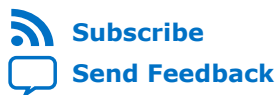




# 25G Ethernet Intel® Stratix® 10 FPGA IP User Guide

Updated for Intel® Quartus® Prime Design Suite: **21.1**

IP Version: **19.4.0**



[Subscribe](#)

[Send Feedback](#)

**UG-20109 | 2021.03.29**

Latest document on the web: [PDF](#) | [HTML](#)

## Contents

---

|  |           |
|--|-----------|
| <b>1. About the 25G Ethernet Intel FPGA IP.....</b>  | <b>4</b>  |
| 1.1. Release Information.....  | 8         |
| 1.2. 25G Ethernet Intel FPGA IP Supported Features.....                                    | 9         |
| 1.3. 25G Ethernet Intel FPGA IP Core Device Family and Speed Grade Support.....            | 10        |
| 1.3.1. 25G Ethernet Intel FPGA IP Core Device Family Support.....                          | 10        |
| 1.3.2. 25G Ethernet Intel FPGA IP Core Device Speed Grade Support.....                     | 11        |
| 1.4. IP Core Verification.....   | 11        |
| 1.4.1. Simulation Environment.....   | 12        |
| 1.4.2. Compilation Checking.....   | 12        |
| 1.4.3. Hardware Testing.....   | 12        |
| 1.5. Performance and Resource Utilization.....   | 12        |
| <b>2. Getting Started.....</b>   | <b>15</b> |
| 2.1. Installing and Licensing Intel FPGA IP Cores.....                                     | 15        |
| 2.1.1. Intel FPGA IP Evaluation Mode.....  | 16        |
| 2.2. Specifying the Intel Stratix 10 IP Core Parameters and Options.....                   | 18        |
| 2.3. Simulating the IP Core.....   | 18        |
| 2.4. Generated File Structure.....   | 20        |
| 2.5. Integrating Your IP Core in Your Design.....  | 22        |
| 2.5.1. Pin Assignments.....  | 22        |
| 2.5.2. Adding the Transceiver PLL .....  | 22        |
| 2.5.3. Adding the External Time-of-Day Module for Variations with 1588 PTP<br>Feature..... | 25        |
| 2.5.4. Placement Settings for the 25G Ethernet Intel FPGA IP Core.....                     | 27        |
| 2.6. Compiling the Full Design and Programming the FPGA.....                               | 27        |
| <b>3. 25G Ethernet Intel FPGA IP Parameters.....</b>                                       | <b>28</b> |
| <b>4. Functional Description.....</b>  | <b>31</b> |
| 4.1. 25G Ethernet Intel FPGA IP Core Functional Description.....                           | 31        |
| 4.1.1. 25G Ethernet Intel FPGA IP Core TX MAC Datapath.....                                | 32        |
| 4.1.2. 25 GbE TX PCS.....  | 34        |
| 4.1.3. TX RS-FEC.....  | 34        |
| 4.1.4. 25G Ethernet Intel FPGA IP Core RX MAC Datapath.....                                | 34        |
| 4.1.5. Link Fault Signaling Interface.....   | 39        |
| 4.1.6. 25 GbE RX PCS.....  | 40        |
| 4.1.7. RX RS-FEC.....  | 41        |
| 4.1.8. Flow Control.....   | 41        |
| 4.1.9. 1588 Precision Time Protocol Interfaces.....  | 44        |
| 4.2. User Interface to Ethernet Transmission.....  | 53        |
| 4.2.1. Order of Transmission.....  | 53        |
| 4.2.2. Bit Order For TX and RX Datapaths.....  | 54        |
| <b>5. Reset.....</b>   | <b>55</b> |
| <b>6. Interfaces and Signal Descriptions.....</b>  | <b>56</b> |
| 6.1. TX MAC Interface to User Logic.....   | 57        |
| 6.2. RX MAC Interface to User Logic.....   | 59        |
| 6.3. Transceivers.....   | 61        |

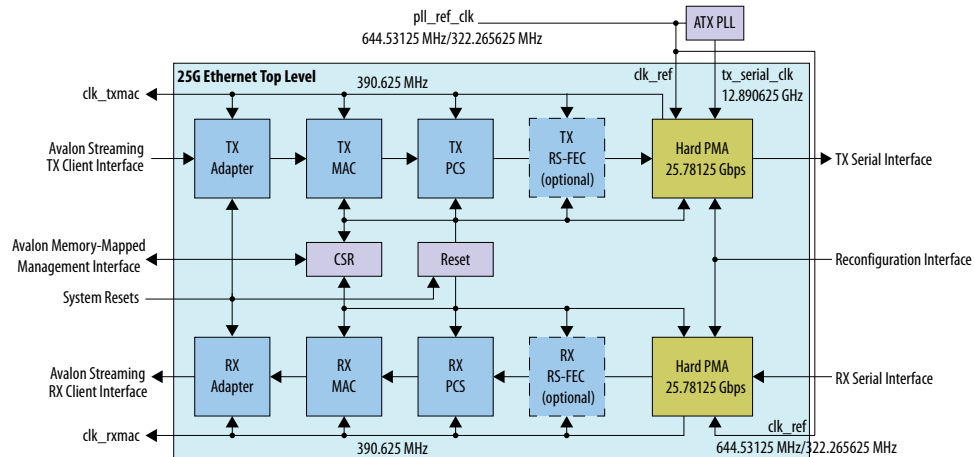
|  |           |
|--|-----------|
| 6.4. Transceiver Reconfiguration Signals.....  | 62        |
| 6.4.1. Accessing the Native PHY Registers in H-Tile Devices.....                                   | 63        |
| 6.4.2. Accessing the Native PHY Registers in L-Tile Devices.....                                   | 63        |
| 6.5. Avalon Memory-Mapped Management Interface.....  | 65        |
| 6.6. PHY Interface Signals.....  | 66        |
| 6.7. 1588 PTP Interface Signals.....   | 67        |
| 6.8. Miscellaneous Status and Debug Signals.....   | 72        |
| 6.9. Reset Signals.....  | 73        |
| <b>7. Control, Status, and Statistics Register Descriptions.....</b>                               | <b>74</b> |
| 7.1. PHY Registers.....  | 75        |
| 7.2. TX MAC Registers.....   | 77        |
| 7.3. RX MAC Registers.....   | 78        |
| 7.4. Pause/PFC Flow Control Registers.....   | 79        |
| 7.5. Statistics Registers.....   | 84        |
| 7.5.1. TX Statistics Registers.....  | 85        |
| 7.5.2. RX Statistics Registers.....  | 88        |
| 7.6. 1588 PTP Registers.....   | 92        |
| 7.7. TX Reed-Solomon FEC Registers.....  | 94        |
| 7.8. RX Reed-Solomon FEC Registers.....  | 95        |
| <b>8. Debugging the Link.....</b>  | <b>96</b> |
| 8.1. Error Insertion Test and Debugging.....   | 97        |
| <b>9. 25G Ethernet Intel Stratix 10 FPGA IP User Guide Archives.....</b>                           | <b>98</b> |
| <b>10. Document Revision History for the 25G Ethernet Intel Stratix 10 FPGA IP User Guide.....</b> | <b>99</b> |

## 1. About the 25G Ethernet Intel FPGA IP

The 25G Ethernet Intel FPGA IP implements the *25G & 50G Ethernet Specification, Draft 1.6* from the 25 Gigabit Ethernet Consortium and the *IEEE 802.3by 25Gb Ethernet* specification. The IP includes an option to support unidirectional transport as defined in *Clause 66* of the *IEEE 802.3-2012 Ethernet Standard*. The MAC client side interface for the 25G Ethernet Intel FPGA IP is a 64-bit Avalon® streaming interface. It maps to one 25.78125 Gbps transceiver. The IP optionally includes the *IEEE 802.3-2018 Clause 108* Reed-Solomon forward error correction (RS-FEC) for support of *IEEE802.3-2018 Clause 107* 25GBASE-R PCS. *IEEE 802.3 Clause 73* Auto-Negotiation and *IEEE 802.3 Clause 74* CR/KR-FEC are not supported. Transceiver interface to 25GBASE-SR optical Physical Medium Dependent (PMD) transceiver is supported.

The IP provides standard media access control (MAC) and physical coding sublayer (PCS), Reed-Solomon Forward Error Correction (RS-FEC), and PMA functions shown in the following block diagrams. The PHY comprises the PCS, optional RS-FEC, and elective PMA.

**Figure 1. 25G Ethernet MAC, PCS, and PMA IP Block Diagram**



Intel Corporation. All rights reserved. Agilix, Altera, Arria, Cyclone, eASIC, Intel, the Intel logo, MAX, Nios, Quartus and Stratix words and logos are trademarks of Intel Corporation or its subsidiaries in the U.S. and/or other countries. Intel warrants performance of its FPGA and semiconductor products to current specifications in accordance with Intel's standard warranty, but reserves the right to make changes to any products and services at any time without notice. Intel assumes no responsibility or liability arising out of the application or use of any information, product, or service described herein except as expressly agreed to in writing by Intel. Intel customers are advised to obtain the latest version of device specifications before relying on any published information and before placing orders for products or services.

\*Other names and brands may be claimed as the property of others.

Figure 2. 10G/25G Ethernet MAC, PCS, and PMA IP Block Diagram

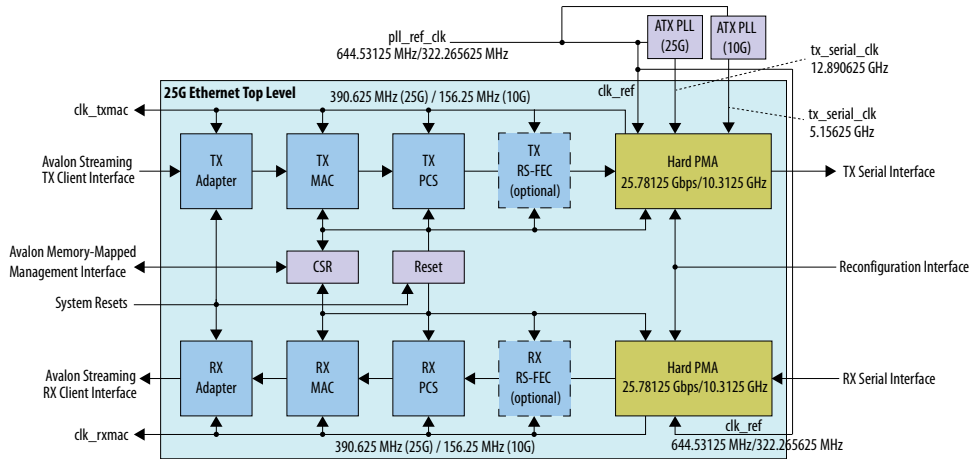


Figure 3. 25G Ethernet MAC and PCS IP Block Diagram

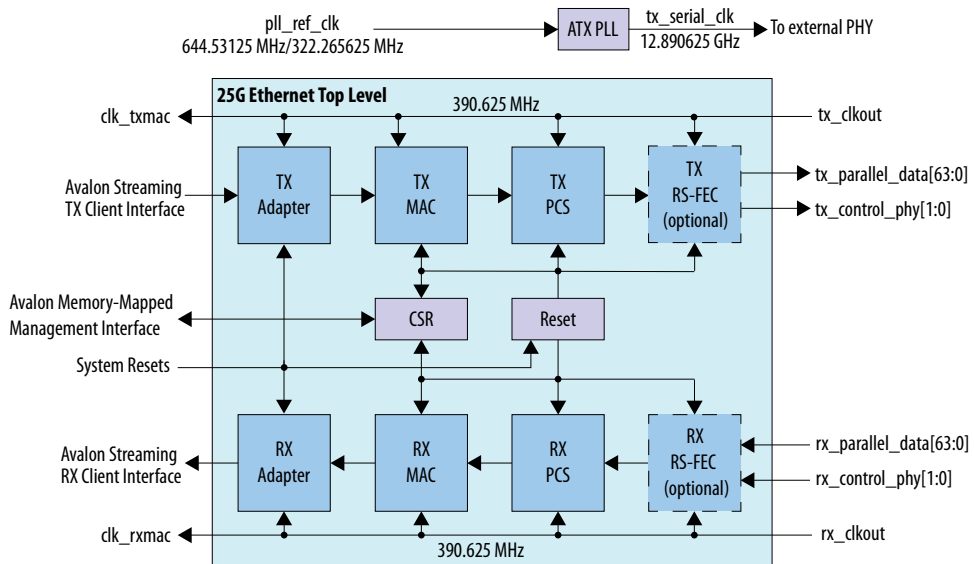
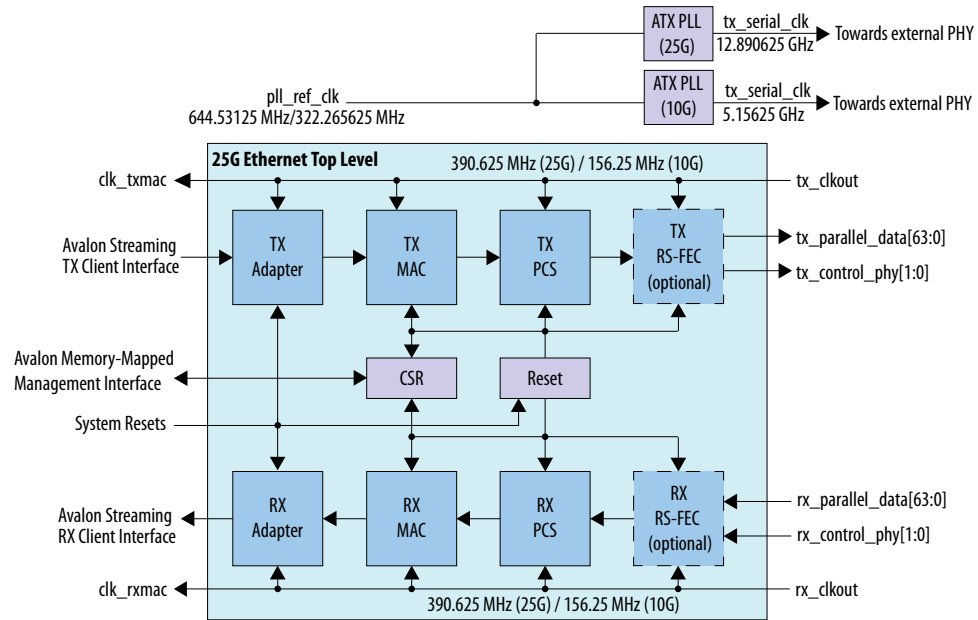


Figure 4. 10G/25G Ethernet MAC and PCS IP Block Diagram

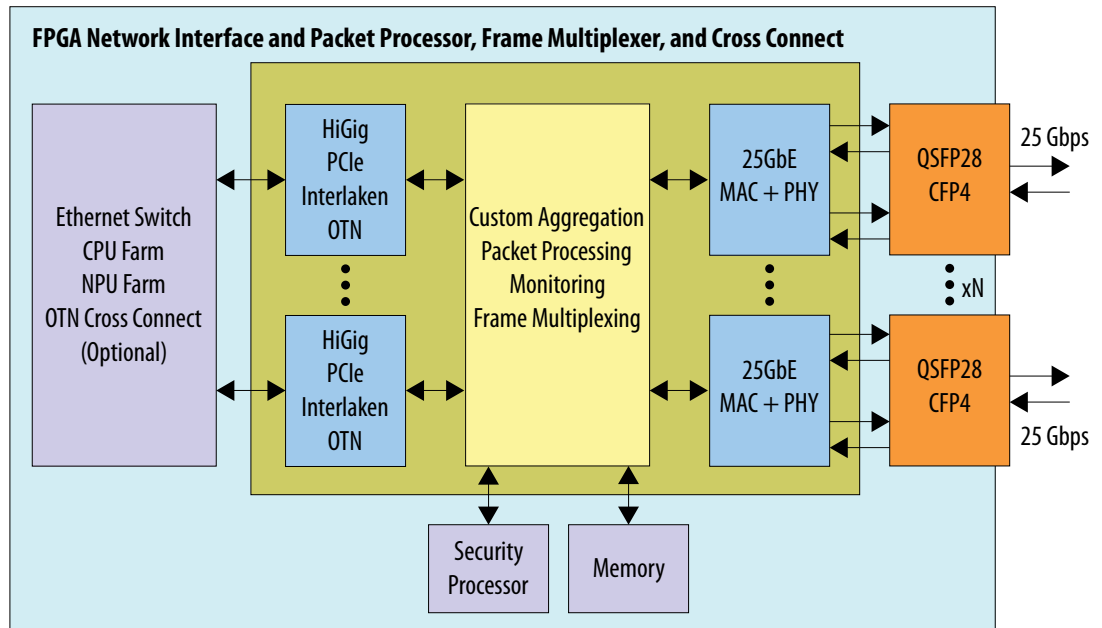


Note:

1. To configure the IP between 10G and 25G, follow the reconfiguration sequence as defined in the *L- and H-Tile Transceiver PHY User Guide*. For simplification, refer to the reconfiguration sequencer module from the design example, which is not part the IP.
2. For MAC + PCS core variant, follow the reset sequence guideline as defined in *Recommended Reset Sequence* of the *L- and H-Tile Transceiver PHY User Guide* to ensure the 25G Ethernet Intel FPGA IP is having a proper reset sequence.

The following block diagram shows an example of a network application with 25G Ethernet Intel FPGA IP MAC and PHY.

Figure 5. Example Network Application



#### Related Information

- [25 Gigabit Ethernet Consortium](#)
- [L- and H-Tile Transceiver PHY User Guide](#)
- [25G Ethernet Intel® Stratix® 10 FPGA IP Design Example User Guide](#)

## 1.1. Release Information

Intel® FPGA IP versions match the Intel Quartus® Prime Design Suite software versions until v19.1. Starting in Intel Quartus Prime Design Suite software version 19.2, Intel FPGA IP has a new versioning scheme.

The Intel FPGA IP version (X.Y.Z) number can change with each Intel Quartus Prime software version. A change in:

- X indicates a major revision of the IP. If you update the Intel Quartus Prime software, you must regenerate the IP.
- Y indicates the IP includes new features. Regenerate your IP to include these new features.
- Z indicates the IP includes minor changes. Regenerate your IP to include these changes.

**Table 1. 25G Ethernet Intel FPGA IP Core Current Release Information**

| Item                        | Description  |
|-----------------------------|--|
| IP Version                  | 19.4.0   |
| Intel Quartus Prime Version | 20.1   |
| Release Date                | 2020.04.13   |
| Ordering Codes              | Variations without 1588 PTP option and without FEC option:<br>IP-25GEUMACPHY (IPR-25GEUMACPHY for renewal)<br>Variations with 1588 PTP option and without FEC option:<br>IP-25GEUMACPHYF (IPR-25GEUMACPHYF for renewal)<br>Variations without 1588 PTP option and with FEC option:<br>IP-25GEUMACPHYFC (IPR-25GEUMACPHYFC for renewal)<br>Variations with 1588 PTP option and with FEC option:<br>IP-25GEUMACPHYFFC (IPR-25GEUMACPHYFFC for renewal) |

### Related Information

[25G Ethernet Intel FPGA IP Release Note](#)

Describes changes to the IP in a particular release.



## 1.2. 25G Ethernet Intel FPGA IP Supported Features

The 25G Ethernet Intel FPGA IP is designed to the *25G & 50G Ethernet Specification, Draft 1.6* from the 25 Gigabit Ethernet Consortium and designed to the *IEEE 802.3by 25Gb Ethernet* specification, as well as the *IEEE 802.3ba-2012 High Speed Ethernet Standard* available on the IEEE website ([www.ieee.org](http://www.ieee.org)). The MAC provides RX cut-through frame processing to optimize latency. The IP supports the following features:

- PHY features:
  - *IEEE 802.3-2018 Ethernet Standard Clause 107* for 25GBASE-R and Clause 49 for 10GBASE-R compliant soft PCS logic that interfaces seamlessly to Intel Stratix® 10 FPGA 25.78125 gigabits per second (Gbps) or 10.3125 Gbps serial transceivers.
  - Support for dynamic reconfiguration between the Ethernet data rates of 25.78125 Gbps and 10.3125 Gbps.
  - *IEEE 802.3-2018 Ethernet Standard Clause 108* optional soft Reed-Solomon forward error correction (RS-FEC).
  - *IEEE 802.3-2018 Ethernet Standard Clause 109* elective physical medium attachment (PMA) for interface to 25GBASE-SR optical PMD transceiver.
  - Supports adaptive mode for RX PMA Adaptation.
- Frame structure control features:
  - Support for jumbo packets, defined as packets greater than 1500 bytes.
  - Receive (RX) CRC removal and pass-through control.
  - Transmit (TX) CRC generation and insertion.
  - RX and TX preamble pass-through option for applications that require proprietary user management information transfer.
  - TX automatic frame padding to meet the 64-byte minimum Ethernet frame length.
- Frame monitoring and statistics:
  - RX CRC checking and error reporting.
  - RX malformed packet checking per IEEE specification.
  - Optional statistics counters.
  - Optional fault signaling detects and reports local fault and generates remote fault, with *IEEE 802.3ba-2012 Ethernet Standard Clause 46* support.
  - Unidirectional transport as defined in *Clause 66 of the IEEE 802.3-2012 Ethernet Standard*.
- Flow control:
  - Standard *IEEE 802.3 Clause 31* and Priority-Based *IEEE 802.1Qbb* flow control.

- Precision Time Protocol support:
  - Optional support for the IEEE Standard 1588-2008 Precision Clock Synchronization Protocol (1588 PTP). This feature supports PHY operating speed with a constant timestamp accuracy of  $\pm 4$  ns and a dynamic timestamp accuracy of  $\pm 1$  ns.
- Debug and testability features:
  - Programmable serial PMA local loopback (TX to RX) at the serial transceiver for self-diagnostic testing.
  - TX error insertion capability.
  - RSFEC TX error injection capability.
  - Optional access to Native PHY Debug Master Endpoint (NPDME) for serial link debugging or monitoring PHY signal integrity.
- User system interfaces:
  - Avalon memory-mapped management interface to access the IP control and status registers.
  - Avalon streaming data path interface connects to client logic.
  - Configurable ready latency of 0 or 3 clock cycles for Avalon streaming TX interface.
  - Hardware and software reset control.

For a detailed specification of the Ethernet protocol refer to the *IEEE 802.3 Ethernet Standard*.

**Related Information**

[IEEE website](#)

The *IEEE 802.3 Ethernet Standard* is available on the IEEE website.

## 1.3. 25G Ethernet Intel FPGA IP Core Device Family and Speed Grade Support

### 1.3.1. 25G Ethernet Intel FPGA IP Core Device Family Support

**Table 2. Intel FPGA IP Core Device Support Levels**

| Device Support Level | Definition  |
|----------------------|---|
| <b>Advance</b>       | The IP core is available for simulation and compilation for this device family. Timing models include initial engineering estimates of delays based on early post-layout information. The timing models are subject to change as silicon testing improves the correlation between the actual silicon and the timing models. You can use this IP core for system architecture and resource utilization studies, simulation, pinout, system latency assessments, basic timing assessments (pipeline budgeting), and I/O transfer strategy (datapath width, burst depth, I/O standards tradeoffs). |
| <b>Preliminary</b>   | The IP core is verified with preliminary timing models for this device family. The IP core meets all functional requirements, but might still be undergoing timing analysis for the device family. It can be used in production designs with caution.   |
| <b>Final</b>         | The IP core is verified with final timing models for this device family. The IP core meets all functional and timing requirements for the device family and can be used in production designs.  |

**Table 3. 25G Ethernet Intel FPGA IP Core Device Family Support**

Shows the level of support offered by the 25G Ethernet Intel FPGA IP core for each Intel FPGA device family.

| Device Family         | Support    |
|-----------------------|------------|
| Intel Stratix 10      | Final      |
| Other device families | No support |

#### Related Information

##### [Timing and Power Models](#)

Reports the default device support levels in the current version of the Quartus Prime Pro Edition software.

### 1.3.2. 25G Ethernet Intel FPGA IP Core Device Speed Grade Support

**Table 4. Supported Device Speed Grades**

| IP Core                    | Device Family  | Supported Speed Grades  |
|----------------------------|--|---|
| 25G Ethernet Intel FPGA IP | Intel Stratix 10 L- and H-tile <sup>(1)</sup> <sup>(2)</sup> | <ul style="list-style-type: none"><li>• Transceiver speed grade: -1 or -2</li><li>• Core speed grade: -1 and -2</li></ul> |

#### Related Information

##### [Stratix 10 GX/SX Device Overview](#)

Provides more information on the sample ordering code and available options for Intel Stratix 10 devices.

## 1.4. IP Core Verification

To ensure functional correctness of the 25G Ethernet Intel FPGA IP core, Intel performs extensive validation through both simulation and hardware testing. Before releasing a version of the 25G Ethernet Intel FPGA IP core, Intel runs comprehensive regression tests in the current version of the Intel Quartus Prime Pro Edition software.

Intel verifies that the current version of the Intel Quartus Prime Pro Edition software compiles the previous version of each IP core. Any exceptions to this verification are reported in the *Intel FPGA IP Release Notes*. Intel does not verify compilation with IP core versions older than the previous release.

#### Related Information

- [Knowledge Base Issues for IP core](#)  
Exceptions to functional correctness are documented in the 25G Ethernet Intel FPGA IP core errata.
- [25G Ethernet Intel FPGA IP Release Notes](#)
- [Intel Quartus Prime Design Suite Update Release Notes](#)  
Includes changes in minor releases (updates).

---

(1) Only Intel Stratix 10 devices ending with "VG", VGS3", and "LG" suffixes in the part number are supported.

(2) Intel Stratix 10 devices with both E- and H-tile transceivers are supported. However, the IP core can only utilize the H-tile transceiver.

### 1.4.1. Simulation Environment

Intel performs the following tests on the 25G Ethernet Intel FPGA IP core in the simulation environment using internal and third-party standard bus functional models (BFM):

- Constrained random tests that cover randomized frame size and contents.
- Assertion based tests to confirm proper behavior of the IP core with respect to the specification.
- Extensive coverage of our runtime configuration space and proper behavior in all possible modes of operation.

### 1.4.2. Compilation Checking

Intel performs compilation testing on an extensive set of 25G Ethernet Intel FPGA IP core variations and designs to ensure the Intel Quartus Prime Pro Edition software places and routes the IP core ports correctly.

### 1.4.3. Hardware Testing

Intel performs hardware testing of the key functions of the 25G Ethernet Intel FPGA IP core using internal loopback and standard 25 Gbps Ethernet network test equipment. The hardware tests also ensure reliable solution coverage for hardware related areas such as performance, link synchronization, and reset recovery.

## 1.5. Performance and Resource Utilization

The following table shows the typical device resource utilization for selected configurations using the current version of the Intel Quartus Prime software. With the exception of M20K memory blocks, the numbers of ALMs and logic registers are rounded up to the nearest 100. The timing margin for this IP core is a minimum of 15%.

**Table 5. IP Core Variation Encoding for Resource Utilization Table for MAC+PCS+PMA Core Variant**

"On" indicates the parameter is turned on. The symbol "—" indicates the parameter is turned off or not available.

| IP Core Variation              | A           | B                              | C                              | D                              |
|--------------------------------|-------------|--------------------------------|--------------------------------|--------------------------------|
| Parameter                      |             |                                |                                |                                |
| Ready Latency                  | 0           | 0                              | 3                              | 3                              |
| Enable RS-FEC                  | —           | On                             | —                              | —                              |
| Core Variant                   | MAC+PCS+PMA |                                |                                |                                |
| Enable flow control            | —           | Standard flow control, 1 queue | Standard flow control, 1 queue | Standard flow control, 1 queue |
| Enable link fault generation   | —           | —                              | On                             | On                             |
| Enable preamble passthrough    | —           | —                              | On                             | On                             |
| Enable TX CRC passthrough      | On          | —                              | —                              | —                              |
| Enable MAC statistics counters | —           | On                             | On                             | On                             |
| <i>continued...</i>            |             |                                |                                |                                |

| IP Core Variation                               | A | B | C  | D  |
|---|---|---|----|----|
| Parameter                                       |   |   |    |    |
| Enable IEEE 1588                                | — | — | On | —  |
| Enable 10G/25G Dynamic Rate Switching           | — | — | —  | On |
| Enable Native PHY Debug Master Endpoint (NPDME) | — | — | —  | On |

**Table 6. IP Core FPGA Resource Utilization for 25G Ethernet Intel FPGA IP Core with MAC+PCS+PMA Core Variant for Intel Stratix 10 Devices**

Lists the resources and expected performance for selected variations of the 25G Ethernet Intel FPGA IP core.

These results were obtained using the Intel Quartus Prime software v20.1.

- The transceiver PLL reference clock frequency is 644.531250 MHz.
- The numbers of ALMs and logic registers are rounded up to the nearest 100.
- The numbers of ALMs, before rounding, are the **ALMs needed** numbers from the Intel Quartus Prime Fitter Report.

| IP Core Variation | ALMs  | Dedicated Logic Registers | Block Memory Bits |
|-------------------|-------|---------------------------|-------------------|
| A                 | 4300  | 9200                      | 0                 |
| B                 | 17700 | 45200                     | 114880            |
| C                 | 14700 | 38400                     | 11912             |
| D                 | 8700  | 18700                     | 1024              |

**Table 7. IP Core Round Trip Latency**

The round trip latency values are based on the following definitions and assumptions:

- Round trip latency is measured as the time taken for a packet to travel from TX Avalon streaming interface to the RX Avalon streaming interface with the IP core in serial loopback mode.
- Latency values are obtained via simulation of the IP Core's example design generated using Intel Quartus Prime software v20.1. These values are expected to be different across different builds.
- Synopsys's VCS simulator is used when measuring the following values. These values may differ across different simulators.

| IP Core Variation | Latency (ns)             |
|-------------------|--------------------------|
| A                 | 210.0                    |
| B                 | 1002.2                   |
| C                 | 465.2                    |
| D                 | 10G: 668.8<br>25G: 265.5 |

**Table 8. IP Core Variation Encoding for Resource Utilization Table for MAC+PCS Core Variant**

"On" indicates the parameter is turned on. The symbol "—" indicates the parameter is turned off or not available.

| IP Core Variation                               | A       | B                              | C                              | D                              |
|---|---------|--------------------------------|--------------------------------|--------------------------------|
| Parameter                                       |         |                                |                                |                                |
| Ready Latency                                   | 0       | 0                              | 3                              | 3                              |
| Enable RS-FEC                                   | —       | On                             | —                              | —                              |
| Core Variant                                    | MAC+PCS |                                |                                |                                |
| Enable flow control                             | —       | Standard flow control, 1 queue | Standard flow control, 1 queue | Standard flow control, 1 queue |
| Enable link fault generation                    | —       | —                              | On                             | On                             |
| Enable preamble passthrough                     | —       | —                              | On                             | On                             |
| Enable TX CRC passthrough                       | On      | —                              | —                              | —                              |
| Enable MAC statistics counters                  | —       | On                             | On                             | On                             |
| Enable IEEE 1588                                | —       | —                              | On                             | —                              |
| Enable 10G/25G Dynamic Rate Switching           | —       | —                              | —                              | On                             |
| Enable Native PHY Debug Master Endpoint (NPDME) | —       | —                              | —                              | On                             |

**Table 9. IP Core FPGA Resource Utilization for 25G Ethernet Intel FPGA IP Core with MAC+PCS Core Variant for Intel Stratix 10 Devices**

Lists the resources and expected performance for selected variations of the 25G Ethernet Intel FPGA IP core.

These results were obtained using the Intel Quartus Prime software v20.1.

- The transceiver PLL reference clock frequency is 644.531250 MHz.
- The numbers of ALMs and logic registers are rounded up to the nearest 100.
- The numbers of ALMs, before rounding, are the **ALMs needed** numbers from the Intel Quartus Prime Fitter Report.

| IP Core Variation | ALMs  | Dedicated Logic Registers | Block Memory Bits |
|-------------------|-------|---------------------------|-------------------|
| A                 | 4300  | 9200                      | 0                 |
| B                 | 17700 | 45600                     | 114880            |
| C                 | 14600 | 37800                     | 11912             |
| D                 | 8600  | 19500                     | 1024              |

**Related Information**

- [25G Ethernet Intel FPGA IP Parameters](#) on page 28  
Information about the parameters and values in the IP core variations.
- [Fitter Resources Reports in the Quartus Prime Pro Edition Help](#)

## 2. Getting Started

### Related Information

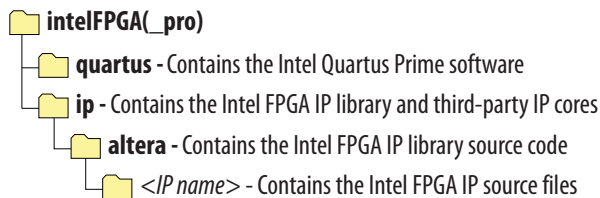
- [Introduction to Intel FPGA IP Cores](#)  
Provides general information about all Intel FPGA IP cores, including parameterizing, generating, upgrading, and simulating IP cores.
- [Creating Version-Independent IP and Platform Designer Simulation Scripts](#)  
Create simulation scripts that do not require manual updates for software or IP version upgrades.
- [Project Management Best Practices](#)  
Guidelines for efficient management and portability of your project and IP files.

### 2.1. Installing and Licensing Intel FPGA IP Cores

The Intel Quartus Prime Pro Edition software installation includes the Intel FPGA IP library. This library provides many useful IP cores for your production use without the need for an additional license. Some Intel FPGA IP cores require purchase of a separate license for production use. The Intel FPGA IP Evaluation Mode allows you to evaluate these licensed Intel FPGA IP cores in simulation and hardware, before deciding to purchase a full production IP core license. You only need to purchase a full production license for licensed Intel IP cores after you complete hardware testing and are ready to use the IP in production.

The Intel Quartus Prime software installs IP cores in the following locations by default:

**Figure 6. IP Core Installation Path**



**Table 10. IP Core Installation Locations**

| Location  | Software                        | Platform |
|---|---------------------------------|----------|
| <drive>:\intelFPGA_pro\quartus\ip\altera          | Intel Quartus Prime Pro Edition | Windows* |
| <home directory>:/intelFPGA_pro/quartus/ip/altera | Intel Quartus Prime Pro Edition | Linux*   |

### 2.1.1. Intel FPGA IP Evaluation Mode

The free Intel FPGA IP Evaluation Mode allows you to evaluate licensed Intel FPGA IP cores in simulation and hardware before purchase. Intel FPGA IP Evaluation Mode supports the following evaluations without additional license:

- Simulate the behavior of a licensed Intel FPGA IP core in your system.
- Verify the functionality, size, and speed of the IP core quickly and easily.
- Generate time-limited device programming files for designs that include IP cores.
- Program a device with your IP core and verify your design in hardware.

Intel FPGA IP Evaluation Mode supports the following operation modes:

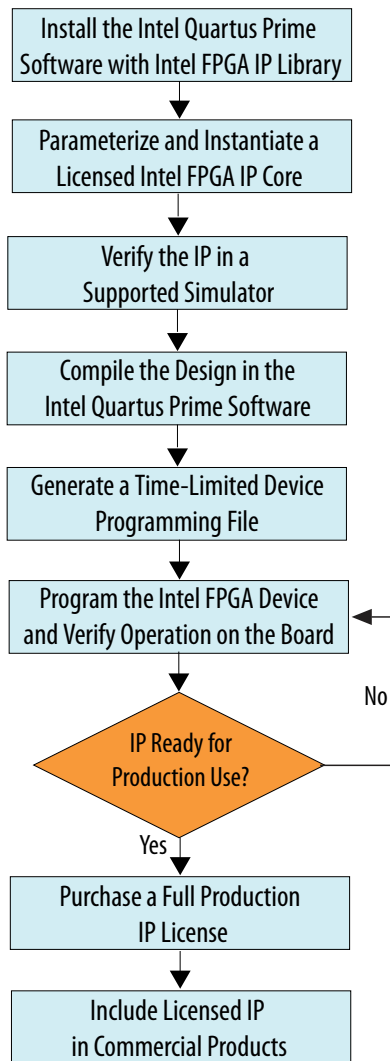
- **Tethered**—Allows running the design containing the licensed Intel FPGA IP indefinitely with a connection between your board and the host computer. Tethered mode requires a serial joint test action group (JTAG) cable connected between the JTAG port on your board and the host computer, which is running the Intel Quartus Prime Programmer for the duration of the hardware evaluation period. The Programmer only requires a minimum installation of the Intel Quartus Prime software, and requires no Intel Quartus Prime license. The host computer controls the evaluation time by sending a periodic signal to the device via the JTAG port. If all licensed IP cores in the design support tethered mode, the evaluation time runs until any IP core evaluation expires. If all of the IP cores support unlimited evaluation time, the device does not time-out.
- **Untethered**—Allows running the design containing the licensed IP for a limited time. The IP core reverts to untethered mode if the device disconnects from the host computer running the Intel Quartus Prime software. The IP core also reverts to untethered mode if any other licensed IP core in the design does not support tethered mode.

When the evaluation time expires for any licensed Intel FPGA IP in the design, the design stops functioning. All IP cores that use the Intel FPGA IP Evaluation Mode time out simultaneously when any IP core in the design times out. When the evaluation time expires, you must reprogram the FPGA device before continuing hardware verification. To extend use of the IP core for production, purchase a full production license for the IP core.

You must purchase the license and generate a full production license key before you can generate an unrestricted device programming file. During Intel FPGA IP Evaluation Mode, the Compiler only generates a time-limited device programming file (`<project name>_time_limited.sof`) that expires at the time limit.



Figure 7. Intel FPGA IP Evaluation Mode Flow



**Note:** Refer to each IP core's user guide for parameterization steps and implementation details.

Intel licenses IP cores on a per-seat, perpetual basis. The license fee includes first-year maintenance and support. You must renew the maintenance contract to receive updates, bug fixes, and technical support beyond the first year. You must purchase a full production license for Intel FPGA IP cores that require a production license, before generating programming files that you may use for an unlimited time. During Intel FPGA IP Evaluation Mode, the Compiler only generates a time-limited device programming file (*<project name>\_time\_limited.sof*) that expires at the time limit. To obtain your production license keys, visit the [Self-Service Licensing Center](#).

The [Intel FPGA Software License Agreements](#) govern the installation and use of licensed IP cores, the Intel Quartus Prime design software, and all unlicensed IP cores.

### Related Information

- [Intel FPGA Licensing Support Center](#)
- [Introduction to Intel FPGA Software Installation and Licensing](#)

## 2.2. Specifying the Intel Stratix 10 IP Core Parameters and Options

The 25G Ethernet Intel FPGA IP parameter editor allows you to quickly configure your custom IP variation. Use the following steps to specify IP core options and parameters in the Intel Quartus Prime Pro Edition software.

1. In the Intel Quartus Prime Pro Edition, click **File > New Project Wizard** to create a new Quartus Prime project, or **File > Open Project** to open an existing Quartus Prime project. The wizard prompts you to specify a device.
2. In the IP Catalog (**Tools > IP Catalog**), locate and double-click the name of the IP core to customize. The New IP Variation window appears.
3. In the **New IP Variation** dialog box, specify a top-level name for your custom IP variation. The parameter editor saves the IP variation settings in a file named `<your_ip>.ip`.
4. Click **Create**. The parameter editor appears.
5. On the **IP** tab, specify the parameters for your IP core variation. Refer to [25G Ethernet Intel FPGA IP Parameters](#) on page 28 for information about specific IP core parameters.
6. Optionally, to generate a simulation testbench or compilation and hardware design example, follow the instructions in the *25G Ethernet Intel Stratix 10 FPGA IP Design Example User Guide*.
7. Click **Generate HDL**. The **Generation** dialog box appears.
8. Specify output file generation options, and then click **Generate**. The IP variation files generate according to your specifications.
9. Click **Finish**. The parameter editor adds the top-level `.ip` file to the current project automatically. If you are prompted to manually add the `.ip` file to the project, click **Project > Add/Remove Files in Project** to add the file.
10. After generating and instantiating your IP variation, make appropriate pin assignments to connect ports.

### Related Information

#### [25G Ethernet Intel Stratix 10 FPGA IP Design Example User Guide](#)

Information about the **Example Design** tab in the 25G Ethernet Intel FPGA IP parameter editor for Intel Stratix 10 devices.

## 2.3. Simulating the IP Core

You can simulate your 25G Ethernet Intel FPGA IP core variation with the functional simulation model and the testbench generated with the IP core. The functional simulation model is a cycle-accurate model that allows for fast functional simulation of your IP core instance using industry-standard Verilog HDL simulators. You can simulate the Intel-provided testbench or create your own testbench to exercise the IP core functional simulation model.

The functional simulation model and testbench files are generated in project subdirectories. These directories also include scripts to compile and run the design example.

**Note:** Use the simulation models only for simulation and not for synthesis or any other purposes. Using these models for synthesis creates a nonfunctional design.

In the top-level wrapper file for your simulation project, you can set the following RTL parameters to enable simulation optimization. These optimizations significantly decrease the time to reach link initialization.

- **SIM\_SHORT\_RST:** Shortens the reset times to speed up simulation.
- **SIM\_SHORT\_AM:** Shortens the interval between alignment markers to accelerate alignment marker lock. Alignment markers are used when Reed-Solomon FEC is enabled.
  - **SIM\_SHORT\_AM = 1'b1:** The TX RS-FEC inserts alignment marker at every 1280 64b/66b blocks or 320 257-bit transcoded blocks. The RX RS-FEC expects alignment marker at every 1280 64b/66b blocks or 320 257-bit transcoded blocks.
  - **SIM\_SHORT\_AM = 1'b0:** The TX RS-FEC inserts alignment marker at every 81920 64b/66b blocks or 20480 257-bit transcoded blocks. The RX RS-FEC expects alignment marker at every 81920 64b/66b blocks or 20480 257-bit transcoded blocks.
- **SIM\_SIMPLE\_RATE:** Sets the PLL reference clock (`clk_ref`) to 625 MHz instead of 644.53125 MHz to optimize PLL simulation model behavior.

In general, parameters are set through the IP core parameter editor and you should not change them manually. The only exceptions are these simulation optimization parameters.

To set these parameters on the PHY blocks, add the following lines to the top-level wrapper file:

```
defparam <dut instance>.SIM_SHORT_RST = 1'b1;  
defparam <dut instance>.SIM_SHORT_AM = 1'b1;  
defparam <dut instance>.SIM_SIMPLE_RATE = 1'b1;
```

**Note:** You can use the example testbench as a guide for setting the simulation parameters in your own simulation environment. These lines are already present in the Intel-provided testbench for the IP core.

### Related Information

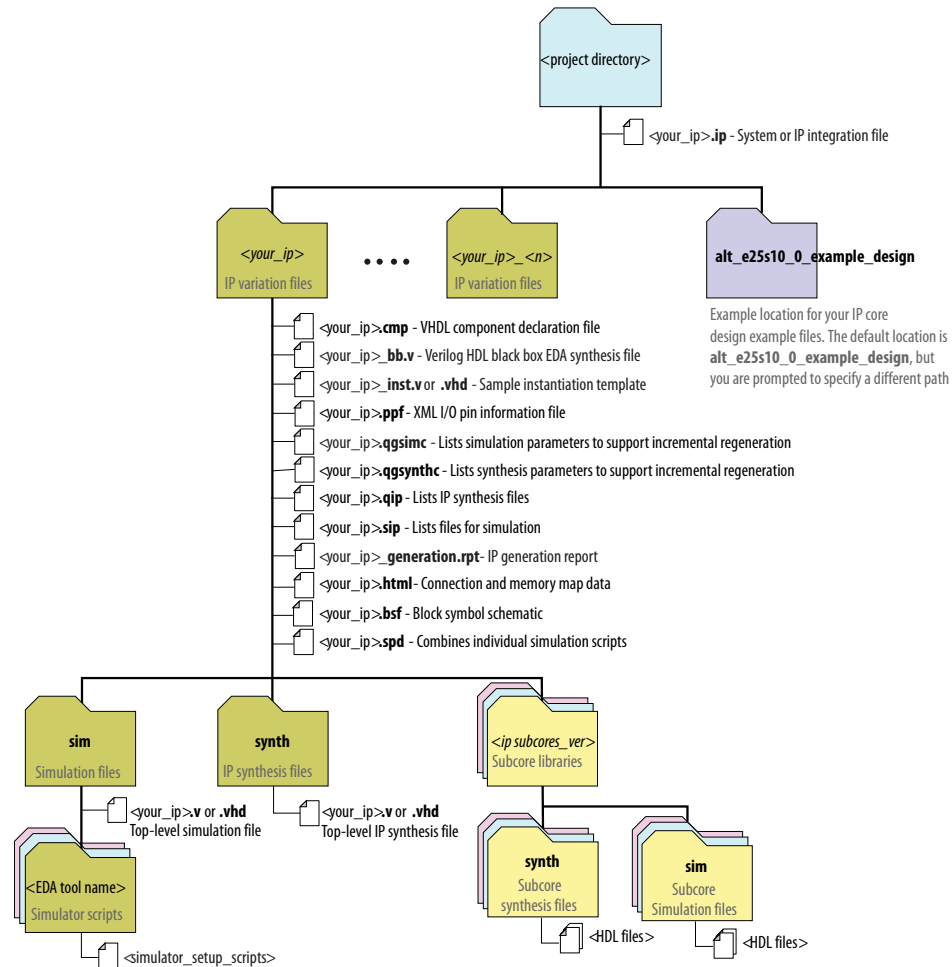
- [Simulating Intel FPGA Designs](#)  
*Intel Quartus Prime Pro Edition User Guide: Third-party Simulation* chapter that provides information about simulating Intel FPGA IP cores.
- [25G Ethernet Intel Stratix 10 FPGA IP Design Example User Guide](#)  
Information about generating and simulating the Intel-provided 25G Ethernet Intel FPGA IP testbench. This testbench demonstrates a basic test of the IP core. It is not intended to be a substitute for a full verification environment.

## 2.4. Generated File Structure

The Intel Quartus Prime Pro Edition software generates the following IP core output file structure.

For information about the file structure of the design example, refer to the *25G Ethernet Intel Stratix 10 FPGA IP Design Example User Guide*.

**Figure 8. IP Core Generated Files**



**Table 11. IP Core Generated Files**

| File Name         | Description  |
|-------------------|--|
| <your_ip>.ip      | The Platform Designer system or top-level IP variation file. <your_ip> is the name that you give your IP variation.  |
| <system>.sopcinfo | Describes the connections and IP component parameterizations in your Platform Designer system. You can parse its contents to get requirements when you develop software drivers for IP components. |

*continued...*

| File Name  | Description  |
|--|--|
|  | Downstream tools such as the Nios® II Gen 2 tool chain use this file. The <code>.sopcinfo</code> file and the <code>system.h</code> file generated for the Nios II Gen 2 tool chain include address map information for each slave relative to each master that accesses the slave. Different masters may have a different address map to access a particular slave component.   |
| <code>&lt;your_ip&gt;.cmp</code>   | The VHDL Component Declaration ( <code>.cmp</code> ) file is a text file that contains local generic and port definitions that you can use in VHDL design files. This IP core does not support VHDL. However, the Intel Quartus Prime software generates this file.  |
| <code>&lt;your_ip&gt;.html</code>  | A report that contains connection information, a memory map showing the address of each slave with respect to each master to which it is connected, and parameter assignments.   |
| <code>&lt;your_ip&gt;.generation.rpt</code>  | IP or Platform Designer generation log file. A summary of the messages during IP generation.   |
| <code>&lt;your_ip&gt;.qgsimc</code>  | Lists simulation parameters to support incremental regeneration.   |
| <code>&lt;your_ip&gt;.qgsynthc</code>  | Lists synthesis parameters to support incremental regeneration.  |
| <code>&lt;your_ip&gt;.qip</code>   | Contains all the required information about the IP component to integrate and compile the IP component in the Intel Quartus Prime Pro Edition software.  |
| <code>&lt;your_ip&gt;.csv</code>   | Contains information about the upgrade status of the IP component.   |
| <code>&lt;your_ip&gt;.bsf</code>   | A Block Symbol File ( <code>.bsf</code> ) representation of the IP variation for use in Intel Quartus Prime Pro Edition Block Diagram Files ( <code>.bdf</code> ).   |
| <code>&lt;your_ip&gt;.spd</code>   | Required input file for <code>ip-make-simscript</code> to generate simulation scripts for supported simulators. The <code>.spd</code> file contains a list of files generated for simulation, along with information about memories that you can initialize.   |
| <code>&lt;your_ip&gt;.ppf</code>   | The Pin Planner File ( <code>.ppf</code> ) stores the port and node assignments for IP components created for use with the Pin Planner.  |
| <code>&lt;your_ip&gt;_bb.v</code>  | You can use the Verilog black-box ( <code>_bb.v</code> ) file as an empty module declaration for use as a black box.   |
| <code>&lt;your_ip&gt;_inst.v</code> and <code>_inst.vhd</code>                     | HDL example instantiation template. You can copy and paste the contents of this file into your HDL file to instantiate the IP variation. This IP core does not support VHDL. However, the Intel Quartus Prime Pro Edition software generates the <code>_inst.vhd</code> file.  |
| <code>&lt;your_ip&gt;.regmap</code>  | If IP contains register information, <code>.regmap</code> file generates. The <code>.regmap</code> file describes the register map information of master and slave interfaces. This file complements the <code>.sopcinfo</code> file by providing more detailed register information about the system. This enables register display views and user customizable statistics in the System Console.   |
| <code>&lt;your_ip&gt;.svd</code>   | Allows hard processor system (HPS) System Debug tools to view the register maps of peripherals connected to HPS within a Platform Designer system. During synthesis, the <code>.svd</code> files for slave interfaces visible to System Console masters are stored in the <code>.sof</code> file in the debug section. System Console reads this section, which Platform Designer can query for register map information. For system slaves, Platform Designer can access the registers by name. |
| <code>synth/&lt;your_ip&gt;.v</code> or <code>&lt;synth/&lt;your_ip&gt;.vhd</code> | Top-level IP synthesis HDL files that instantiate each submodule or child IP core for synthesis. This IP core does not support VHDL. However, the Intel Quartus Prime software generates this file.  |
| <code>sim/&lt;your_ip&gt;.v</code> or <code>.vhd</code>                            | Top-level simulation files that instantiate each submodule or child IP core for simulation. This IP core does not support VHDL. However, the Intel Quartus Prime Pro Edition software generates this file.   |
| <b>continued...</b>  |  |

| File Name                                | Description  |
|--|--|
| sim/mentor/                              | Contains a ModelSim script <code>msim_setup.tcl</code> to set up and run a simulation.   |
| sim/aldec/                               | Contains a Riviera-PRO script <code>rivierapro_setup.tcl</code> to setup and run a simulation.   |
| sim/synopsys/vcs/<br>sim/synopsys/vcsmx/ | Contains a shell script <code>vcs_setup.sh</code> to set up and run a VCS® simulation.<br>Contains a shell script <code>vcsmx_setup.sh</code> and <code>synopsys_sim.setup</code> file to set up and run a VCS MX® simulation. |
| sim/cadence/                             | Contains a shell script <code>ncsim_setup.sh</code> and other setup files to set up and run an NCSIM simulation.   |
| sim/xcelium/                             | Contains a shell script <code>xcelium_setup.sh</code> and other setup files to set up and run an xcelium simulation.   |
| <child IP cores>/                        | For each generated child IP core directory, Platform Designer generates <code>synth/</code> and <code>sim/</code> sub-directories.   |

### Related Information

#### [25G Ethernet Intel Stratix 10 FPGA IP Design Example User Guide](#)

Information about the 25G Ethernet Intel FPGA IP core design example file structure.

## 2.5. Integrating Your IP Core in Your Design

### 2.5.1. Pin Assignments

When you integrate your 25G Ethernet Intel FPGA IP core instance in your design, you must make appropriate pin assignments. While compiling the IP core alone, you can create virtual pins to avoid making specific pin assignments for top-level signals. When you are ready to map the design to hardware, you can change to the correct pin assignments.

### Related Information

#### [Intel Quartus Prime Help](#)

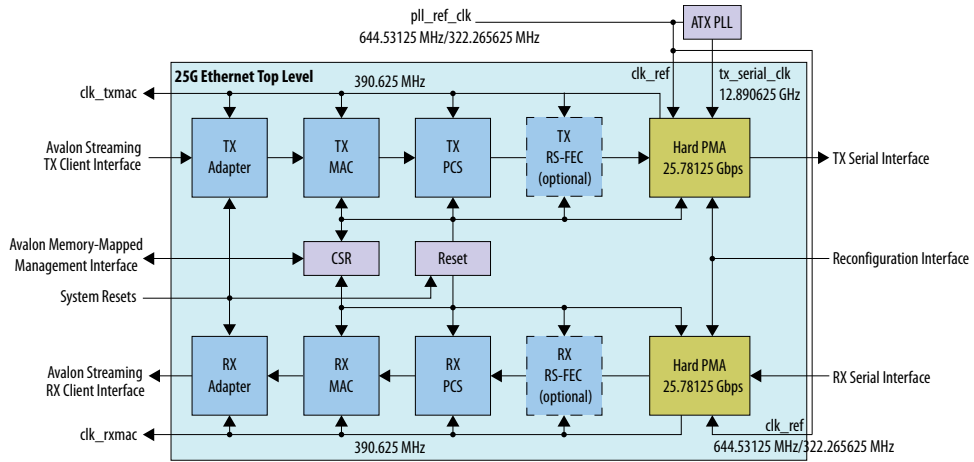
For information about the Intel Quartus Prime software, including virtual pins.

### 2.5.2. Adding the Transceiver PLL

The transceiver channels in the Intel Stratix 10 devices require an external PLL to drive the TX transceiver serial clock, in order to compile and to function correctly in hardware. In many cases, the same PLL can be shared with an additional transceiver in your design.

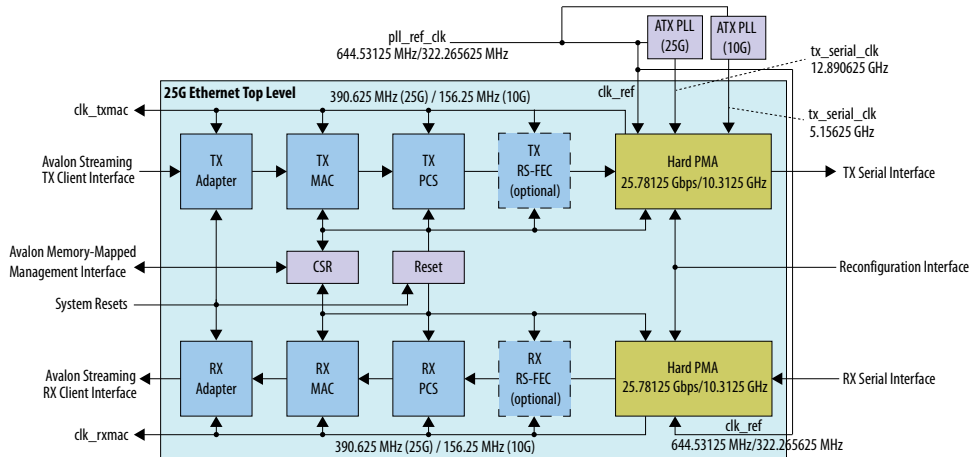
**Figure 9. PLL Configuration Example for 25G Configuration**

The TX transceiver PLL is instantiated with an ATX PLL IP core. The TX transceiver PLL must always be instantiated outside the 25G Ethernet Intel FPGA IP core.



**Figure 10. PLL Configuration Example for 10G/25G Configuration**

The TX transceiver PLL is instantiated with an ATX PLL IP core. The TX transceiver PLL must always be instantiated outside the 25G Ethernet Intel FPGA IP core.



You can use the IP Catalog to create a transceiver PLL.

- Select **L-Tile/H-Tile Transceiver ATX PLL Intel Stratix 10 FPGA IP**.
- In the parameter editor, set the following parameter values:
  - For 25G configuration:
    - **PLL output frequency** to **12890.625** MHz. The transceiver performs dual edge clocking, using both the rising and falling edges of the input clock from the PLL. Therefore, this PLL output frequency setting supports a 25.78125 Gbps data rate through the transceiver.
    - **Primary PLL clock output buffer** to **GXT clock output buffer**.
    - Turn on **Enable GXT local clock output port (tx\_serial\_clk\_gxt)**.
  - For 10G configuration:
    - **PLL output frequency** to **5156.25** MHz. The transceiver performs dual edge clocking, using both the rising and falling edges of the input clock from the PLL. Therefore, this PLL output frequency setting supports a 10.3125 Gbps data rate through the transceiver.
    - **Primary PLL clock output buffer** to **GX clock output buffer**.
    - Turn on **Enable GX local clock output port (tx\_serial\_clk)**.
  - **PLL auto mode reference clock frequency (integer)** to **644.53125** or **322.265625** MHz.

You must connect the ATX PLL to the 25G Ethernet Intel FPGA IP core as follows:

- Connect the clock output port of the ATX PLL to the `tx_serial_clk` input port of the 25G Ethernet Intel FPGA IP core.
- If the **Enable 10G/25G dynamic rate switching** option is turned on:
  - Connect the clock output port of the ATX PLL with 25G configuration to `tx_serial_clk0` input port of the 25G Ethernet Intel FPGA IP core.
  - Connect the clock output port of the ATX PLL with 10G configuration to `tx_serial_clk1` input port of the 25G Ethernet Intel FPGA IP core.
- Connect the `pll_locked` output port of the ATX PLL to the `tx_pll_locked` input port of the 25G Ethernet Intel FPGA IP core.
- Drive the ATX PLL reference clock port and the 25G Ethernet Intel FPGA IP core `clk_ref` input port with the same clock. The clock frequency must be the frequency you specify for the ATX PLL IP core **PLL auto mode reference clock frequency (integer)** parameter.

### Related Information

- [Transceivers](#) on page 61
- [L- and H-Tile Transceiver PHY User Guide](#)  
Information about the correspondence between PLLs and transceiver channels, and information about how to configure an external transceiver PLL for your own design. You specify the clock network to which the PLL output connects by setting the clock network in the PLL parameter editor.



### 2.5.3. Adding the External Time-of-Day Module for Variations with 1588 PTP Feature

25G Ethernet Intel FPGA IP cores that include the 1588 PTP module require an external time-of-day (TOD) module to provide a continuous flow of current time-of-day information. The TOD module must update the time-of-day output value on every clock cycle, and must provide the TOD value in the V2 format (96 bits) or the 64-bit TOD format, or both.

Intel provides the following components that you can combine to create the TOD module the 25G Ethernet Intel FPGA IP core requires:

- A simple TOD clock module, available from the IP Catalog (**Interface Protocols > Ethernet > Reference Design Components > Ethernet IEEE 1588 Time of Day Clock Intel FPGA IP**). You can instantiate two of these clock modules and connect one to the TX MAC and the other to the RX MAC.
- A single-format TOD synchronizer, available from the IP Catalog (**Interface Protocols > Ethernet > Reference Design Components > Ethernet IEEE 1588 TOD Synchronizer Intel FPGA IP**). This component can handle only a single TOD format. Therefore, if you set the **Time of day format** parameter to the value of **Enable both formats**, you must instantiate and connect two TOD synchronizer modules. If your IP core supports only a single TOD format, your design requires only a single TOD synchronizer module.

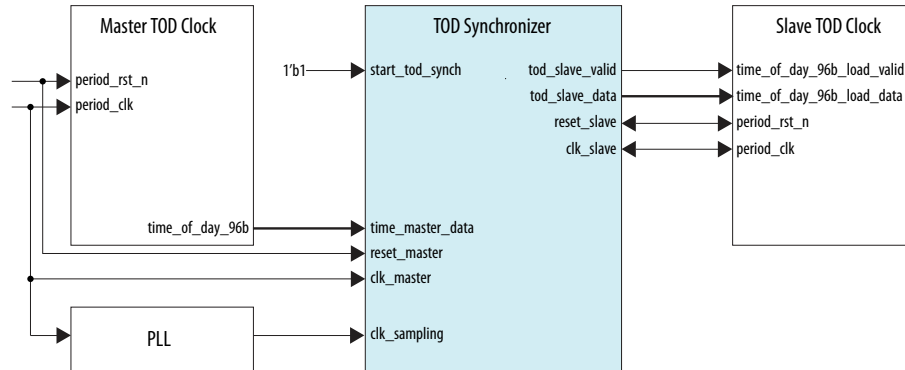
Each TOD synchronizer connects a master TOD clock and a slave TOD clock.

- If you create your TOD module with a single TOD synchronizer, the master TOD clock connects to the TX MAC of the 25G Ethernet Intel FPGA IP core and the slave TOD clock connects to the RX MAC of the 25G Ethernet Intel FPGA IP core.
- Alternatively, you can drive both the TX and RX TOD clocks from a single master TOD clock. In that case, your design must include two TOD synchronizers, one to connect the master TOD clock and the slave TX TOD clock and one to connect the master TOD clock and the slave RX TOD clock.

If your IP core supports both TOD formats, double the number of TOD synchronizers in your TOD module. The configuration you implement depends on your system design requirements for 1588 PTP functionality.

**Figure 11. TOD Synchronizer and TOD Clocks in 96-Bit TOD Format Design**

Shows the required connections between two TOD clock components and a TOD synchronizer component in a single TOD format design. In a simple TOD module, the master TOD clock connects to the TX MAC of the IP core, and the slave TOD clock connects to the RX MAC of the IP core. If your 25G Ethernet Intel FPGA IP core supports both TOD formats, a second TOD synchronizer connects to the corresponding 64-bit time-of-day signals of the same master and slave TOD clocks.



For information about the Ethernet IEEE 1588 Time of Day Clock and Ethernet IEEE 1588 TOD Synchronizer components, and the requirements for the PLL that connects to the TOD synchronizer, refer to the *Ethernet Design Example Components User Guide*.

**Table 12. TOD Module Required Connections to 25G Ethernet Intel FPGA IP Core**

Lists the required connections between the TOD module and the 25G Ethernet Intel FPGA IP core, using signal names for TOD modules that provide both a 96-bit TOD and a 64-bit TOD. If you create your own TOD module it must have the output signals required by the 25G Ethernet Intel FPGA IP core. However, its signal names could be different than the TOD module signal names in the table. The signals that the IP core includes depend on the value you set for **Time of day format** in the parameter editor. For example, an RX TOD module might require only a 96-bit TOD out signal. This table does not list required connections between the TOD module and additional parts of your design.

| TOD Module Signal  | 25GbE IP Core Signal   |
|--|--|
| rst_n (input to TX and RX TOD clocks)  | Drive this signal from the same source as the <code>csr_rst_n</code> input signal to the 25G Ethernet Intel FPGA IP core.  |
| period_rst_n (input to RX TOD clock)<br>reset_slave (input to Synchronizer)  | Drive these signals from the same source as the <code>rx_rst_n</code> input signal to the 25G Ethernet Intel FPGA IP core. |
| period_rst_n (input to TX TOD clock)<br>reset_master (input to Synchronizer) | Drive these signals from the same source as the <code>tx_rst_n</code> input signal to the 25G Ethernet Intel FPGA IP core. |
| time_of_day_96b[95:0] (output from TX TOD clock)                             | <code>tx_time_of_day_96b_data[95:0]</code> (input)   |
| time_of_day_64b[63:0] (output from TX TOD clock)                             | <code>tx_time_of_day_64b_data[63:0]</code> (input)   |
| time_of_day_96b[95:0] (output from RX TOD clock)                             | <code>rx_time_of_day_96b_data[95:0]</code> (input)   |
| time_of_day_64b[63:0] (output from RX TOD clock)                             | <code>rx_time_of_day_64b_data[63:0]</code> (input)   |
| period_clk (input to TX TOD clock)<br>clk_master (input to Synchronizer)     | <code>clk_txmac</code> (output)  |
| period_clk (input to RX TOD clock)<br>clk_slave (input to Synchronizer)      | <code>clk_rxmac</code> (output)  |

**Related Information**

- [External Time-of-Day Module for 1588 PTP Variations](#) on page 51

- [Ethernet Design Example Components User Guide](#)  
Describes the Ethernet IEEE 1588 Time of Day Clock component and the Ethernet IEEE 1588 TOD Synchronizer component available in the Intel Quartus Prime software from the IP Catalog.

#### 2.5.4. Placement Settings for the 25G Ethernet Intel FPGA IP Core

The Quartus Prime software provides the options to specify design partitions and Logic Lock (Standard) or Logic Lock regions for incremental compilation, to control placement on the device. To achieve timing closure for your design, you might need to provide floorplan guidelines using one or both of these features.

The appropriate floorplan is always design-specific, and depends on your design.

##### Related Information

[Intel Quartus Prime Pro Edition User Guide: Design Constraints](#)

Describes incremental compilation, design partitions, and Logic Lock regions.

### 2.6. Compiling the Full Design and Programming the FPGA

You can use the **Start Compilation** command on the Processing menu in the Intel Quartus Prime software to compile your design. After successfully compiling your design, program the targeted Intel FPGA with the Programmer and verify the design in hardware.

*Note:* The 25G Ethernet Intel FPGA IP core design example synthesis directories include Synopsys Constraint (.sdc) files that you can copy and modify for your own design.

##### Related Information

- [Incremental Compilation for Hierarchical and Team-Based Design](#)
- [Programming Intel Devices](#)
- [25G Ethernet Intel Stratix 10 FPGA IP Design Example User Guide](#)  
Information about generating the design example and the design example directory structure.

### 3. 25G Ethernet Intel FPGA IP Parameters

The 25G Ethernet Intel FPGA IP parameter editor provides the parameters you can set to configure the 25G Ethernet Intel FPGA IP core and design example.

The 25G Ethernet Intel FPGA IP parameter editor includes an **Example Design** tab. For information about that tab, refer to the *25G Ethernet Intel Stratix 10 FPGA IP Design Example User Guide*.

**Table 13. IP Core Parameters**

| Parameter                   | Range                                 | Default Setting         | Description   |
|-----------------------------|---------------------------------------|-------------------------|---|
| <b>General Options</b>      |                                       |                         |   |
| <b>Device Family</b>        | <b>Stratix 10</b>                     | <b>Stratix 10</b>       | Selects the device family.  |
| <b>Ready Latency</b>        | <b>0, 3</b>                           | <b>0</b>                | Selects the readyLatency value on the TX client interface. readyLatency is an Avalon streaming interface property that defines the number of clock cycles of delay from when the IP core asserts the <code>l1_tx_ready</code> signal to the clock cycle in which the IP core can accept data on the TX client interface. Refer to the <i>Avalon Interface Specifications</i> .<br>Selecting a latency of 3 eases timing closure at the expense of increased latency for the datapath.<br>If you set the readyLatency to 3 and turn on standard flow control, data might be delayed in the IP core while the IP core is backpressured. |
| <b>Core Variant</b>         | <b>MAC+PCS<br/>+PMA, MAC<br/>+PCS</b> | <b>MAC+PCS<br/>+PMA</b> | Selects the primary blocks to include in the IP core variation. <ul style="list-style-type: none"> <li><b>MAC+PCS+PMA</b>—When enabled, the IP core generates with capability of MAC, PCS, and PMA protocol layers.</li> <li><b>MAC+PCS</b>—When enabled, the IP core generates with the capability of MAC and PCS only.</li> </ul>   |
| <b>PCS/PMA Options</b>      |                                       |                         |   |
| <b>Enable RS-FEC</b>        | <b>Enabled,<br/>Disabled</b>          | <b>Disabled</b>         | When enabled, the IP core implements Reed-Solomon forward error correction (FEC).   |
| <b>Flow Control Options</b> |                                       |                         |   |
| <b>Enable flow control</b>  | <b>Enabled,<br/>Disabled</b>          | <b>Disabled</b>         | When enabled, the IP core implements flow control. When either link partner experiences congestion, the respective transmit control sends pause frames. Register settings in <a href="#">Table 28</a> on page 80 and <a href="#">Table 29</a> on page 83 control flow control behavior, including whether the IP core implements standard flow control or priority-based flow control.<br>If you turn on standard flow control and set the readyLatency to 3, data might be delayed in the IP core while the IP core is backpressured.  |
| <i>continued...</i>         |                                       |                         |   |

| Parameter   | Range   | Default Setting     | Description  |
|---|---|---------------------|--|
| Number of queues                                  | 1-8   | 8                   | Specifies the number of queues used in managing flow control.  |
| <b>MAC Options</b>                                |   |                     |  |
| Enable link fault generation                      | Enabled, Disabled   | Disabled            | When enabled, the IP core implements link fault signaling as defined in the <i>IEEE 802.3-2012 IEEE Standard for Ethernet</i> . The MAC includes a Reconciliation Sublayer (RS) to manage local and remote faults. When enabled, the local RS TX logic can transmit remote fault sequences in case of a local fault and can transmit IDLE control words in case of a remote fault.                                   |
| Enable preamble passthrough                       | Enabled, Disabled   | Disabled            | When enabled, the IP core is in RX and TX preamble pass-through mode. In RX preamble pass-through mode, the IP core passes the preamble and Start Frame Delimiter (SFD) to the client instead of stripping them out of the Ethernet packet. In TX preamble pass-through mode, the client specifies the preamble and provides the SFD to be sent in the Ethernet frame.   |
| Enable TX CRC passthrough                         | Enabled, Disabled   | Disabled            | When enabled, TX MAC does not insert the CRC-32 checksum in the out-going frame. In pass-through mode, the client must provide frames with at least 64 bytes, including the Frame Check Sequence (FCS). When disabled, the TX MAC computes and inserts a 32-bit FCS in the TX MAC frame.<br><br>This parameter is not available if you turn on <b>Enable IEEE 1588</b> .   |
| Enable MAC statistics counters                    | Enabled, Disabled   | Enabled             | When enabled, the IP core includes statistics counters that characterize TX and RX traffic.  |
| <b>IEEE 1588 Options</b>                          |   |                     |  |
| Enable IEEE 1588                                  | Enabled, Disabled   | Disabled            | If enabled, the IP core supports the IEEE Standard 1588-2008 Precision Clock Synchronization Protocol, by providing the hooks to implement the Precise Timing Protocol (PTP).<br><br>This parameter is not available if you turn on <b>Enable TX CRC passthrough</b> .   |
| Time of day format                                | Enable 96-bit timestamp format, Enable 64-bit timestamp format, Enable both formats | Enable both formats | Specifies the interface to the Time of Day module. If you select <b>Enable both formats</b> , the IP core includes both the 64-bit interface and the 96-bit interface.<br><br>This parameter is available only in variations with <b>Enable IEEE 1588</b> turned on. The IP core provides the Time of Day interface; the IP core does not include Time of Day and synchronizer modules to connect to this interface. |
| Fingerprint width                                 | 1-32  | 4                   | Specifies the number of bits in the fingerprint that the IP core handles.<br><br>This parameter is available only in variations with <b>Enable IEEE 1588</b> turned on.  |
| <b>10G/25G Rate Switching</b>                     |   |                     |  |
| Enable 10G/25G dynamic rate switching             | Enabled, Disabled   | Disabled            | If enabled, the IP core supports dynamic reconfiguration between the 10 Gbps and the 25 Gbps data rates.   |
| <b>Configuration, Debug and Extension Options</b> |   |                     |  |
| Enable Native PHY Debug Master Endpoint (NPDME)   | Enabled, Disabled   | Disabled            | If enabled, the Transceiver Native PHY IP includes an embedded Native PHY Debug Master Endpoint (NPDME) that connects internally to the Avalon memory-mapped   |

continued...

| Parameter   | Range                         | Default Setting | Description  |
|---|-------------------------------|-----------------|--|
|   |                               |                 | slave interface. The NPDME can access the reconfiguration space of the transceiver. It can perform certain test and debug functions via JTAG using the System Console.   |
| <b>Reference clock frequency</b>                                  | <b>644.531250, 322.265625</b> | 644.531250      | Specifies the frequency of the transceiver CDR reference clock input in MHz.   |
| <b>Enable auto adaptation triggering for RX PMA CTLE/DFE mode</b> | <b>Enabled, Disabled</b>      | <b>Enabled</b>  | If enabled, additional logic is instantiated to automatically request adaptation once RX data is unlocked.<br>If disabled, refer to <i>Adaptation Control - Start</i> section of the <i>L- and H-Tile Transceiver PHY User Guide</i> for more information about how to start adaptation. |

### Related Information

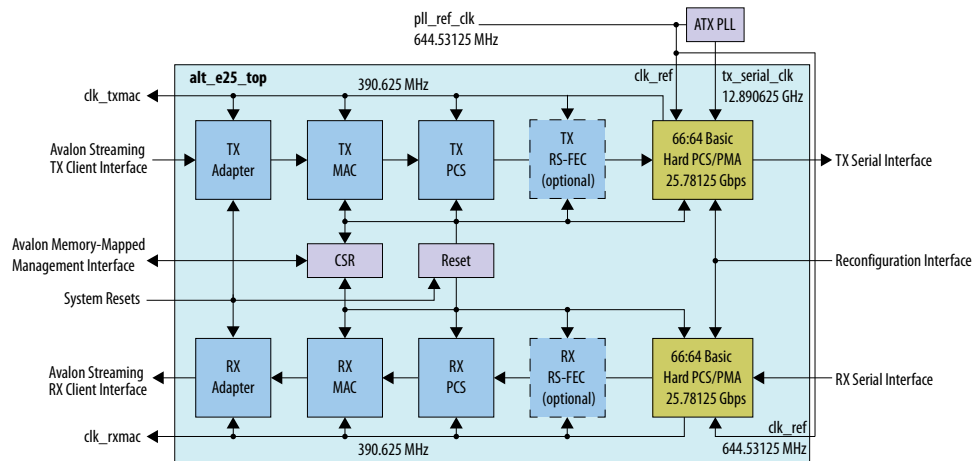
- [25G Ethernet Intel Stratix 10 FPGA IP Design Example User Guide](#)  
Information about the Example Design tab in the 25GbE parameter editor.
- [Avalon Interface Specifications](#)  
Detailed information about Avalon streaming interfaces and the Avalon streaming readLatency parameter.
- [L- and H-Tile Transceiver PHY User Guide](#)  
Information about Intel Stratix 10 Native PHY IP features, including NPDME.

## 4. Functional Description

### 4.1. 25G Ethernet Intel FPGA IP Core Functional Description

The 25G Ethernet Intel FPGA IP core implements an Ethernet MAC in accordance with the *25G & 50G Ethernet Specification*. The IP core implements an Ethernet PCS and PMA (PHY) that handles the frame encapsulation and flow of data between a client logic and Ethernet network.

**Figure 12. 25G Ethernet Intel FPGA IP Core with MAC, PCS, and PMA Clock Diagram**



Note:  
1. 66:64 encode/decode function is implemented as part of the soft PCS.

In the TX direction, the MAC assembles packets and sends them to the PHY. It completes the following tasks:

- Accepts client frames.
- Inserts the inter-packet gap (IPG), preamble, start of frame delimiter (SFD), and padding. The source of the preamble and SFD depends on whether the IP core is in preamble-pass-through mode.
- Adds the CRC bits if enabled.
- Updates statistics counters if enabled.

The PCS encodes MAC frames. The PHY, if selected, will perform reliable transmission over the media to the remote end.

In the RX direction, the PMA, if selected, passes frames to the PCS that sends them to the MAC. The MAC completes the following tasks:

- Performs CRC and malformed packet checks.
- Updates statistics counters if enabled.
- Strips out the CRC, preamble, and SFD.
- Passes the remainder of the frame to the client.

In preamble pass-through mode, the MAC passes on the preamble and SFD to the client instead of stripping them out. In RX CRC pass-through mode, the MAC passes on the CRC bytes to the client and asserts the end-of-packet signal in the same clock cycle as the final CRC byte.

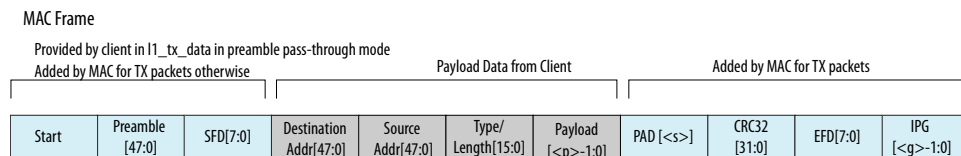
### 4.1.1.1. 25G Ethernet Intel FPGA IP Core TX MAC Datapath

The TX MAC module receives the client payload data with the destination and source addresses. It then adds, appends, or updates various header fields in accordance with the configuration specified. The MAC does not modify the destination address, the source address, or the payload received from the client. However, the TX MAC module adds a preamble, if the IP core is not configured to receive the preamble from user logic. It pads the payload of frames greater than eight bytes to satisfy the minimum Ethernet frame payload of 46 bytes. By default, the MAC inserts the CRC bytes. The TX MAC module inserts IDLE bytes to maintain an average IPG of 12.

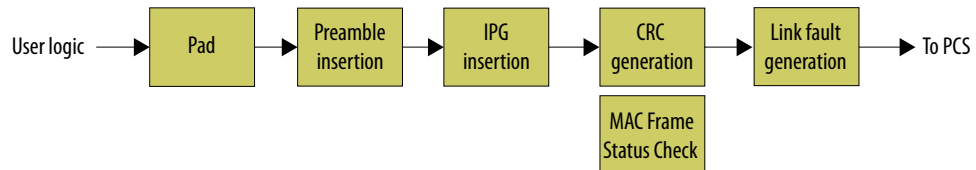
**Figure 13. Typical Client Frame at the Transmit Interface**

Illustrates the changes that the TX MAC makes to the client frame. This figure uses the following notational conventions:

- $\langle p \rangle$  = payload size, which is arbitrarily large
- $\langle s \rangle$  = number of padding bytes (0–46)
- $\langle g \rangle$  = number of IPG bytes



**Figure 14. TX MAC Functions**



#### 4.1.1.1.1. Frame Padding

When the length of the client frame is less than 64 bytes, the TX MAC module inserts pad bytes (0x00) after the payload to create a frame length equal to the minimum size of 64 bytes (including CRC).



The IP core filters out all client frames with lengths less than 9 bytes. The IP core drops these frames silently.

#### 4.1.1.2. Preamble Insertion

In the TX datapath the MAC prepends an eight-byte preamble to the client frame. If you turn on **Enable link fault generation**, this MAC module also incorporates the functions of the reconciliation sublayer (RS).

The source of the 7-byte preamble (including a Start byte) and 1-byte SFD depends on whether you turn on **Enable preamble passthrough** in the parameter editor.

If the preamble pass-through feature is enabled, the client provides the eight-byte preamble (including the 0xFB Start byte and final 1-byte SFD) on `l1_tx_data`. The client is responsible for providing the correct Start byte (0xFB) and an appropriate SFD byte. If the preamble pass-through feature is disabled, the MAC inserts the standard Ethernet preamble in the transmitted Ethernet frame.

Note that a single parameter in the 25G Ethernet Intel FPGA IP parameter editor turns on both RX and TX preamble passthrough.

#### 4.1.1.3. Inter-Packet Gap Generation and Insertion

The TX MAC maintains the minimum inter-packet gap (IPG) between transmitted frames required by the IEEE 802.3 Ethernet standard. The deficit idle counter (DIC) maintains the average IPG of 12 bytes.

#### 4.1.1.4. Frame Check Sequence (CRC32) Insertion

The component GUI includes the **Enable TX CRC passthrough** parameter to control CRC generation. When enabled, TX MAC does not insert the CRC32 checksum in the out-going frame. In pass-through mode, the client must provide frames with at least 64 bytes, so that the IP core does not pad them. When disabled, the TX MAC computes and inserts a 32-bit Frame Check Sequence (FCS) in the TX MAC frame. The MAC computes the CRC32 over the frame bytes that include the source address, destination address, length, data, and pad (if applicable). The CRC checksum computation excludes the preamble, SFD, and FCS.

In pass-through mode, the `l1_tx_endofpacket`, `l1_rx_endofpacket`, `l1_tx_empty[2:0]`, and `l1_rx_empty` are asserted in the same clock cycle with the final FCS byte. When pass-through mode is disabled, the `l1_tx_endofpacket`, `l1_rx_endofpacket`, `l1_tx_empty[2:0]`, and `l1_rx_empty` are asserted in the same clock cycle with the byte before the first FCS bytes.

The encoding is defined by the following generating polynomial:

$$FCS(X) = X^{32} + X^{26} + X^{23} + X^{22} + X^{16} + X^{12} + X^{11} + X^{10} + X^8 + X^7 + X^5 + X^4 + X^2 + X + 1$$

CRC bits are transmitted with MSB first.

Note that you control whether the IP core implements TX CRC insertion or passthrough with a parameter in the 25G Ethernet Intel FPGA IP parameter editor. You control RX CRC forwarding dynamically with the `MAC_CRC_CONFIG` register.

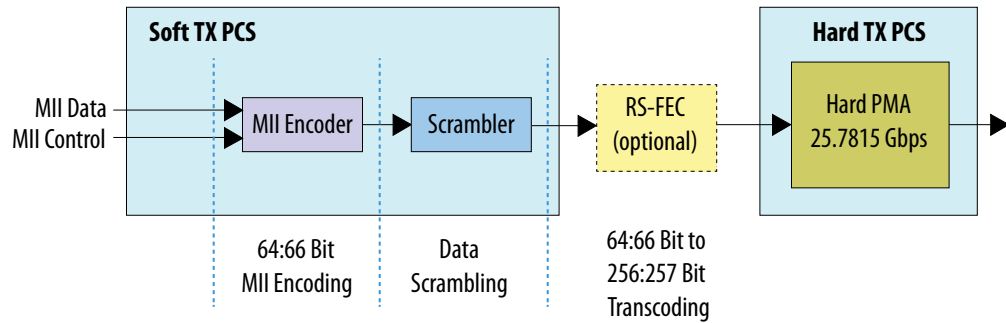
#### Related Information

[Order of Transmission](#) on page 53

### 4.1.2. 25 GbE TX PCS

The soft TX PCS implements MII encoding and scrambling. The 66-bit output stream is input to the hard PCS and PMA block.

Figure 15. High Level Block Diagram of the TX PCS with Optional RS-FEC



The Hard PCS and PMA blocks are configured in 66:64 bit basic generic 10G PCS mode whose status can be read through Control and Status registers. These blocks use FIFOs in elastic-buffer mode. The PMA operates at 25.78125 Gbps.

#### Related Information

[Ethernet section of the L- and H-Tile Transceiver PHY User Guide](#)

Provides more information about the PMA and PCS for Ethernet protocols.

### 4.1.3. TX RS-FEC

If you turn on **Enable RS-FEC** in the 25G Ethernet Intel FPGA IP parameter editor, the IP core includes Reed-Solomon forward error correction (FEC) in both the receive and transmit datapaths.

The IP core implements Reed-Solomon FEC per Clause 108 of the IEEE Standard 802.3by. The Reed-Solomon FEC algorithm includes the following modules:

- 64B/66B to 256B/257B Transcoding
- 257:80 gearbox
- High-Speed Reed-Solomon Encoder
- 80:66 gearbox

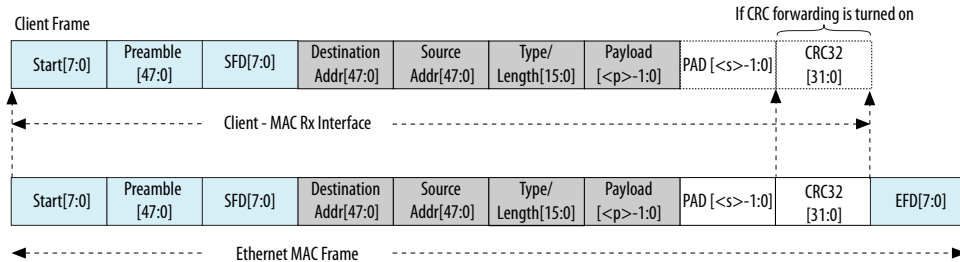
### 4.1.4. 25G Ethernet Intel FPGA IP Core RX MAC Datapath

The RX MAC receives Ethernet frames and forwards the payload with relevant header bytes to the client after performing some MAC functions on header bytes. The RX MAC processes all incoming valid frames.

**Figure 16. Flow of Client Frame With Preamble Pass-Through Turned On**

This figure uses the following notational conventions:

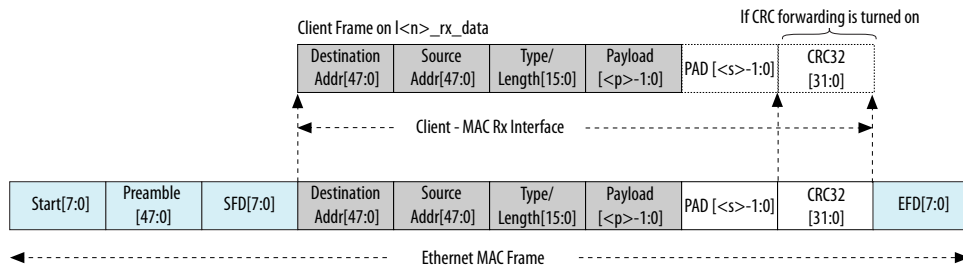
- $\langle p \rangle$  = payload size, which is arbitrarily large.
- $\langle s \rangle$  = number of padding bytes (0–46).



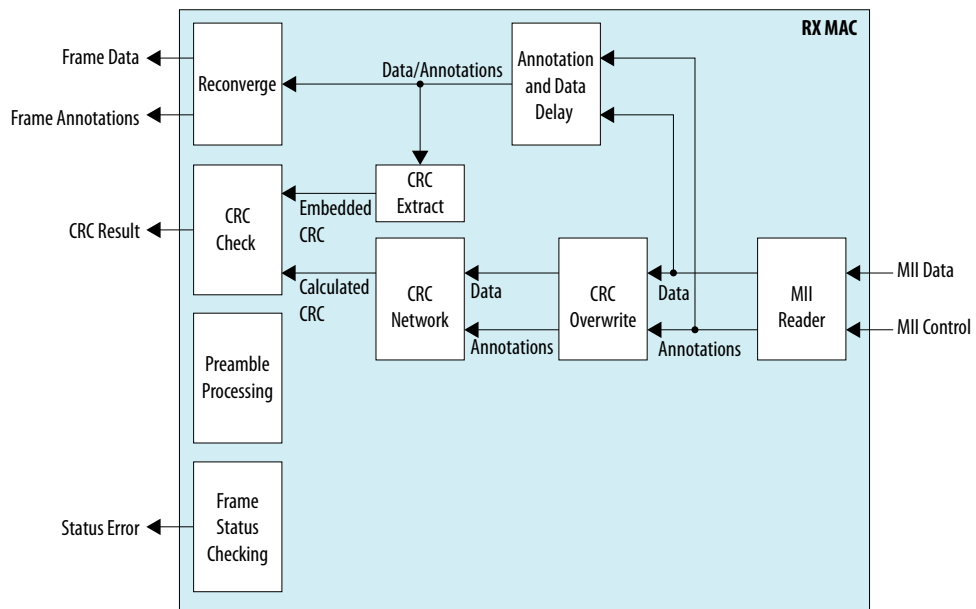
**Figure 17. Flow of Client Frame With Preamble Pass-Through Turned Off**

This figure uses the following notational conventions:

- $\langle p \rangle$  = payload size, which is arbitrarily large.
- $\langle s \rangle$  = number of padding bytes (0–46).



**Figure 18. RX MAC Datapath**



#### 4.1.4.1. IP Core Preamble Processing

If you turn on **Enable preamble passthrough** in the parameter editor, the RX MAC forwards preamble bytes. The TX MAC requires the preamble bytes to be included in the frames at the Avalon Streaming interface.

If you turn off **Enable preamble passthrough**, the IP core removes the preamble bytes. `l1_rx_startofpacket` is aligned to the MSB of the destination address.

Note that a single parameter in the 25G Ethernet Intel FPGA IP parameter editor turns on both RX and TX preamble passthrough.

#### 4.1.4.2. IP Core Malformed Packet Handling

While receiving an incoming packet from the Ethernet link, the 25G Ethernet Intel FPGA IP core expects to detect a terminate character at the end of the packet. When it detects an expected terminate character, the IP core generates an EOP on the client interface. However, sometimes the IP core detects an unexpected control character when it expects a terminate character.

If the 25G Ethernet Intel FPGA IP core detects an Error character, a Start character, an IDLE character, or any other non-terminate control character, when it expects a terminate character, it performs the following actions:

- Generates an EOP.
- Asserts a malformed packet error (`l1_rx_error[0]`).
- Asserts an FCS error (`l1_rx_error[1]`).

If the IP core subsequently detects a terminate character, it does not generate another EOP indication.

When the IP core receives a packet that contains an error deliberately introduced on the Ethernet link using the 25G Ethernet Intel FPGA IP TX error insertion feature, the IP core identifies it as a malformed packet.

At this time, the 25G Ethernet Intel FPGA IP core does not recognize non-zero 4-bit ordered set types as an error.

#### 4.1.4.3. Length/Type Field Processing

This two-byte header represents either the length of the payload or the type of MAC frame.

- Length/type < 0x600—The field represents the payload length of a basic Ethernet frame. The MAC RX continues to check the frame and payload lengths.
- Length/type >= 0x600—The field represents the frame type. The following frame types are possible:
  - Length/type = 0x8100—VLAN or stacked VLAN tagged frames (up to a total of two tags with value 0x8100). The MAC RX continues to check the frame and payload lengths.
  - Length/type = 0x8808—Control frames. The next two bytes are the Opcode field that indicates the type of control frame. For pause frames (Opcode = 0x0001) and PFC frames (Opcode = 0x0101), the MAC RX proceeds with pause frame processing. In addition to processing any pause request, the IP core passes these frames to the RX client interface and updates the appropriate `ll_rxstatus_data` bits.
  - For other field values, the MAC RX forwards the receive frame to the client.

#### 4.1.4.3.1. Length Checking

The MAC function checks the frame and payload lengths of basic, VLAN tagged, and stacked VLAN tagged frames.

The IP core checks that the frame length is valid—is neither undersized nor oversized. A valid frame length is at least 64 (0x40) bytes and does not exceed the following maximum value for the different frame types:

- Basic frames—The number of bytes specified in the `MAX_RX_SIZE_CONFIG` register.
- VLAN tagged frames—The value specified in the `MAX_RX_SIZE_CONFIG` register plus four bytes.
- Stacked VLAN tagged frames—The value specified in the `MAX_RX_SIZE_CONFIG` register plus eight bytes.

If the length/type field in a basic MAC frame or the client length/type field in a VLAN tagged frame has a value less than 0x600, the IP core also checks the payload length. The IP core keeps track of the payload length as it receives a frame, and checks the length against the relevant frame field. The payload length is valid if it satisfies the following conditions:

- The actual payload length matches the value in the length/type or client length/type field.
- Normal frames:
  - Basic frames—the payload length is between 46 (0x2E) and 1536 (0x0600) bytes, excluding 1536.
  - VLAN tagged frames—the payload length is between 42 (0x2A) and 1536 (0x0600), excluding 1536.
  - Stacked VLAN tagged frames—the payload length is between 38 (0x26) and 1536 (0x0600), excluding 1536.
- Jumbo frames:
  - Jumbo basic frames—the payload length is between 46 (0x2E) and the value specified in the MAX\_RX\_SIZE\_CONFIG register minus 18 bytes.
  - Jumbo VLAN tagged frames—the payload length is between 42 (0x2A) and the value specified in the MAX\_RX\_SIZE\_CONFIG register minus 22 bytes.
  - Jumbo stacked VLAN tagged frames—the payload length is between 38 (0x26) and the value specified in the MAX\_RX\_SIZE\_CONFIG register minus 26 bytes.

The RX MAC does not drop frames with invalid length or invalid payload length. If the frame or payload length is not valid, the MAC function asserts output error bits.

- `l1_rx_error[2]`—Undersized frame.
- `l1_rx_error[3]`—Oversized frame.
- `l1_rx_error[4]`—Payload length error.

If the length field value is greater than the actual payload length, the IP core asserts `l1_rx_error[4]`. If the length field value is less than the actual payload length, the MAC RX considers the frame to have excessive padding and does not assert `l1_rx_error[4]`.

**Note:** Starting from Intel Quartus Prime Pro Edition software version 20.3 onwards, the MAC has a counter limit of 0xFFFF. For frames with size larger than this value, the MAC asserts oversized frame error regardless of the value programmed into MAX\_RX\_SIZE\_CONFIG. This applies to the TX datapath as well.

#### 4.1.4.4. RX CRC Checking and Dynamic Forwarding

The RX MAC checks the incoming CRC32 for errors. It asserts `l1_rx_error[1]` in the same cycle as `l1_rx_endofpacket` when it detects an error. CRC checking takes several cycles. The packet frame is delayed to align the CRC output with the end of the frame.

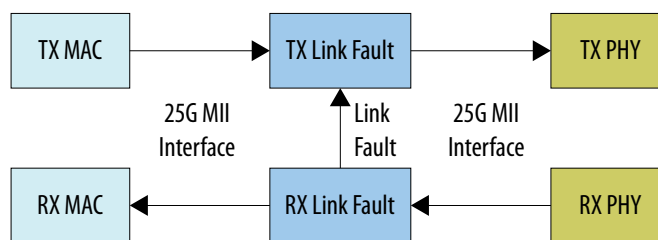
By default, the RX MAC strips off the CRC bytes before forwarding the packet to the MAC client. You can configure the core to retain the RX CRC and forward it to the client by updating the MAC\_CRC\_CONFIG register.

### 4.1.5. Link Fault Signaling Interface

Link fault signaling reflects the health of the link. It operates between the remote Ethernet device Reconciliation Sublayer (RS) and the local Ethernet device RS. The link fault modules communicate status during the interframe period.

You enable link fault signaling by turning on **Enable link fault generation** in the parameter editor. For bidirectional fault signaling, the IP core implements the functionality defined in the *IEEE 802.3ba 10G Ethernet Standard* and *Clause 46* based on the `LINK_FAULT` configuration register settings. For unidirectional fault signaling, the core implements *Clause 66 of the IEEE 802.3-2012 Ethernet Standard*.

Figure 19. Link Fault Block Diagram



#### Local Fault (LF)

If an Ethernet PHY sublayer detects a fault that makes the link unreliable, it notifies the RS of the local fault condition. If unidirectional is not enabled, the core follows *Clause 46*. The RS stops sending MAC data, and continuously generates a remote fault status on the TX datapath. After a local fault is detected, the RX PCS modifies the MII data and control to send local fault sequence ordered sets. Refer to *Link Fault Signaling Based On Configuration and Status* below.

The RX PCS cannot recognize the link fault under the following conditions:

- The RX PCS is not fully aligned.
- The bit error rate (BER) is high.

#### Remote Fault (RF)

If unidirectional is not enabled, the core follows *Clause 46*. If the RS receives a remote fault status, the TX datapath stops sending MAC data and continuously generates idle control characters. If the RS stops receiving fault status messages, it returns to normal operation, sending MAC client data. Refer to *Link Fault Signaling Based On Configuration and Status* below.

#### Link Status Signals

The MAC RX generates two link fault signals: `local_fault_status` and `remote_fault_status`.

**Note:** These signals are real time status signals that reflect the status of the link regardless of the settings in the link fault configuration register.

This register is generated only if you turn on **Enable link fault generation**. The MAC TX interface uses the link fault status signals for additional link fault signaling.

**Table 14. Link Fault Signaling Based On Configuration and Status**

For more information about the LINK\_FAULT register, refer to TX MAC Registers.

| LINK_FAULT Register (0x405) |            |            |            | Real Time Link Status |             | Configured TX Behavior |       | Comment   |
|-----------------------------|------------|------------|------------|-----------------------|-------------|------------------------|-------|---|
| Bit [0]                     | Bit [3]    | Bit [1]    | Bit [2]    | LF Received           | RF Received | TX Data                | TX RF |   |
| 1'b0                        | Don't care | Don't care | Don't care | Don't care            | Don't care  | On                     | Off   | Disable Link fault signaling on TX.<br>RX still reports link status.<br>TX side Link fault signaling disabled on the link.<br>TX data and idle. |
| 1'b1                        | 1'b1       | Don't care | Don't care | Don't care            | Don't care  | Off                    | On    | Force RF.<br>TX: Stop data. Transmit RF only  |
| 1'b1                        | 1'b0       | 1'b1       | 1'b1       | Don't care            | Don't care  | On                     | Off   | Unidir: Backwards compatible.<br>TX: Transmit data and idle. No RF.   |
| 1'b1                        | 1'b0       | 1'b1       | 1'b0       | 1'b1                  | 1'b0        | On                     | On    | Unidir: LF received.<br>TX: Transmit data 1 column<br>IDLE after end of packet and RF   |
| 1'b1                        | 1'b0       | 1'b1       | 1'b0       | 1'b0                  | 1'b1        | On                     | Off   | Unidir: RF receives<br>TX: Transmit data and idle. No RF.   |
| 1'b1                        | 1'b0       | 1'b1       | 1'b0       | 1'b0                  | 1'b0        | On                     | Off   | Unidir: No link fault<br>TX: Transmit data and idle. No RF.   |
| 1'b1                        | 1'b0       | 1'b0       | Don't care | 1'b1                  | 1'b0        | Off                    | On    | Bidir: LF received<br>TX: Stop data. Transmit RF only.  |
| 1'b1                        | 1'b0       | 1'b0       | Don't care | 1'b0                  | 1'b1        | Off                    | Off   | Bidir: RF received<br>TX: Stop data. Idle only. No RF.  |
| 1'b1                        | 1'b0       | 1'b0       | Don't care | 1'b0                  | 1'b0        | On                     | Off   | Bidir: No link fault<br>TX: Transmit data and idle. No RF.  |

At this time, the 25G Ethernet Intel FPGA IP core does not recognize received non-zero 4-bit ordered set types as an error.

**Related Information**

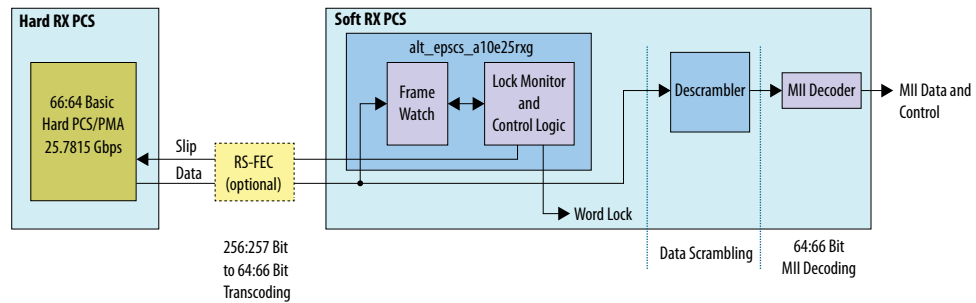
- [TX MAC Registers](#) on page 77  
Information about the LINK\_FAULT register.
- [IEEE website](#)  
The Ethernet specifications are available on the IEEE website.

**4.1.6. 25 GbE RX PCS**

The soft RX PCS interfaces to the hard PCS and PMA blocks configured in 66:64 10G PCS Basic Generic Mode with bitslip enabled. The hard PCS drives a 66-bit output stream to the soft RX PCS. The soft RX PCS implements word lock, descrambling, and MII decoding. It drives output data to the MAC. You can read the status of FIFOs at the interface of Hard RX PCS using the Control and Status registers.



Figure 20. High Level Block Diagram of the RX PCS with Optional RS-FEC Datapath



### 4.1.7. RX RS-FEC

If you turn on **Enable RS-FEC** in the 25G Ethernet Intel FPGA IP parameter editor, the IP core includes Reed-Solomon forward error correction (FEC) in both the receive and transmit datapaths.

The IP core implements Reed-Solomon FEC per Clause 108 of the IEEE Standard 802.3by. The Reed-Solomon FEC algorithm includes the following modules:

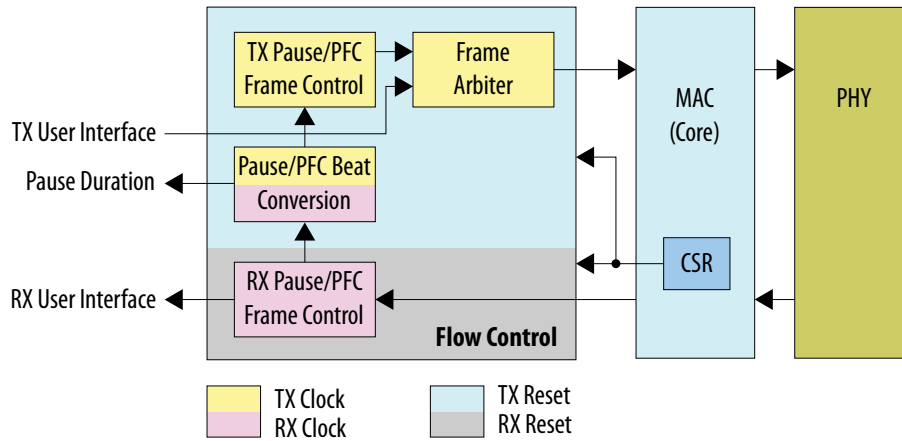
- Alignment marker lock
- 66:80 gearbox
- High-speed Reed-Solomon decoder
- 80:257 gearbox
- 256B/257B to 64B/66B Transcoding

### 4.1.8. Flow Control

Flow control reduces congestion at the local or remote link partner. When either link partner experiences congestion, the respective transmit control sends pause frames. XOFF Pause frames stop the remote transmitter. XON Pause frames let the remote transmitter resume data transmission. The 25G Ethernet Intel FPGA IP core supports both standard and Priority-based Flow Control (PFC) control frames.

**Figure 21. Flow Control Module Conceptual Overview**

The flow control module acts as a buffer between client logic and the TX and RX MAC.



Standard Flow Control (Pause Frame Flow Control):

- Inhibits the next client frame transmission on the reception of a valid Pause frame.

Priority-based Flow Control (PFC):

- PFC frame transmission follows a priority-based arbitration scheme, where the Frame Type indication is provided for the usage of external downstream logic.
- Inhibits the per queue client frame transmission on the reception of a valid PFC frame from the client. Includes per-queue PFC Pause quanta duration indicator

Flow Control includes the following features:

| Feature   | Standard Flow Control | Priority-based Flow Control (PFC) |
|---|-----------------------|-----------------------------------|
| <b>Generation and Transmission</b>  |                       |                                   |
| Programmable 1-bit or 2-bit XON/XOFF request mode                                 | Supported             | Supported                         |
| In 2-bit request mode, programmable selection of register or signal-based control | Supported             | Supported                         |
| Programmable destination and source addresses                                     | Supported             | Supported                         |
| Programmable pause quanta   | Supported             | Supported                         |
| Programmable per-queue XOFF frame separation                                      | —                     | Supported                         |
| <b>Reception and Decode</b>   |                       |                                   |
| Programmable destination address for filtering incoming pause and PFC frames      | Supported             | Supported                         |
| Configurable enable, directing the IP core to ignore incoming flow control frames | Supported             | Supported                         |
| Per-queue client frame transmission pause duration indicator                      | —                     | Supported                         |

**Caution:** The 25G Ethernet Intel FPGA IP core supports the flow control feature for either value of the **Ready Latency** parameter. However, in standard flow control you might experience data delay if you select the value of 3 for this parameter. The IP core might still hold user data packet in its internal buffer if transmission of the IP core stops due to flow control. This issue does not occur in priority-based flow control.

#### Related Information

[Pause/PFC Flow Control Registers](#) on page 79

Describes the registers that the IP core uses to implement the flow control functionality.

#### 4.1.8.1. TX Pause/PFC Flow Control Frame Transmission Request

An XON/XOFF request triggers the IP core to transmit a Pause or PFC flow control frame on the Ethernet link. You can control XON/XOFF requests using the TX flow control registers or the `pause_insert_tx0` and `pause_insert_tx1` input signals.

You can specify whether the IP core accepts XON/XOFF requests in 1-bit or 2-bit format by updating the TX Flow Control Request Mode register field. By default the IP core assumes 1-bit requests.

### 4.1.8.2. XON/XOFF Pause Frames

#### Priority-based Flow Control

You can trigger the 25G Ethernet Intel FPGA IP core to transmit PFC XOFF frame with a pause duration that is specified in TX Flow Control Quanta register by updating the `pause_insert_tx0` and `pause_insert_tx1` input signals or TX flow control registers. If an enabled priority queue is in the XOFF condition, a new PFC frame is transmitted after the minimum time gap. You specify the minimum time gap in the per priority queue TX Flow Control Signal XOFF Request Hold Quanta register. The minimum time gap between two consecutive PFC frames is 1 pause quanta or 512-bit times. PFC frame transmission ends when none of the PFC interfaces of all enabled priority queues is requesting PFC frames.

A transition from XOFF to XON in any enabled priority queue triggers the IP core to transmit a PFC frame with pause quanta of 0 for the associated priority queue. The IP core sends a single XON flow control frame. In the rare case that the XON frame is lost or corrupted, the remote partner should still be able to resume transmission. The remote partner resumes transmission after the duration or the pause quanta value specified in the previous XOFF flow control frame expires.

#### Standard Flow Control

In the case of standard flow control, the IP core transmits Pause frames instead of PFC frames. The transmission behavior is identical.

When the IP core is in standard flow control mode and receives a Pause frame, the IP core stops processing TX client data, either immediately or at the next frame boundary. Client data transmission resumes when all of the following conditions are true:

- The time specified by the pause quanta has elapsed and there is no new quanta value.
- A valid pause frame with 0 pause duration has been received.

A Pause frame has no effect if the associated TX Flow Control Enable register bit is set to disable XON and XOFF flow control.

### 4.1.9. 1588 Precision Time Protocol Interfaces

If you turn on **Enable IEEE 1588**, the 25G Ethernet Intel FPGA IP core processes and provides 1588 Precision Time Protocol (PTP) timestamp information as defined in the *IEEE 1588-2008 Precision Clock Synchronization Protocol for Networked Measurement and Control Systems Standard*. This feature supports PHY operating speed with a constant timestamp accuracy of  $\pm 4$  ns and a dynamic timestamp accuracy of  $\pm 1$  ns.

1588 PTP packets carry timestamp information. The 25G Ethernet Intel FPGA IP core updates the incoming timestamp information in a 1588 PTP packet to transmit a correct updated timestamp with the data it transmits on the Ethernet link, using a one-step or two-step clock.

A fingerprint can accompany a 1588 PTP packet. You can use this information for client identification and other client uses. If provided fingerprint information, the IP core passes it through unchanged.

The IP core connects to a time-of-day (TOD) module that continuously provides the current time of day based on the input clock frequency. Because the module is outside the 25G Ethernet Intel FPGA IP core, you can use the same module to provide the current time of day for multiple modules in your system.

#### Related Information

- [1588 PTP Registers](#) on page 92
- [IEEE website](#)  
The *IEEE 1588-2008 Precision Clock Synchronization Protocol for Networked Measurement and Control Systems Standard* is available on the IEEE website.

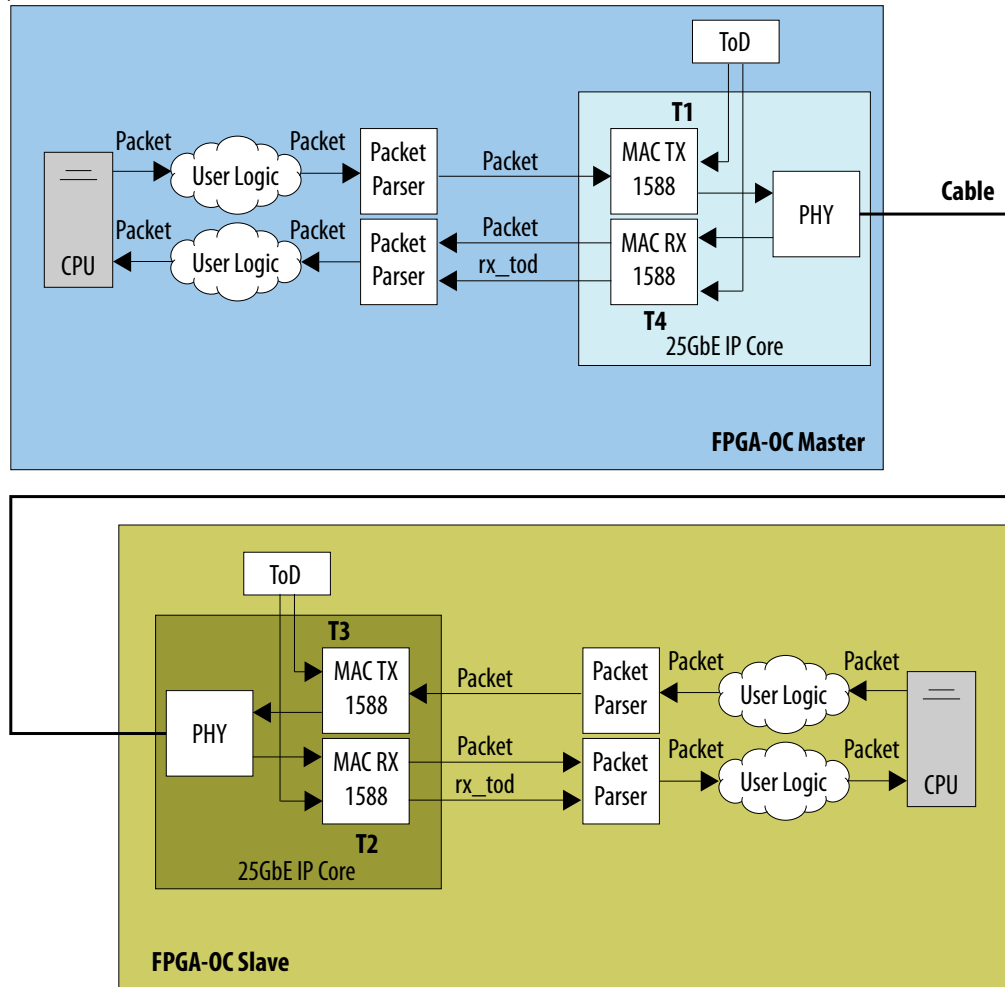
#### 4.1.9.1. Implementing a 1588 System That Includes a 25G Ethernet Intel FPGA IP Core

The 1588 specification in *IEEE 1588-2008 Precision Clock Synchronization Protocol for Networked Measurement and Control Systems Standard* describes various systems you can implement in hardware and software to synchronize clocks in a distributed system by communicating offset and frequency correction information between master and slave clocks in arbitrarily complex systems. A 1588 system that includes the 25G Ethernet Intel FPGA IP core with 1588 PTP functionality uses the incoming and outgoing timestamp information from the IP core and the other modules in the system to synchronize clocks across the system.

The 25G Ethernet Intel FPGA IP core with 1588 PTP functionality provides the timestamp manipulation and basic update capabilities required to integrate your IP core in a 1588 system. You can specify that packets are PTP packets, and how the IP core should update incoming timestamps from the client interface before transmitting them on the Ethernet link. The IP core does not implement the event messaging layers of the protocol, but rather provides the basic hardware capabilities that support a system in implementing the full 1588 protocol.

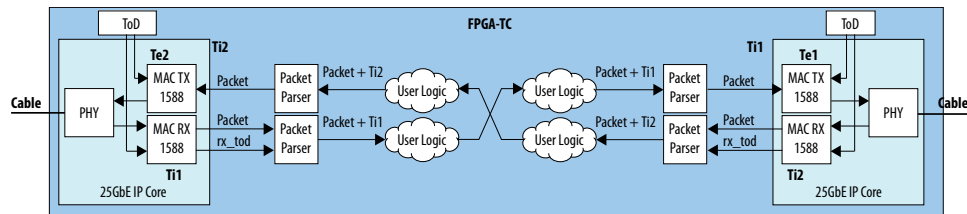
**Figure 22. Example Ethernet System with Ordinary Clock Master and Ordinary Clock Slave**

You can implement both master and slave clocks using the 25G Ethernet Intel FPGA IP core with 1588 PTP functionality. Refer to [Adding the External Time-of-Day Module for Variations with 1588 PTP Feature](#) for implementation of the TOD module.



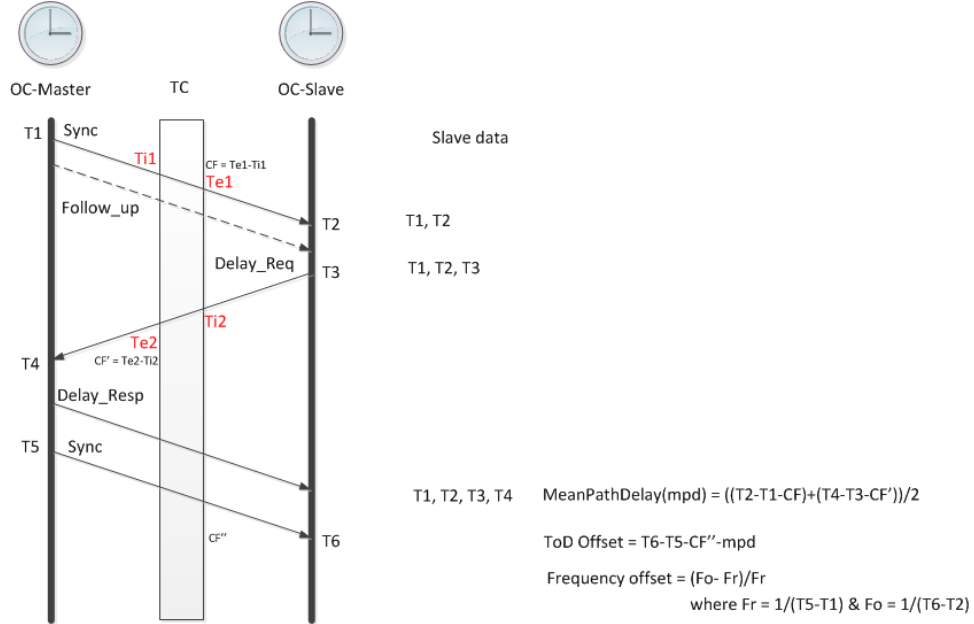
**Figure 23. Hardware Configuration Example Using 25G Ethernet Intel FPGA IP core in a 1588 System in Transparent Clock Mode**

Refer to [Adding the External Time-of-Day Module for Variations with 1588 PTP Feature](#) for implementation of the TOD module.



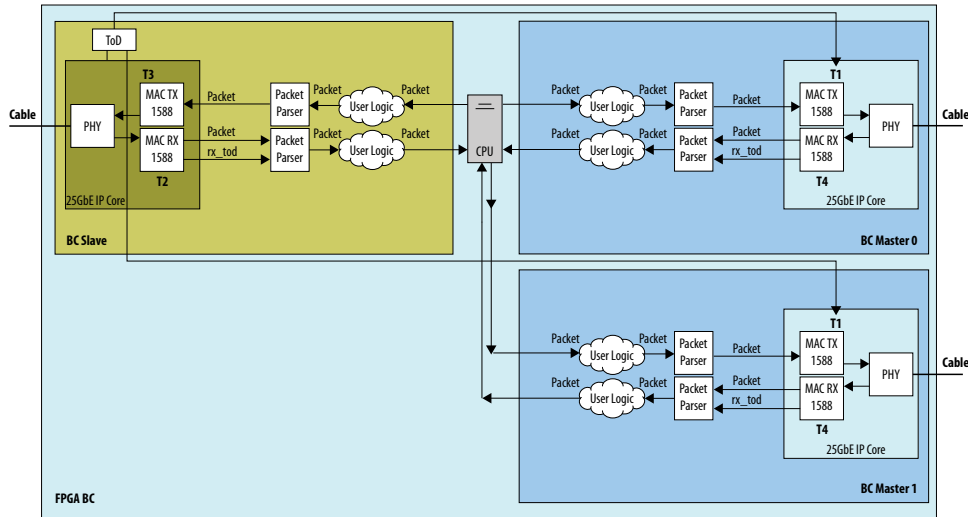
**Figure 24. Software Flow Using Transparent Clock Mode System**

This figure from the 1588 standard is augmented with the timestamp labels shown in the transparent clock system figure. A precise description of the software requirements is beyond the scope of this document. Refer to the 1588 standard.



**Figure 25. Example Boundary Clock with One Slave Port and Two Master Ports**

You can implement a 1588 system in boundary clock mode using the 25G Ethernet Intel FPGA IP core with 1588 PTP functionality. Refer to [Adding the External Time-of-Day Module for Variations with 1588 PTP Feature](#) for implementation of the TOD module.



**Related Information**

[IEEE website](#)

The *IEEE 1588-2008 Precision Clock Synchronization Protocol for Networked Measurement and Control Systems Standard* is available on the IEEE website.

#### 4.1.9.2. PTP Transmit Functionality

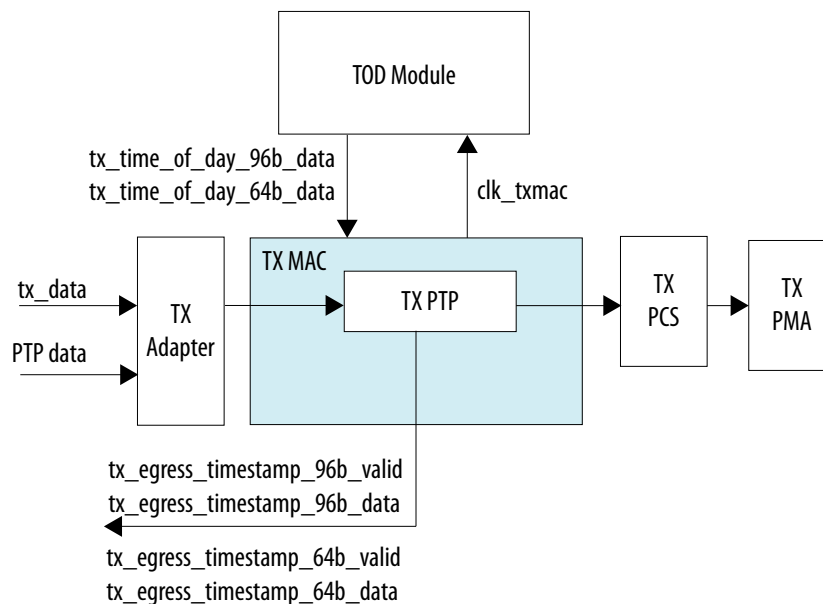
When you send a 1588 PTP packet to a 25G Ethernet Intel FPGA IP core with **Enable IEEE 1588** turned on in the parameter editor, you should assert the following respective input signals with the TX SOP signal to tell the IP core the PTP operations or processes that the IP core should perform to the packet:

- `tx_egress_timestamp_request_valid`: assert this signal to tell the IP core to process the current packet in two-step processing mode.
- `tx_etstamp_ins_ctrl_timestamp_insert`: assert this signal to tell the IP core to process the current packet in one-step processing mode and to insert the exit timestamp for the packet in the packet (insertion mode).
- `tx_etstamp_ins_ctrl_residence_time_update`: assert this signal to tell the IP core to process the current packet in one-step processing mode and to update the timestamp in the packet by adding the latency through the IP core (the residence time in the IP core) to the cumulative delay field maintained in the packet (correction mode). This mode supports transparent clock systems.

*Note:* If `tx_etstamp_ins_ctrl_residence_time_update` is asserted, you should not assert `tx_egress_timestamp_request_valid` or `tx_etstamp_ins_ctrl_timestamp_insert` as the result will be undefined.

The IP core transmits the 1588 PTP packet in an Ethernet frame after PTP processing.

**Figure 26. PTP Transmit Block Diagram**



In one-step mode, the IP core either overwrites the timestamp information provided at the user-specified offset with the packet exit timestamp (insertion mode), or adds the residence time in this system to the value at the specified offset (correction mode). You tell the IP core how to process the timestamp by asserting the appropriate signal with the TX SOP signal. You must specify the offset of the timestamp in the packet (`tx_etstamp_ins_ctrl_offset_timestamp`) in insertion mode, or the offset of the correction field in the packet



(`tx_etstamp_ins_ctrl_offset_correction_field`) in correction mode. In addition, the IP core zeroes out or updates the UDP checksum, or leaves the UDP checksum as is, depending on the mutually exclusive `tx_etstamp_ins_ctrl_checksum_zero` and `tx_etstamp_ins_ctrl_checksum_correct` signals. Checksum calculation is mandatory for the UDP/IPv6 protocol. You must extend 2 bytes at the end of the UDP payload of the PTP packet. The MAC function modifies the extended bytes to ensure that the UDP checksum remains uncompromised.

Two-step PTP processing ignores the values on the one-step processing signals. In two-step processing mode, the IP core does not modify the current timestamp in the packet. Instead, the IP core transmits a two-step derived timestamp on the separate `tx_egress_timestamp_96b_data[95:0]` or `tx_egress_timestamp_64b_data[63:0]` bus, when it begins transmitting the Ethernet frame. The value on the `tx_egress_timestamp_{96b,64b}_data` bus is the packet exit timestamp. The `tx_egress_timestamp_{96b,64b}_data` bus holds a valid value when the corresponding `tx_egress_timestamp_{96b,64b}_valid` signal is asserted.

In addition, to help the client to identify the packet, you can specify a fingerprint to be passed by the IP core in the same clock cycle with the timestamp. To specify the number of distinct fingerprint values the IP core can handle, set the **Fingerprint width** parameter to the desired number of bits  $w$ . You provide the fingerprint value to the IP core in the `tx_egress_timestamp_request_fingerprint[(w-1):0]` signal. The IP core then drives the fingerprint on the appropriate `tx_egress_timestamp_{96b,64b}_fingerprint[(w-1):0]` port with the corresponding output timestamp, when it asserts the `tx_egress_timestamp_{96b,64b}_valid` signal.

The IP core calculates the packet exit timestamp.

exit TOD = entry TOD + IP core maintained expected latency + user-configured PMA latency

- entry TOD is the value in `tx_time_of_day_96b_data` or `tx_time_of_day_64b_data` when the packet enters the IP core.
- The expected latency through the IP core is a static value. The IP core maintains this value internally.
- The IP core reads the user-configured PMA latency from the `TX_PTP_PMA_LATENCY` register. This option is provided for user flexibility.

The IP core provides the exit TOD differently in different processing modes.

- In two-step mode, the IP core drives the exit TOD on `tx_egress_timestamp_96b_data` and on `tx_egress_timestamp_64b_data`, as available.
- In one-step processing insertion mode, the IP core inserts the exit TOD in the timestamp field of the packet at the offset you specify in `tx_etstamp_ins_ctrl_offset_timestamp`.
- In one-step processing correction mode, the IP core calculates the exit TOD and uses it only to calculate the residence time.

In one-step processing correction mode, the IP core calculates the updated correction field value:

exit correction field value = entry correction field value + residence time + asymmetry extra latency

- residence time = exit TOD - entry (ingress) timestamp.
- entry (ingress) timestamp is the value on `tx_etstamp_ins_ctrl_ingress_timestamp_{95,64}b` in the SOP cycle when the IP core received the packet on the TX client interface. The application is responsible to drive this signal with the correct value for the cumulative calculation. The correct value depends on system configuration.
- The IP core reads the asymmetry extra latency from the `TX_PTP_ASYM_DELAY` register if the `tx_egress_asymmetry_update` signal is asserted. This option is provided for additional user-defined precision. You can set the value of this register and set the `tx_egress_asymmetry_update` signal to indicate the register value should be included in the latency calculation.

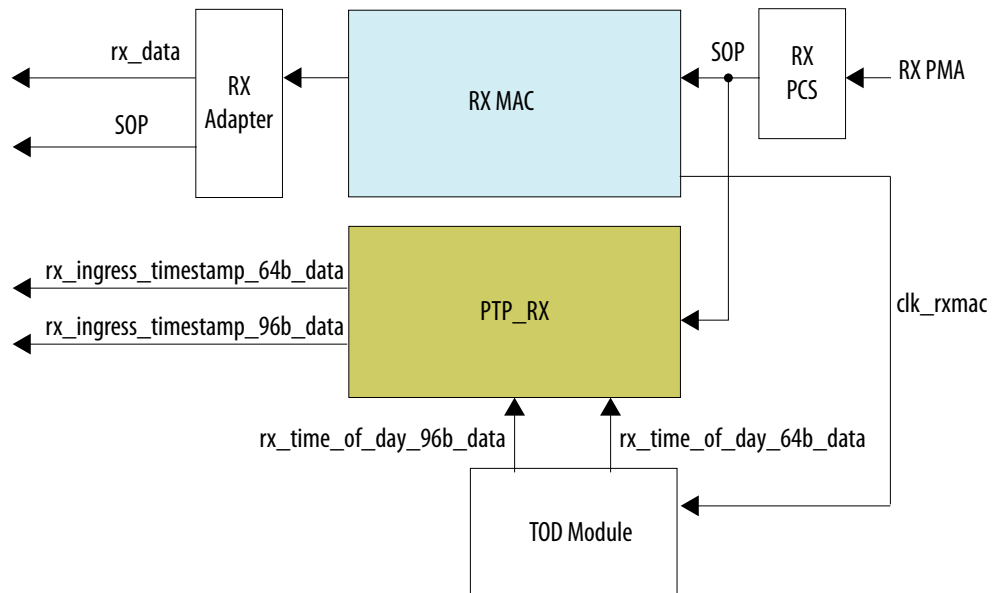
#### Related Information

- [1588 PTP Registers](#) on page 92
- [IEEE website](#)  
The *IEEE 1588-2008 Precision Clock Synchronization Protocol for Networked Measurement and Control Systems Standard* is available on the IEEE website.

#### 4.1.9.3. PTP Receive Functionality

If you turn on **Enable IEEE 1588** in the 25G Ethernet Intel FPGA IP parameter editor, the IP core provides a 96-bit (V2 format) or 64-bit timestamp with every packet on the RX client interface, whether it is a 1588 PTP packet or not. The value on the timestamp bus (`rx_ingress_timestamp_96b_data[95:0]` or `rx_ingress_timestamp_64b_data[63:0]` or both, if present) is valid in the same clock cycle as the RX SOP signal. The value on the timestamp bus is not the current timestamp; instead, it is the timestamp from the time when the IP core received the packet on the Ethernet link. The IP core captures the time-of-day from the TOD module on `rx_time_of_data_96b_data` or `rx_time_of_day_64b_data` at the time it receives the packet on the Ethernet link, and sends that timestamp to the client on the RX SOP cycle on the timestamp bus `rx_ingress_timestamp_96b_data[95:0]` or `rx_ingress_timestamp_64b_data[63:0]` or both, if present. User logic can use this timestamp or ignore it.

Figure 27. PTP Receive Block Diagram



#### Related Information

[IEEE website](#)

The *IEEE 1588-2008 Precision Clock Synchronization Protocol for Networked Measurement and Control Systems Standard* is available on the IEEE website.

#### 4.1.9.4. External Time-of-Day Module for 1588 PTP Variations

25G Ethernet Intel FPGA IP cores that include the 1588 PTP module require an external time-of-day (TOD) module to provide the current time-of-day in each clock cycle, based on the incoming clock. The TOD module must update the time-of-day output value on every clock cycle, and must provide the TOD value in the V2 format (96 bits) or the 64-bit TOD format, or both.

#### Related Information

[Adding the External Time-of-Day Module for Variations with 1588 PTP Feature](#) on page 25

#### 4.1.9.5. PTP Timestamp and TOD Formats

The 25G Ethernet Intel FPGA IP core supports a 96-bit timestamp (V2 format) or a 64-bit timestamp (correction-field format) in PTP packets. The 64-bit timestamp and TOD signals of the IP core are in an Intel-defined 64-bit format that is distinct from the V1 format, for improved efficiency in one-step processing correction mode. Therefore, if your system need not handle any packets in one-step processing correction mode, you should set the **Time of day format** parameter to the value of **Enable 96-bit timestamp format**.

You control the format or formats the IP core supports with the **Time of day format** parameter. If you set the value of this parameter to **Enable 96-bit timestamp format** or **Enable both formats**, your IP core can support two-step processing mode, one-step processing insertion mode, and one-step processing correction mode, and can support both V1 and V2 formats. You can set the parameter value to **Enable**

**64-bit timestamp format** to support one-step processing correction mode more efficiently. However, if you do so, your IP core variation cannot support two-step processing mode and cannot support one-step processing insertion mode. If you turn on both of these parameters, the value you drive on the `tx_estamp_ins_ctrl_timestamp_format` or `tx_etstamp_ins_ctrl_residence_time_calc_format` signal determines the format the IP core supports for the current packet.

The IP core completes all internal processing in the V2 format. However, if you specify V1 format for a particular PTP packet in one-step insertion mode, the IP core inserts the appropriate V1-format timestamp in the outgoing packet on the Ethernet link.

### V2 Format

The IP core maintains the time-of-day (TOD) in V2 format according to the IEEE specification:

- Bits [95:48]: Seconds (48 bits).
- Bits [47:16]: Nanoseconds (32 bits). This field overflows at 1 billion.
- Bits [15:0]: Fractions of nanosecond (16 bits). This field is a true fraction; it overflows at 0xFFFF.

The IP core can receive time-of-day information from the TOD module in V2 format or in 64-bit TOD format, or both, depending on your setting for the **Time of day format** parameter.

### V1 Format

V1 timestamp format is specified in the IEEE specification:

- Bits [63:32]: Seconds (32 bits).
- Bits [31:0]: Nanoseconds (32 bits). This field overflows at 1 billion.

### Intel 64-Bit TOD Format

The Intel 64-bit TOD format is distinct from the V1 format and supports a longer time delay. It is intended for use in transparent clock systems, in which each node adds its own residence time to a running total latency through the system. This format matches the format of the correction field in the packet, as used in transparent clock mode.

- Bits [63:16]: Nanoseconds (48 bits). This field can specify a value greater than 4 seconds.
- Bits [15:0]: Fractions of nanosecond (16 bits). This field is a true fraction; it overflows at 0xFFFF.

The TOD module provides 64-bit TOD information to the IP core in this 64-bit TOD format. The expected format of all 64-bit input timestamp and TOD signals to the IP core is the Intel 64-bit TOD format. The format of all 64-bit output timestamp and TOD signals from the IP core is the Intel 64-bit TOD format. If you build your own TOD module that provides 64-bit TOD information to the IP core, you must ensure it provides TOD information in the Intel 64-bit TOD format.

### Related Information

[IEEE website](#)

The *IEEE 1588-2008 Precision Clock Synchronization Protocol for Networked Measurement and Control Systems Standard* is available on the IEEE website.

#### 4.1.9.6. Design Considerations in PTP

- When the PTP option is enabled together with RS-FEC option, there is no accuracy loss by neglecting bit shift due to transcode effect with the assumption transcode effect will be totally reversed at the receiver side.
- When the PTP option is enabled together with 10/25G switching option, `tx_period`, `rx_period`, `tx_pma_delay`, and `rx_pma_delay` need to be reconfigured according to the running speed. Refer to the *1588 PTP Registers* section for the correct value.

### Related Information

[1588 PTP Registers](#) on page 92

## 4.2. User Interface to Ethernet Transmission

The IP core reverses the bit stream for transmission per Ethernet requirements. The transmitter handles the insertion of the inter-packet gap, frame delimiters, and padding with zeros as necessary. The transmitter also handles FCS computation and insertion.

The IP core transmits complete packets. After transmission begins, it must complete with no IDLE insertions. Between the end of one packet and the beginning of the next packet, the data input is not considered and the transmitter sends IDLE characters. An unbounded number of IDLE characters can be sent between packets.

### 4.2.1. Order of Transmission

The IP core transmits bytes on the Ethernet link starting with the preamble and ending with the FCS in accordance with the IEEE 802.3 standard. On the transmit client interface, the IP core expects the client to send the most significant bytes of the frame first, and to send each byte in big-endian format. Similarly, on the receive client interface, the IP core sends the client the most significant bytes of the frame first, and orders each byte in big-endian format.

**Figure 28. Byte Order on the Client Interface Lanes**

Describes the byte order on the Avalon streaming interface. Destination Address[40] is the broadcast/multicast bit (a type bit), and Destination Address[41] is a locally administered address bit.

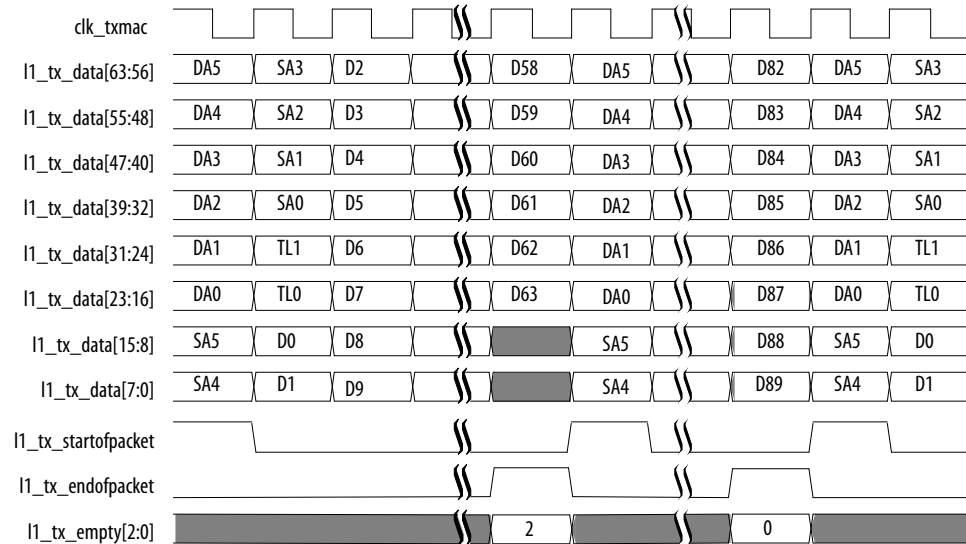
|       | Destination Address (DA) |         |         |         |        |       | Source Address (SA) |         |         |         |        |       | Type/<br>Length (TL) |       | Data (D) |     |          |
|-------|--------------------------|---------|---------|---------|--------|-------|---------------------|---------|---------|---------|--------|-------|----------------------|-------|----------|-----|----------|
| Octet | 5                        | 4       | 3       | 2       | 1      | 0     | 5                   | 4       | 3       | 2       | 1      | 0     | 1                    | 0     | 00       | ... | NN       |
| Bit   | [47:40]                  | [39:32] | [31:24] | [23:16] | [15:8] | [7:0] | [47:40]             | [39:32] | [31:24] | [23:16] | [15:8] | [7:0] | [15:8]               | [7:0] | MSB[7:0] | ..  | LSB[7:0] |

For example, the destination MAC address includes the following six octets AC-DE-48-00-00-80. The first octet transmitted (octet 0 of the MAC address described in the 802.3 standard) is AC and the last octet transmitted (octet 5 of the MAC address) is 80. The first bit transmitted is the low-order bit of AC, a zero. The last bit transmitted is the high order bit of 80, a one.

The preceding table and the following figure show that in this example, 0xAC is driven on DA5 (DA[47:40]) and 0x80 is driven on DA0 (DA[7:0]).

**Figure 29. Octet Transmission on the 25GbE Avalon Streaming Signals**

In the following diagram Preamble pass through and CRC pass through mode are disabled.



### 4.2.2. Bit Order For TX and RX Datapaths

The TX bit order matches the placement shown in the PCS lanes as illustrated in *IEEE Standard for Ethernet, Section 4, Figure 49-5*. The RX bit order matches the placement shown in *IEEE Standard for Ethernet, Section 4, Figure 49-6*.

#### Related Information

[IEEE website](#)

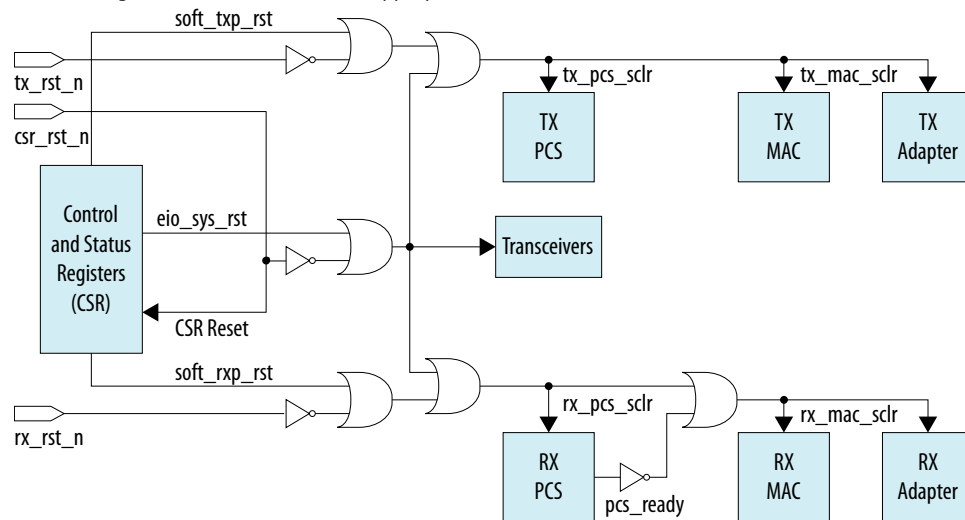
The *IEEE Standard for Ethernet, Section 4* is available on the IEEE website.

## 5. Reset

Control and Status registers control three parallel soft resets. These soft resets are not self-clearing. Software clears them by writing to the appropriate register. Asserting the external hard reset `csr_rst_n` returns Control and Status registers to their original values.

**Figure 30. Conceptual Overview of Reset Logic**

The three hard resets are top-level ports. The soft resets are internal signals which are outputs of the `PHY_CONFIG` register. Software writes the appropriate bit of the `PHY_CONFIG` to assert a soft reset.



The internal soft reset signals reset the following functions:

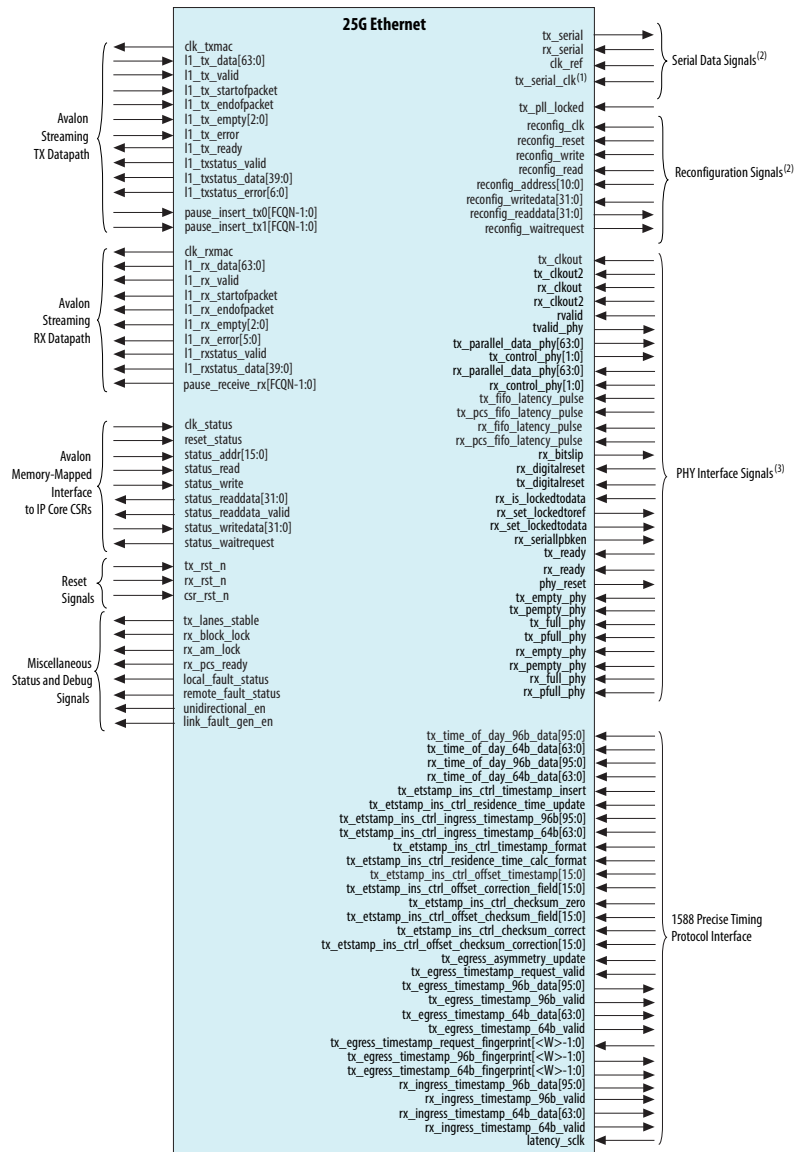
- `soft_txp_rst`: Resets the IP core in TX direction. Resets the TX PCS, MAC, and adapter. This soft reset leads to deassertion of `tx_lanes_stable` output signal.
- `soft_rxp_rst`: Resets the IP core in RX direction. Resets the RX PCS, MAC, and adapter. This soft reset leads to the deassertion of `rx_pcs_ready` output signal.
- `eio_sys_rst`: Resets the IP core. Resets the TX and RX MACs, PCS, adapters, and transceivers. Does not reset the Control and Status registers. This soft reset leads to the deassertion of `tx_lanes_stable` and `rx_pcs_ready` output signal.

### Related Information

- [Reset Signals](#) on page 73
- [PHY Registers](#) on page 75

## 6. Interfaces and Signal Descriptions

Figure 31. 25G Ethernet Intel FPGA IP Signals and Interfaces



Notes:

- When 10/25G dynamic rate switching is enabled, the tx\_serial\_clk signal is changed to tx\_serial\_clk0 and tx\_serial\_clk1 signals.
- These signals are applicable only for MAC+PCS+PMA core variant.
- These signals are applicable only for MAC+PCS core variant.

Intel Corporation. All rights reserved. Agilex, Altera, Arria, Cyclone, eASIC, Intel, the Intel logo, MAX, Nios, Quartus and Stratix words and logos are trademarks of Intel Corporation or its subsidiaries in the U.S. and/or other countries. Intel warrants performance of its FPGA and semiconductor products to current specifications in accordance with Intel's standard warranty, but reserves the right to make changes to any products and services at any time without notice. Intel assumes no responsibility or liability arising out of the application or use of any information, product, or service described herein except as expressly agreed to in writing by Intel. Intel customers are advised to obtain the latest version of device specifications before relying on any published information and before placing orders for products or services.

\*Other names and brands may be claimed as the property of others.



## 6.1. TX MAC Interface to User Logic

The TX MAC provides an Avalon streaming interface to the FPGA fabric. The minimum packet size is nine bytes.

**Table 15. Avalon Streaming TX Datapath**

All interface signals are clocked by the `clk_txmac` clock. The value you specify for **Ready Latency** in the 25G Ethernet Intel FPGA IP parameter editor is the Avalon streaming readyLatency value on this interface.

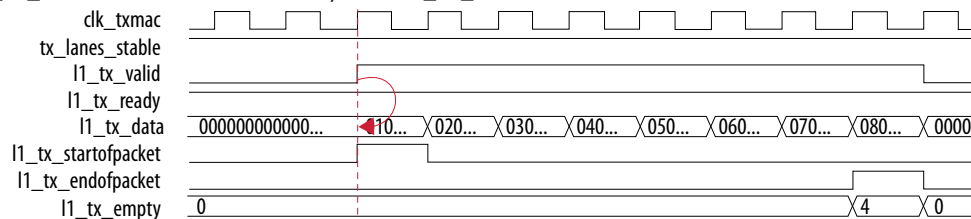
| Signal                           | Direction | Description   |
|----------------------------------|-----------|---|
| <code>clk_txmac</code>           | Output    | Clock for the TX logic. Derived from <code>pll_refclk</code> , and is an output from the 25G Ethernet Intel FPGA IP core. <code>clk_txmac</code> is guaranteed to be stable when <code>tx_lanes_stable</code> is asserted. The frequency of this clock is 390.625 MHz. <b>All TX MAC interface signals are synchronous to <code>clk_txmac</code>.</b>   |
| <code>l1_tx_data[63:0]</code>    | Input     | Data input to MAC. Bit 63 is the MSB and bit 0 is the LSB. Bytes are read in the usual left to right order.<br>The 25G Ethernet Intel FPGA IP core does not process incoming frames of less than nine bytes correctly. You must ensure such frames do not reach the TX client interface.<br>You must send each TX data packet without intermediate idle cycles. Therefore, you must ensure your application can provide the data for a single packet in consecutive clock cycles. If data might not be available otherwise, you must buffer the data in your design and wait to assert <code>l1_tx_startofpacket</code> when you are assured the packet data to send on <code>l1_tx_data[63:0]</code> is available or will be available on time.<br>If <code>readyLatency = 0</code> , ensure that no data transition at the <code>l1_tx_data</code> bus at the same clock cycle <code>l1_tx_ready</code> is deasserted. You can transition the data at the <code>l1_tx_data</code> bus at the same clock cycle <code>l1_tx_ready</code> is asserted.<br>If <code>readyLatency = 3</code> , ensure that no data transition at the <code>l1_tx_data</code> bus at the third clock cycle after <code>l1_tx_ready</code> is deasserted. You can transition the data at the <code>l1_tx_data</code> bus at the third clock cycles after <code>l1_tx_ready</code> is asserted. |
| <code>l1_tx_valid</code>         | Input     | When asserted, indicates valid data is available on <code>l1_tx_data[63:0]</code> . You must assert this signal continuously between the assertions of <code>l1_tx_startofpacket</code> and <code>l1_tx_endofpacket</code> for the same packet regardless of the <code>l1_tx_ready</code> status.   |
| <code>l1_tx_startofpacket</code> | Input     | When asserted, indicates the first byte of a frame. When <code>l1_tx_startofpacket</code> is asserted, the MSB of <code>l1_tx_data</code> drives the start of packet.<br>Packets that drive <code>l1_tx_startofpacket</code> and <code>l1_tx_endofpacket</code> in the same cycle are ignored.  |
| <code>l1_tx_endofpacket</code>   | Input     | When asserted, indicates the end of a packet.<br>Packets that drive <code>l1_tx_startofpacket</code> and <code>l1_tx_endofpacket</code> in the same cycle are ignored.  |
| <code>l1_tx_empty[2:0]</code>    | Input     | Specifies the number of empty bytes on <code>l1_tx_data</code> when <code>l1_tx_endofpacket</code> is asserted.   |
| <code>l1_tx_error</code>         | Input     | When asserted in the same cycle as <code>l1_tx_endofpacket</code> , indicates the current packet should be treated as an error packet. Assertion at any other position in the packet is ignored.<br>The TX statistics counters do not reflect errors the IP core creates in response to this signal.  |
| <code>l1_tx_ready</code>         | Output    | When asserted, indicates that the MAC can accept the data. When the <code>readyLatency = 0</code> , the IP core accepts valid data in the same clock cycle in which it asserts <code>l1_tx_ready</code> . When the <code>readyLatency = 3</code> , the IP core accepts valid data 3 clock cycles after it asserts <code>l1_tx_ready</code> .  |

*continued...*

| Signal   | Direction | Description  |
|--|-----------|--|
| l1_txstatus_valid  | Output    | When asserted, indicates that l1_txstatus_data[39:0] is driving valid data.  |
| l1_txstatus_data[39:0]                                   | Output    | Specifies information about the transmit frame. The following fields are defined: <ul style="list-style-type: none"> <li>• Bit[39]: When asserted, indicates a PFC frame</li> <li>• Bit[38]: When asserted, indicates a unicast frame</li> <li>• Bit[37]: When asserted, indicates a multicast frame</li> <li>• Bit[36]: When asserted, indicates a broadcast frame</li> <li>• Bit[35]: When asserted, indicates a pause frame</li> <li>• Bit[34]: When asserted, indicates a control frame</li> <li>• Bit[33]: When asserted, indicates a VLAN frame</li> <li>• Bit[32]: When asserted, indicates a stacked VLAN frame</li> <li>• Bits[31:16]: Specifies the frame length from the first byte of the destination address to the last byte of the FCS</li> <li>• Bits[15:0]: Specifies the payload length</li> </ul>   |
| l1_txstatus_error[6:0]                                   | Output    | Specifies the error type in the transmit frame. The following fields are defined: <ul style="list-style-type: none"> <li>• Bits[6:3]: Reserved</li> <li>• Bit[2]: Payload length error</li> <li>• Bit[1]: Oversized frame</li> <li>• Bit[0]: Reserved</li> </ul>   |
| pause_insert_tx0[FCQN-1:0]<br>pause_insert_tx1[FCQN-1:0] | Input     | Available if you specify Pause or PFC. Indicates to the MAC if an XON, XOFF, Pause or PFC frame should be sent. FCQN equals 1 for Pause and 1-8 for PFC.<br>In 1-bit programming mode, the IP core ignores pause_insert_tx1[FCQN-1:0]. In 2-bit programming mode, the higher-order bit is in pause_insert_tx1[FCQN-1:0] and the lower-order bit is in pause_insert_tx0[FCQN-1:0].<br>The following encodings are defined for 1-bit programming mode: <ul style="list-style-type: none"> <li>• 0 = No request</li> <li>• 0 to 1 = Generate XOFF request</li> <li>• 1 = Continue to generate XOFF request</li> <li>• 1 to 0 = Generate XON request</li> </ul> The following encodings are defined for the 2-bit programming model: <ul style="list-style-type: none"> <li>• 2'b00: No further XON/XOFF request. If there is a XON/XOFF flow control frame in progress, it is sent</li> <li>• 2'b01: Generate XON flow control frame</li> <li>• 2'b10: Generate XOFF request</li> <li>• 2'b11: Invalid</li> </ul> |

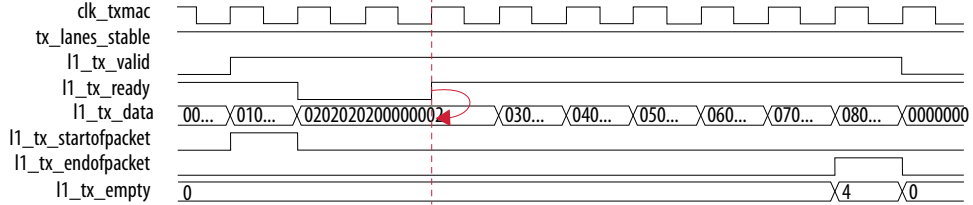
**Figure 32. Client to 25G Ethernet Intel FPGA IP MAC Avalon Streaming Interface when Ready Latency is 0 (1 of 2)**

This timing diagram shows l1\_tx\_ready asserts before l1\_tx\_valid asserts. TX MAC captures data at l1\_tx\_data bus at the same clock cycle as l1\_tx\_valid asserts.



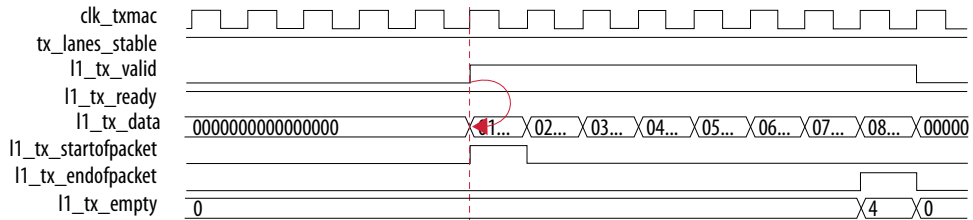
**Figure 33. Client to 25G Ethernet Intel FPGA IP MAC Avalon Streaming Interface when Ready Latency is 0 (2 of 2)**

This timing diagram shows l1\_tx\_valid asserts before l1\_tx\_ready asserts. TX MAC captures data at l1\_tx\_data bus at the same clock cycle as l1\_tx\_ready asserts.

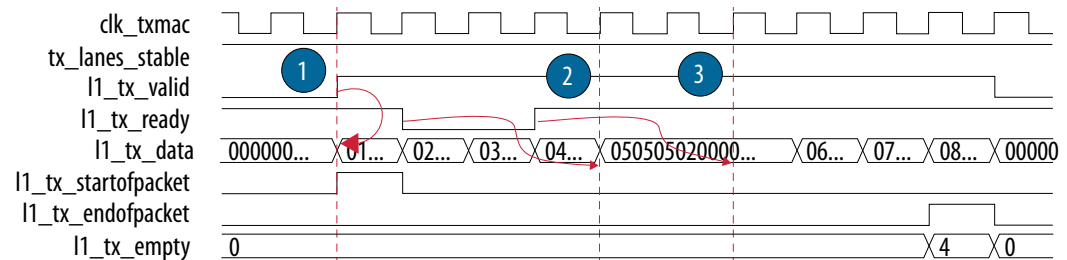


**Figure 34. Client to 25G Ethernet Intel FPGA IP MAC Avalon Streaming Interface when Ready Latency is 3 (1 of 2)**

This timing diagram shows l1\_tx\_ready asserts before l1\_tx\_valid asserts. TX MAC captures data at l1\_tx\_data bus at the same clock cycle as l1\_tx\_valid asserts.



**Figure 35. Client to 25G Ethernet Intel FPGA IP MAC Avalon Streaming Interface when Ready Latency is 3 (2 of 2)**



**Notes:**

1. TX MAC captures data at l1\_tx\_data bus at the same clock cycle as l1\_tx\_valid asserts.
2. TX MAC stops capturing data at l1\_tx\_data bus 3 clock cycles after l1\_tx\_ready de-asserts.
3. TX MAC captures data at l1\_tx\_data bus 3 clock cycles after l1\_tx\_ready asserts.

**Related Information**

[Avalon Interface Specifications](#)

Detailed information about Avalon streaming interfaces and the Avalon streaming readyLatency parameter.

**6.2. RX MAC Interface to User Logic**

The RX MAC provides an Avalon streaming interface to the FPGA fabric. The datapath consists of a single 64-bit word.

**Table 16. Avalon Streaming RX Datapath**

All interface signals are clocked by the `clk_rxmac` clock.

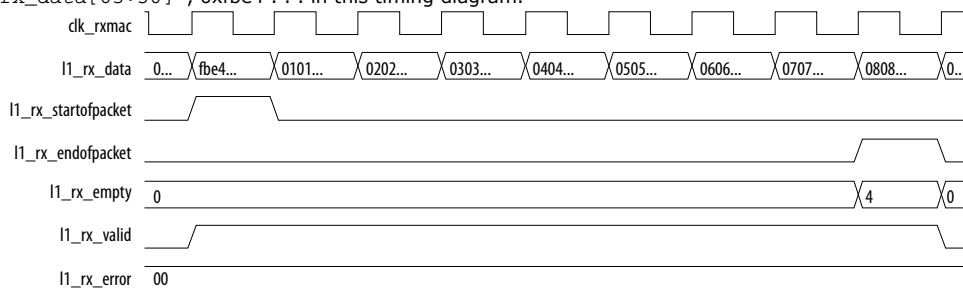
| Signal                              | Direction | Description  |
|-------------------------------------|-----------|--|
| <code>clk_rxmac</code>              | Output    | Clock for the RX MAC. Recovered from the incoming data. This clock is guaranteed stable when <code>rx_pcs_ready</code> is asserted. The frequency of this clock is 390.625 MHz for 25G data rate and 156.25 MHz for 10G data rate.. <b>All RX MAC interface signals are synchronous to <code>clk_rxmac</code>.</b>   |
| <code>l1_rx_data[63:0]</code>       | Output    | Data output from the MAC. Bit[63] is the MSB and bit[0] is the LSB. Bytes are read in the usual left to right order. The IP core reverses the byte order to meet the requirements of the Ethernet standard.  |
| <code>l1_rx_valid</code>            | Output    | When asserted, indicates that <code>l1_rx_data[63:0]</code> is driving valid data. If you turn off <b>Enable RS-FEC</b> , the IP core asserts this signal continuously between the assertions of <code>l1_tx_startofpacket</code> and <code>l1_tx_endofpacket</code> for the same packet. However, if you turn on <b>Enable RS-FEC</b> , the IP core drives IDLE cycles during alignment marker cycles.  |
| <code>l1_rx_startofpacket</code>    | Output    | When asserted, indicates the first byte of a frame.  |
| <code>l1_rx_endofpacket</code>      | Output    | When asserted, indicates the last data byte of a frame, before the frame check sequence (FCS). In CRC pass-through mode, it is the last byte of the FCS. The packet can end at any byte position.  |
| <code>l1_rx_empty[2:0]</code>       | Output    | Specifies the number of empty bytes when <code>l1_rx_endofpacket</code> is asserted. The packet can end at any byte position. The empty bytes are the low-order bytes.   |
| <code>l1_rx_error[5:0]</code>       | Output    | When asserted in the same cycle as <code>l1_rx_endofpacket</code> , indicates the current packet should be treated as an error packet. The 6 bits of <code>l1_rx_error</code> specify the following errors: <ul style="list-style-type: none"> <li><code>l1_rx_error[5]</code>: Unused.</li> <li><code>l1_rx_error[4]</code>: Payload length error. If the length field is &lt;1535 bytes (0x600 bytes), the received payload length is less than what is advertised in the payload length field.</li> <li><code>l1_rx_error[3]</code>: Oversized frame. The frame size is greater than the value specified in the <code>MAX_RX_SIZE_CONFIG</code> register.</li> <li><code>l1_rx_error[2]</code>: Undersized frame – The frame size is less than 64 bytes. Frame size = header size + payload size.</li> <li><code>l1_rx_error[1]</code>: CRC Error. The computed CRC value differs from the received CRC.</li> <li><code>l1_rx_error[0]</code>: Malformed packet. The packet is terminated with a non-terminate control character. When this bit is asserted, <code>l1_rx_error[1]</code> is also asserted.</li> </ul> |
| <code>l1_rxstatus_valid</code>      | Output    | When asserted, indicates that <code>l1_rxstatus_data</code> is driving valid data.   |
| <code>l1_rxstatus_data[39:0]</code> | Output    | Specifies information about the received frame. The following fields are defined: <ul style="list-style-type: none"> <li>Bit[39]: When asserted, indicates a PFC frame</li> <li>Bit[38]: When asserted, indicates a unicast frame</li> <li>Bit[37]: When asserted, indicates a multicast frame</li> <li>Bit[36]: When asserted, indicates a broadcast frame</li> <li>Bit[35]: When asserted, indicates a pause frame</li> <li>Bit[34]: When asserted, indicates a control frame</li> <li>Bit[33]: When asserted, indicates a VLAN frame</li> </ul>   |

*continued...*

| Signal                     | Direction | Description  |
|----------------------------|-----------|--|
|                            |           | <ul style="list-style-type: none"> <li>Bit[32]: When asserted, indicates a stacked VLAN frame</li> <li>Bits[31:16]: Specifies the frame length from the first byte of the destination address to the last byte of the FCS</li> <li>Bits[15:0]: Specifies the payload length</li> </ul> |
| pause_receive_rx[FCQN-1:0] | Output    | Each bit of pause_receive_rx[FCQN-1:0] indicates that the corresponding queue is being paused.   |

**Figure 36. 25G Ethernet Intel FPGA IP MAC to Client Avalon Streaming Interface**

l1\_rx\_data reception order is highest byte to lowest byte. The first byte of the destination address is on l1\_rx\_data[65:56], 0xfbe4 . . . in this timing diagram.



### Related Information

#### [Avalon Interface Specifications](#)

Detailed information about Avalon streaming interfaces.

## 6.3. Transceivers

The transceivers require a separately instantiated advanced transmit (ATX) PLL to generate the high speed serial clock. In many cases, the same ATX PLL can serve as input to an additional transceiver that has similar input clocking requirements. In comparison to the fractional PLL (fPLL) and clock multiplier unit PLL, the ATX PLL has the best jitter performance and supports the highest frequency operation.

**Table 17. Transceiver Signals**

| Signal         | Direction | Description  |
|----------------|-----------|--|
| tx_serial      | Output    | TX transceiver signal. Each tx_serial bit becomes two physical pins that form a differential pair.   |
| rx_serial      | Input     | RX transceiver signals. Each rx_serial bit becomes two physical pins that form a differential pair.  |
| clk_ref        | Input     | The PLL reference clock. Input to the clock data recovery (CDR) circuitry in the RX PMA. The frequency of this clock is 644.53125 MHz or 322.265625 MHz. |
| tx_serial_clk  | Input     | High speed serial clock driven by the ATX PLL. The frequency of this clock is 12.890625 GHz.   |
| tx_serial_clk0 | Input     | High speed serial clock driven by the ATX PLL for 25G data rate. The frequency of this clock is 12.890625 GHz.   |
| tx_serial_clk1 | Input     | High speed serial clock driven by the ATX PLL for 10G data rate. The frequency of this clock is 5.15625 GHz.   |
| tx_pll_locked  | Input     | Lock signal from ATX PLL. Indicates all ATX PLL(s) are locked.   |

**Note:**

1. The integrated transceivers supports adaptation mode by setting the RX PMA Adaptation Mode parameter in the internal generated transceiver IP to **Adaptive CTLE, Adaptive VGA, All-Tap Adaptive DFE** mode. Refer to the *Analog PMA Settings Parameters* and *RX PMA Use Model* sections of the *L- and H-Tile Transceiver PHY User Guide* for more information.
2. Intel Stratix 10 devices use the `OSC_CLK_1` pin to provide the transceiver calibration clock source. You must provide a 25, 100, or 125 MHz free running and stable clock to `OSC_CLK_1`. The FPGA's **Internal Oscillator** cannot be used for transceiver calibration. Do not select this clock source as the **Configuration clock source** in the Intel Quartus Prime settings. For more information, refer to the *Calibration* section of the *L- and H-Tile Transceiver PHY User Guide*.

**Related Information**

- [Adding the Transceiver PLL on page 22](#)
- [Ethernet section of the L- and H-Tile Transceiver PHY User Guide](#)  
Provides more information about the PMA and PCS for Ethernet protocols.
- [Analog PMA Settings Parameters section of the L- and H-Tile Transceiver PHY User Guide](#)
- [RX PMA Use Model section of the L- and H-Tile Transceiver PHY User Guide](#)
- [Calibration section of the L- and H-Tile Transceiver PHY User Guide](#)

## 6.4. Transceiver Reconfiguration Signals

You access the transceiver control and status registers using the transceiver reconfiguration interface. This is an Avalon memory-mapped interface.

The Avalon memory-mapped interface implements a standard memory-mapped protocol. You can connect an Avalon master to this bus to access the registers of the embedded Transceiver PHY IP core.

**Table 18. Reconfiguration Interface Ports with Shared Native PHY Reconfiguration Interface**

All interface signals are clocked by the `reconfig_clk` clock.

| Port Name                             | Direction | Description  |
|---------------------------------------|-----------|--|
| <code>reconfig_clk</code>             | Input     | Avalon clock. The clock frequency is 100-125 MHz. <b>All signals transceiver reconfiguration interface signals are synchronous to <code>reconfig_clk</code>.</b>                         |
| <code>reconfig_reset</code>           | Input     | Resets the Avalon memory-mapped interface and all of the registers to which it provides access.  |
| <code>reconfig_write</code>           | Input     | Write enable signal. Signal is active high.  |
| <code>reconfig_read</code>            | Input     | Read enable signal. Signal is active high.   |
| <code>reconfig_address[10:0]</code>   | Input     | Address bus.   |
| <code>reconfig_writedata[31:0]</code> | Input     | A 32-bit data write bus. <code>reconfig_address</code> specifies the address.  |
| <code>reconfig_readdata[31:0]</code>  | Output    | A 32-bit data read bus. Drives read data from the specified address. Signal is valid after <code>reconfig_waitrequest</code> is deasserted.  |
| <code>reconfig_waitrequest</code>     | Output    | Indicates the Avalon memory-mapped interface is busy. Keep the <code>reconfig_write</code> or <code>reconfig_read</code> asserted until <code>reconfig_waitrequest</code> is deasserted. |

### 6.4.1. Accessing the Native PHY Registers in H-Tile Devices

For Intel Stratix 10 H-tile production devices, disable the background calibration first prior to accessing the transceiver core reconfiguration register. The Intel Stratix 10 H-tile ES devices do not have background calibration.

In Intel Quartus Prime software version 19.2 onwards, use the following steps to access the transceiver core reconfiguration registers:

1. Write 0x1 into register 0x343[0] of the Avalon memory-mapped control and status interface to hold the auto adaptation module in an Idle state. If you have disabled the **Enable auto adaptation triggering for RX PMA CTLE/DFE mode** parameter, you can skip this step.
2. Write 0x0 into register 0x542[0] of the transceiver control and status registers using the transceiver reconfiguration Avalon memory-mapped interface to disable background calibration.
3. Access the transceiver register, for example, to perform the transceiver reconfiguration.
4. Once completed, write 0x1 into register 0x542[0] of the transceiver control and status registers using the transceiver reconfiguration Avalon memory-mapped interface to enable background calibration.
5. Write 0x0 into register 0x343[0] of the Avalon memory-mapped control and status interface to release the auto adaptation module from the Idle state. If you have disabled the **Enable auto adaptation triggering for RX PMA CTLE/DFE mode** parameter, you can skip this step.

*Note:* If you do not select the **Enable auto adaptation triggering for RX PMA CTLE/DFE mode** parameter, refer to *Adaptation Control - Start* section of the *Intel Stratix 10 L- and H-Tile Transceiver PHY User Guide* for more information about how to start adaptation.

### 6.4.2. Accessing the Native PHY Registers in L-Tile Devices

All variants of Intel Stratix 10 L-tile devices (ES and production) do not have background calibration. If the **Enable auto adaptation triggering for RX PMA CTLE/DFE mode** option is enabled, the auto adaptation module FSM needs to be held in IDLE state prior to accessing the transceiver core reconfiguration register. If the **Enable auto adaptation triggering for RX PMA CTLE/DFE mode** option is disabled, skip the steps below.

In Intel Quartus Prime software version 20.2 onwards, follow these steps to access the transceiver core reconfiguration registers:

1. Write 0x1 into register 0x343[0] of the memory-mapped control and status interface to hold the auto adaptation module in an idle state.
2. Access the transceiver register, for example, to perform the transceiver reconfiguration.
3. Once completed, write 0x0 into register 0x343[0] of the Avalon memory-mapped control and status interface to release the auto adaptation module.

*Note:* If you do not select the **Enable auto adaptation triggering for RX PMA CTLE/DFE mode** parameter, refer to *Adaptation Control - Start* section of the *L- and H-Tile Transceiver PHY User Guide* for more information about how to start adaptation.

**Related Information**

[L- and H-Tile Transceiver PHY User Guide](#)



## 6.5. Avalon Memory-Mapped Management Interface

You access control and status registers using an Avalon memory-mapped management interface. The interface responds regardless of the link status. It also responds when the IP core is in a reset state driven by any reset signal or soft reset other than the `csr_rst_n` signal. Asserting the `csr_rst_n` signal resets all control, status, and statistics registers; while this reset is in process, the Avalon memory-mapped management interface does not respond.

**Note:** This interface cannot handle multiple pending read transfers. Despite the presence of the `status_readdata_valid` signal, this Avalon memory-mapped interface is non-pipelined with variable latency.

**Table 19. Avalon Memory-Mapped Management Interface**

**Note:** All `status_*` signals are synchronous to `clk_status` signal.

| Signal                              | Direction | Description   |
|-------------------------------------|-----------|---|
| <code>clk_status</code>             | Input     | The clock that drives the control and status registers. The frequency of this clock is 100 MHz.   |
| <code>reset_status</code>           | Input     | Connect this signal to 1'b0. This signal remains visible as a top-level signal for backward compatibility.  |
| <code>status_addr[15:0]</code>      | Input     | Drives the Avalon memory-mapped register address.   |
| <code>status_read</code>            | Input     | When asserted, specifies a read request.  |
| <code>status_write</code>           | Input     | When asserted, specifies a write request.   |
| <code>status_readdata[31:0]</code>  | Output    | Drives read data. Valid when <code>status_readdata_valid</code> is asserted.  |
| <code>status_readdata_valid</code>  | Output    | When asserted, indicates that <code>status_read_data[31:0]</code> is valid.   |
| <code>status_writedata[31:0]</code> | Input     | Drives the write data. The packet can end at any byte position. The empty bytes are the low-order bytes.  |
| <code>status_waitrequest</code>     | Output    | Indicates that the control and status interface is not ready to complete the transaction. <code>status_waitrequest</code> is only used for read transactions. |

### Related Information

- [Control, Status, and Statistics Register Descriptions](#) on page 74  
Information about the 25G Ethernet Intel FPGA IPcore registers you can access through the Avalon Memory-Mapped management interface.
- [Typical Read and Write Transfers](#) section in the *Avalon Interface Specifications*  
Describes typical Avalon memory-mapped read and write transfers with a slave-controlled waitrequest signal.

## 6.6. PHY Interface Signals

**Table 20. Signals of the PHY Interface**

This table lists the PHY interface signals. These interface signals are only applicable to the 25G Ethernet Intel FPGA IP with MAC and PCS core variant. For more information on these interface signals, refer to the *Intel Stratix 10 L- and H-Tile Transceiver PHY User Guide*.

| Signal                     | Direction | Description   |
|----------------------------|-----------|---|
| tx_clkout                  | Input     | When you turn on <b>Enable IEEE 1588</b> , this clock is a TX transceiver parallel clock. The frequency of this clock is 402.832 MHz for 25G data rate and 161.132 MHz for 10G rate.<br>When <b>Enable IEEE 1588</b> is turned off, this clock is used by the MAC transmitter (TX MAC). The frequency of this clock is 390.625 MHz for 25G data rate and 156.25 MHz for 10G data rate.  |
| tx_clkout2                 | Input     | Clock for the TX MAC when you turn on <b>Enable IEEE 1588</b> . The frequency of this clock is 390.625 MHz for 25G data rate and 156.25 MHz for 10G data rate.<br>This port is unused when <b>Enable IEEE 1588</b> is turned off.   |
| rx_clkout                  | Input     | When you turn on <b>Enable IEEE 1588</b> , this clock is a RX transceiver parallel clock. The frequency of this clock is 402.832 MHz for 25G data rate and 161.132 MHz for 10G rate.<br>When <b>Enable IEEE 1588</b> is turned off, this clock is used by the MAC receiver (RX MAC). The frequency of this clock is 390.625 MHz for 25G data rate and 156.25 MHz for 10G data rate.   |
| rx_clkout2                 | Input     | Clock for the RX MAC when you turn on <b>Enable IEEE 1588</b> . The frequency of this clock is 390.625 MHz for 25G data rate and 156.25 MHz for 10G data rate.<br>This port is unused when <b>Enable IEEE 1588</b> is turned off.   |
| rvalid                     | Input     | Indication for RX valid data.   |
| tvalid_phy                 | Output    | Indicates valid data output towards PHY.  |
| tx_parallel_data_phy[63:0] | Output    | TX parallel data output from the FPGA fabric to PHY.  |
| tx_control_phy[1:0]        | Output    | TX control character output from the FPGA fabric to PHY.<br>When you turn on <b>Enable RS-FEC</b> , the tx_control_phy does not transmit the control character to link partner. The 66-bits output from the RS-FEC is split into tx_parallel_data_phy[63:0] and tx_control_phy[1:0], where the tx_control_phy[1:0] is the most upper 2-bits of the 66-bits data bus, for example, {tx_control_phy[1:0], tx_parallel_data_phy[63:0]}.<br>For details about the TX RS-FEC, refer to the <i>TX RS-FEC</i> section. |
| rx_parallel_data_phy[63:0] | Input     | RX parallel data input from the PHY to FPGA fabric.   |
| rx_control_phy[1:0]        | Input     | RX control character input from the PHY to FPGA fabric.<br>When you turn on <b>Enable RS-FEC</b> , the rx_control_phy does not receive the control character from link partner. The 66-bits input to the RS-FEC is split into rx_parallel_data_phy[63:0] and rx_control_phy[1:0], where the rx_control_phy[1:0] is the most upper 2-bits of the 66-bits data bus, for example, {rx_control_phy[1:0], rx_parallel_data_phy[63:0]}.<br>For details about the RX RS-FEC, refer to the <i>RX RS-FEC</i> section.    |
| tx_fifo_latency_pulse      | Input     | Latency pulse for TX Core FIFO.   |
| tx_pcs_fifo_latency_pulse  | Input     | Latency pulse for TX PCS FIFO.  |
| rx_fifo_latency_pulse      | Input     | Latency pulse for RX Core FIFO.   |

*continued...*

| Signal                    | Direction | Description   |
|---------------------------|-----------|---|
| rx_pcs_fifo_latency_pulse | Input     | Latency pulse for RX PCS FIFO.                          |
| rx_bitslip                | Output    | Indicates bit slip enable status.                       |
| rx_digitalreset           | Input     | Resets the PCS RX portion of the transceiver PHY.       |
| tx_digitalrest            | Input     | Resets the PCS TX portion of the transceiver PHY.       |
| rx_is_lockedtoata         | Input     | Indicates the status of RX CDR lock on data.            |
| rx_set_locktoeref         | Output    | Indicates the status of RX CDR lock to reference clock. |
| rx_serialpbken            | Output    | Status for Internal Serial Loopback.                    |
| tx_ready                  | Input     | Indication for external PMA TX Ready.                   |
| rx_ready                  | Input     | Indication for external PMA RX Ready                    |
| phy_reset                 | Output    | Reset signal for PHY.                                   |
| tx_empty_phy              | Input     | Indication for TX Core FIFO empty.                      |
| tx_pempty_phy             | Input     | Indication for TX Core FIFO partially empty.            |
| tx_full_phy               | Input     | Indication for TX Core FIFO full.                       |
| tx_pfull_phy              | Input     | Indication for TX Core FIFO partially full.             |
| rx_empty_phy              | Input     | Indication for RX Core FIFO empty.                      |
| rx_pempty_phy             | Input     | Indication for RX Core FIFO partially empty.            |
| rx_full_phy               | Input     | Indication for RX Core FIFO full.                       |
| rx_pfull_phy              | Input     | Indication for RX Core FIFO partially full.             |

#### Related Information

- [TX RS-FEC](#) on page 34
- [RX RS-FEC](#) on page 41
- [Intel Stratix 10 L- and H-Tile Transceiver PHY User Guide](#)

## 6.7. 1588 PTP Interface Signals

**Table 21. Signals of the 1588 Precision Time Protocol Interface**

Signals are clocked by `clk_rxmac` or `clk_txmac`, as specified. All 64-bit output signals are in the Intel 64-bit TOD format, and you are expected to drive all 64-bit input signals in this format.

| Signal Name                        | Direction | Description   |
|------------------------------------|-----------|---|
| latency_sclk                       | Input     | Latency measurement input sampling clock.<br>For 25G Ethernet Intel FPGA IP with the IEEE 1588v2 feature, Intel recommends that the frequency of this clock is set to 156.25 MHz. Refer to <i>25G Ethernet Intel Stratix 10 FPGA IP Design Example User Guide</i> and <i>L- and H-Tile Transceiver PHY User Guide</i> for more details. |
| <b>PTP Interface to TOD module</b> |           |   |
| tx_time_of_day_96b_data[95:0]      | Input     | Current V2-format (96-bit) TOD in <code>clk_txmac</code> clock domain. Connect this signal to the external TOD module.  |

*continued...*

| Signal Name                                     | Direction | Description  |
|---|-----------|--|
|   |           | This signal is available only if you set the <b>Time of day format</b> parameter to the value of <b>Enable 96-bit timestamp format</b> or <b>Enable both formats</b> .   |
| tx_time_of_day_64b_data[63:0]                   | Input     | Current 64-bit TOD in clk_txmac clock domain. Connect this signal to the external TOD module.<br>This signal is available only if you set the <b>Time of day format</b> parameter to the value of <b>Enable 64-bit timestamp format</b> or <b>Enable both formats</b> .  |
| rx_time_of_day_96b_data[95:0]                   | Input     | Current V2-format (96-bit) TOD in clk_rxmac clock domain. Connect this signal to the external TOD module.<br>This signal is available only if you set the <b>Time of day format</b> parameter to the value of <b>Enable 96-bit timestamp format</b> or <b>Enable both formats</b> .  |
| rx_time_of_day_64b_data[63:0]                   | Input     | Current 64-bit TOD in clk_rxmac clock domain. Connect this signal to the external TOD module.<br>This signal is available only if you set the <b>Time of day format</b> parameter to the value of <b>Enable 64-bit timestamp format</b> or <b>Enable both formats</b> .  |
| <b>PTP Interface to Client</b>                  |           |  |
| TX Signals Related to One Step Processing       |           |  |
| tx_etstamp_ins_ctrl_timestamp_insert            | Input     | Indicates the current packet on the TX client interface is a 1588 PTP packet, and directs the IP core to process the packet in one-step processing insertion mode. In this mode, the IP core overwrites the timestamp of the packet with the timestamp field when the packet appears on the TX Ethernet link.<br>The TX client must assert and deassert this signal synchronously with the TX SOP signal for the 1588 PTP packet.<br>If the TX client asserts this signal simultaneously with tx_etstamp_ins_ctrl_residence_time_update, the results are undefined.                                      |
| tx_etstamp_ins_ctrl_residence_time_update       | Input     | Indicates the current packet on the TX client interface is a 1588 PTP packet, and directs the IP core to process the packet in one-step processing correction mode. In this mode, the IP core adds the latency through the IP core (residence time) to the current contents of the timestamp field.<br>The TX client must assert and deassert this signal synchronously with the TX SOP signal for the 1588 PTP packet.<br>If the TX client asserts this signal simultaneously with either of tx_etstamp_ins_ctrl_timestamp_insert or tx_egress_timestamp_request_valid, the results are undefined.      |
| tx_etstamp_ins_ctrl_ingress_timestamp_96b[95:0] | Input     | Indicates the V2-format TOD when the packet entered the system.<br>The TX client must ensure this signal is valid in each TX SOP cycle when it asserts tx_etstamp_ins_ctrl_residence_time_update. The TX client must maintain the desired value on this signal while the TX SOP signal is asserted. This signal is useful only in transparent clock mode when the TX client asserts tx_etstamp_ins_ctrl_residence_time_update.<br>This signal is available only if you set the <b>Time of day format</b> parameter to the value of <b>Enable 96-bit timestamp format</b> or <b>Enable both formats</b> . |
| tx_etstamp_ins_ctrl_ingress_timestamp_64b[63:0] | Input     | Indicates the TOD (in Intel 64-bit format) when the packet entered the system.<br>The TX client must ensure this signal is valid in each TX SOP cycle when it asserts tx_etstamp_ins_ctrl_residence_time_update. The TX client must maintain the desired value on this signal while the TX SOP   |
| <i>continued...</i>                             |           |  |

| Signal Name  | Direction | Description   |
|--|-----------|---|
|  |           | <p>signal is asserted. This signal is useful only in transparent clock mode when the TX client asserts <code>tx_etstamp_ins_ctrl_residence_time_update</code>.</p> <p>This signal is available only if you set the <b>Time of day format</b> parameter to the value of <b>Enable 64-bit timestamp format</b> or <b>Enable both formats</b>.</p>   |
| <code>tx_etstamp_ins_ctrl_timestamp_format</code>              | Input     | <p>Specifies the timestamp format (V1 or V2 format) for the current packet if the TX client simultaneously asserts <code>tx_etstamp_ins_ctrl_timestamp_insert</code>. Values are:</p> <ul style="list-style-type: none"> <li>1'b0: 96-bit timestamp format (V2)</li> <li>1'b1: 64-bit timestamp format (V1)</li> </ul> <p>The TX client must maintain the desired value on this signal while the TX SOP signal is asserted.</p> <p>If the client specifies the V1 format, you read and write the V1 format TOD (32 bits of seconds and 32 bits of nanoseconds) in bits [79:16] of the 96-bit timestamp and TOD signals.</p> <p><i>Note:</i> If you set the <b>Time of day format</b> parameter to the value of <b>Enable 64-bit timestamp format</b>, the results of asserting <code>tx_etstamp_ins_ctrl_timestamp_insert</code> are undefined. Therefore, the timestamp in any case maps to the 96-bit signals.</p>  |
| <code>tx_etstamp_ins_ctrl_residence_time_calc_format</code>    | Input     | <p>Specifies the TOD format (64-bit TOD format or the V2 96-bit TOD format) for the current packet if the TX client simultaneously asserts <code>tx_etstamp_ins_ctrl_residence_time_update</code>. Values are:</p> <ul style="list-style-type: none"> <li>1'b0: 96-bit TOD format (V2)</li> <li>1'b1: 64-bit TOD format (48-bit ns and 16-bit fns)</li> </ul> <p>The TX client must maintain the desired value on this signal while the TX SOP signal is asserted.</p> <p>If you set the <b>Time of day format</b> parameter to the value of <b>Enable 64-bit timestamp format</b> or <b>Enable both formats</b>, and the client specifies the 64-bit TOD format, the IP core uses the 64-bit TOD format for residence time calculation.</p> <p>If you set the <b>Time of day format</b> parameter to the value of <b>Enable 64-bit timestamp format</b> and the client specifies the 96-bit format (V2), the results are undefined.</p>  |
| <code>tx_etstamp_ins_ctrl_offset_timestamp[15:0]</code>        | Input     | <p>Specifies the byte offset of the timestamp information in the current packet if the TX client simultaneously asserts <code>tx_etstamp_ins_ctrl_timestamp_insert</code>. The IP core overwrites the value at this offset. The TX client must maintain the desired value on this signal while the TX SOP signal is asserted.</p> <p>If the packet supports V2 format, the timestamp has 96 bits. In this case, the IP core inserts ten bytes (bits [95:16]) of the timestamp at this offset and the remaining two bytes (bits [15:0]) of the timestamp at the offset specified in <code>tx_etstamp_ins_ctrl_offset_correction_field</code>.</p> <p>The TX client must ensure that:</p> <ul style="list-style-type: none"> <li>The offset includes the entire timestamp in a single packet.</li> <li>If the packet is more than 256 bytes, the offset supports inclusion of the entire timestamp in the first 256 bytes of the packet.</li> <li>The timestamp bytes do not overlap with the bytes in any other field, including the UDP checksum field. (If these particular two fields overlap, the result is undefined).</li> </ul> |
| <code>tx_etstamp_ins_ctrl_offset_correction_field[15:0]</code> | Input     | <p>If the TX client simultaneously asserts <code>tx_etstamp_ins_ctrl_residence_time_update</code>, this signal specifies the byte offset of the correction field in the current packet.</p> <p>If the TX client simultaneously asserts <code>tx_etstamp_ins_ctrl_timestamp_insert</code> and deasserts (sets to the value of 0) the <code>tx_etstamp_ins_ctrl_timestamp_format</code> signal, this signal specifies the byte offset of bits [15:0] of the timestamp.</p>  |

continued...

| Signal Name  | Direction | Description  |
|--|-----------|--|
|  |           | <p>The TX client must maintain the desired value on this signal while the TX SOP signal is asserted.</p> <p>In addition, the TX client must ensure that:</p> <ul style="list-style-type: none"> <li>• The offset includes the entire correction field or timestamp in a single packet.</li> <li>• If the packet is more than 256 bytes, the offset supports inclusion of the entire timestamp or correction field in the first 256 bytes of the packet.</li> <li>• The correction field or timestamp bytes do not overlap with the bytes in any other field, including the UDP checksum field. (If these particular two fields overlap, the result is undefined).</li> </ul>   |
| tx_etstamp_ins_ctrl_checksum_zero                    | Input     | <p>The TX client asserts this signal during a TX SOP cycle to tell the IP core to zero the UDP checksum in the current packet.</p> <p>If the TX client asserts the tx_etstamp_ins_ctrl_checksum_correct signal, it cannot assert this signal. This signal is meaningful only in one-step clock mode.</p> <p>A zeroed UDP checksum indicates the checksum value is not necessarily correct. This information is useful to tell the application to skip checksum checking of UDP IPv4 packets. This function is illegal for UDP IPv6 packets.</p>  |
| tx_etstamp_ins_ctrl_offset_checksum_field[15:0]      | Input     | <p>Indicates the byte offset of the UDP checksum in the current packet. The TX client must ensure this signal has a valid value during each TX SOP cycle when it also asserts the tx_etstamp_ins_ctrl_checksum_zero signal. Holds the byte offset of the two bytes in the packet that the IP core should reset. This signal is meaningful only in one-step clock mode.</p> <p>The TX client must ensure that:</p> <ul style="list-style-type: none"> <li>• The offset includes the entire checksum in a single packet.</li> <li>• The checksum bytes do not overlap with the bytes in any other field, including the timestamp bytes. (If these particular two fields overlap, the result is undefined).</li> </ul>  |
| tx_etstamp_ins_ctrl_checksum_correct                 | Input     | <p>The TX client asserts this signal during a TX SOP cycle to tell the IP core to update (correct) the UDP checksum in the current packet.</p> <p>If the TX client asserts the tx_etstamp_ins_ctrl_checksum_zero signal, it cannot assert this signal. This signal is meaningful only in one-step clock mode.</p> <p>The application must assert this signal for correct processing of UDP IPv6 packets.</p>   |
| tx_etstamp_ins_ctrl_offset_checksum_correction[15:0] | Input     | <p>Indicates the byte offset of the UDP checksum correction field in the current packet represented by the extended bytes before CRC. The TX client must ensure this signal has a valid value during each TX SOP cycle when it also asserts the tx_etstamp_ins_ctrl_checksum_correct signal. Holds the byte offset of the two bytes in the packet that the IP core should correct. This signal is meaningful only in one-step clock mode.</p> <p>The TX client must ensure that:</p> <ul style="list-style-type: none"> <li>• The offset and length of the checksum correction field includes the entire two bytes of the checksum correction field in a single packet.</li> <li>• The checksum bytes do not overlap with the bytes in any other field, including the timestamp bytes. (If these particular two fields overlap, the result is undefined).</li> <li>• The end of the UDP payload of the PTP packet is extended by 2 bytes. The MAC function modifies the extended bytes to ensure that the UDP checksum remains uncompromised.</li> </ul> |
| tx_egress_asymmetry_update                           | Input     | <p>Indicates the IP core should include the value in the TX_PTP_ASYM_DELAY register in its correction calculations. The TX client must maintain the desired value on this signal while the TX SOP signal is asserted.</p> <p>This option is useful in one-step correction mode.</p>  |
| <i>continued...</i>                                  |           |  |

| Signal Name   | Direction | Description   |
|---|-----------|---|
| TX Signals Related to Two Step Processing   |           |   |
| tx_egress_timestamp_request_valid   | Input     | Indicates the current packet on the TX client interface is a 1588 PTP packet, and directs the IP core to process the packet in two-step processing mode. In this mode, the IP core outputs the timestamp of the packet when it exits the IP core, and does not modify the packet timestamp information.<br>The TX client must assert and deassert this signal synchronously with the TX SOP signal for the 1588 PTP packet.<br>If the TX client asserts this signal simultaneously with tx_etstamp_ins_ctrl_residence_time_update, the results are undefined. |
| tx_egress_timestamp_96b_data[95:0]  | Output    | Provides the V2-format timestamp when a 1588 PTP frame begins transmission on the Ethernet link. Value is valid when the tx_egress_timestamp_96b_valid signal is asserted. This signal is meaningful only in two-step clock mode.<br>This signal is available only if you set the <b>Time of day format</b> parameter to the value of <b>Enable 96-bit timestamp format</b> or <b>Enable both formats</b> .   |
| tx_egress_timestamp_96b_valid   | Output    | Indicates that the tx_egress_timestamp_96b_data and tx_egress_timestamp_96b_fingerprint signals are valid in the current clk_txmac clock cycle. This signal is meaningful only in two-step clock mode.<br>This signal is available only if you set the <b>Time of day format</b> parameter to the value of <b>Enable 96-bit timestamp format</b> or <b>Enable both formats</b> .  |
| tx_egress_timestamp_64b_data[63:0]  | Output    | Provides the timestamp when a 1588 PTP frame begins transmission on the Ethernet link. Value is valid when the tx_egress_timestamp_64b_valid signal is asserted. This signal is meaningful only in two-step clock mode.<br>This signal is available only if you set the <b>Time of day format</b> parameter to the value of <b>Enable 64-bit timestamp format</b> or <b>Enable both formats</b> .   |
| tx_egress_timestamp_64b_valid   | Output    | Indicates that the tx_egress_timestamp_64b_data and tx_egress_timestamp_64b_fingerprint signals are valid in the current clk_txmac clock cycle. This signal is meaningful only in two-step clock mode.<br>This signal is available only if you set the <b>Time of day format</b> parameter to the value of <b>Enable 64-bit timestamp format</b> or <b>Enable both formats</b> .  |
| tx_egress_timestamp_request_fingerprint[(W-1):0]<br>where W is the value between 1 and 32, inclusive, that you specify for the <b>Fingerprint width</b> parameter | Input     | Fingerprint of the current packet.<br>The TX client must assert and deassert this signal synchronously with the TX SOP signal for the 1588 PTP packet.  |
| tx_egress_timestamp_96b_fingerprint[(W-1):0]<br>where W is the value between 1 and 32, inclusive, that you specify for the <b>Fingerprint width</b> parameter     | Output    | Provides the fingerprint of the 1588 PTP frame currently beginning transmission on the Ethernet link. Value is valid when the tx_egress_timestamp_96b_valid signal is asserted.<br>This signal is available only if you set the <b>Time of day format</b> parameter to the value of <b>Enable 96-bit timestamp format</b> or <b>Enable both formats</b> .   |
| tx_egress_timestamp_64b_fingerprint[(W-1):0]<br>where W is the value between 1 and 32, inclusive, that you specify for the <b>Fingerprint width</b> parameter     | Output    | Provides the fingerprint of the 1588 PTP frame currently beginning transmission on the Ethernet link. Value is valid when the tx_egress_timestamp_64b_valid signal is asserted.<br>This signal is available only if you set the <b>Time of day format</b> parameter to the value of <b>Enable 64-bit timestamp format</b> or <b>Enable both formats</b> .   |
| <i>continued...</i>   |           |   |

| Signal Name                         | Direction | Description  |
|-------------------------------------|-----------|--|
| RX Signals                          |           |  |
| rx_ingress_timestamp_96b_data[95:0] | Output    | Whether or not the current packet on the RX client interface is a 1588 PTP packet, indicates the V2-format timestamp when the IP core received the packet on the Ethernet link. The IP core provides a valid value on this signal in the same cycle it asserts the RX SOP signal for 1588 PTP packets.<br>This signal is available only if you set the <b>Time of day format</b> parameter to the value of <b>Enable 96-bit timestamp format</b> or <b>Enable both formats</b> .                 |
| rx_ingress_timestamp_96b_valid      | Output    | Indicates that the rx_ingress_timestamp_96b_data signal is valid in the current cycle. This signal is redundant with the RX SOP signal for 1588 PTP packets.<br>This signal is available only if you set the <b>Time of day format</b> parameter to the value of <b>Enable 96-bit timestamp format</b> or <b>Enable both formats</b> .   |
| rx_ingress_timestamp_64b_data[63:0] | Output    | Whether or not the current packet on the RX client interface is a 1588 PTP packet, indicates the 64-bit TOD (in Intel 64-bit format) when the IP core received the packet on the Ethernet link. The IP core provides a valid value on this signal in the same cycle it asserts the RX SOP signal for 1588 PTP packets.<br>This signal is available only if you set the <b>Time of day format</b> parameter to the value of <b>Enable 64-bit timestamp format</b> or <b>Enable both formats</b> . |
| rx_ingress_timestamp_64b_valid      | Output    | Indicates that the rx_ingress_timestamp_64b_data signal is valid in the current cycle. This signal is redundant with the RX SOP signal for 1588 PTP packets.<br>This signal is available only if you set the <b>Time of day format</b> parameter to the value of <b>Enable 64-bit timestamp format</b> or <b>Enable both formats</b> .   |

**Related Information**

- [1588 PTP Registers](#) on page 92
- [25G Ethernet Intel Stratix 10 FPGA IP Design Example User Guide](#)
- [L- and H-Tile Transceiver PHY User Guide](#)

## 6.8. Miscellaneous Status and Debug Signals

The miscellaneous status and debug signals are asynchronous.

**Table 22. Miscellaneous Status and Debug Signals**

| Signal              | Direction | Description  |
|---------------------|-----------|--|
| tx_lanes_stable     | Output    | Asserted when all TX lanes are stable and ready to transmit data.  |
| rx_block_lock       | Output    | Signal is asserted when 64B/66B sync header is found consecutively for at least 64 clock cycles by the RX PCS.   |
| rx_am_lock          | Output    | If you turn on <b>Enable RS-FEC</b> in the parameter editor, this signal is asserted when alignment marker lock status is achieved.<br>If you turn off <b>Enable RS-FEC</b> in the parameter editor, this signal behaves the same as the rx_block_lock signal. |
| rx_pcs_ready        | Output    | Signal is asserted when rx_block_lock is asserted.   |
| local_fault_status  | Output    | Asserted when the RX MAC detects a local fault. This signal is available if you turn on <b>Enable link fault generation</b> in the parameter editor.   |
| <i>continued...</i> |           |  |



| Signal              | Direction | Description  |
|---------------------|-----------|--|
| remote_fault_status | Output    | Asserted when the RX MAC detects a remote fault. This signal is available if you turn on <b>Enable link fault generation</b> in the parameter editor.                              |
| unidirectional_en   | Output    | Asserted if the IP core includes <i>Clause 66</i> for unidirectional support. This signal is available if you turn on <b>Enable link fault generation</b> in the parameter editor. |
| link_fault_gen_en   | Output    | Asserted if the IP core includes <i>Clause 66</i> for unidirectional support. This signal is available if you turn on <b>Enable link fault generation</b> in the parameter editor. |

### Related Information

[Debugging the Link](#) on page 96

## 6.9. Reset Signals

The IP core has three external hard reset inputs. These resets are asynchronous and are internally synchronized. Assert resets for ten cycles or until you observe the effect of their specific reset. Asserting the external hard reset `csr_rst_n` returns control and status registers to their original values. `rx_pcs_ready` and `tx_lanes_stable` are asserted when the IP core has exited reset successfully.

**Table 23. Reset Signals**

| Signal        | Direction | Description   |
|---------------|-----------|---|
| tx_rst_n      | Input     | Active low hard reset signal. Resets the TX interface, including the TX PCS and TX MAC. This reset leads to the deassertion of the <code>tx_lanes_stable</code> output signal.  |
| rx_rst_n      | Input     | Active low hard reset signal. Resets the RX interface, including the RX PCS and RX MAC. This reset leads to the deassertion of the <code>rx_pcs_ready</code> output signal.   |
| csr_rst_n     | Input     | Active low hard global reset. Resets the full IP core. Resets the TX MAC, RX MAC, TX PCS, RX PCS, adapters, transceivers, and control, status, and statistic registers. This reset leads to the deassertion of the <code>tx_lanes_stable</code> and <code>rx_pcs_ready</code> output signals. |
| channel_reset | Input     | This port is only present if the <b>Enable 10G/25G Dynamic Rate Switching</b> parameter is enabled. Before initiating reconfiguration between speeds, assert this signal to hold the TX or RX data paths in reset.  |

## 7. Control, Status, and Statistics Register Descriptions

This section provides information about the memory-mapped registers. You access these registers using the IP core Avalon memory-mapped control and status interface. The registers use 32-bit addresses; they are not byte addressable.

Write operations to a read-only register field have no effect. Read operations that address a Reserved register return an unspecified result. Write operations to Reserved registers have no effect. Accesses to registers that do not exist in your IP core variation, or to register bits that are not defined in your IP core variation, have an unspecified result. You should consider these registers and register bits Reserved. Although you can only access registers in 32-bit read and write operations, you should not attempt to write or ascribe meaning to values in undefined register bits.

**Table 24. Register Base Addresses**

| Word Offset | Register Type                                   |
|-------------|---|
| 0x300-0x3FF | PHY registers                                   |
| 0x400-0x4FF | TX MAC registers                                |
| 0x500-0x5FF | RX MAC registers                                |
| 0x600-0x708 | Pause and Priority-Based Flow Control registers |
| 0x800-0x8FF | Statistics Counter registers - TX direction     |
| 0x900-0x9FF | Statistics Counter registers - RX direction     |
| 0xA00-0xAFF | TX 1588 PTP registers                           |
| 0xB00-0xBFF | RX 1588 PTP registers                           |
| 0xC00-0xCFF | TX Reed-Solomon FEC registers                   |
| 0xD00-0xDFF | RX Reed-Solomon FEC registers                   |

- Note:**
- Do not attempt to access any register address that is Reserved or undefined. Accesses to registers that do not exist in your IP core variation have unspecified results.
  - For Intel Stratix 10 H-tile production device, disable the background calibration prior to accessing the transceiver core reconfiguration register, as described in the *Disabling Background Calibration* section of this user guide.

### Related Information

- [Avalon Memory-Mapped Management Interface](#) on page 65  
Interface to access the 25G Ethernet Intel FPGA IP core registers.
- [Accessing the Native PHY Registers in H-Tile Devices](#) on page 63
- [Accessing the Native PHY Registers in H-Tile Devices](#) on page 63
- [Accessing the Native PHY Registers in L-Tile Devices](#) on page 63

## 7.1. PHY Registers

**Table 25. PHY Registers**

The global hard reset `csr_rst_n` resets all of these registers. The TX reset `tx_rst_n` and RX reset `rx_rst_n` signals do not reset these registers.

| Addr  | Name         | Description  | Reset                               | Access |
|-------|--------------|--|-------------------------------------|--------|
| 0x300 | REVID        | IP core PHY module revision ID   | 0x0504 2018                         | RO     |
| 0x301 | SCRATCH      | Scratch register available for testing   | 0x0000 0000                         | RW     |
| 0x302 | PHY_NAME_0   | First characters of IP core variation identifier string, "0025". The "00" is unprintable.  | 0x0000 3235                         | RO     |
| 0x303 | PHY_NAME_1   | Next characters of IP core variation identifier string, "00GE". The "00" is unprintable.   | 0x0000 4745                         | RO     |
| 0x304 | PHY_NAME_2   | Final characters of IP core variation identifier string, "0pcs". The "0" is unprintable.   | 0x0070 6373                         | RO     |
| 0x310 | PHY_CONFIG   | PHY configuration registers. The following bit fields are defined: <ul style="list-style-type: none"> <li>Bit[0]: <code>eio_sys_rst</code>. Full system reset (except registers). Set this bit to initiate the internal reset sequence.</li> <li>Bit[1]: <code>soft_txp_rst</code>. TX soft reset. Resets TX PCS, MAC, and adapter.</li> <li>Bit[2]: <code>soft_rxp_rst</code>. RX soft reset. Resets RX PCS, MAC, and adapter.</li> <li>Bit[3]: Reserved.</li> <li>Bit[4]: <code>set_ref_lock</code>. Directs the RX CDR PLL to lock to the reference clock.</li> <li>Bit[5]: <code>set_data_lock</code>. Directs the RX CDR PLL to lock to data.</li> <li>Bits[31:6]: Reserved.</li> </ul> | 26'hX_2'b0_1'bX_3'b0 <sup>(3)</sup> | RW     |
| 0x312 | WORD_LOCK    | When asserted, indicates that the virtual channel has identified 66 bit block boundaries in the serial data stream.  | 31'hX1'b0 <sup>(3)</sup>            | RO     |
| 0x313 | EIO_SLOOP    | Serial PMA loopback. Setting a bit puts the corresponding transceiver in serial loopback mode. In serial loopback mode, the TX lane loops back to the RX lane on an internal loopback path.  | 31'hX1'b0 <sup>(3)</sup>            | RW     |
| 0x314 | EIO_FLAG_SEL | Supports indirect addressing of individual FIFO flags in the PCS Native PHY IP core. Program this register with the encoding for a specific FIFO flag. The flag values (one per transceiver) are then accessible in the <code>EIO_FLAGS</code> register.<br>The value in the <code>EIO_FLAG_SEL</code> register directs the IP core to make available the following FIFO flag: <ul style="list-style-type: none"> <li>3'b000: TX FIFO full</li> <li>3'b001: TX FIFO empty</li> <li>3b010: TX FIFO partially full</li> <li>3'b011: TX FIFO partially empty</li> <li>3b100: RX FIFO full</li> </ul>  | 29'hX3'b0 <sup>(3)</sup>            | RW     |

*continued...*

<sup>(3)</sup> X means "Don't Care".

| Addr  | Name                   | Description  | Reset                     | Access |
|-------|------------------------|--|---------------------------|--------|
|       |                        | <ul style="list-style-type: none"> <li>3b101: RX FIFO empty</li> <li>3b110: RX FIFO partially full</li> <li>3b111: RX FIFO partially empty</li> </ul>  |                           |        |
| 0x315 | EIO_FLAGS              | PCS indirect data. To read a FIFO flag, set the value in the EIO_FLAG_SEL register to indicate the flag you want to read. After you specify the flag in the EIO_FLAG_SEL register, each bit [n] in the EIO_FLAGS register has the value of that FIFO flag for the transceiver channel for lane [n].  | 31'hX1'b0 <sup>(3)</sup>  | RO     |
| 0x321 | EIO_FREQ_LOCK          | Each asserted bit indicates that the corresponding lane RX clock data recovery (CDR) phase-locked loop (PLL) is locked.  | 31'hX1'b0 <sup>(3)</sup>  | RO     |
| 0x322 | PHY_CLK                | The following encodings are defined: <ul style="list-style-type: none"> <li>Bit[0]: Indicates if the TX PCS is ready.</li> <li>Bit[1]: Indicates if the TX MAC PLL is locked.</li> <li>Bit[2]: Indicates if the RX CDR PLL is locked.</li> </ul>   | 29'hX3'b00 <sup>(3)</sup> | RO     |
| 0x323 | FRM_ERR                | The IP core asserts bit [0] if it identifies a frame error. You can read this register to determine if the IP core sustains a low number of frame errors, below the threshold to lose word lock. This bit is sticky, unless the IP core loses word lock. Write 1'b1 to the SCLR_FRM_ERR register to clear. If the IP core loses word lock, it clears this register.                                    | 31'hX1'b0 <sup>(3)</sup>  | RO     |
| 0x324 | SCLR_FRM_ERR           | Synchronous clear for FRM_ERR register. Write 1'b1 to this register to clear the FRM_ERR register and bit [1] of the LANE_DESKEWED register. A single bit clears all sticky framing errors. This bit does not auto-clear. Write a 1'b0 to continue logging frame errors.   | 0x0                       | RW     |
| 0x325 | EIO_RX_SOFT_PURGE_S    | Reserved.  | 0x0000                    | RO     |
| 0x326 | RX_PCS_FULLY_ALIGNED_S | Indicates the RX PCS is fully aligned and ready to accept traffic. <ul style="list-style-type: none"> <li>Bit[0]: RX PCS fully aligned status.</li> <li>Bit[1]: RX PCS bit error rate status. A bit value of 1 indicates a bit error rate higher than 10<sup>-4</sup> or there are at least 16 errors within 50 us. This bit value is only valid when the link fault generation is enabled.</li> </ul> | 31'hX1'b0 <sup>(3)</sup>  | RO     |
| 0x329 | LANE_DESKEWED          | The following encodings are defined: <ul style="list-style-type: none"> <li>Bit[0]: Indicates all lanes are deskewed.</li> <li>Bit[1]: When asserted indicates a change in lanes deskewed status. To clear this sticky bit, write 1'b1 to the corresponding bit of the SCLR_FRM_ERR register. This is a latched signal.</li> </ul>   | 30'hX2'b00 <sup>(3)</sup> | RO     |
| 0x340 | Reserved               |  |                           |        |
| 0x341 | KHZ_RX                 | The register indicates the value of RX clock (clk_rxmac) frequency. Apply the following definition for the frequency value:  | 0x0000 0000               | RO     |

*continued...*

| Addr  | Name             | Description  | Reset       | Access |
|-------|------------------|--|-------------|--------|
|       |                  | [(Register value <sup>(4)</sup> * clk_status)/10] KHZ  |             |        |
| 0x342 | KHZ_TX           | The register indicates the value of TX clock (clk_txmac) frequency. Apply the following definition for the frequency value:<br>[(Register value <sup>(4)</sup> * clk_status)/10] KHZ   | 0x0000 0000 | RO     |
| 0x343 | PHY_TLKIT_ACCESS | If you turn on the <b>Enable auto adaptation triggering for RX PMA CTLE/DFE mode</b> option, write 1'b1 to bit[0] of this register to hold the auto adaptation module FSM to an idle state.<br><br><i>Note:</i> For H-tile production devices, write 1'b1 to bit[0] before you launch the Transceiver Toolkit so that the transceiver channel appears in the Transceiver Toolkit. Close the Transceiver Toolkit before you write 1'b0 to bit[0] to restart the auto adaptation module FSM so that the System Console does not hang. For more information, refer to <i>Disabling Background Calibration and Accessing the Native PHY in L- and H-Tile Devices</i> . | 31'hX 1'b0  | RW     |

## 7.2. TX MAC Registers

Table 26. TX MAC Registers

| Addr  | Name          | Description  | Reset       | Access |
|-------|---------------|--|-------------|--------|
| 0x400 | TXMAC_REVID   | TX MAC revision ID for 25G TX MAC CSRs.  | 0x0504 2018 | RO     |
| 0x401 | TXMAC_SCRATCH | Scratch register available for testing.  | 0x0000 0000 | RW     |
| 0x402 | TXMAC_NAME_0  | First 4 characters of IP core variation identifier string, "25gMACTxCSR".                  | 0x3235 674D | RO     |
| 0x403 | TXMAC_NAME_1  | Next 4 characters of IP core variation identifier string, "ACTx".                          | 0x4143 5478 | RO     |
| 0x404 | TXMAC_NAME_2  | Final 4 characters of IP core variation identifier string, "0CSR". The "0" is unprintable. | 0x0043 5352 | RO     |

*continued...*

<sup>(4)</sup> Register value convert in decimal.

<sup>(5)</sup> X means "Don't Care".

| Addr  | Name               | Description   | Reset                        | Access |
|-------|--------------------|---|------------------------------|--------|
| 0x405 | LINK_FAULT         | <p>Link Fault Configuration Register. The following bits are defined:</p> <ul style="list-style-type: none"> <li>Force Remote Fault bit[3]: When link fault generation is enabled, stops data transmission and forces transmission of a remote fault.</li> <li>Disable Remote Fault bit[2]: When both link fault reporting and unidirectional transport are enabled, the core transmits data and does not transmit remote faults (RF). This bit takes effect when the value of this register is 28'hX4'b0111.</li> <li>Unidir Enable bit[1]: When asserted, the core includes Clause 66 support for the remote link fault reporting on the Ethernet link.</li> <li>Link Fault Reporting Enable bit[0]: The following encodings are defined: <ul style="list-style-type: none"> <li>1'b1: The PCS generates the proper fault sequence on Ethernet link, when conditions are met.</li> <li>1'b0: The PCS does not generate the fault sequence.</li> </ul> </li> </ul> | 28'hX_4'b0001 <sup>(5)</sup> | RW     |
| 0x407 | MAX_TX_SIZE_CONFIG | <p>Specifies the maximum TX frame length. Frames that are longer are considered oversized. They are transmitted, but also increment the CNTR_TX_OVERSIZE register. Bits [31:16] of this register are Reserved.</p>  | 0xXXXX 2580 <sup>(5)</sup>   | RW     |
| 0x40A | TXMAC_CONTROL      | <p>TX MAC Control Register. A single bit is defined:</p> <ul style="list-style-type: none"> <li>Bit[1]: VLAN detection disabled. This bit is deasserted by default, implying VLAN detection is enabled.</li> </ul>  | 30'hX2'b0X <sup>(5)</sup>    | RW     |

### 7.3. RX MAC Registers

Table 27. RX MAC Registers

| Addr  | Name          | Description  | Reset       | Access |
|-------|---------------|--|-------------|--------|
| 0x500 | RXMAC_REVID   | RX MAC revision ID for 25G Ethernet IP core.   | 0x0504 2018 | RO     |
| 0x501 | RXMAC_SCRATCH | Scratch register available for testing.  | 0x0000 0000 | RW     |
| 0x502 | RXMAC_NAME_0  | First 4 characters of IP core variation identifier string, "25gMACRxCSR".                  | 0x3235 674D | RO     |
| 0x503 | RXMAC_NAME_1  | Next 4 characters of IP core variation identifier string, "ACRx".                          | 0x4143 5278 | RO     |
| 0x504 | RXMAC_NAME_2  | Final 4 characters of IP core variation identifier string, "0CSR". The "0" is unprintable. | 0x0043 5352 | RO     |

*continued...*

<sup>(5)</sup> X means "Don't Care".

<sup>(6)</sup> X means "Don't Care".

| Addr  | Name               | Description   | Reset                         | Access |
|-------|--------------------|---|-------------------------------|--------|
| 0x506 | MAX_RX_SIZE_CONFIG | Specifies the maximum frame length available. The MAC asserts <code>l1_rx_error[3]</code> when the length of the received frame exceeds the value of this register.<br>If the IP core receives an Ethernet frame of size greater than the number of bytes specified in this register, and the IP core includes statistics registers, the IP core increments the 64-bit <code>CNTR_RX_OVERSIZE</code> counter.   | 0xXXXX 2580 <sup>(6)</sup>    | RW     |
| 0x507 | MAC_CRC_CONFIG     | The RX CRC forwarding configuration register. The following encodings are defined: <ul style="list-style-type: none"> <li>1'b0: Remove RX CRC, do not forward it to the RX client interface</li> <li>1'b1: Retain RX CRC, forward it to the RX client interface</li> </ul> In either case, the IP core checks the incoming RX CRC and flags errors.   | 31'hX1'b0 <sup>(6)</sup>      | RW     |
| 0x508 | LINK_FAULT         | Link Fault Status Register.<br>For regular (non-unidirectional) Link Fault, implements <i>IEEE 802.3 Ethernet Clause 46</i> .<br>For unidirectional Link Fault, implements <i>IEEE 802.3 Ethernet Clause 66</i> .<br>If you turn on <b>Enable link fault generation</b> , the following bit fields are defined: <ul style="list-style-type: none"> <li>Bit[0]: A bit value of 1 indicates local fault is detected.</li> <li>Bit[1]: A bit value of 1 indicates remote fault is detected.</li> </ul> If you disable <b>Enable link fault generation</b> , bit[0] and [1] are always to zero. | 30'hX2'b00 <sup>(6)</sup>     | RO     |
| 0x50A | RXMAC_CONTROL      | RX MAC Control Register. A single bit is defined: <ul style="list-style-type: none"> <li>Bit [1]: VLAN detection disabled. This bit is deasserted by default implying VLAN detection is enabled.</li> <li>Bit [4]: Enable check for Preamble. By default, Preamble check is turned off. Write 1'b1 to this bit to enable preamble checking. This bit is a don't care when you turn on <b>Enable Preamble Passthrough</b>.</li> </ul>  | 27'hX_5'b0XX0X <sup>(6)</sup> | RW     |

## 7.4. Pause/PFC Flow Control Registers

Some of the registers in this table cannot be updated during normal operation. To ensure correct operation, perform a soft reset by writing Bit[0] of the `PHY_CONFIG` (0x310) after updating registers that cannot be changed dynamically.

<sup>(6)</sup> X means "Don't Care".

**Table 28. TX Flow Control Registers**

| Addr  | Bit        | Name  | Description   | Reset  | Access |
|-------|------------|---|---|--|--------|
| 0x600 | 31:0       | TX Flow Control Revision ID                                 | Specifies the revision ID, "25GEFCTx CSR".  | 0x0916_2016                                  | RO     |
| 0x601 | 31:0       | TX Flow Control Scratch Pad                                 | Scratch register for testing.   | 0  | RW     |
| 0x602 | 31:0       | TX Flow Control IP Core Variant 0                           | Specifies first 4 characters of IP core variation identifier ASCII string, "25GE".  | 0x3235_4745                                  | RO     |
| 0x603 | 31:0       | TX Flow Control IP Core Variant 1                           | Next 4 characters of IP core variation identifier ASCII string, "FCTx".   | 0x4643_5478                                  | RO     |
| 0x604 | 31:0       | TX Flow Control IP Core Variant 2                           | Final 4 characters of IP core variation identifier ASCII string, "0CSR". The "0" is unprintable.  | 0x0043_5352                                  | RO     |
| 0x605 | (FCQN-1):0 | TX Flow Control Enable                                      | <p>Enables the IP core to generate XON and XOFF Pause/PFC flow control frames to the remote partner. The following encodings are defined:</p> <ul style="list-style-type: none"> <li>1'b0: XON or XOFF Pause/PFC flow control is disabled.</li> <li>1'b1: XON or XOFF Pause/PFC flow control is enabled.</li> </ul> <p>You can change this field dynamically.</p>   | 1'b1 in each bit that corresponds to a queue | RW     |
|       | 31:FCQN    | Reserved  | Reserved  | 0  | RO     |
| 0x606 | (FCQN-1):0 | TX Flow Control CSR XON/XOFF Request 1<br>One bit per queue | <p>XON/XOF flow control frame request bit 0. Interpretation depends on whether the IP core is in 1-bit FC request mode or in 2-bit FC request mode. This register affects a flow control queue only if the corresponding bit of the TX Flow Control Enable register has the value of 1.</p> <p>In 2-bit mode, in addition, this register is active for a specific flow control queue only if the corresponding bit in the TX 2-bit Flow Control Request Mode register field (bits [(FCQN-1):0] of the register at offset 0x641) specifies that the flow control logic accepts input from this register.</p> <p>The following encodings are defined for 1-bit mode. The IP core reads the 1-bit mode value in TX Flow Control CSR XON/XOFF Request 0.</p> <ul style="list-style-type: none"> <li>0 = No request</li> <li>0 to 1 = Generate XOFF request</li> <li>1 = Continue to generate XOFF request</li> <li>1 to 0 = Generate XON request</li> </ul> | 0  | RW     |

*continued...*



| Addr  | Bit           | Name   | Description  | Reset      | Access |
|-------|---------------|--|--|------------|--------|
|       |               |  | <p>The following encodings are defined for 2-bit mode. The IP core reads the 2-bit mode value in {TX Flow Control CSR XON/XOFF Request 1, TX Flow Control CSR XON/XOFF Request 1}.</p> <ul style="list-style-type: none"> <li>• 00 = No request</li> <li>• 01 = XON request</li> <li>• 10 = XOFF request</li> <li>• 11 = Invalid</li> </ul> <p>You can modify the value of this field dynamically.</p> |            |        |
|       | 15:FCQN       | Reserved   | Reserved   | 0          | RO     |
|       | (FCQN +15):16 | TX Flow Control CSR XON/XOFF Request 1 1-bit per queue | In conjunction with Flow Control XON/XOFF Request 0 specifies a 2-bit request for XON/XOFF flow control frame transmission. This bit is the upper bit of the 2-bit control field. You can change the value of this field dynamically.  | 0          | RW     |
|       | 31:(FCQN +16) | Reserved   | Reserved   | 0          | RO     |
| 0x607 | 31:0          | Reserved   | Reserved   | N/A        | RO     |
| 0x608 | 31:0          | Reserved   | Reserved   | N/A        | RO     |
| 0x609 | 31:0          | Reserved   | Reserved   | N/A        | RO     |
| 0x60A | 0             | TX Pause Enable 1-bit                                  | <p>Determines whether receiving a valid Pause frame stops TX user data transmission.</p> <p>1'b0: Transmission is not stopped<br/>1'b1: Transmission stops</p> <p>You cannot change the value of this field dynamically.</p>   | 0          | RW     |
|       | 31:1          | Reserved   | Reserved   | 0          | RO     |
| 0x60B | 31:0          | Reserved   | Reserved   | N/A        | RO     |
| 0x60C | 31:0          | Reserved   | Reserved   | N/A        | RO     |
| 0x60D | 31:0          | TX Flow Control Destination Address Lower              | Specifies the 48-bit Destination Address of the flow control frame. Contains the 32 LSB of the address field. You cannot modify the value of this field dynamically.   | 0xC2000001 | RW     |
| 0x60E | 15:0          | TX Flow Control Destination Address Upper              | Specifies the 48-bit Destination Address of flow control frame. Contains the 16 MSB of the address field. You cannot modify the value of this field dynamically.   | 0x0180     | RW     |
|       | 31:16         | Reserved   | Reserved   | 0          | RO     |
| 0x60F | 31:0          | TX Flow Control Source Address Lower                   | Specifies the 48-bit Source Address of flow control frame. Contains the 32 LSB of the address field.   | 0xCBFC5ADD | RW     |
| 0x610 | 15:0          | TX Flow Control Source Address Upper                   | Specifies the 48-bit Source Address of flow control frame. Contains the 16 MSB of the address field.   | 0xE100     | RW     |

*continued...*

| Addr  | Bit            | Name  | Description   | Reset  | Access |
|---|----------------|---|---|--------|--------|
|   |                |   | You cannot modify the value of this field dynamically.  |        |        |
|   | 31:16          | Reserved  | Reserved  | 0      | RO     |
| 0x620,<br>0x621, ...,<br>0x620+<br>(FCQN-1) | 15:0           | TX Flow Control Quanta<br>16-bit per queue                          | Specifies the pause quanta of Pause/PFC flow control frames to be sent to remote partner.<br>You cannot modify the value of this field dynamically.   | 0xFFFF | RW     |
|   | 31:16          | Reserved  | Reserved  | 0      | RO     |
| 0x628,<br>0x629, ...,<br>0x628+<br>(FCQN-1) | 15:0           | TX Flow Control Signal XOFF Request Hold Quanta<br>16-bit per queue | Specifies the separation between 2 consecutive XOFF flow control frames.<br>You cannot modify the value of this field dynamically.  | 0xFFFF | RW     |
|   | 31:16          | Reserved  | Reserved  | 0      | RO     |
| 0x640                                       | 0              | TX Flow Control Select<br>1-bit                                     | Specifies whether the TX hardware generates Pause or PFC frames. Affects only PFC Queue 0.<br>Usage example:<br>You can synthesize a single PFC queue and use it for both Pause and PFC purpose.<br>1'b0: Pause<br>1'b1: PFC<br>You cannot modify the value of this field dynamically.  | 1      | RW     |
|   | 31:1           | Reserved.   | Reserved.   | 0      | RO     |
| 0x641                                       | (FCQN-1):<br>0 | TX 2-bit Flow Control Request Mode<br>1-bit per queue               | Determines whether the TX Flow Control CSR XON/XOFF Request register or the pause_insert_tx0 and pause_insert_tx1 signals control XON/XOFF mode in 2-bit control mode.<br>1'b0: The pause_insert_tx0 and pause_insert_tx1 signals control requests<br>1'b1: The TX Flow Control CSR XON/XOFF Request register fields control requests<br>You cannot modify the value of this field dynamically. | 0      | RW     |
|   | 16             | TX Flow Control Request Mode<br>1 bit for all queues                | Determines whether the IP core is in TX flow control 1-bit mode or 2-bit mode.<br>1'b0: Use 1-bit mode to make TX flow control requests<br>1'b1: Use 2-bit mode to make TX flow control requests  | 0      | RW     |
|   | 31:17          | Reserved  | Reserved  | 0      | RO     |

**Table 29. RX Flow Control Registers**

| Addr  | Bit          | Name                                      | Description   | Reset  | Access |
|-------|--------------|---|---|--|--------|
| 0x700 | 31:0         | RX Flow Control Revision ID               | Provides the flow control revision, "25GEFCRx CSR".   | 0x0916_2016                                  | RO     |
| 0x701 | 31:0         | RX Flow Control Scratch Pad               | Provides a register for debug.  | 0  | RW     |
| 0x702 | 31:0         | RX Flow Control IP Core Variant 0         | First 4 characters of IP core variation identifier ASCII string, "25GE".  | 0x3235_4745                                  | RO     |
| 0x703 | 31:0         | RX Flow Control IP Core Variant 1         | Next 4 characters of IP core variation identifier ASCII string, "FCRx".   | 0x4643_5278                                  | RO     |
| 0x704 | 31:0         | RX Flow Control IP Core Variant 2         | Final 4 characters of IP core variation identifier ASCII string, "0CSR". The "0" is unprintable.  | 0x0043_5352                                  | RO     |
| 0x705 | (FCQN-1): 0] | RX PFC Enable<br>1 bit per queue          | Determines whether receiving a valid PFC frame causes the PFC duration user interface to indicate a valid pause quantity duration to the user logic.<br>1'b0: Disable<br>1'b1: Enable<br>You cannot modify the value of this field dynamically. | 1'b1 in each bit that corresponds to a queue | RW     |
|       | 31:FCQN8     | Reserved                                  | Reserved  | 0  | RO     |
| 0x706 | 31:0         | Reserved                                  | Reserved  | N/A  | RO     |
| 0x707 | 31:0         | RX Flow Control Destination Address Lower | Specifies the 48-bit Destination Address of the flow control frame. Contains the 32 LSB of the address field. You cannot modify the value of this field dynamically.  | 0xC2000001                                   | RW     |
| 0x708 | 15:0         | RX Flow Control Destination Address Upper | Specifies the 48-bit Destination Address of flow control frame. Contains the 16 MSB of the address field. You cannot modify the value of this field dynamically.  | 0x0180                                       | RW     |
|       | 31:16        | Reserved                                  | Reserved  | 0  | RO     |

**Related Information**

[Flow Control](#) on page 41

Describes how the IP core uses the information in these registers to provide flow control functionality.

## 7.5. Statistics Registers

The 25G Ethernet Intel FPGA IP statistics registers count Ethernet traffic and errors. The 64-bit statistics registers are designed to roll over, to ensure timing closure on the FPGA. However, these registers should never roll over if the link is functioning properly. The statistics registers check the size of frames, which includes the following fields:

- Size of the destination address
- Size of the source address
- Size of the data
- Four bytes of CRC

The statistics counters module is a synthesis option. The statistics registers are counters that are implemented inside the CSR. When you turn on the **Enable MAC statistics counters** parameter in the 25G Ethernet Intel FPGA IP parameter editor, the counters are implemented in the CSR. When you turn off the **Enable MAC statistics counters** parameter in the 25G Ethernet Intel FPGA IP parameter editor, the counters are not implemented in the CSR, and read access to the counters returns undefined results.

After system power-up, the statistics counters have random values. You must clear these registers and confirm the system is stable before using their values. To clear the registers, use any of the following methods:

1. Assert `csr_rst_n` to clear both the TX and RX statistic counters.
2. Assert `tx_rst_n` to clear the TX statistic counters.
3. Assert `rx_rst_n` to clear the RX statistic counters.
4. Write 1'b1 to bit[0], `eio_sys_rst` of the PHY\_CONFIG (0x310) register to clear both the TX and RX statistic counters.
5. Write 1'b1 to bit[1], `soft_txp_rst` of the PHY\_CONFIG (0x310) register to clear the TX statistic counters.
6. Write 1'b1 to bit[2], `soft_rxp_rst` of the PHY\_CONFIG (0x310) register to clear the RX statistic counters.
7. Write 1'b1 to bit[0] of the CNTR\_TX\_CONFIG (0x845) to clear the TX statistic counters.
8. Write 1'b1 to bit[0] of the CNTR\_RX\_CONFIG (0x945) to clear the RX statistic counters.

The configuration register at offset 0x845 allows you to clear all of the TX statistics counters. The configuration register at offset 0x945 allows you to clear all of the RX statistics counters. If you exclude these registers, you can monitor the statistics counter increment vectors that the IP core provides at the client side interface and maintain your own counters.

Reading the value of a statistics register does not affect its value.

To ensure that the counters you read are consistent, you should issue a shadow request to create a snapshot of all of the TX or RX statistics registers, by setting bit [2] of the configuration register at offset 0x845 or 0x945, respectively. Until you reset

this bit, the counters continue to increment but the readable values remain constant. You can read bit [1] of the status register at offset 0x846 or 0x946, respectively, to confirm your shadow request has been granted or released.

### 7.5.1. TX Statistics Registers

**Table 30. Transmit Side Statistics Registers**

The TX statistics counters do not reflect TX CRC errors the user forces by asserting the `ll_tx_error` signal.

| Address | Name-                     | Description   | Access |
|---------|---------------------------|---|--------|
| 0x800   | CNTR_TX_FRAGMENT_S_LO     | Number of transmitted frames less than 64 bytes and reporting a CRC error (lower 32 bits). The value of this register is always zero. The IP core does not transmit frames of length less than nine bytes. The IP core pads frames of length nine bytes to 64 bytes to extend them to 64 bytes. The CRC field of the client frames is not verified by TX MAC when the <b>Enable TX CRC passthrough</b> option is disabled or enabled. | RO     |
| 0x801   | CNTR_TX_FRAGMENT_S_HI     | Number of transmitted frames less than 64 bytes and reporting a CRC error (upper 32 bits). The value of this register is always zero. The IP core does not transmit frames of length less than nine bytes. The IP core pads frames of length nine bytes to 64 bytes to extend them to 64 bytes. The CRC field of the client frames is not verified by TX MAC when the <b>Enable TX CRC passthrough</b> option is disabled or enabled. | RO     |
| 0x802   | CNTR_TX_JABBERS_LO        | Number of transmitted oversized frames reporting a CRC error (lower 32 bits). The value of this register is always zero. The CRC field of the client frames is not verified by TX MAC when the <b>Enable TX CRC passthrough</b> option is disabled or enabled.  | RO     |
| 0x803   | CNTR_TX_JABBERS_HI        | Number of transmitted oversized frames reporting a CRC error (upper 32 bits). The value of this register is always zero. The CRC field of the client frames is not verified by TX MAC when the <b>Enable TX CRC passthrough</b> option is disabled or enabled.  | RO     |
| 0x804   | CNTR_TX_FCS_LO            | Number of transmitted packets with FCS errors. (lower 32 bits). The value of this register is always zero. The CRC field of the client frames is not verified by TX MAC when the <b>Enable TX CRC passthrough</b> option is disabled or enabled.  | RO     |
| 0x805   | CNTR_TX_FCS_HI            | Number of transmitted packets with FCS errors. (upper 32 bits). The value of this register is always zero. The CRC field of the client frames is not verified by TX MAC when the <b>Enable TX CRC passthrough</b> option is disabled or enabled.  | RO     |
| 0x806   | CNTR_TX_CRCERR_LO         | Number of transmitted frames with a frame of length at least 64 reporting a CRC error (lower 32 bits).  | RO     |
| 0x807   | CNTR_TX_CRCERR_HI         | Number of transmitted frames with a frame of length at least 64 reporting a CRC error (upper 32 bits).  | RO     |
| 0x808   | CNTR_TX_MCAST_DATA_ERR_LO | Number of errored multicast frames transmitted, excluding control frames (lower 32 bits).   | RO     |
| 0x809   | CNTR_TX_MCAST_DATA_ERR_HI | Number of errored multicast frames transmitted, excluding control frames (upper 32 bits).   | RO     |
| 0x80A   | CNTR_TX_BCAST_DATA_ERR_LO | Number of errored broadcast frames transmitted, excluding control frames (lower 32 bits).   | RO     |
| 0x80B   | CNTR_TX_BCAST_DATA_ERR_HI | Number of errored broadcast frames transmitted, excluding control frames (upper 32 bits).   | RO     |
| 0x80C   | CNTR_TX_UCAST_DATA_ERR_LO | Number of errored unicast frames transmitted, excluding control frames (lower 32 bits).   | RO     |

*continued...*

| Address | Name-                     | Description   | Access |
|---------|---------------------------|---|--------|
| 0x80D   | CNTR_TX_UCAST_DATA_ERR_HI | Number of errored unicast frames transmitted, excluding control frames (upper 32 bits).   | RO     |
| 0x80E   | CNTR_TX_MCAST_CTL_ERR_LO  | Number of errored multicast control frames transmitted (lower 32 bits).   | RO     |
| 0x80F   | CNTR_TX_MCAST_CTL_ERR_HI  | Number of errored multicast control frames transmitted (upper 32 bits).   | RO     |
| 0x810   | CNTR_TX_BCAST_CTL_ERR_LO  | Number of errored broadcast control frames transmitted (lower 32 bits).   | RO     |
| 0x811   | CNTR_TX_BCAST_CTL_ERR_HI  | Number of errored broadcast control frames transmitted (upper 32 bits).   | RO     |
| 0x812   | CNTR_TX_UCAST_CTL_ERR_LO  | Number of errored unicast control frames transmitted (lower 32 bits).   | RO     |
| 0x813   | CNTR_TX_UCAST_CTL_ERR_HI  | Number of errored unicast control frames transmitted (upper 32 bits).   | RO     |
| 0x814   | CNTR_TX_PAUSE_ERR_LO      | Number of errored pause frames transmitted (lower 32 bits).   | RO     |
| 0x815   | CNTR_TX_PAUSE_ERR_HI      | Number of errored pause frames transmitted (upper 32 bits).   | RO     |
| 0x816   | CNTR_TX_64B_LO            | Number of 64-byte transmitted frames (lower 32 bits), including the CRC field but excluding the preamble and SFD bytes.                       | RO     |
| 0x817   | CNTR_TX_64B_HI            | Number of 64-byte transmitted frames (upper 32 bits), including the CRC field but excluding the preamble and SFD bytes.                       | RO     |
| 0x818   | CNTR_TX_65to127B_LO       | Number of transmitted frames between 65–127 bytes (lower 32 bits).  | RO     |
| 0x819   | CNTR_TX_65to127B_HI       | Number of transmitted frames between 65–127 bytes (upper 32 bits).  | RO     |
| 0x81A   | CNTR_TX_128to255B_LO      | Number of transmitted frames between 128–255 bytes (lower 32 bits).   | RO     |
| 0x81B   | CNTR_TX_128to255B_HI      | Number of transmitted frames between 128–255 bytes (upper 32 bits).   | RO     |
| 0x81C   | CNTR_TX_256to511B_LO      | Number of transmitted frames between 256–511 bytes (lower 32 bits).   | RO     |
| 0x81D   | CNTR_TX_256to511B_HI      | Number of transmitted frames between 256–511 bytes (upper 32 bits).   | RO     |
| 0x81E   | CNTR_TX_512to1023B_LO     | Number of transmitted frames between 512–1023 bytes (lower 32 bits).  | RO     |
| 0x81F   | CNTR_TX_512to1023B_HI     | Number of transmitted frames between 512–1023 bytes (upper 32 bits).  | RO     |
| 0x820   | CNTR_TX_1024to1518B_LO    | Number of transmitted frames between 1024–1518 bytes (lower 32 bits).   | RO     |
| 0x821   | CNTR_TX_1024to1518B_HI    | Number of transmitted frames between 1024–1518 bytes (upper 32 bits).   | RO     |
| 0x822   | CNTR_TX_1519toMAXB_LO     | Number of transmitted frames of size between 1519 bytes and the number of bytes specified in the MAX_TX_SIZE_CONFIG register (lower 32 bits). | RO     |

*continued...*

| Address     | Name                     | Description  | Access |
|-------------|--------------------------|--|--------|
| 0x823       | CNTR_TX_1519toMAX_HI     | Number of transmitted frames of size between 1519 bytes and the number of bytes specified in the MAX_TX_SIZE_CONFIG register (upper 32 bits).  | RO     |
| 0x824       | CNTR_TX_OVERSIZE_LO      | Number of oversized frames (frames with more bytes than the number specified in the MAX_TX_SIZE_CONFIG register) transmitted (lower 32 bits).  | RO     |
| 0x825       | CNTR_TX_OVERSIZE_HI      | Number of oversized frames (frames with more bytes than the number specified in the MAX_TX_SIZE_CONFIG register) transmitted (upper 32 bits).  | RO     |
| 0x826       | CNTR_TX_MCAST_DATA_OK_LO | Number of valid multicast frames transmitted, excluding control frames (lower 32 bits).  | RO     |
| 0x827       | CNTR_TX_MCAST_DATA_OK_HI | Number of valid multicast frames transmitted, excluding control frames (upper 32 bits).  | RO     |
| 0x828       | CNTR_TX_BCAST_DATA_OK_LO | Number of valid broadcast frames transmitted, excluding control frames (lower 32 bits).  | RO     |
| 0x829       | CNTR_TX_BCAST_DATA_OK_HI | Number of valid broadcast frames transmitted, excluding control frames (upper 32 bits).  | RO     |
| 0x82A       | CNTR_TX_UCAST_DATA_OK_LO | Number of valid unicast frames transmitted, excluding control frames (lower 32 bits).  | RO     |
| 0x82B       | CNTR_TX_UCAST_DATA_OK_HI | Number of valid unicast frames transmitted, excluding control frames (upper 32 bits).  | RO     |
| 0x82C       | CNTR_TX_MCAST_COUNTER_LO | Number of valid multicast frames transmitted, excluding data frames (lower 32 bits).   | RO     |
| 0x82D       | CNTR_TX_MCAST_COUNTER_HI | Number of valid multicast frames transmitted, excluding data frames (upper 32 bits).   | RO     |
| 0x82E       | CNTR_TX_BCAST_COUNTER_LO | Number of valid broadcast frames transmitted, excluding data frames (lower 32 bits).   | RO     |
| 0x82F       | CNTR_TX_BCAST_COUNTER_HI | Number of valid broadcast frames transmitted, excluding data frames (upper 32 bits).   | RO     |
| 0x830       | CNTR_TX_UCAST_COUNTER_LO | Number of valid unicast frames transmitted, excluding data frames (lower 32 bits).   | RO     |
| 0x831       | CNTR_TX_UCAST_COUNTER_HI | Number of valid unicast frames transmitted, excluding data frames (upper 32 bits).   | RO     |
| 0x832       | CNTR_TX_PAUSE_LO         | Number of valid pause frames transmitted (lower 32 bits).  | RO     |
| 0x833       | CNTR_TX_PAUSE_HI         | Number of valid pause frames transmitted (upper 32 bits).  | RO     |
| 0x834       | CNTR_TX_RUNT_LO          | Number of transmitted runt packets (lower 32 bits). The value of this register is always zero. The IP core does not transmit frames of length less than nine bytes. The IP core pads frames of length nine bytes to 64 bytes to extend them to 64 bytes. | RO     |
| 0x835       | CNTR_TX_RUNT_HI          | Number of transmitted runt packets (upper 32 bits). The value of this register is always zero. The IP core does not transmit frames of length less than nine bytes. The IP core pads frames of length nine bytes to 64 bytes to extend them to 64 bytes. | RO     |
| 0x836-0x844 | Reserved                 |  |        |
| 0x845       | CNTR_TX_CONFIG           | Bits[2:0]: Configuration of TX statistics counters:  | RW     |

*continued...*

| Address     | Name-                | Description  | Access |
|-------------|----------------------|--|--------|
|             |                      | <ul style="list-style-type: none"> <li>Bit[2]: Shadow request (active high): When set to the value of 1, TX statistics collection is paused. The underlying counters continue to operate, but the readable values reflect a snapshot at the time the pause flag was activated. Write a 0 to release.</li> <li>Bit[1]: Parity-error clear. When software sets this bit, the IP core clears the parity bit CNTR_TX_STATUS[0]. This bit (CNTR_TX_CONFIG[1]) is self-clearing.</li> <li>Bit[0]: Software can set this bit to the value of 1 to reset all of the TX statistics registers at the same time. This bit is self-clearing.</li> </ul> Bits[31:3] are Reserved. |        |
| 0x846       | CNTR_TX_STATUS       | <ul style="list-style-type: none"> <li>Bit[1]: Indicates that the TX statistics registers are paused (while CNTR_TX_CONFIG[2] is asserted).</li> <li>Bit[0]: Indicates the presence of at least one parity error in the TX statistics counters.</li> </ul> Bits[31:2] are Reserved.  | RO     |
| 0x847–0x85F | Reserved             |  |        |
| 0x860       | TxPayloadOctetsOK_LO | Number of transmitted payload bytes in frames with no FCS, undersized, oversized, or payload length errors. If VLAN detection is turned off for the TX MAC (bit[1] of the TX_MAC_CONTROL register at offset 0x40A has the value of 1), the IP core counts the VLAN header bytes (4 bytes for VLAN and 8 bytes for stacked VLAN) as payload bytes. This register is compliant with the requirements for aOctetsTransmittedOK in section 5.2.2.1.8 of the <i>IEEE Standard 802.3-2008</i> .  | RO     |
| 0x861       | TxPayloadOctetsOK_HI |  | RO     |
| 0x862       | TxFrameOctetsOK_LO   | Number of transmitted bytes in frames with no FCS, undersized, oversized, or payload length errors. This register is compliant with the requirements for ifOutOctets in RFC3635 (Managed Objects for Ethernet-like Interface Types) and TX etherStatsOctets in RFC2819(Remote Network Monitoring Management Information Base (RMON)).  | RO     |
| 0x863       | TxFrameOctetsOK_HI   |  | RO     |

### 7.5.2. RX Statistics Registers

Table 31. Receive Side Statistics Registers

| Address | Name                 | Description  | Access |
|---------|----------------------|--|--------|
| 0x900   | CNTR_RX_FRAGMENTS_LO | Number of received frames less than 64 bytes and reporting a CRC error (lower 32 bits)   | RO     |
| 0x901   | CNTR_RX_FRAGMENTS_HI | Number of received frames less than 64 bytes and reporting a CRC error (upper 32 bits)   | RO     |
| 0x902   | CNTR_RX_JABBERS_LO   | Number of received oversized frames reporting a CRC error (lower 32 bits)  | RO     |
| 0x903   | CNTR_RX_JABBERS_HI   | Number of received oversized frames reporting a CRC error (upper 32 bits)  | RO     |
| 0x904   | CNTR_RX_FCS_LO       | Number of received packets with FCS errors. This register maintains a count of the number of pulses on the l<n>_rx_fcs_error or rx_fcs_error output signal (lower 32 bits) | RO     |
| 0x905   | CNTR_RX_FCS_HI       | Number of received packets with FCS errors. This register maintains a count of the number of pulses on the l<n>_rx_fcs_error output signal (upper 32 bits)                 | RO     |

*continued...*



## 7. Control, Status, and Statistics Register Descriptions

UG-20109 | 2021.03.29



| Address | Name                      | Description   | Access |
|---------|---------------------------|---|--------|
| 0x906   | CNTR_RX_CRCERR_LO         | Number of received frames with a frame of length at least 64, with CRC error (lower 32 bits)                        | RO     |
| 0x907   | CNTR_RX_CRCERR_HI         | Number of received frames with a frame of length at least 64, with CRC error (upper 32 bits)                        | RO     |
| 0x908   | CNTR_RX_MCAST_DATA_ERR_LO | Number of errored multicast frames received, excluding control frames (lower 32 bits)                               | RO     |
| 0x909   | CNTR_RX_MCAST_DATA_ERR_HI | Number of errored multicast frames received, excluding control frames (upper 32 bits)                               | RO     |
| 0x90A   | CNTR_RX_BCAST_DATA_ERR_LO | Number of errored broadcast frames received, excluding control frames (lower 32 bits)                               | RO     |
| 0x90B   | CNTR_RX_BCAST_DATA_ERR_HI | Number of errored broadcast frames received, excluding control frames (upper 32 bits)                               | RO     |
| 0x90C   | CNTR_RX_UCAST_DATA_ERR_LO | Number of errored unicast frames received, excluding control frames (lower 32 bits)                                 | RO     |
| 0x90D   | CNTR_RX_UCAST_DATA_ERR_HI | Number of errored unicast frames received, excluding control frames (upper 32 bits)                                 | RO     |
| 0x90E   | CNTR_RX_MCAST_CTRL_ERR_LO | Number of errored multicast control frames received (lower 32 bits)   | RO     |
| 0x90F   | CNTR_RX_MCAST_CTRL_ERR_HI | Number of errored multicast control frames received (upper 32 bits)   | RO     |
| 0x910   | CNTR_RX_BCAST_CTRL_ERR_LO | Number of errored broadcast control frames received (lower 32 bits)   | RO     |
| 0x911   | CNTR_RX_BCAST_CTRL_ERR_HI | Number of errored broadcast control frames received (upper 32 bits)   | RO     |
| 0x912   | CNTR_RX_UCAST_CTRL_ERR_LO | Number of errored unicast control frames received (lower 32 bits)   | RO     |
| 0x913   | CNTR_RX_UCAST_CTRL_ERR_HI | Number of errored unicast control frames received (upper 32 bits)   | RO     |
| 0x914   | CNTR_RX_PAUSE_ERR_LO      | Number of errored pause frames received (lower 32 bits)   | RO     |
| 0x915   | CNTR_RX_PAUSE_ERR_HI      | Number of errored pause frames received (upper 32 bits)   | RO     |
| 0x916   | CNTR_RX_64B_LO            | Number of 64-byte received frames (lower 32 bits), including the CRC field but excluding the preamble and SFD bytes | RO     |
| 0x917   | CNTR_RX_64B_HI            | Number of 64-byte received frames (upper 32 bits), including the CRC field but excluding the preamble and SFD bytes | RO     |
| 0x918   | CNTR_RX_65to127B_LO       | Number of received frames between 65–127 bytes (lower 32 bits)  | RO     |
| 0x919   | CNTR_RX_65to127B_HI       | Number of received frames between 65–127 bytes (upper 32 bits)  | RO     |
| 0x91A   | CNTR_RX_128to255B_LO      | Number of received frames between 128 –255 bytes (lower 32 bits)  | RO     |
| 0x91B   | CNTR_RX_128to255B_HI      | Number of received frames between 128 –255 bytes (upper 32 bits)  | RO     |

**continued...**

| Address | Name                     | Description   | Access |
|---------|--------------------------|---|--------|
| 0x91C   | CNTR_RX_256to511B_LO     | Number of received frames between 256 –511 bytes (lower 32 bits)  | RO     |
| 0x91D   | CNTR_RX_256to511B_HI     | Number of received frames between 256 –511 bytes (upper 32 bits)  | RO     |
| 0x91E   | CNTR_RX_512to1023B_LO    | Number of received frames between 512–1023 bytes (lower 32 bits)  | RO     |
| 0x91F   | CNTR_RX_512to1023B_HI    | Number of received frames between 512 –1023 bytes (upper 32 bits)   | RO     |
| 0x920   | CNTR_RX_1024to1518B_LO   | Number of received frames between 1024–1518 bytes (lower 32 bits)   | RO     |
| 0x921   | CNTR_RX_1024to1518B_HI   | Number of received frames between 1024–1518 bytes (upper 32 bits)   | RO     |
| 0x922   | CNTR_RX_1519toMAXB_LO    | Number of received frames between 1519 bytes and the maximum size defined in the MAX_RX_SIZE_CONFIG register (lower 32 bits)              | RO     |
| 0x923   | CNTR_RX_1519toMAXB_HI    | Number of received frames between 1519 bytes and the maximum size defined in the MAX_RX_SIZE_CONFIG register (upper 32 bits)              | RO     |
| 0x924   | CNTR_RX_OVERSIZE_LO      | Number of oversized frames (frames with more bytes than the number specified in the MAX_RX_SIZE_CONFIG register) received (lower 32 bits) | RO     |
| 0x925   | CNTR_RX_OVERSIZE_HI      | Number of oversized frames (frames with more bytes than the number specified in the MAX_RX_SIZE_CONFIG register) received (upper 32 bits) | RO     |
| 0x926   | CNTR_RX_MCAST_DATA_OK_LO | Number of valid multicast frames received, excluding control frames (lower 32 bits)   | RO     |
| 0x927   | CNTR_RX_MCAST_DATA_OK_HI | Number of valid multicast frames received, excluding control frames (upper 32 bits)   | RO     |
| 0x928   | CNTR_RX_BCAST_DATA_OK_LO | Number of valid broadcast frames received, excluding control frames (lower 32 bits)   | RO     |
| 0x929   | CNTR_RX_BCAST_DATA_OK_HI | Number of valid broadcast frames received, excluding control frames (upper 32 bits)   | RO     |
| 0x92A   | CNTR_RX_UCAST_DATA_OK_LO | Number of valid unicast frames received, excluding control frames (lower 32 bits)   | RO     |
| 0x92B   | CNTR_RX_UCAST_DATA_OK_HI | Number of valid unicast frames received, excluding control frames (upper 32 bits)   | RO     |
| 0x92C   | CNTR_RX_MCAST_CTRL_LO    | Number of valid multicast frames received, excluding data frames (lower 32 bits)  | RO     |
| 0x92D   | CNTR_RX_MCAST_CTRL_HI    | Number of valid multicast frames received, excluding data frames (upper 32 bits)  | RO     |
| 0x92E   | CNTR_RX_BCAST_CTRL_LO    | Number of valid broadcast frames received, excluding data frames (lower 32 bits)  | RO     |
| 0x92F   | CNTR_RX_BCAST_CTRL_HI    | Number of valid broadcast frames received, excluding data frames (upper 32 bits)  | RO     |
| 0x930   | CNTR_RX_UCAST_CTRL_LO    | Number of valid unicast frames received, excluding data frames (lower 32 bits)  | RO     |

*continued...*

| Address     | Name                  | Description  | Access |
|-------------|-----------------------|--|--------|
| 0x931       | CNTR_RX_UCAST_CTRL_HI | Number of valid unicast frames received, excluding data frames (upper 32 bits)   | RO     |
| 0x932       | CNTR_RX_PAUSE_LO      | Number of received pause frames, with or without error (lower 32 bits)   | RO     |
| 0x933       | CNTR_RX_PAUSE_HI      | Number of received pause frames, with or without error (upper 32 bits)   | RO     |
| 0x934       | CNTR_RX_RUNT_LO       | Number of received runt packets (lower 32 bits)<br>A runt is a packet of size less than 64 bytes but greater than eight bytes. If a packet is eight bytes or smaller, it is considered a decoding error and not a runt frame, and the IP core does not flag it nor count it as a runt.   | RO     |
| 0x935       | CNTR_RX_RUNT_HI       | Number of received runt packets (upper 32 bits)<br>A runt is a packet of size less than 64 bytes but greater than eight bytes. If a packet is eight bytes or smaller, it is considered a decoding error and not a runt frame, and the IP core does not flag it nor count it as a runt.   | RO     |
| 0x936-0x944 | Reserved              |  |        |
| 0x945       | CNTR_RX_CONFIG        | Bits[2:0]: Configuration of RX statistics counters: <ul style="list-style-type: none"> <li>Bit[2]: Shadow request (active high): When set to the value of 1, RX statistics collection is paused. The underlying counters continue to operate, but the readable values reflect a snapshot at the time the pause flag was activated. Write a 0 to release.</li> <li>Bit[1]: Parity-error clear. When software sets this bit, the IP core clears the parity bit CNTR_RX_STATUS[0]. This bit (CNTR_RX_CONFIG[1]) is self-clearing.</li> <li>Bit[0]: Software can set this bit to the value of 1 to reset all of the RX statistics registers at the same time. This bit is self-clearing.</li> </ul> Bits[31:3] are Reserved. | RW     |
| 0x946       | CNTR_RX_STATUS        | <ul style="list-style-type: none"> <li>Bit[1]: Indicates that the RX statistics registers are paused (while CNTR_RX_CONFIG[2] is asserted).</li> <li>Bit[0]: Indicates the presence of at least one parity error in the RX statistics counters.</li> </ul> Bits [31:2] are Reserved.   | RO     |
| 0x947-0x95F | Reserved              |  |        |
| 0x960       | RxPayloadOctetsOK_LO  | Number of received payload bytes in frames with no FCS, undersized, oversized, or payload length errors. If VLAN detection is turned off for the RX MAC (bit [1] of the RXMAC_CONTROL register at offset 0x50A has the value of 1), the IP core counts the VLAN header bytes (4 bytes for VLAN and 8 bytes for stacked VLAN) as payload bytes. This register is compliant with the requirements for aOctetsReceivedOK in section 5.2.2.1.14 of the <i>IEEE Standard 802.3-2008</i> .   | RO     |
| 0x961       | RxPayloadOctetsOK_HI  |  | RO     |
| 0x962       | RxFrameOctetsOK_LO    | Number of received bytes in frames with no FCS, undersized, oversized, or payload length errors. This register is compliant with the requirements for ifInOctets in RFC3635 (Managed Objects for Ethernet-like Interface Types) and RX etherStatsOctets in RFC2819 (Remote Network Monitoring Management Information Base (RMON)).   | RO     |
| 0x963       | RxFrameOctetsOK_HI    |  | RO     |

## 7.6. 1588 PTP Registers

The 1588 PTP registers together with the 1588 PTP signals process and provide Precision Time Protocol (PTP) timestamp information as defined in the *IEEE 1588-2008 Precision Clock Synchronization Protocol for Networked Measurement and Control Systems Standard*. The 1588 PTP module provides you the support to implement the 1588 Precision Time Protocol in your design.

**Table 32. TX 1588 PTP Registers**

| Addr        | Name               | Bit    | Description  | HW Reset Value | Access |
|-------------|--------------------|--------|--|----------------|--------|
| 0xA00       | TXPTP_REVID        | [31:0] | IP core revision ID.   | 0x0504_2018    | RO     |
| 0xA01       | TXPTP_SCRATCH      | [31:0] | Scratch register available for testing.  | 32'b0          | RW     |
| 0xA02       | TXPTP_NAME_0       | [31:0] | First 4 characters of IP core variation identifier string "25GETXPTPCSR"   | 0x3235_4745    | RO     |
| 0xA03       | TXPTP_NAME_1       | [31:0] | Next 4 characters of IP core variation identifier string "25GETXPTPCSR"  | 0x5458_5054    | RO     |
| 0xA04       | TXPTP_NAME_2       | [31:0] | Final 4 characters of IP core variation identifier string "25GETXPTPCSR"   | 0x5043_5352    | RO     |
| 0xA05       | TX_PTP_CLK_PERIOD  | [19:0] | clk_txmac clock period.<br>Bits[19:16]: nanoseconds (ns)<br>Bits[15:0]: fraction of nanosecond<br>The value of TX_PTP_CLK_PERIOD is speed dependent and needs to be reconfigured during speed switching.<br><ul style="list-style-type: none"> <li>• 25G speed: 2.56 ns</li> <li>• 10G speed: 6.4 ns</li> </ul>  | 0x28F5C        | RW     |
| 0xA06–0xA0A | Reserved           |        | Reserved   | 96'b0          | RO     |
| 0xA0B       | TX_PTP_ASYM_DELAY  | [18:0] | Asymmetry adjustment as required for delay measurement. The IP core adds this value to the final delay.<br><ul style="list-style-type: none"> <li>• Bit[18]: The value of 1 enables the feature and the value of 0 disables the feature.</li> <li>• Bit[17]: The value of 1 indicates subtraction and the value of 0 indicates addition. Depending on the value of this bit, the value in bits [16:0] is added to or subtracted from the final delay.</li> <li>• Bits[16:0]: Asymmetry adjustment in nanoseconds.</li> </ul> | 19'b0          | RW     |
| 0xA0C       | TX_PTP_PMA_LATENCY | [31:0] | Latency through the TX PMA. This is the delay from the TX PCS output to the tx_serial pin.<br><ul style="list-style-type: none"> <li>• Bits[31:16]: Full nanoseconds (ns)</li> <li>• Bits[15:0]: Fraction of a nanosecond</li> </ul> In Intel Stratix 10 devices, the TX_PTP_PMA_LATENCY value is speed dependent and needs to be reconfigured during speed switching. The following are the TX PMA latency values for 25G and 10G speed rates:<br>25G speed:  | 32'b0          | RW     |

*continued...*

| Addr | Name | Bit | Description   | HW Reset Value | Access |
|------|------|-----|---|----------------|--------|
|      |      |     | <ul style="list-style-type: none"> <li>Digital delay: 187 UI<br/><i>Note:</i> 1 UI is approximately 38.8 ps.</li> <li>Analog delay: <ul style="list-style-type: none"> <li>ES silicon: 3.2 ns</li> <li>Production silicon: 4.44 ns</li> </ul> </li> <li>Total delay: <ul style="list-style-type: none"> <li>ES silicon: 10.456 ns</li> <li>Production silicon: 11.7 ns</li> </ul> </li> </ul> 10G speed: <ul style="list-style-type: none"> <li>Digital delay: 187 UI<br/><i>Note:</i> 1 UI is approximately 97 ps.</li> <li>Analog delay: <ul style="list-style-type: none"> <li>ES silicon: 2.196 ns</li> <li>Production silicon: 5.3 ns</li> </ul> </li> <li>Total delay: <ul style="list-style-type: none"> <li>ES silicon: 20.335 ns</li> <li>Production silicon: 23.439 ns</li> </ul> </li> </ul> |                |        |

Table 33. RX 1588 PTP Registers

| Addr  | Name               | Bit    | Description  | HW Reset Value | Access |
|-------|--------------------|--------|--|----------------|--------|
| 0xB00 | RXPTP_REVID        | [31:0] | IP core revision ID.   | 0x0504_2018    | RO     |
| 0xB01 | RXPTP_SCRATCH      | [31:0] | Scratch register available for testing.  | 32'b0          | RW     |
| 0xB02 | RXPTP_NAME_0       | [31:0] | First 4 characters of IP core variation identifier string "25GERXPTPCSR"   | 0x3235_4745    | RO     |
| 0xB03 | RXPTP_NAME_1       | [31:0] | Next 4 characters of IP core variation identifier string "25GERXPTPCSR"  | 0x5258_5054    | RO     |
| 0xB04 | RXPTP_NAME_2       | [31:0] | Final 4 characters of IP core variation identifier string "25GERXPTPCSR"   | 0x5043_5352    | RO     |
| 0xB05 | RX_PTP_CLK_PERIOD  | [19:0] | clk_rxmac clock period.<br>Bits [19:16]: Full nanoseconds (ns)<br>Bits [15:0]: Fraction of a nanosecond<br>The value of RX_PTP_CLK_PERIOD is speed dependent and needs to be reconfigured during speed switching. <ul style="list-style-type: none"> <li>25G speed: 2.56 ns</li> <li>10G speed: 6.4 ns</li> </ul>  | 0x28F5C        | RW     |
| 0xB06 | RX_PTP_PMA_LATENCY | [31:0] | Latency through the RX PMA. This is the delay from the rx_serial pin to the RX PCS input. <ul style="list-style-type: none"> <li>Bits[31:16]: Full nanoseconds</li> <li>Bits[15:0]: Fraction of a nanosecond</li> </ul> In Intel Stratix 10 devices, the RX_PTP_PMA_LATENCY value is speed dependent and needs to be reconfigured during speed switching. The following are the RX PMA latency values for 25G and 10G speed rates:<br>25G speed: | 32'b0          | RW     |

continued...

| Addr | Name | Bit | Description  | HW Reset Value | Access |
|------|------|-----|--|----------------|--------|
|      |      |     | <ul style="list-style-type: none"> <li>Digital delay: 104.5 UI<br/><i>Note:</i> 1 UI is approximately 38.8 ps.</li> <li>Analog delay:               <ul style="list-style-type: none"> <li>ES silicon: 0.139 ns</li> <li>Production silicon: 1.38 ns</li> </ul> </li> <li>Total delay:               <ul style="list-style-type: none"> <li>ES silicon: 4.194 ns</li> <li>Production silicon: 5.435 ns</li> </ul> </li> </ul> 10G speed: <ul style="list-style-type: none"> <li>Digital delay: 104.5 UI<br/><i>Note:</i> 1 UI is approximately 97 ps.</li> <li>Analog delay:               <ul style="list-style-type: none"> <li>ES silicon: -1.194 ns</li> <li>Production silicon: 1.91 ns</li> </ul> </li> <li>Total delay:               <ul style="list-style-type: none"> <li>ES silicon: 8.943 ns</li> <li>Production silicon: 12.047 ns</li> </ul> </li> </ul> |                |        |

**Related Information**

- [1588 PTP Interface Signals](#) on page 67
- [1588 Precision Time Protocol Interfaces](#) on page 44
- [PTP Transmit Functionality](#) on page 48

## 7.7. TX Reed-Solomon FEC Registers

**Table 34. TX Reed-Solomon FEC Registers**

| Addr  | Name             | Description  | Reset       | Access |
|-------|------------------|--|-------------|--------|
| 0xC00 | REVID            | Reed-Solomon FEC TX module revision ID.  | 0x0504_2018 | RO     |
| 0xC01 | TX_RSFECC_NAME_0 | First 4 characters of IP core variation identifier string, "25geRSFECCoTX".  | 0x3235_6765 | RO     |
| 0xC02 | TX_RSFECC_NAME_1 | Middle 4 characters of IP core variation identifier string, "25geRSFECCoTX".   | 0x5253_4645 | RO     |
| 0xC03 | TX_RSFECC_NAME_2 | Final 4 characters of IP core variation identifier string, "25geRSFECCoTX".  | 0x436F_5458 | RO     |
| 0xC04 | ERR_INS_EN       | Configuration register to enable error insertion in RS-FEC transmitter. Writing 1'b1 enables the feature. Writing 1'b0 disables it. The following encodings are defined: <ul style="list-style-type: none"> <li>• Bit[4]: Enable error insertion for single FEC codeword. Bit self-clears after error is inserted.</li> <li>• Bit[0]: Enable error insertion for every FEC codeword.</li> <li>• All other bits: Reserved.</li> </ul> | 0x00000000  | RW     |
| 0xC05 | ERR_MASK         | Specifies the bit masks for symbols and bits in a group for error injection. Each FEC codeword consists of 528 symbols of 10 bits each. The encoder works on groups of 8 symbols (80 bits). Therefore, each FEC  | 0x00000000  | RW     |

*continued...*

| Addr  | Name          | Description  | Reset      | Access |
|-------|---------------|--|------------|--------|
|       |               | <p>codeword consists of 66 groups. Writing 1'b1 enables the feature. Writing 1'b0 disables it. The following encodings are defined:</p> <ul style="list-style-type: none"> <li>• Bits[25:16]: Bit mask.</li> <li>• Bits[15:8]: Symbol mask.</li> <li>• Bits[6:0]: Group number (0-65).</li> <li>• Other bits: Reserved.</li> </ul> |            |        |
| 0xC06 | BYPASS_RSFECC | <p>Bypass RS-FEC core. Used by both TX and RX RS-FEC cores. Writing 1'b1 enables the feature. Writing 1'b0 disables it. The following encodings are defined:</p> <ul style="list-style-type: none"> <li>• Bit[0]: Bypass RS-FEC core.</li> <li>• All other bits: Reserved.</li> </ul>  | 0x00000000 | RW     |

## 7.8. RX Reed-Solomon FEC Registers

**Table 35. RX Reed-Solomon FEC Registers**

| Addr  | Name             | Description   | Reset       | Access |
|-------|------------------|---|-------------|--------|
| 0xD00 | REVID            | RS-FEC TX module revision ID  | 0x0504_2018 | RO     |
| 0xD01 | RX_RSFECC_NAME0  | First 4 characters of IP core variation identifier string, "25geRSFECCoRX".   | 0x3235_6765 | RO     |
| 0xD02 | RX_RSFECC_NAME1  | Middle 4 characters of IP core variation identifier string, "25geRSFECCoRX".  | 0x5253_4645 | RO     |
| 0xD03 | RX_RSFECC_NAME2  | Final 4 characters of IP core variation identifier string, "25geRSFECCoRX".   | 0x436F_5258 | RO     |
| 0xD04 | BYPASS_RESTART   | <p>Configuration register to bypass error correction and to restart alignment marker synchronization. Writing 1'b1 enables the feature. Writing 1'b0 disables it. The following encodings are defined:</p> <ul style="list-style-type: none"> <li>• Bit[0]: Bypass error correction. The RS-FEC core remains enabled but does not correct errors.</li> <li>• Bit[4]: Restarts FEC alignment marker synchronization. Bit clears after alignment marker synchronization is restarted.</li> <li>• All other bits: Reserved.</li> </ul> | 0x0000_0000 | RW     |
| 0xD05 | FEC_ALIGN_STATUS | <p>Alignment marker lock status. The following encodings are defined:</p> <ul style="list-style-type: none"> <li>• Bit[0]: Indicates alignment marker lock status. When 1'b1, indicates alignment has been achieved.</li> <li>• All other bits: reserved</li> </ul>   | 0x0000_0000 | RO     |
| 0xD06 | CORRECTED_CW     | 32-bit counter that contains the number of corrected FEC codewords processed. The value resets to zero upon read and holds at max count.  | 0x0000_0000 | RO     |
| 0xD07 | UNCORRECTED_CW   | 32-bit counter that contains the number of uncorrected FEC codewords processed. The value resets to zero upon read and holds at max count.  | 0x0000_0000 | RO     |

## 8. Debugging the Link

---

Begin debugging your link at the most basic level, with word lock. Then, consider higher level issues.

The following steps should help you identify and resolve common problems that occur when bringing up a 25G Ethernet Intel FPGA IP core link:

1. Establish word lock—The RX lanes should be able to achieve word lock even in the presence of extreme bit error rates. If the IP core is unable to achieve word lock, check the transceiver clocking and data rate configuration. Check for cabling errors such as the reversal of the TX and RX lanes. Check the clock frequency monitors (`KHZ_TX`, `KHZ_RX` PHY registers) in the Control and Status registers.

To check for word lock: Clear the `FRM_ERR` register by writing the value of 1 followed by another write of 0 to the `SCLR_FRM_ERR` register at offset 0x324. Then read the `FRM_ERR` register at offset 0x323. If the value is zero, the core has word lock. If non-zero the status is indeterminate

2. When having problems with word lock, check the `EIO_FREQ_LOCK` register at address 0x321. The values in this register define the status of the recovered clock. In normal operation, all the bits should be asserted. A non-asserted (value-0) or toggling logic value on the bit that corresponds to any lane, indicates a clock recovery problem. Clock recovery difficulties are typically caused by the following problems:
  - Bit errors
  - Failure to establish the link
  - Incorrect clock inputs to the IP core
3. Check the PMA FIFO levels by selecting appropriate bits in the `EIO_FLAG_SEL` register and reading the values in the `EIO_FLAGS` register. During normal operation, the TX and RX FIFOs should be nominally filled. Observing a the TX FIFO is either empty or full typically indicates a problem with clock frequencies. The RX FIFO should never be full, although an empty RX FIFO can be tolerated.
4. Establish lane integrity—When operating properly, the lanes should not experience bit errors at a rate greater than roughly one per hour per day. Bit errors within data packets are identified as FCS errors. Bit errors in control information, including IDLE frames, generally cause errors in XL/CGMII decoding.
5. Verify packet traffic—The Ethernet protocol includes automatic lane reordering so the higher levels should follow the PCS. If the PCS is locked, but higher level traffic is corrupted, there may be a problem with the remote transmitter virtual lane tags.
6. Tuning—You can adjust transceiver analog parameters to improve the bit error rate.



In addition, your IP core can experience loss of signal on the Ethernet link after it is established. In this case, the TX functionality is unaffected, but the RX functionality is disrupted. The following symptoms indicate a loss of signal on the Ethernet link:

- The IP core deasserts the `rx_pcs_ready` signal, indicating the IP core has lost alignment marker lock.
- The IP core deasserts the RX PCS fully aligned status bit (bit [0]) of the `RX_PCS_FULLY_ALIGNED_S` register at offset 0x326. This change is linked to the change in value of the `rx_pcs_ready` signal.
- If **Enable link fault generation** is turned on, the IP core sets `local_fault_status` to the value of 1.
- The IP core asserts the `Local Fault Status` bit (bit [0]) of the `Link_Fault` register at offset 0x508. This change is linked to the change in value of the `local_fault_status` signal.
- The IP core triggers the RX digital reset process by asserting `soft_rxp_rst`.

#### Related Information

[L- and H-Tile Transceiver PHY User Guide](#)

For information about the analog parameters for Intel Stratix 10 devices.

## 8.1. Error Insertion Test and Debugging

Error insertion allows you to test 25G Ethernet Intel FPGA IP core test error handling.

To use this feature, the Avalon streaming TX client asserts `l1_tx_error` in the same cycle as `l1_tx_endofpacket`. The error appears as a 66-bit error block that consists of eight `/E/` characters (`EBLOCK_T`) in the Ethernet frame. The 25G Ethernet Intel FPGA IP core overwrites Ethernet frame data with an `EBLOCK_T` error block when it transmits the Ethernet frame that corresponds to the packet EOP. The RX interface detects the frame corruption resulting in a CRC error output.

## 9. 25G Ethernet Intel Stratix 10 FPGA IP User Guide Archives

---

IP versions are the same as the Intel Quartus Prime Design Suite software versions up to v19.1. From Intel Quartus Prime Design Suite software version 19.2 or later, IP cores have a new IP versioning scheme.

If an IP core version is not listed, the user guide for the previous IP core version applies.

| Intel Quartus Prime Version | IP Core Version | User Guide   |
|-----------------------------|-----------------|--|
| 20.3                        | 19.4.0          | <a href="#">25G Ethernet Intel Stratix 10 FPGA IP User Guide</a> |
| 20.1                        | 19.4.0          | <a href="#">25G Ethernet Intel Stratix 10 FPGA IP User Guide</a> |
| 19.4                        | 19.4.0          | <a href="#">25G Ethernet Intel Stratix 10 FPGA IP User Guide</a> |
| 19.3                        | 19.3.0          | <a href="#">25G Ethernet Intel Stratix 10 FPGA IP User Guide</a> |
| 19.2                        | 19.2.0          | <a href="#">25G Ethernet Intel Stratix 10 FPGA IP User Guide</a> |
| 19.1                        | 19.1            | <a href="#">25G Ethernet Intel Stratix 10 FPGA IP User Guide</a> |
| 18.1                        | 18.1            | <a href="#">25G Ethernet Intel Stratix 10 FPGA IP User Guide</a> |
| 18.0                        | 18.0            | <a href="#">25G Ethernet Intel Stratix 10 FPGA IP User Guide</a> |

## 10. Document Revision History for the 25G Ethernet Intel Stratix 10 FPGA IP User Guide

| Document Version | Intel Quartus Prime Version | IP Version | Changes  |
|------------------|-----------------------------|------------|--|
| 2021.03.29       | 21.1                        | 19.4.0     | <p>Updated the descriptions for the following signals in Table: <i>Signals of the 1588 Precision Time Protocol Interface</i>:</p> <ul style="list-style-type: none"> <li>tx_etstamp_ins_ctrl_residence_time_calc_format</li> <li>tx_egress_timestamp_64b_data[63:0]</li> <li>tx_egress_timestamp_96b_fingerprint[(W-1):0]</li> <li>tx_egress_timestamp_64b_fingerprint[(W-1):0]</li> </ul>   |
| 2021.01.29       | 20.3                        | 19.4.0     | <p>Updated the descriptions for the following signals in Table: <i>Signals of the PHY Interface</i>:</p> <ul style="list-style-type: none"> <li>tx_clkout</li> <li>tx_clkout2</li> <li>rx_clkout</li> <li>rx_clkout2</li> </ul>  |
| 2020.10.12       | 20.3                        | 19.4.0     | <ul style="list-style-type: none"> <li>Added a note to the <i>Length Checking</i> section to state that the MAC has a counter limit of 0xFFFF starting from Intel Quartus Prime Pro Edition software version 20.3 onward.</li> <li>Added a note to the <i>Transceivers</i> section to state that the Intel Stratix 10 devices use the OSC_CLK_1 pin to provide the transceiver calibration clock source.</li> <li>Made editorial updates throughout the document.</li> </ul> |
| 2020.07.29       | 20.1                        | 19.4.0     | Added the channel_reset signal to Table: <i>Reset Signals</i> .  |
| 2020.06.22       | 20.1                        | 19.4.0     | <ul style="list-style-type: none"> <li>Added a new section—<i>Accessing the Native PHY Registers in L-Tile Devices</i>.</li> <li>Renamed section title <i>Disabling Background Calibration</i> to <i>Accessing the Native PHY Registers in H-Tile Devices</i>.</li> <li>Updated the <i>Length/Type Field Processing</i> section.</li> </ul>  |

**continued...**

| Document Version | Intel Quartus Prime Version | IP Version | Changes   |
|------------------|-----------------------------|------------|---|
|                  |                             |            | <ul style="list-style-type: none"> <li>Update the descriptions to the following signals in Table: <i>Avalon Streaming TX Datapath</i>:               <ul style="list-style-type: none"> <li>11_tx_data[63:0]</li> <li>11_tx_valid</li> <li>11_tx_ready</li> </ul> </li> <li>Removed Figure: <i>Client to 25G Ethernet Intel FPGA IP MAC Avalon Streaming Interface</i>.</li> <li>Added the following Figures:               <ul style="list-style-type: none"> <li><i>Client to 25G Ethernet Intel FPGA IP MAC Avalon Streaming Interface when Ready Latency is 0 (1 of 2)</i></li> <li><i>Client to 25G Ethernet Intel FPGA IP MAC Avalon Streaming Interface when Ready Latency is 0 (2 of 2)</i></li> <li><i>Client to 25G Ethernet Intel FPGA IP MAC Avalon Streaming Interface when Ready Latency is 3 (1 of 2)</i></li> <li><i>Client to 25G Ethernet Intel FPGA IP MAC Avalon Streaming Interface when Ready Latency is 3 (2 of 2)</i></li> </ul> </li> </ul>  |
| 2020.04.13       | 20.1                        | 19.4.0     | <ul style="list-style-type: none"> <li>Added a new Table: <i>IP Core Round Trip Latency</i>.</li> <li>Updated the following tables:               <ul style="list-style-type: none"> <li><i>IP Core FPGA Resource Utilization for 25G Ethernet Intel FPGA IP Core with MAC+PCS +PMA Core Variant for Intel Stratix 10 Devices</i>.</li> <li><i>IP Core FPGA Resource Utilization for 25G Ethernet Intel FPGA IP Core with MAC+PCS Core Variant for Intel Stratix 10 Devices</i>.</li> </ul> </li> <li>Updated the <i>Simulating the IP Core</i> section.</li> <li>Updated the <i>Length Checking</i> section.</li> <li>Updated the description for 11_rx_error[5:0] in Table: <i>Avalon Streaming RX Datapath</i>.</li> <li>Updated the descriptions for CNTR_RX_1519 to MAXB_HI, CNTR_RX_OVERSIZE_LO, and CNTR_RX_OVERSIZE_HI in Table: <i>Receive Side Statistics Registers</i>.</li> <li>Updated the description for latency_sclk in Table: <i>Signals of the 1588 Precision Time Protocol Interface</i>.</li> <li>Updated the descriptions for tx_control_phy[1:0] and rx_control_phy[1:0] in Table: <i>Signals of the PHY Interface</i>.</li> <li>Added a note to the description of PHY_TLKIT_ACCESS in Table: <i>PHY Registers</i>.</li> </ul> |
| 2020.02.21       | 19.4                        | 19.4.0     | <ul style="list-style-type: none"> <li>Updated the description for frame monitoring and statistics in the <i>25G Ethernet Intel FPGA IP Core Supported Features</i> section.</li> <li>Updated the <i>Debugging the Link</i> section.</li> </ul>   |
|                  |                             |            | <b>continued...</b>   |

| Document Version | Intel Quartus Prime Version | IP Version | Changes  |
|------------------|-----------------------------|------------|--|
| 2019.12.16       | 19.4                        | 19.4.0     | <ul style="list-style-type: none"> <li>• Updated the description in the <i>About the 25G Ethernet Intel FPGA IP Core</i> section.</li> <li>• Updated the description for debug and testability features in the <i>25G Ethernet Intel FPGA IP Core Supported Features</i> section.</li> <li>• Updated the following tables:               <ul style="list-style-type: none"> <li>– <i>IP Core FPGA Resource Utilization for 25G Ethernet Intel FPGA IP Core with MAC+PCS +PMA Core Variant for Intel Stratix 10 Devices.</i></li> <li>– <i>IP Core FPGA Resource Utilization for 25G Ethernet Intel FPGA IP Core with MAC+PCS Core Variant for Intel Stratix 10 Devices.</i></li> </ul> </li> <li>• Updated the description in the <i>PTP Transmit Functionality</i> section.</li> <li>• Updated the descriptions for the following signals in Table: <i>Signals of the 1588 Precision Time Protocol Interface</i>:               <ul style="list-style-type: none"> <li>– tx_etstamp_ins_ctrl_offset_timestamp[15:0]</li> <li>– tx_etstamp_ins_ctrl_offset_correction_field[15:0]</li> <li>– tx_etstamp_ins_ctrl_offset_checksum_field[15:0]</li> <li>– tx_etstamp_ins_ctrl_offset_checksum_correction[15:0]</li> </ul> </li> <li>• Added rx_am_lock to Table: <i>Miscellaneous Status and Debug Signals</i>.</li> <li>• Updated the description and reset value for RXMAC_CONTROL and description for LINK_FAULT in Table: <i>RX MAC Registers</i>.</li> <li>• Added reset_status signal to Table: <i>Avalon Memory-Mapped Management Interface</i>.</li> </ul> |

**continued...**

| Document Version | Intel Quartus Prime Version | IP Version | Changes   |
|------------------|-----------------------------|------------|---|
|                  |                             |            | <ul style="list-style-type: none"> <li>Updated the <i>Avalon Memory-Mapped Management Interface</i> section.</li> <li>Updated the <i>Statistics Registers</i> section.</li> <li>Updated for latest Intel branding standards.</li> </ul>   |
| 2019.10.11       | 19.3                        | 19.3.0     | <ul style="list-style-type: none"> <li>Updated the description in the <i>About the 25G Ethernet Intel FPGA IP Core</i> section.</li> <li>Updated the PHY feature description in the <i>25G Ethernet Intel FPGA IP Core Supported Features</i> section.</li> <li>Updated the description in the <i>Hardware Testing</i> section.</li> <li>Updated the description for 0x800, 0x801, 0x802, 0x803, 0x804, 0x805, 0x834, and 0x835 in Table: <i>Transmit Side Statistics Registers</i>.</li> <li>Updated the descriptions for rx_block_lock and rx_pcs_ready in Table: <i>Miscellaneous Status and Debug Signals</i>.</li> </ul> |
| 2019.08.29       | 19.2                        | 19.2.0     | <ul style="list-style-type: none"> <li>Added PHY_TLKIT_ACCESS register to Table: <i>PHY Registers</i>.</li> <li>Updated the description for CNTR_RX_RUNT_LO and CNTR_RX_RUNT_HI in Table: <i>Receive Side Statistics Registers</i>.</li> <li>Updated the l2_rxstatus_data bits to l1_rxstatus_data bits in the <i>Length/Type Field Processing</i> section.</li> <li>Updated l2_rx_error[2], l2_rx_error[3], and l2_rx_error[4] to l1_rx_error[2], l1_rx_error[3], and l1_rx_error[4] in the <i>Length Checking</i> section.</li> <li>Updated the steps in the <i>Disabling Background Calibration</i> section.</li> </ul>    |

| Document Version | Intel Quartus Prime Version | Changes   |
|------------------|-----------------------------|---|
| 2019.04.05       | 19.1                        | <ul style="list-style-type: none"> <li>Added a new IP core parameter—<b>Enable auto adaptation triggering for RX PMA CTLE/DFE mode</b>.</li> <li>Added a new Topic: <i>Disabling Background Calibration</i>.</li> <li>Updated the <i>25G Ethernet Intel FPGA IP Core Supported Features</i> to state support for adaptive mode for RX PMA Adaptation.</li> <li>Renamed Altera Debug Master Endpoint (ADME) to Native PHY Debug Master Endpoint (NPDME).</li> <li>Updated the <i>Adding the Transceiver PLL</i> topic.</li> <li>Updated the <i>Placement Settings for the 25G Ethernet Intel FPGA IP Core</i> topic.</li> <li>Updated the <i>Flow Control</i> topic.</li> <li>Updated the <i>XON/XOFF Pause Frames</i> topic.</li> <li>Updated the <i>Transceivers</i> topic.</li> <li>Updated the second note in the <i>Control, Status, and Statistics Register Descriptions</i> topic.</li> </ul> |

**continued...**



| Document Version | Intel Quartus Prime Version | Changes   |
|------------------|-----------------------------|---|
|                  |                             | <ul style="list-style-type: none"> <li>• Updated the following Tables:                             <ul style="list-style-type: none"> <li>— Updated Table: <i>Supported Device Speed Grades</i> to update the second footnote for Intel Stratix 10 L- and H-tile device family.</li> <li>— Updated Table: <i>IP Core FPGA Resource Utilization for 25G Ethernet Intel FPGA IP Core with MAC+PCS+PMA Core Variant for Intel Stratix 10 Devices</i>.</li> <li>— Updated Table: <i>IP Core FPGA Resource Utilization for 25G Ethernet Intel FPGA IP Core with MAC+PCS Core Variant for Intel Stratix 10 Devices</i>.</li> <li>— Updated Table: <i>Transceiver Signals</i> to update the direction values for tx_serial_clk0 and tx_serial_clk1.</li> </ul> </li> <li>• Made minor topic restructuring to the <i>Core Functional Description</i> section.</li> <li>• Made editorial updates throughout the document.</li> </ul>   |
| 2019.01.02       | 18.1                        | <ul style="list-style-type: none"> <li>• Removed the reference to Intel Stratix 10 E-tile devices because 25G Ethernet Intel FPGA IP core supports Intel Stratix 10 H-tile and L-tile devices only.</li> <li>• Updated Table: <i>Supported Device Speed Grades</i> to add a footnote to clarify that Intel Stratix 10 devices with both E- and H-tile transceivers are supported if the IP core is only utilizing the H-tile transceiver.</li> <li>• Added a note to <i>Control, Status, and Statistics Register Descriptions</i> topic.</li> </ul>   |
| 2018.10.05       | 18.1                        | <p>Updated Table: <i>PHY Registers</i> to correct the bit[1] description for RX_PCS_FULLY_ALIGNED_S.</p>  |
| 2018.10.03       | 18.1                        | <ul style="list-style-type: none"> <li>• Added a new feature—Elective PMA.</li> <li>• Added a new signal for 1588 Precision Time Protocol Interface—latency_sclk.</li> <li>• Updated the <i>About the 25G Ethernet Intel FPGA IP Core</i> topic:                             <ul style="list-style-type: none"> <li>— Updated notes in the topic.</li> <li>— Updated Figure title <i>25G Ethernet Intel FPGA IP MAC IP Clock Diagram to 25G Ethernet MAC, PCS, and PMA IP Clock Diagram</i>.</li> <li>— Updated Figure title <i>10G/25G Ethernet MAC IP Clock Diagram to 10G/25G Ethernet MAC, PCS, and PMA IP Clock Diagram</i>.</li> <li>— Added new Figures:                                     <ul style="list-style-type: none"> <li>• <i>25G Ethernet MAC and PCS IP Clock Diagram</i>.</li> <li>• <i>10G/25G Ethernet MAC and PCS IP Clock Diagram</i>.</li> </ul> </li> <li>— Updated Table: <i>IP Core Parameters</i> to include <b>Core Variants</b> parameter.</li> <li>— Added new topic: <i>PHY Interface Signals</i>.</li> <li>— Updated Figure: <i>25G Ethernet Intel FPGA IP Signals and Interfaces</i> to include PHY interface signals.</li> </ul> </li> <li>• Updated the <i>Performance and Resource Utilization</i> topic:                             <ul style="list-style-type: none"> <li>— Added new Tables:                                     <ul style="list-style-type: none"> <li>• <i>IP Core Variation Encoding for Resource Utilization for MAC+PCS Core Variant</i>.</li> <li>• <i>IP Core FPGA Resource Utilization for 25G Ethernet Intel FPGA IP Core with MAC+PCS Core Variant for Intel Stratix 10 Devices</i>.</li> </ul> </li> <li>— Updated Table title <i>IP Core Variation Encoding for Resource Utilization</i> to <i>IP Core Variation Encoding for Resource Utilization for MAC+PCS+PMA Core Variant</i>.</li> <li>— Updated Table title <i>IP Core FPGA Resource Utilization for 25G Ethernet Intel FPGA IP Core for Intel Stratix 10 Devices</i> to <i>IP Core FPGA Resource Utilization for 25G Ethernet Intel FPGA IP Core with MAC+PCS+PMA Core Variant for Intel Stratix 10 Devices</i>.</li> <li>— Updated Table: <i>IP Core Parameters</i> to update the description for <b>Enable flow control</b>.</li> </ul> </li> </ul> |

continued...

| Document Version | Intel Quartus Prime Version | Changes  |
|------------------|-----------------------------|--|
|                  |                             | <ul style="list-style-type: none"> <li>• Updated the descriptions of the following topics:               <ul style="list-style-type: none"> <li>– <i>Flow Control</i></li> <li>– <i>XON/XOFF Pause Frames</i></li> </ul> </li> <li>• Updated the <i>Installing and Licensing Intel FPGA IP Cores</i> topic to remove Intel Quartus Prime Standard Edition software references.</li> <li>• Updated the following Tables:               <ul style="list-style-type: none"> <li>– Updated Table: <i>Supported Device Speed Grades</i>:                   <ul style="list-style-type: none"> <li>• Updated Table title <i>Slowest Supported Device Speed Grades</i> to <i>Supported Device Speed Grades</i>.</li> <li>• Updated the Intel Stratix 10 device family to include L-tile, H-tile, and E-tile support.</li> <li>• Added a footnote for Intel Stratix 10 device family to state that only Intel Stratix 10 devices ending with "VG", VGS3", and "LG" suffixes in the part number are supported.</li> </ul> </li> <li>– Updated Table: <i>25G Ethernet Intel FPGA IP Core Current Release Information</i>.</li> <li>– Updated Table: <i>IP Core Generated Files</i> to remove &lt;your_ip&gt;.debuginfo filename.</li> <li>– Updated Table: <i>PHY Registers</i>:                   <ul style="list-style-type: none"> <li>• Added bit[1] description for RX_PCS_FULLY_ALIGNED_S.</li> <li>• Updated the descriptions for KHZ_RX and KHZ_TX.</li> </ul> </li> </ul> </li> <li>• Updated the following Figures:               <ul style="list-style-type: none"> <li>– Updated Figure: <i>IP Core Generated Files</i></li> <li>– Updated Figure: <i>25G Ethernet Intel FPGA IP Core with MAC, PCS, and PMA Clock Diagram</i></li> <li>– Updated Figure: <i>High Level Block Diagram of the TX PCS with Optional RS-FEC Datapath.</i> :                   <ul style="list-style-type: none"> <li>• Updated figure to include RS-FEC block.</li> <li>• Updated figure title from <i>High Level Block Diagram of the Soft TX PCS</i> to <i>High Level Block Diagram of the TX PCS with Optional RS-FEC Datapath</i>.</li> <li>• Updated Figure: <i>High Level Block Diagram of the RX PCS with Optional RS-FEC Datapath.</i> :                       <ul style="list-style-type: none"> <li>– Updated figure to include RS-FEC block.</li> <li>– Updated figure title from <i>High Level Block Diagram of the Soft RX PCS</i> to <i>High Level Block Diagram of the RX PCS with Optional RS-FEC Datapath</i>.</li> </ul> </li> </ul> </li> </ul> </li> <li>• Made editorial updates throughout the document.</li> </ul> |
| 2018.07.17       | 18.0                        | <ul style="list-style-type: none"> <li>• Updated Table: <i>TX 1588 PTP Registers</i> to correct the HW reset value of the TX_PTP_CLK_Period register to 0x28F5C.</li> <li>• Updated Table: <i>RX 1588 PTP Registers</i> to update the description and correct the HW reset value of the RX_PTP_CLK_Period register to 0x28F5C.</li> </ul>  |
| 2018.06.06       | 18.0                        | Initial release.   |