]</td>
<td>0x0058</td>
</tr>
<tr>
<td>0x484</td>
<td>Priority 1 Quanta Register</td>
<td>CONFIGURATION_TX_FLOW_CONTROL_QUANTA_REG1 [31:16]</td>
<td>0x0058</td>
</tr>
<tr>
<td>0x488</td>
<td>Priority 2 Quanta Register</td>
<td>CONFIGURATION_TX_FLOW_CONTROL_QUANTA_REG2 [15:0]</td>
<td>0x005C</td>
</tr>
<tr>
<td>0x48C</td>
<td>Priority 3 Quanta Register</td>
<td>CONFIGURATION_TX_FLOW_CONTROL_QUANTA_REG2 [31:16]</td>
<td>0x005C</td>
</tr>
<tr>
<td>0x490</td>
<td>Priority 4 Quanta Register</td>
<td>CONFIGURATION_TX_FLOW_CONTROL_QUANTA_REG3 [15:0]</td>
<td>0x0060</td>
</tr>
<tr>
<td>0x494</td>
<td>Priority 5 Quanta Register</td>
<td>CONFIGURATION_TX_FLOW_CONTROL_QUANTA_REG3 [31:16]</td>
<td>0x0060</td>
</tr>
<tr>
<td>0x498</td>
<td>Priority 6 Quanta Register</td>
<td>CONFIGURATION_TX_FLOW_CONTROL_QUANTA_REG4 [15:0]</td>
<td>0x0064</td>
</tr>
<tr>
<td>0x49C</td>
<td>Priority 7 Quanta Register</td>
<td>CONFIGURATION_TX_FLOW_CONTROL_QUANTA_REG4 [31:16]</td>
<td>0x0064</td>
</tr>
<tr>
<td>0x4A0</td>
<td>Legacy Pause Refresh Register</td>
<td>Does not support Legacy PFC</td>
<td></td>
</tr>
</tbody>
</table>
MDIO Control Registers

The 10G/25G High Speed Ethernet IP does not provide an MDIO station master and thus does not have any of the MDIO control registers.

Interrupt Registers

Typically interrupts are generated after an MDIO operation to indicate completion and because there is no MDIO master there are no interrupt registers.

PCS/PMA MDIO register map

Again, because there is no MDIO interface provided, there are no MDIO registers for the PCS/PMA interface.
Debugging

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

**TIP:** If the IP generation halts with an error, there might be a license issue. See License Checkers in Chapter 1 for more details.

### Finding Help on Xilinx.com

To help in the design and debug process when using the 10G/25G Ethernet, the [Xilinx Support web page](https://www.xilinx.com) 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 10G/25G Ethernet. This guide, along with documentation related to all products that aid in the design process, can be found on the [Xilinx Support web page](https://www.xilinx.com) or by using the Xilinx Documentation Navigator.

Download the Xilinx Documentation Navigator from the [Downloads page](https://www.xilinx.com). For more information about this tool and the features available, open the online help after installation.

### Solution Centers

See the [Xilinx Solution Centers](https://www.xilinx.com) for support on devices, software tools, and intellectual property at all stages of the design cycle. Topics include design assistance, advisories, and troubleshooting tips.

Refer to the [Xilinx Ethernet IP Solution Center](https://www.xilinx.com).
Answer Records

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

Answer Records for this subsystem 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 10G/25G Ethernet

AR: 64710

Technical Support

Xilinx provides technical support in the Xilinx Support web page for this 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, go to the Xilinx Support web page.
Debug Tools

There are many tools available to address 10G/25G Ethernet design issues. It is important to know which tools are useful for debugging various situations.

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 IDE is used for logic debugging and validation of a design running in Xilinx devices.

The Vivado logic analyzer is used with the logic debug IP cores, including:

- ILA 2.0 (and later versions)
- VIO 2.0 (and later versions)

See the Vivado Design Suite User Guide: Programming and Debugging (UG908) [Ref 8].

Reference Boards

Various Xilinx development boards support the 10G/25G Ethernet. These boards can be used to prototype designs and establish that the core can communicate with the system.

- UltraScale™ FPGA evaluation boards
  - VCU107
  - VCU108

Simulation Debug

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.
Appendix B: Debugging

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.

1. Use a faster computer with more memory.
2. Make use of a Platform Load Sharing Facility (LSF) if available in your organization.
3. Bypass the Xilinx transceiver (this might require that the customer create their own test bench.
4. Send fewer packets.
5. If using the example design, see Simulation Speed Up in Chapter 4 to speed up wait timers in the example design.

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.

The simulation debug flow for Questa® SIM is illustrated in Figure B-1. A similar approach can be used with other simulators.
Appendix B: Debugging

Figure B-1: QuestaSim Simulation Debug Flow
Appendix B: Debugging

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 debug feature for debugging the specific problems.

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.

Timing

Ensure that timing is met according to the Vivado tools before attempting to implement the IP in hardware.

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.
**Appendix B: Debugging**

### Ethernet Specific Checks

A number of issues can commonly occur during the first hardware test of an 10G/25G Ethernet. These should be checked as indicated below.

It is assumed that the 10G/25G Ethernet has already passed all simulation testing which is being implemented in hardware. This is a pre-requisite 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 the SerDes achieves clock data recovery (CDR) lock.
3. Check that the 10/25G IP has achieved word sync.

### Signal Integrity

When bringing up a board for the first time and the 10/25G Ethernet does not seem to be achieving word sync, the most likely problem is related to signal integrity. Signal integrity issues must be addressed before any other debugging can take place.

Signal integrity should be debugged independently from the 10G/25G Ethernet. The following procedures should be carried out. (Note that 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 802.3 specification.)

- Transceiver Settings
- Checking For Noise
- Bit Error Rate Testing

If assistance is required for transceiver and signal integrity debugging, contact Xilinx technical support.

### N/P Swapping

If the positive and negative signals of a differential pair are swapped, then data cannot be correctly received on that lane. You should verify that the link has the correct polarity of each differential pair.
Appendix B: Debugging

Clocking and Resets

Refer to the Clocking and Resets in Chapter 3 for these requirements.

Ensure that the clock frequencies for both the 10G/25G Ethernet as well as the Xilinx Transceiver reference clock match the configuration requested when the subsystem 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 10G/25G Ethernet is taken out of reset. This applies to both the SerDes clock as well as the core clock.

If any subsequent instability is detected in a clock, the 10G/25G Ethernet must be reset. One example of such instability is a loss of CDR lock. The user logic should determine all external conditions which would require a reset (e.g. clock glitches, loss of CDR lock, power supply glitches, etc.).

The GT requires a GTRXRESET after the serial data becomes valid to ensure correct CDR lock to the data. This is required after powering on, resetting or reconnecting the link partner. At the core level to avoid interruption on the TX side of the link, the reset can be triggered using gtwiz_reset_rx_datapath. If available, signal detect or inversion of loss of signal from the optics can be used to trigger the reset. If signal detect or loss of signal are not available, timeout logic can be added to monitor if alignment has not completed and issue the gtwiz_reset_rx_datapath reset.

Configuration changes cannot be made unless the subsystem 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.
RX Debug

Consult the port list section for a description of the diagnostic signals which are available to debug the RX.

\textbf{stat\_rx\_block\_lock}

This signal indicates that the receiver has detected and locked to the word boundaries as defined by a 01 or 10 control or data header. This is the first step to ensure that the 10/25G Ethernet IP is functioning normally.

\textbf{CAUTION!} Under some conditions of no signal input, the SerDes receiver exhibits a steady pattern of alternating 1010101.... This can cause erroneous block lock, but still indicates that the receiver has detected the pattern.

\textbf{stat\_rx\_bad\_fcs}

A bad FCS indicates a bit error in the received packet. An FCS error could be due to any number of causes of packet corruption such as noise on the line.

\textbf{stat\_rx\_local\_fault}

A local fault indication can be locally generated or received. Some causes of a local fault are:

\begin{itemize}
  \item block lock not complete
  \item high bit error rate
  \item overflow or underflow
\end{itemize}

\textbf{Loopback Check}

If the Ethernet packets are being transmitted properly according to 802.3, there should not be RX errors. However, the signal integrity of the received signals must be verified first.

To aid in debug, a local loopback can be performed with the signal \texttt{ctl\_local\_loopback}. This connects the TX SerDes to the RX SerDes, effectively bypassing potential signal integrity problems. The transceiver is placed into "PMA loopback", which is fully described in the transceiver product guide. In this way, the received data can be checked against the transmitted packets to verify that the logic is operating properly.
Protocol Interface Debug

To achieve error-free data transfers with the 10G/25G Ethernet, the 802.3 specification should be followed. Note that signal integrity should always be ensured before proceeding to the protocol debug.

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. Make sure there are no descrambler state errors.
3. Eliminate CRC32 errors, if any.
4. Make sure the protocol is being followed correctly.
5. Ensure that there are no overflow or underflow conditions when packets are sent.

Statistics Counters

After error-free communication has been achieved, the statistics indicators can be monitored to ensure that traffic characteristics meet expectations. Note that some signals are strobes only, which means that the counters are not part of the subsystem. This is done so that the counter size can be customized. Counters are optionally available with the AXI interface.
Appendix C

Additional Resources and Legal Notices

Xilinx Resources

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

References

These documents provide supplemental material useful with this product guide:

1. 25G and 50G Ethernet Consortium Schedule 3 version 1.6 (August 18, 2015) (http://25gethernet.org/)
2. IEEE Standard 802.3-2015
12. 25G IEEE 802.3by Reed-Solomon Forward Error Correction LogiCORE IP Product Guide (PG217) Registration required for access
Revision History

The following table shows the revision history for this document.

<table>
<thead>
<tr>
<th>Date</th>
<th>Version</th>
<th>Revision</th>
</tr>
</thead>
</table>
| 11/30/2016 | 2.0     | • Modified “tx_reset and rx_reset” to “s_axi_arstn’ and “active-High” to “active-Low” in the Configuration Register Map section in Chapter 2.  
• Added text about clearing status registers to the first paragraph of the Status Register Map section in Chapter 2.  
• Added text about clearing statistics counters to the first paragraph of the Statistics Counters section in Chapter 2.  
• Updated table notes 3 and 4 and added table note 5 for Table 4-1, Configuration Options.  
• Added Note about Auto Negotiation/Link training to the Overview section in Chapter 5.  
• Added many ports and deleted many ports in Table 5-2, Core xci Top Level Port List. See the Migrating and Updating appendix.  
• Removed the text “GT Selection and” throughout the Descriptions in Table 2, Core xci Top Level Port List. |
| 10/05/2016 | 2.0     | • Added migration from legacy 10G EMAC to Appendix A Migrating and Updating.  
• Added AUTOMOTIVE APPLICATIONS DISCLAIMER.  
• Added references to tick_reg_mode_sel throughout.  
• Updated the following figures: 3-6, 3-7, 3-8, 3-9, 4-1, 4-2, 4-3, 4-4  
• Updated the description of tx_ptp_1588op_in[1:0] and rx_ptp_tstamp_out[80-1:0] in Table 3-1.  
• Added support for one-step operation.  
• Updated several of the port descriptions in Table 3-3.  
• Added Ethernet MAC value to Select Core option in Table 4-1.  
• Replaced Include FEC Logic option with Clause 74 (BASE-KR FEC) in Table 4-1.  
• Added Clause 108 (RS-FEC) option to Table 4-1.  
• Added the new subsection Simulation Speed Up to the Simulation section in Chapter 4.  
• Updated the description of ctl_rx_rate_10g_25gn_* in Table 5-2.  
• Added the rx_ptp_tstamp_valid_out_* to Table 5-2.  
• Added several new subsection under the AXI4-Lite Interface Implementation section in Chapter 5.  
• Added Step 5 in the Slow Simulation section in the Debugging appendix.  
• Added a new paragraph about GTRXRESET in the Clocking and Resets section in the Debugging appendix.  
• Added the Core xci Top Level Port List section to Chapter 5.  
• Updated IEEE references to 2015 instead of 2012. |
<table>
<thead>
<tr>
<th>Date</th>
<th>Version</th>
<th>Revision</th>
</tr>
</thead>
<tbody>
<tr>
<td>06/08/2016</td>
<td>1.3</td>
<td></td>
</tr>
</tbody>
</table>

- Changed 10 Gb/s to 10.3125 Gb/s throughout
- Updated Figures 2-2, 3-16, 4-1, 4-2, 4-3, 4-4, 5-1, 5-2, 5-3, 5-4, 5-5
- Added XGMII to XVGMII throughout.
- Changed XXVMII to XVGMII throughout.
- Added notes for addresses that support MAC+PCS throughout.
- Added Hex Addresses and links in Table 2-24.
- Added Bits to Tables 2-25, 2-28, 2-86
- Added new register tables for STAT_TX_RSFEC_STATUS_REG: 044C, STAT_RX_ERROR_LSB: 0668, STAT_RX_ERROR_MSB: 066C, STAT_RX_RSFEC_ERR_COUNT0_INC_LSB: 0680, STAT_RX_RSFEC_ERR_COUNT0_INC_MSB: 0684
- Removed General Design Guidelines section in Chapter 3.
- Added tx_ptp_rxtstamp_in to Table 3-1
- Changed “HSEC” to “10G/25G High Speed Ethernet subsystem” throughout.
- Added Control and Statistics Interface section to Table 4-1
- Added GT Location section to Table 4-3 and updated options in the Others section
- Updated some descriptions in Table 4-3, GT Clock Options
- Updated Overview in the Chapter 5, Example Design
- Updated the descriptions of the optional modules.
- Added the Example Design Hierarchy (GT in example design), Runtime Switchable, and IEEE Clause 108 (RS-FEC) Integration sections to Chapter 5.
- Completely revised the Shared Logic Implementation section in Chapter 5.
- Added descriptions of the modules that are part of the shared logic wrapper.
- Completely revised the Simulation Debug section in Appendix B, Debugging.
- Changed 802.3-2012 to 802.3-2015 throughout.
- Added one-step operation throughout.
- Updated several port descriptions in Table 3-3.
- Added Simulation Speed Up section to Chapter 4.
- Added new port rx_ptp_tstamp_valid_out_* to Table 5-2.
- Added Step 5 to the Slow Simulation section in the Debugging appendix.
- Added a paragraph about GTRXRESET to the Clocking and Resets section the Debugging appendix.
### Appendix C: Additional Resources and Legal Notices

<table>
<thead>
<tr>
<th>Date</th>
<th>Version</th>
<th>Revision</th>
</tr>
</thead>
<tbody>
<tr>
<td>04/06/2016</td>
<td>1.2</td>
<td>- Added UltraScale+ support.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Added new section that has RSFEC, 1588 1-step and 2-step support.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Added new IEEE 1588 Timestamping section.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Added rx_preambleout [55:0] for both AXI4-Stream interfaces.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Added tx_preamblein [55:0] for AXI4-Stream interface.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Added registers to the Configuration, Status, and Counter register maps.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Changed custom preamble from in-band to out-of-band.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Added text about pm_tick and TIC_REG to Statistics Counter section</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Changed polarity of the tx_axis_tuser and rx_axis_tuser signals.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Updated Figure 3-13 and Figure 3-14.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Removed VLane Adjust Mode from Table 4-2.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Removed LBUS material.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Added ctl_tx_ipg_value[3:0] to Table C-4.</td>
</tr>
<tr>
<td>12/02/2015</td>
<td>1.1</td>
<td>- Updated the performance and resource utilization data link.</td>
</tr>
<tr>
<td>11/18/2015</td>
<td>1.1</td>
<td>- Added a link to the performance and resource utilization data on the web</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Added the stat_rx_valid_ctrl_code, ctl_tx_custom_preamble_enable, and</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ctl_rx_custom_preamble_enable signal.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Updated the tx_axis_tuser signal description.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Updated the Normal Transmission and Aborting a Transmission</td>
</tr>
<tr>
<td></td>
<td></td>
<td>information in the Transmit AXI4-Stream Interface section.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Added Vivado IDE option details in Design Flow Steps chapter.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Added new information in Example Design chapter.</td>
</tr>
<tr>
<td>09/30/2015</td>
<td>1.0</td>
<td>- Initial Xilinx release.</td>
</tr>
</tbody>
</table>
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 https://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 https://www.xilinx.com/legal.htm#tos.

AUTOMOTIVE APPLICATIONS DISCLAIMER

AUTOMOTIVE PRODUCTS (IDENTIFIED AS “XA” IN THE PART NUMBER) ARE NOT WARRANTED FOR USE IN THE DEPLOYMENT OF AIRBAGS OR FOR USE IN APPLICATIONS THAT AFFECT CONTROL OF A VEHICLE (“SAFETY APPLICATION”) UNLESS THERE IS A SAFETY CONCEPT OR REDUNDANCY FEATURE CONSISTENT WITH THE ISO 26262 AUTOMOTIVE SAFETY STANDARD (“SAFETY DESIGN”). CUSTOMER SHALL, PRIOR TO USING OR DISTRIBUTING ANY SYSTEMS THAT INCORPORATE PRODUCTS, THOROUGHLY TEST SUCH SYSTEMS FOR SAFETY PURPOSES. USE OF PRODUCTS IN A SAFETY APPLICATION WITHOUT A SAFETY DESIGN IS FULLY AT THE RISK OF CUSTOMER, SUBJECT ONLY TO APPLICABLE LAWS AND REGULATIONS GOVERNING LIMITATIONS ON PRODUCT LIABILITY.

© Copyright 2015–2016 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.