Low Latency 100-Gbps Ethernet IP Core User Guide

Updated for Intel® Quartus® Prime Design Suite: **16.1**

Online Version
Send Feedback

UG-20044

ID: 683160
Version: 2021.04.15

Contents

1. About the LL 100GbE IP Core

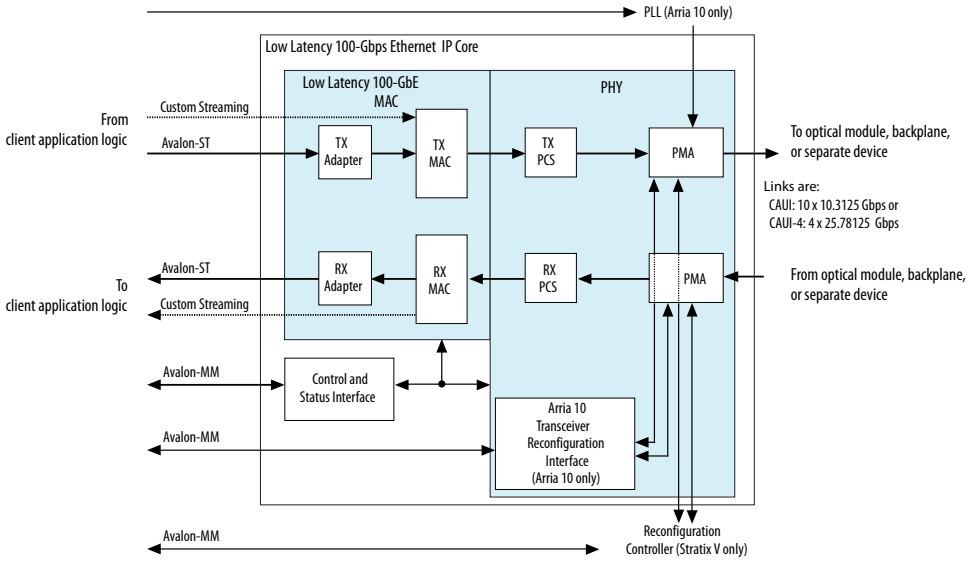
The Intel® Low Latency 100-Gbps Ethernet (LL 100GbE) media access controller (MAC) and PHY Intel FPGA IP functions offer the lowest round-trip latency and smallest size to implement the *IEEE 802.3ba High Speed Ethernet Standard* with an option to support the *IEEE 802.3ap-2007 Backplane Ethernet Standard*.

The version of this product that supports Intel Arria® 10 devices is included in the Intel FPGA IP Library and is available from the Intel Quartus® Prime IP Catalog.

Note: The full product name, Low Latency 100-Gbps Ethernet MAC and PHY Intel FPGA IP Function, is shortened to Low Latency (LL) 100GbE (LL 100GbE) IP core in this document. In addition, although multiple variations are available from the parameter editor, this document refers to this product as a single IP core, because all variations are configurable from the same parameter editor.

Figure 1. LL 100GbE IP Core

Main blocks, internal connections, and external block requirements.



As illustrated, on the MAC client side you can choose a wide, standard Avalon® streaming interface (Avalon-ST), or a narrower, custom streaming interface. The MAC client side Avalon streaming interface data bus is 512 bits wide. The MAC client side custom streaming interface data bus is 256 bits wide. The client-side data maps to ten 10.3125 Gbps transceiver PHY links or to four 25.78125 Gbps transceiver PHY links.

The 100GbE (CAUI) interface has 10x10.3125 Gbps links. For Intel Arria 10 10 GT devices only, you can configure a 100GbE CAUI-4 option, with 4x25.78125 Gbps links.

The FPGA serial transceivers are compliant with the IEEE 802.3ba standard CAUI and CAUI-4 specifications. The IP core configures the transceivers to implement the relevant specification for your IP core variation. You can connect the transceiver interfaces directly to an external physical medium dependent (PMD) optical module or to another device.

The IP core provides standard MAC and physical coding sublayer (PCS) functions with a variety of configuration and status registers. You can exclude the statistics registers. 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.

Related Information

- LL 100GbE IP Core Functional Description on page 37
  Provides detailed descriptions of LL 100GbE IP core operation and functions.
- 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 Qsys 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.
- LL 100GbE IP Core User Guide Archives on page 124
- Low Latency 100G Ethernet Design Example User Guide

1.1. LL 100GbE IP Core Supported Features

All LL 100GbE IP core variations include both a MAC and a PHY, and all variations are in full-duplex mode. These IP core variations offer the following features:

- Designed to the *IEEE 802.3ba-2010 High Speed Ethernet Standard* available on the IEEE website (www.ieee.org).
- Soft PCS logic that interfaces seamlessly to Intel FPGA 10.3125 Gbps and 25.78125 Gbps serial transceivers.
- Standard CAUI external interface consisting of ten FPGA hard serial transceiver lanes operating at 10.3125 Gbps , or the CAUI-4 external interface consisting of four FPGA hard serial transceiver lanes operating at 25.78125 Gbps.
- Supports Synchronous Ethernet (SyncE) by providing an optional CDR recovered clock output signal to the device fabric.
- Avalon memory mapped (Avalon-MM) management interface to access the IP core control and status registers.
- Avalon streaming (Avalon-ST) data path interface connects to client logic with the start of frame in the most significant byte (MSB) when optional adapters are used. Interface has data width 512 bits.
- Optional custom streaming data path interface with narrower bus width and a start frame possible on 64-bit word boundaries without the optional adapters. Interface has data width 256 bits.
- Support for jumbo packets.

- TX and RX CRC pass-through control.
- Optional TX CRC generation and insertion.
- RX CRC checking and error reporting.
- TX error insertion capability supports test and debug.
- RX and TX preamble pass-through options for applications that require proprietary user management information transfer.
- TX automatic frame padding to meet the 64-byte minimum Ethernet frame length at the LL 100GbE Ethernet connection.
- RX malformed packet checking per IEEE specification.
- Hardware and software reset control.
- Pause frame filtering control.
- Received control frame type indication.
- MAC provides cut-through frame processing.
- Optional deficit idle counter (DIC) options to maintain a finely controlled 8-byte or 12-byte inter-packet gap (IPG) minimum average.
- Optional IEEE 802.3 Clause 31 Ethernet flow control operation using the pause registers or pause interface.
- Optional priority-based flow control that complies with the *IEEE Standard 802.1Qbb-2011—Amendment 17: Priority-based Flow Control*, using the pause registers for fine control.
- 1000 bits RX PCS lane skew tolerance, which exceeds the IEEE 802.3-2012 Ethernet standard clause 82.2.12 requirements.
- Optional support for the IEEE Standard 1588-2008 Precision Clock Synchronization Protocol (1588 PTP).
- Optional statistics counters.
- Optional fault signaling: detects and reports local fault and generates remote fault, with *IEEE 802.3ba-2012 Ethernet Standard* Clause 66 support.
- Optional serial PMA loopback (TX to RX) at the serial transceiver for self-diagnostic testing.
- Optional access to Native PHY Debug Master Endpoint (NPDME) for debugging or monitoring PHY signal integrity.

The LL 100GbE IP core can support full wire line speed with a 64-byte frame length and back-to-back or mixed length traffic with no dropped packets.

For a detailed specification of the Ethernet protocol refer to the *IEEE 802.3ba-2010 High Speed Ethernet Standard*.

Related Information

IEEE website

The *IEEE 802.3ba-2010 High Speed Ethernet Standard* and the *IEEE Standard 802.1Qbb-2011—Amendment 17: Priority-based Flow Control* are available on the IEEE website.

1.2. IP Core Device Family and Speed Grade Support

The following sections list the device family and device speed grade support offered by the Low Latency 100-Gbps Ethernet Intel FPGA IP:

LL 100GbE IP Core Device Family Support on page 7
LL 100GbE IP Core Device Speed Grade Support on page 7

1.2.1. LL 100GbE IP Core Device Family Support

Table 1. Intel FPGA IP Core Device Support Levels

Device Support Level	Definition
Preliminary	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.
Final	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 2. LL 100GbE IP Core Device Family Support

Shows the level of support offered by the LL 100GbE IP core for each Intel FPGA device family.

Device Family	Support
Stratix® V (GX, GT, and GS)	Final
Arria 10 (GX, GT, and SX)	Default support level provided in the Intel Quartus Prime software. Refer to the *Quartus Prime Standard Edition Software and Device Support Release Notes* and the *Quartus Prime Pro Edition Software and Device Support Release Notes.*
Other device families	Not supported

Related Information

- Timing and Power Models
  Reports the default device support levels in the current version of the Quartus Prime Standard Edition software.
- Timing and Power Models
  Reports the default device support levels in the current version of the Quartus Prime Pro Edition software.
- LL 100GbE IP Core Device Speed Grade Support on page 7
  Shows which IP core variations support which device family speed grades.

1.2.2. LL 100GbE IP Core Device Speed Grade Support

Table 3. Slowest Supported Device Speed Grades

Lists the slowest supported device speed grades for standard variations of the LL 100GbE IP core. IP core variations that include a 1588 PTP module might require Intel Quartus Prime seed sweeping to achieve a comfortable timing margin.

Intel FPGA IP Function	Device Family	Supported Speed Grades
LL 100GbE	Stratix V (GX)	I2, C2
	Stratix V (GT)	I2, C2

continued...

Intel FPGA IP Function	Device Family	Supported Speed Grades
	Stratix V (GS)	I2, C2
	Arria 10 (GX, GT, SX)	I2, C2
LL 100GbE (CAUI–4 option)	Arria 10 GT	I2, C2

1.3. IP Core Verification

To ensure functional correctness of the LL 100GbE IP core, Intel performs extensive validation through both simulation and hardware testing. Before releasing a version of the LL 100GbE IP core, Intel runs comprehensive regression tests in the current or associated version of the Intel Quartus Prime software.

Intel verifies that the current version of the Intel Quartus Prime 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 Errata for Low Latency 40-100GbE IP core
  Exceptions to functional correctness that first manifest in software releases prior to the 16.1 software release are documented in the Low Latency 40-100GbE IP core errata.
- Knowledge Base Errata for LL 100GbE IP core
  Exceptions to functional correctness that first manifest in software releases 16.0 and later are documented in the Low Latency 100GbE IP core errata.
- Intel FPGA IP Release Notes: Low Latency 100-Gbps Ethernet IP Core Release Notes
  Changes to the Low Latency 100GbE IP core in software releases 16.1 and earlier are noted in the Intel FPGA IP Release Notes.

1.3.1. Simulation Environment

Intel performs the following tests on the LL 100GbE 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
- Randomized error injection tests that inject Frame Check Sequence (FCS) field errors, runt packets, and corrupt control characters, and then check for the proper response from the IP core
- 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

1.3.2. Compilation Checking

Intel performs compilation testing on an extensive set of LL 100GbE IP core variations and designs that target different devices, to ensure the Intel Quartus Prime software places and routes the IP core ports correctly.

1.3.3. Hardware Testing

Intel performs hardware testing of the key functions of the LL 100GbE IP core using standard 100Gbps Ethernet network test equipment and optical modules. The Intel hardware tests of the LL 100GbE IP core also ensure reliable solution coverage for hardware related areas such as performance, link synchronization, and reset recovery.

1.4. Performance and Resource Utilization

The following sections provide performance and resource utilization data for the LL 100GbE IP core.

Table 4. IP Core Variation Encoding for Resource Utilization Tables

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

IP Core Variation / Parameter	A	B	C	D
Data interface	Custom-ST	Avalon-ST	Avalon-ST	Avalon-ST
Flow control mode	No flow control	No flow control	Standard flow control	Standard flow control
Average interpacket gap	12	12	12	12
Enable 1588 PTP	—	—	—	On
Enable link fault generation	—	—	On	On
Enable TX CRC insertion	—	On	On	On
Enable preamble passthrough	—	—	On	On
Enable alignment EOP on FCS word	—	On	On	On
Enable TX statistics	—	On	On	On
Enable RX statistics	—	On	On	On

1.4.1. Arria 10 Resource Utilization

Resource utilization changes depending on the parameter settings you specify in the LL 100GbE parameter editor. For example, if you turn on pause functionality or statistics counters in the LL 100GbE parameter editor, the IP core requires additional resources to implement the additional functionality.

Table 5. IP Core FPGA Resource Utilization in Arria 10 Devices

Lists the resources and expected performance for selected variations of the LL 100GbE IP core in an Arria 10 device.

These results were obtained using the Intel Quartus Prime software v16.1.

- 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.

LL 100GbE Variation	ALMs	Dedicated Logic Registers	Memory M20K
LL 100GbE variation A	13000	22800	29
LL 100GbE variation B	22500	47100	73
LL 100GbE variation C	22700	47900	73
LL 100GbE variation D	29600	64500	109

CAUI-4 Variation	ALMs	Dedicated Logic Registers	Memory M20K
CAUI-4 variation B	24200	50800	77
CAUI-4 variation B with RS-FEC	54100	113200	143

Related Information

Fitter Resources Reports in the Quartus Prime Help
Information about Quartus Prime resource utilization reporting, including **ALMs needed**.

1.4.2. Stratix V Resource Utilization

Resource utilization changes depending on the parameter settings you specify in the LL 100GbE parameter editor. For example, if you turn on pause functionality or statistics counters in the LL 100GbE parameter editor, the IP core requires additional resources to implement the additional functionality.

Table 6. IP Core FPGA Resource Utilization in Stratix V Devices

Lists the resources and expected performance for selected variations of the LL 100GbE IP core in a Stratix V device.

These results were obtained using the Quartus II software v14.1.

Note: Please note that at the time of publication, the LL 100GbE IP core that targets a Stratix V device has not been updated since the version compatible with the Intel Quartus Prime Standard Edition software v16.0.

- 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.

100GbE Variation	ALMs	Dedicated Logic Registers	Memory M20K
100GbE variation A	9500	23000	29
100GbE variation B	20900	48400	61
100GbE variation C	22100	52500	61
100GbE variation D	26900	63700	65

Related Information

Fitter Resources Reports in the Quartus Prime Help
Information about Quartus Prime resource utilization reporting, including **ALMs needed**.

1.5. Release Information

Table 7. LL 100GbE IP Core Current Release Information

Item	Description
Version	16.1
Release Date	2016.10.31
Ordering Codes	Low Latency 100G Ethernet MAC and PHY: IP-100GEUMACPHY Low Latency 100G Ethernet MAC and PHY with 1588: IP-100GEUMACPHYF
Vendor ID	6AF7

2. Getting Started

The following sections explain how to install, parameterize, simulate, and initialize the LL 100GbE IP core:

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 Qsys 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. Installation and Licensing for LL 100GbE IP Core for Stratix V Devices

The LL 100GbE IP core that targets the Stratix V device family is an extended IP core which is not included with the Intel Quartus Prime release. This section provides a general overview of the Intel extended FPGA IP core installation process to help you quickly get started with any Intel extended FPGA IP core.

The Intel extended FPGA IP cores are available from the Self-Service Licensing Center (SSLC). Refer to Related Links below for the correct link for this IP core.

Figure 2. IP Core Directory Structure

Directory structure after you install the LL 100GbE IP core that targets a Stratix V device.

Note: At the time of publication, the most recent Stratix V LL 100GbE IP core available from the SSLC is a combined Low Latency 40-100GbE IP core compatible with the Intel Quartus Prime Standard Edition software v16.0.

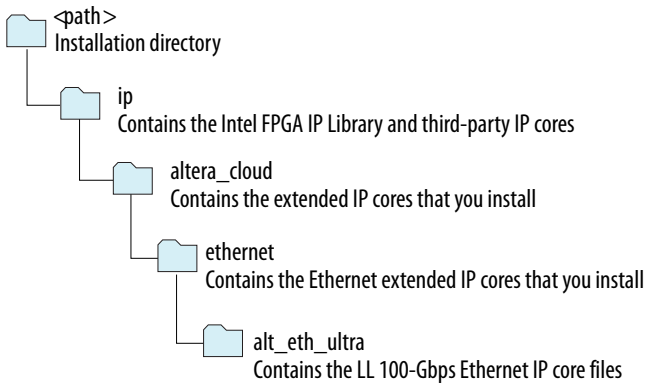


Table 8. Default Intel Quartus Prime Standard Edition Installation Locations

Lists the default location of *<path>*. The Stratix V LL 100GbE IP core available in the SSLC at the time of publication is compatible with the Intel Quartus Prime Standard Edition software v16.0.

Default Location in 16.1 Release	Default Location in 16.0 Release	Platform
`<drive>:\intelFPGA\quartus\<version number>\`	`<drive>:\altera\quartus\<version number>\`	Windows
`<home directory>:/intelFPGA/quartus/<version number>`	`<home directory>:/opt/altera/quartus/<version number>`	Linux

You can evaluate an IP core in simulation and in hardware until you are satisfied with its functionality and performance. You must purchase a license for the IP core when you want to take your design to production. After you purchase a license for an Intel FPGA IP core, you can request a license file from the Licensing page of the Intel website and install the license on your computer.

Related Information

- Intel website
- Intel Licensing website

- Intel Self-Service Licensing Center
  After you purchase the LL 100GbE IP core that supports Stratix V devices, the IP core is available for download from the SSLC page in your My Intel account. Intel requires that you create a My Intel account if you do not have one already, and log in to access the SSLC. On the SSLC page, click Run for this IP core. The SSLC provides an installation dialog box to guide your installation of the IP core.

2.2. Installing and Licensing Intel FPGA IP Cores

The Intel Quartus Prime 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 3. IP Core Installation Path

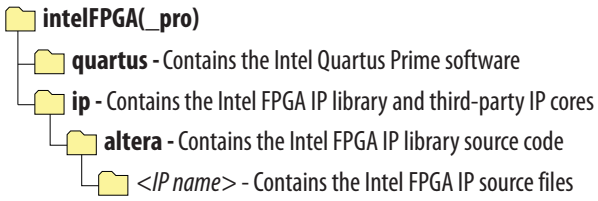


Table 9. IP Core Installation Locations

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

Note: The Intel Quartus Prime software does not support spaces in the installation path.

Related Information

Release Information on page 11
Provides the licensing product codes for the IP core.

2.2.1. 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.

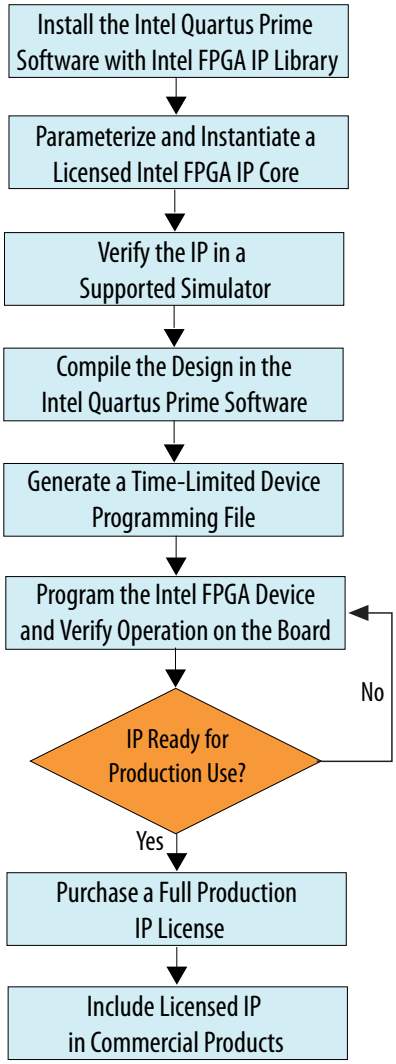
Intel FPGA IP Evaluation Mode supports the following operation modes:

- **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 4. 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

2.3. Specifying the IP Core Parameters and Options

The LL 100GbE parameter editor allows you to quickly configure your custom IP variation. Use the following steps to specify IP core options and parameters in the Quartus Prime software.

1. In the IP Catalog (**Tools ➤ IP Catalog**), select a target device family. The LL 100GbE IP core is not supported in Platform Designer (Standard).
2. In the IP Catalog, locate and double-click the name of the IP core to customize (**Low Latency 100G Ethernet**). The New IP Variation window appears.
3. Specify a top-level name for your custom IP variation. The parameter editor saves the IP variation settings in a file with one of the following names:
   - *<your_ip>*`.qsys` (for Arria 10 variations generated in the Intel Quartus Prime Standard Edition software)
   - *<your_ip>*`.ip` (for Arria 10 variations generated in the Intel Quartus Prime Pro Edition software)
   - *<your_ip>*`.qip` (for Stratix V variations)
4. If your IP core targets the Arria 10 device family, you must select a specific device in the **Device** field or maintain the default device the Quartus Prime software lists. If you target a specific Intel development kit, the hardware design example overwrites the selection with the device on the target board.
5. Click **OK**. The parameter editor appears.
6. Specify the parameters and options for your IP variation in the parameter editor, including one or more of the following. Refer to your IP core user guide for information about specific IP core parameters.
   - Specify parameters defining the IP core functionality, port configurations, and device-specific features.
   - Specify options for processing the IP core files in other EDA tools.
   - A functional VHDL IP core is not available. Specify Verilog HDL only, for your IP core variation.
7. For Arria 10 variations, follow these steps:
   a. Optionally, to generate a simulation testbench or example project, follow the instructions in Generating the LL 100GbE Testbench on page 32.
   b. Click **Generate HDL**. The **Generation** dialog box appears.
   c. Specify output file generation options, and then click **Generate**. The IP variation files generate according to your specifications.
   d. Click **Finish**. The parameter editor adds the top-level `.qsys` file to the current project automatically. If you are prompted to manually add the `.qsys` or `.ip` file to the project, click **Project ➤ Add/Remove Files in Project** to add the file.
8. For Stratix V variations, follow these steps:
   a. Click **Finish**.

b. Optionally, to generate a simulation testbench or example project, follow the instructions in Generating the LL 100GbE Testbench on page 32.
After you click Finish and optionally follow the additional step to generate a simulation testbench and example project, if available for your IP core variation, the parameter editor adds the top-level `.qip` file to the current project automatically. If you are prompted to manually add this file to the project, click **Project ➤ Add/Remove Files in Project** to add the file.

9. After generating and instantiating your IP variation, make appropriate pin assignments to connect ports.

2.4. IP Core Parameters

The LL 100GbE parameter editor provides the parameters you can set to configure the LL 100GbE IP core and simulation and hardware design examples.

LL 100GbE IP core variations that target an Arria 10 device include an **Example Design** tab. For information about that tab, refer to the *Low Latency 100G Ethernet Design Example User Guide*.

Table 10. LL 100GbE Parameters: Main Tab

Describes the parameters for customizing the LL 100GbE IP core on the Main tab of the LL 100GbE parameter editor.

Parameter	Type	Range	Default Setting	Parameter Description
General Options				
Device family	String	• Stratix V • Arria 10	According to the setting in the project or IP Catalog settings.	Selects the device family.
Data interface	String	• Custom–ST • Avalon–ST	Avalon–ST	Selects the Avalon streaming interface or the narrower, custom streaming client interface to the MAC. If you select the custom streaming client interface, the **Flow control mode** and **Enable 1588 PTP** parameters are not available.
PCS/PMA Options				
Enable CAUI4 PCS	Boolean	• True • False	False	If this parameter is turned on, the IP core is a 100GbE CAUI-4 variation, with four 25.78125 Gbps transceiver PHY links. If this parameter is turned off, the IP core is configured with the regular 100 Gbps PHY link option of 10 x 10.3125 Gbps. This parameter is available only in variations that target an Arria 10 device.
Enable RS-FEC for CAUI4	Boolean	• True • False	False	If this parameter is turned on, the IP core implements Reed-Solomon forward error correction (FEC). This parameter is available only in CAUI-4 variations. CAUI-4 variations must target an Arria 10 device.
Enable SyncE	Boolean	• True • False	False	Exposes the RX recovered clock as an output signal. This feature supports the Synchronous Ethernet standard described in the ITU-T G.8261, G.8262, and G.8264 recommendations.

continued...

Parameter	Type	Range	Default Setting	Parameter Description
				This parameter is available only in variations that target an Arria 10 device.
PHY reference frequency	Integer (encoding)	• 322.265625 MHz • 644.53125 MHz	644.53125 MHz	Sets the expected incoming PHY `clk_ref` reference frequency. The input clock frequency must match the frequency you specify for this parameter (± 100ppm).
Use external TX MAC PLL	Boolean	• True • False	False	If you turn this option on, the IP core is configured to expect an input clock to drive the TX MAC. The input clock signal is `clk_txmac_in`.
Flow Control Options				
Flow control mode	String	• No flow control • Standard flow control • Priority-based flow control	No flow control	Configures the flow control mechanism the IP core implements. Standard flow control is Ethernet standard flow control. If you select the custom streaming client interface, the IP core must be configured with no flow control, and this parameter is not available.
Number of PFC queues	Integer	1–8	8	Number of distinct priority queues for priority-based flow control. This parameter is available only if you set **Flow control mode** to **Priority-based flow control**.
Average interpacket gap	String	• Disable deficit idle counter • 8 • 12	12	If you set the value of this parameter to **8** or to **12**, the IP core includes a deficit idle counter (DIC), which maintains an average interpacket gap (IPG) of 8 or 12, as you specify. If you set the value of this parameter to **Disable deficit idle counter**, the IP core is configured without the DIC, and does not maintain the required minimum average IPG. The Ethernet standard requires a minimum average IPG of 12. Turning off the DIC increases bandwidth.
MAC Options				
Enable 1588 PTP	Boolean	• True • False	False	If turned on, the IP core supports the IEEE Standard 1588-2008 Precision Clock Synchronization Protocol, by providing the hooks to implement the Precise Timing Protocol (PTP). If you select the custom streaming client interface, the IP core must be configured without 1588 support, and this parameter is not available.
Enable 96b Time of Day Format	Boolean	• True • False	True	Include the 96-bit interface to the TOD module. If you turn on this parameter, the TOD module that is generated with the IP core has a matching 96-bit timestamp interface. If **Enable 1588 PTP** is turned on, you must turn on at least one of **Enable 96b Time of Day Format** and **Enable 64b Time of Day Format**. You can turn on both **Enable 96b Time of Day Format** and **Enable 64b Time of Day Format** to generate a TOD interface for each format.

continued...

Parameter	Type	Range	Default Setting	Parameter Description
				This parameter is available only in variations with **Enable 1588 PTP** turned on.
Enable 64b Time of Day Format	Boolean	• True • False	False	Include the 64-bit interface to the TOD module. If you turn on this parameter, the TOD module that is generated with the IP core has a matching 64-bit timestamp interface. If **Enable 1588 PTP** is turned on, you must turn on at least one of **Enable 96b Time of Day Format** and **Enable 64b Time of Day Format**. You can turn on both **Enable 96b Time of Day Format** and **Enable 64b Time of Day Format** to generate a TOD interface for each format. This parameter is available only in variations with **Enable 1588 PTP** turned on.
Timestamp fingerprint width	Integer	1–16	1	Specifies the number of bits in the fingerprint that the IP core handles. This parameter is available only in variations with **Enable 1588 PTP** turned on.
Enable link fault generation	Boolean	• True • False	False	If turned on, the IP core includes the link fault signaling modules and relevant signals. If turned off, the IP core is configured without these modules and without these signals. Turning on link fault signaling provides your design a tool to improve reliability, but increases resource utilization.
Enable TX CRC insertion	Boolean	• True • False	True	If turned on, the IP core inserts a 32-bit Frame Check Sequence (FCS), which is a CRC-32 checksum, in outgoing Ethernet frames. If turned off, the IP core does not insert the CRC-32 sequence in outgoing Ethernet communication. Turning on TX CRC insertion improves reliability but increases resource utilization and latency through the IP core. If you turn on flow control, the IP core must be configured with TX CRC insertion, and this parameter is not available.
Enable preamble passthrough	Boolean	• True • False	False	If turned on, 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 SFD to the client instead of stripping them out of the Ethernet packet. In TX preamble pass-through mode, the client specifies the preamble to be sent in the Ethernet frame.
Enable alignment EOP on FCS word	Boolean	• True • False	True	If turned on, the IP core aligns the 32-bit Frame Check Sequence (FCS) error signal with the assertion of the EOP by delaying the RX data bus to match the latency of the FCS computation. If turned off, the IP core does not delay the RX data bus to match the latency of the FCS computation. If the parameter is turned off, the FCS

continued...

Parameter	Type	Range	Default Setting	Parameter Description
				error signal, in the case of an FCS error, is asserted in a later clock cycle than the relevant assertion of the EOP signal. Intel recommends that you turn on this option. Otherwise, the latency between the EOP indication and assertion of the FCS error signal is non-deterministic. You must turn on this parameter if your design relies on the `rx_inc_octetsOK` signal..
Enable TX statistics	Boolean	• True • False	True	If turned on, the IP core includes built–in TX statistics counters. If turned off, the IP core is configured without TX statistics counters. In any case, the IP core is configured with TX statistics counter increment output vectors.
Enable RX statistics	Boolean	• True • False	True	If turned on, the IP core includes built–in RX statistics counters. If turned off, the IP core is configured without RX statistics counters. In any case, the IP core is configured with RX statistics counter increment output vectors.
Configuration, Debug and Extension Options				
Enable Native PHY Debug Master Endpoint (NPDME)	Boolean	• True • False	False	If turned on, the IP core turns on the following features in the Arria 10 PHY IP core that is included in the LL 100GbE IP core: • **Enable Native PHY Debug Master Endpoint (NPDME)** • **Enable capability registers** If turned off, the IP core is configured without these features. This parameter is available only in variations that target an Arria 10 device. For information about these Arria 10 features, refer to the Arria 10 Transceiver PHY User Guide.

Table 11. LL 100GbE PHY Parameter Settings

Lists the PHY parameters that are configured automatically based on parameter values you select in the LL 100GbE parameter editor.

Parameter	LL 100GbE Standard Variations	LL 100GbE CAUI–4 Variations (Arria 10 Devices)
Lanes	10	4
Data rate per lane	10312.5 Mbps	25781.25 Mbps
Available PHY reference clock frequencies	322.265625 MHz 644.53125 MHz	322.265625 MHz 644.53125 MHz

Related Information

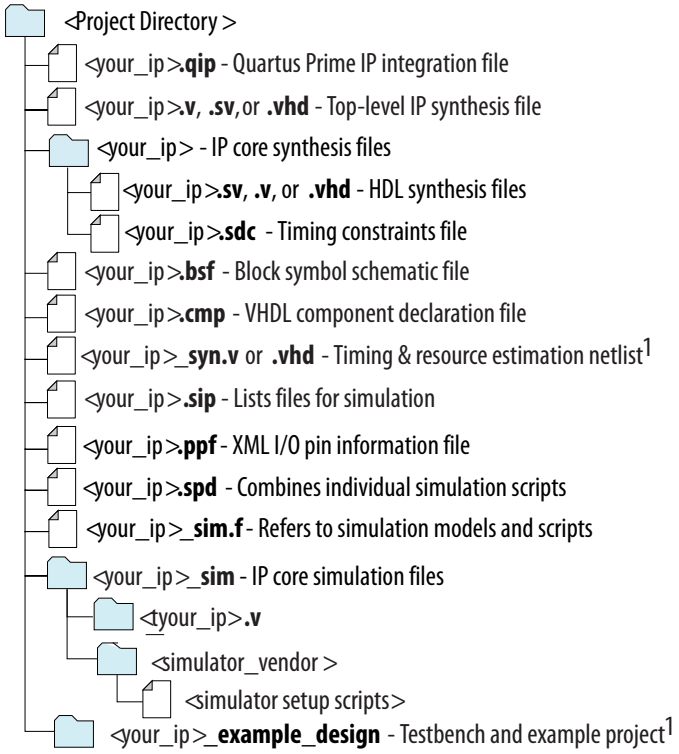
- Clocks on page 84
  The **PHY reference frequency** value is the required frequency of the transceiver reference clock.
- Arria 10 Transceiver PHY User Guide
  Information about Arria 10 Native PHY IP core features, including NPDME.

- Low Latency 100G Ethernet Design Example User Guide
  Information about the **Example Design** tab in the Arria 10 LL 100GbE parameter editor.

2.5. Files Generated for Stratix V Variations

The Intel Quartus Prime Standard Edition software generates the following output for your Stratix V LL 100GbE IP core.

Figure 5. LL 100GbE IP Core Generated Files for Stratix V Variations



Notes:
1. If generated for your IP variation

2.6. Files Generated for Arria 10 Variations

The Intel Quartus Prime software generates the following IP core output file structure when targeting Arria 10 devices.

For information about the file structure of the design example, refer to the *LL 100GbE Design Example User Guide*.

Figure 6. LL 100GbE IP Core Generated Files for Arria 10 Variations

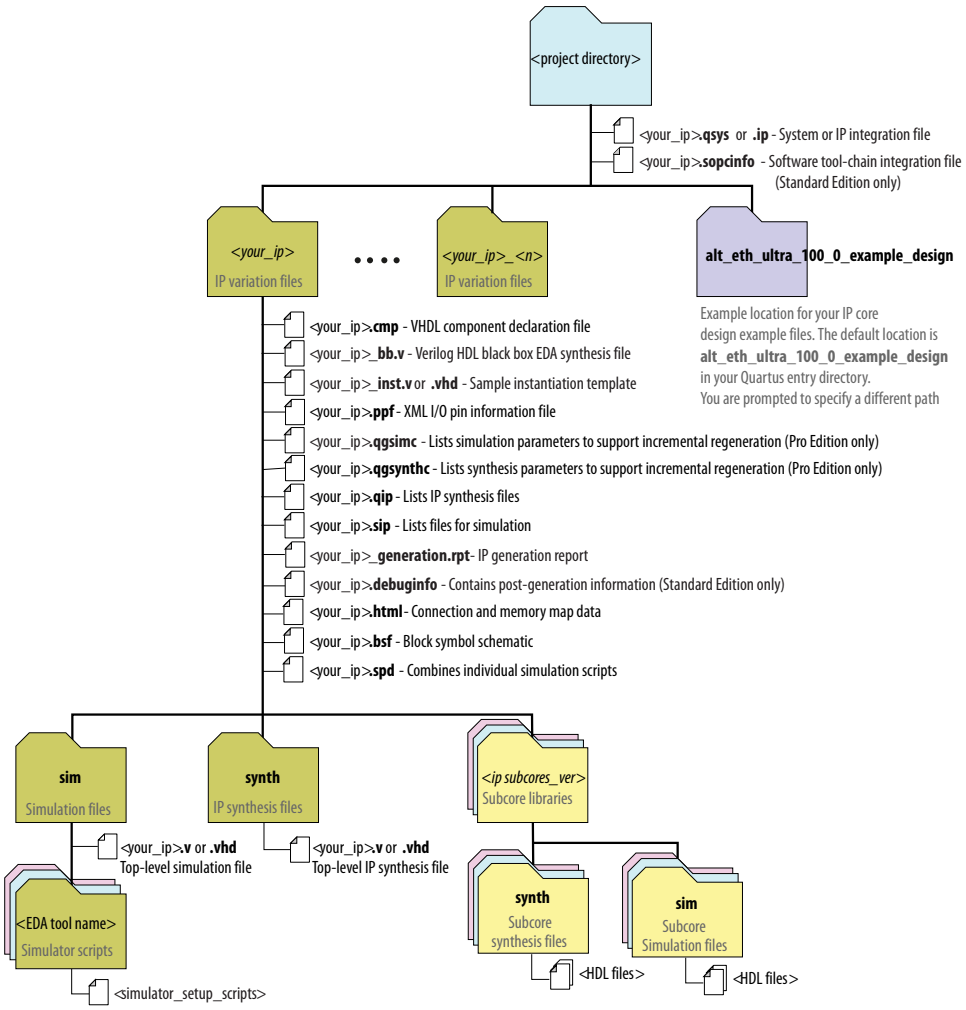


Table 12. IP Core Generated Files

File Name	Description
<*your_ip*>.qsys (Intel Quartus Prime Standard Edition only)	The Platform Designer (Standard) system or top-level IP variation file. <*your_ip*> is the name that you give your IP variation.
<*your_ip*>.ip (Intel Quartus Prime Pro Edition only)	
<*system*>.sopcinfo	Describes the connections and IP component parameterizations in your Platform Designer (Standard) system. You can parse its contents to get requirements when you develop software drivers for IP components. (Intel Quartus Prime Standard Edition only) Downstream tools such as the Nios® II tool chain use this file. The .sopcinfo file and the system.h file generated for the Nios II 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.
<*your_ip*>.cmp	The VHDL Component Declaration (**.cmp**) file is a text file that contains local generic and port definitions that you can use in VHDL design files.

continued...

File Name	Description
	This IP core does not support VHDL. However, the Intel Quartus Prime software generates this file.
<*your_ip*>.html	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.
<*your_ip*>_generation.rpt	IP or Platform Designer (Standard) generation log file. A summary of the messages during IP generation.
<*your_ip*>.debuginfo	Contains post-generation information. Used to pass System Console and Bus Analyzer Toolkit information about the Platform Designer (Standard) interconnect. The Bus Analysis Toolkit uses this file to identify debug components in the Platform Designer (Standard) interconnect. (Intel Quartus Prime Standard Edition only)
<*your_ip*>.qgsimc	Lists simulation parameters to support incremental regeneration. (Intel Quartus Prime Pro Edition only)
<*your_ip*>.qgsynthc	Lists synthesis parameters to support incremental regeneration. (Intel Quartus Prime Pro Edition only)
<*your_ip*>.qip	Contains all the required information about the IP component to integrate and compile the IP component in the Intel Quartus Prime.
<*your_ip*>.csv	Contains information about the upgrade status of the IP component.
<*your_ip*>.bsf	A Block Symbol File (.**bsf**) representation of the IP variation for use in Intel Quartus Prime Block Diagram Files (**.bdf**).
<*your_ip*>.spd	Required input file for `ip-make-simscript` to generate simulation scripts for supported simulators. The **.spd** file contains a list of files generated for simulation, along with information about memories that you can initialize.
<*your_ip*>.ppf	The Pin Planner File (**.ppf**) stores the port and node assignments for IP components created for use with the Pin Planner.
<*your_ip*>_bb.v	You can use the Verilog black-box (**_bb.v**) file as an empty module declaration for use as a black box.
<*your_ip*>.sip	Contains information required for NativeLink simulation of IP components. You must add the **.sip** file to your Intel Quartus Prime project.
<*your_ip*>_inst.v or _inst.vhd	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 software generates this file.
<*your_ip*>.regmap	If IP contains register information, **.regmap** file generates. The **.regmap** file describes the register map information of master and slave interfaces. This file complements the **.sopcinfo** file by providing more detailed register information about the system. This enables register display views and user customizable statistics in the System Console.
<*your_ip*>.svd	Allows hard processor system (HPS) System Debug tools to view the register maps of peripherals connected to HPS within a Platform Designer (Standard) system. During synthesis, the **.svd** files for slave interfaces visible to System Console masters are stored in the **.sof** file in the debug section. System Console reads this section, which Platform Designer (Standard) can query for register map information. For system slaves, Platform Designer (Standard) can access the registers by name.
<*your_ip*>.v or <*your_ip*>.vhd	HDL files that instantiate each submodule or child IP core for synthesis or simulation. This IP core does not support VHDL. However, the Intel Quartus Prime software generates this file.

continued...

File Name	Description
mentor/	Contains a ModelSim* script msim_setup.tcl to set up and run a simulation.
aldec/	Contains a Riviera-PRO* script rivierapro_setup.tcl to setup and run a simulation.
synopsys/vcs/ synopsys/vcsmx/	Contains a shell script vcs_setup.sh to set up and run a VCS® simulation. Contains a shell script vcsmx_setup.sh and synopsys_ sim.setup file to set up and run a VCS MX simulation.
cadence/	Contains a shell script ncsim_setup.sh and other setup files to set up and run an NCSIM simulation.
submodules/	Contains HDL files for the IP core submodule.
<child IP cores>/	For each generated child IP core directory, Platform Designer (Standard) generates synth/ andsim/ sub-directories.

Related Information

Low Latency 100G Ethernet Design Example User Guide
Information about the LL 100GbE design example file structure.

2.7. Integrating Your IP Core in Your Design

When you integrate your IP core instance in your design, you must pay attention to the following items:

Pin Assignments on page 25

External Transceiver Reconfiguration Controller Required in Stratix V Designs on page 26

Transceiver PLL Required in Arria 10 Designs on page 26

Handling Potential Jitter in Intel Arria 10 Devices on page 28

External Time-of-Day Module for Variations with 1588 PTP Feature on page 28

External TX MAC PLL on page 29

Placement Settings for the LL 100GbE Core on page 29

Related Information

Low Latency 100G Ethernet Design Example User Guide

2.7.1. Pin Assignments

When you integrate your LL 100GbE IP core instance in your design, you must make appropriate pin assignments. You can create a virtual pin to avoid making specific pin assignments for top-level signals until you are ready to map the design to hardware.

For the Arria 10 device family, you must configure a transceiver PLL that is external to the LL 100GbE IP core. The transceiver PLLs you configure are physically present in the device transceivers, but the LL 100GbE IP core does not configure and connect them. The required number of transceiver PLLs depends on the distribution of your Ethernet data pins in the different Arria 10 transceiver blocks.

Related Information

- Transceiver PLL Required in Arria 10 Designs on page 26

- Quartus Prime Help
  For information about the Quartus Prime software, including virtual pins and the IP Catalog.

2.7.2. External Transceiver Reconfiguration Controller Required in Stratix V Designs

LL 100GbE IP cores that target Stratix V devices require an external reconfiguration controller to compile and to function correctly in hardware. LL 100GbE IP cores that target Arria 10 devices include a reconfiguration controller block in the PHY component and do not require an external reconfiguration controller.

You can use the IP Catalog to generate an Transceiver Reconfiguration Controller.

When you configure the Transceiver Reconfiguration Controller, you must specify the number of reconfiguration interfaces. Stratix V LL 100GbE IP cores require 20 reconfiguration interfaces.

You can configure your reconfiguration controller with additional interfaces if your design connects with multiple transceiver IP cores. You can leave other options at the default settings or modify them for your preference.

You should connect the `reconfig_to_xcvr`, `reconfig_from_xcvr`, and `reconfig_busy` ports of the LL 100GbE IP core to the corresponding ports of the reconfiguration controller.

You must also connect the `mgmt_clk_clk` and `mgmt_rst_reset` ports of the Transceiver Reconfiguration Controller. The `mgmt_clk_clk` port must have a clock setting in the range of 100–125MHz; this setting can be shared with the LL 100GbE IP core `clk_status` port. The `mgmt_rst_reset` port must be deasserted before, or deasserted simultaneously with, the LL 100GbE IP core `reset_async` port.

Refer to the example project for RTL that connects the transceiver reconfiguration controller to the IP core.

Table 13. External Transceiver Reconfiguration Controller Ports for Connection to LL 100GbE IP Core

Signal Name	Direction	Description
`reconfig_to_xcvr[1399:0]`	Input	The LL 100GbE IP core reconfiguration controller to transceiver port in Stratix V devices.
`reconfig_from_xcvr[919:0]`	Output	The LL 100GbE IP core reconfiguration controller from transceiver port in Stratix V devices.
`reconfig_busy`	Input	Indicates the reconfiguration controller is still in the process of reconfiguring the transceiver.

Related Information

V-Series Transceiver PHY IP Core User Guide
For more information about the Transceiver Reconfiguration Controller.

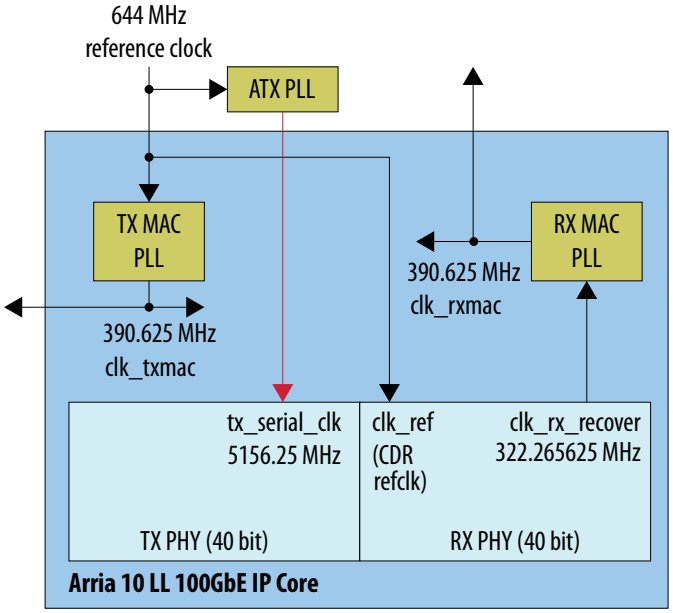
2.7.3. Transceiver PLL Required in Arria 10 Designs

LL 100GbE IP cores that target an Arria 10 device require an external TX transceiver PLL to compile and to function correctly in hardware.

Figure 7. PLL Configuration Example

In this example, the TX transceiver PLL is instantiated with an Arria 10 ATX PLL IP core. The TX transceiver PLL must always be instantiated outside the LL 100GbE IP core.

In this example, **Use external TX MAC PLL** is turned off. Therefore, the TX MAC PLL is in the IP core. If you turn on the **Use external TX MAC PLL** parameter you must also instantiate and connect a TX MAC PLL outside the LL 100GbE IP core.



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

- Select **Arria 10 Transceiver ATX PLL** or **Arria 10 Transceiver CMU PLL**.
- In the parameter editor, set the following parameter values:
  - **PLL output frequency** to **5156.25 MHz** for standard LL 100GbE IP core variations or to **12890.625 MHz** for CAUI-4 variations. 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 or 25.78125 Gbps data rate through the transceiver.
  - **PLL reference clock frequency** to the value you specified for the **PHY reference frequency** parameter.

When you generate a LL 100GbE IP core, the software also generates the HDL code for an ATX PLL, in the file `<variation_name>/arria10_atx_pll.v`. However, the HDL code for the LL 100GbE IP core does not instantiate the ATX PLL. If you choose to use the ATX PLL provided with the LL 100GbE IP core, you must instantiate and connect the instances of the ATX PLL with the LL 100GbE IP core in user logic.

Note: If your Arria 10 design includes multiple instances of the LL 100GbE IP core, do not use the ATX PLL HDL code provided with the IP core. Instead, generate new TX PLL IP cores to connect in your design.

The number of external PLLs you must generate or instantiate depends on the distribution of your Ethernet TX serial lines across physical transceiver channels and banks. You specify the clock network to which each PLL output connects by setting the

clock network in the PLL parameter editor. The example project demonstrates one possible choice, which is compatible with the ATX PLL provided with the LL 100GbE IP core.

You must connect the `tx_serial_clk` input pin for each LL 100GbE IP core PHY link to the output port of the same name in the corresponding external PLL. You must connect the `pll_locked` input pin of the LL 100GbE IP core to the logical AND of the `pll_locked` output signals of the external PLLs for all of the PHY links.

User logic must provide the AND function and connections. Refer to the example compilation project or design example for working user logic that demonstrates one correct method to instantiate and connect an external PLL.

Related Information

- External Transceiver PLL on page 59
- Arria 10 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.
- Low Latency 100G Ethernet Design Example User Guide
  Information about the LL 100GbE design example, which connects an external PLL to the IP core PHY links.

2.7.4. Handling Potential Jitter in Intel Arria 10 Devices

The RX path in the LL 100GbE core includes cascaded PLLs. Therefore, the IP core clocks might experience additional jitter in Intel Arria 10 devices.

Refer to the KDB Answer *How do I compensate for the jitter of PLL cascading or non-dedicated clock path for Arria 10 PLL reference clock?* for a workaround you should apply to the IP core, in your design.

Related Information

https://www.altera.com/support/support-resources/knowledge-base/tools/2017/fb470823.html
KDB Answer: How do I compensate for the jitter of PLL cascading or non-dedicated clock path for Arria 10 PLL reference clock?

2.7.5. External Time-of-Day Module for Variations with 1588 PTP Feature

LL 100GbE 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.

The example project you can generate for your IP core PTP variation includes a TOD module, implemented as two distinct, simple TOD modules, one connected to the TX MAC and one connected to the RX MAC.

Table 14. TOD Module Required Connections

Required connections for TOD module, listed 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 LL 100GbE IP core. However, its signal names could be different than the TOD module signal names in the table. The signals that that the IP core includes depend on the **Enable 96b Time of Day Format** and **Enable 64b Time of Day Format** parameters. For example, an RX TOD module might require only a 96-bit TOD out signal.

TOD Module Signal	LL 100GbE IP Core Signal
`rst_txmac` (input)	Drive this signal from the same source as the `reset_async` input signal to the LL 100GbE IP core.
`rst_rxmac` (input)	Drive this signal from the same source as the `reset_async` input signal to the LL 100GbE IP core.
`tod_txmclk_96b[95:0]` (output)	`tx_time_of_day_96b_data[95:0]` (input)
`tod_txmclk_64b[63:0]` (output)	`tx_time_of_day_64b_data[63:0]` (input)
`tod_rxmclk_96b[95:0]` (output)	`rx_time_of_day_96b_data[95:0]` (input)
`tod_rxmclk_64b[63:0]` (output)	`rx_time_of_day_64b_data[63:0]` (input)
`clk_txmac` (input)	`clk_txmac` (output)
`clk_rxmac` (input)	`clk_rxmac` (output)

Related Information

- PTP Timestamp and TOD Formats on page 76
- External Time-of-Day Module for 1588 PTP Variations on page 76
- 1588 Precision Time Protocol Interfaces on page 69

2.7.6. External TX MAC PLL

If you turn on **Use external TX MAC PLL** in the LL 100GbE parameter editor, you must connect the `clk_txmac_in` input port to a clock source, usually a PLL on the device.

The `clk_txmac_in` signal drives the `clk_txmac` clock in the IP core TX MAC and PHY. If you turn off this parameter, the `clk_txmac_in` input clock signal is not available.

The required TX MAC clock frequency is 390.625 MHz. User logic must drive `clk_txmac_in` from a PLL whose input is the PHY reference clock, `clk_ref`.

2.7.7. Placement Settings for the LL 100GbE 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 Standard Edition User Guide: Design Constraints
  Describes incremental compilation, design partitions, and Logic Lock (Standard) regions.

- Intel Quartus Prime Pro Edition User Guide: Design Constraints
  Describes incremental compilation, design partitions, and Logic Lock regions.

2.8. IP Core Testbenches

Intel provides a testbench, a hardware design example, and a compilation-only design example with most variations of the LL 100GbE IP core. The testbench is available for simulation of your IP core, and the hardware design example can be run on hardware. You can run the testbench to observe the IP core behavior on the various interfaces in simulation.

Intel offers testbenches for all Avalon-ST client interface variations that generate their own TX MAC clock(**Use external TX MAC PLL** is turned off).

Currently, the IP core can generate a testbench and example project for variations that use an external TX MAC PLL, but these testbenches and example projects do not function correctly. Non-functional testbenches and example projects provide an example of the connections you must create in your design to ensure the LL 100GbE IP core functions correctly. However, you cannot simulate them nor run them in hardware.

To generate the testbench, in the LL 100GbE parameter editor, you must first set the parameter values for the IP core variation you intend to generate in your end product. If you do not set the parameter values for your DUT to match the parameter values in your end product, the testbench you generate does not exercise the IP core variation you intend. If your IP core variation does not meet the criteria for a testbench, the generation process does not create a testbench (with the exception of the non-functional testbench generated if an IP core requires an external TX MAC clock signal).

You can simulate the testbench that you generate with your IP core variation. The testbench illustrates packet traffic, in addition to providing information regarding the transceiver PHY.

The testbench demonstrates a basic test of the IP core. It is not intended to be a substitute for a full verification environment.

Related Information

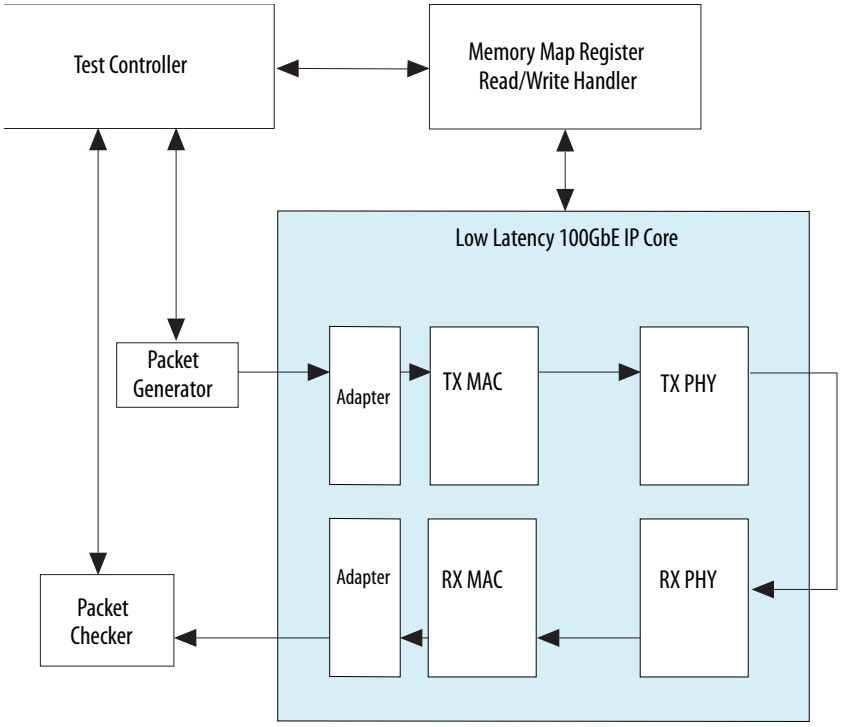
Low Latency 100G Ethernet Design Example User Guide
Information about generating and running the design example and testbench files for LL 100GbE IP cores that target an Arria 10 device. This testbench demonstrates a basic test of the IP core. It is not intended to be a substitute for a full verification environment.

2.8.1. Testbench Behavior

The testbenches send traffic through the IP core in transmit-to-receive loopback mode, exercising the transmit side and receive side of the IP core in the same data flow. These testbenches send traffic to allow the Ethernet lanes to lock, and then send packets to the transmit client data interface and check the data as it returns through the receive client data interface.

Figure 8. LL 100GbE IP Core Testbench

The top-level testbench file consists of a simple packet generator and checker and one IP core in a loopback configuration.



2.8.2. Simulating the LL 100GbE IP Core With the Testbenches

You can simulate the LL 100GbE IP core using the Intel-supported versions of the Mentor Graphics ModelSim SE, Cadence NCSim, and Synopsys VCS simulators for the current version of the Quartus Prime software. The ModelSim - Intel FPGA Edition simulator does not have the capacity to simulate this IP core.

The example testbenches simulate packet traffic at the digital level. The testbenches do not require special SystemVerilog class libraries.

The example testbenches contain the test files and run scripts for the ModelSim, Cadence, and Synopsys simulators. The run scripts use the file lists in the wrapper files. When you launch a simulation from the original directory, the relative filenames in the wrapper files allow the run script to locate the files correctly. When you generate the testbench for a LL 100GbE IP core that targets an Arria 10 device, the software generates a copy of the IP core variation with a specific relative path from the testbench scripts.

Table 15. LL 100GbE IP Core Testbench File Descriptions

Lists the key files that implement the example testbenches.

File Names	Description
Testbench and Simulation Files	
`basic_avl_tb_top.v`	Top-level testbench file. The testbench instantiates the DUT and runs Verilog HDL tasks to generate and accept packets.
Testbench Scripts	
`run_vsim.do`	The ModelSim script to run the testbench. *Note:* The ModelSim - Intel FPGA Edition simulator does not have the capacity to simulate this IP core. You must use another supported ModelSim simulator.
`run_vcs.sh`	The Synopsys VCS script to run the testbench.
`run_ncsim.sh`	The Cadence NCSim script to run the testbench.

Related Information

Low Latency 100G Ethernet Design Example User Guide

2.8.2.1. Generating the LL 100GbE Testbench

A single procedure generates the testbench and example project (for Stratix V variations) or the testbench, compilation-only design example, and hardware design example (for Arria 10 variations). The procedure varies depending on your target device. To generate these demonstration aids:

1. Follow the steps in Specifying the IP Core Parameters and Options on page 17 to parameterize your IP core.
2. If your IP core variation targets a Stratix V device, go to step 4.
3. If your IP core variation targets an Arria 10 device, in the LL 100GbE parameter editor:
   - On the **Example Design** tab, select **Simulation**, **Synthesis**, or both to specify whether you want to generate the simulation-only testbench, the compilation-only and hardware design examples, or all three options.
   - Click the **Generate Example Design** button to generate these options for the IP core variation you intend to generate.

   Tip: You are prompted to locate the new testbench and example project in the directory `<working directory>/alt_eth_ultra_100_0_example_design`. You can accept the default path or modify the path to the new testbench and example project.
4. Generate the IP core by clicking **Generate HDL** for Arria 10 variations or **Generate** for Stratix V variations.

   Note: If your IP core variation targets a Stratix V device, when prompted at the start of generation, you must turn on **Generate example design**. Turning on **Generate example design** is the only process that generates a functional testbench and a functional example project for Stratix V variations.

When the IP core is generated in *<working directory>*, the testbench and example projects are generated in different locations depending on the device family your IP core variation targets.

- For Stratix V variations, the testbench and example project are generated in *<working directory>*/*<IP core variation>*_example_design/alt_eth_ultra.
- For Arria 10 variations, the testench and the compilation-only and hardware design examples are generated in the directory you specify in Step 3. If you do not modify the location text at the prompt, they are generated in *<working directory>*/alt_eth_ultra_100_0_example_design.

The directory with the testbench and design example has four subdirectories for Arria 10 variations and three subdirectories for Stratix V variations:

- example_testbench
- compilation_test_design
- hardware_test_design
- ex_100g or ex_100g_caui4, for Arria 10 variations only

The ex_100g or ex_100g_caui4 directory contains a copy of the IP core variation. The testbench and design examples (compilation-only and hardware design example) for your Arria 10 IP core variation connect to the copy in this directory rather than to the copy you generate in *<working directory>*.

Related Information

Low Latency 100G Ethernet Design Example User Guide
Provides additional details and overview for Arria 10 variations.

2.8.2.2. Optimizing the IP Core Simulation With the Testbenches

In the testbench file, you can set the FASTSIM parameter and force values in the RO_SELS, BPOS, and WPOS parameters to enable simulation optimization. The testbench also specifies another IP core simulation optimization setting that you might need to modify for simulation in your own environment: AM_CNT_BITS.

To derive the values to which to force the RO_SELS, BPOS, and WPOS parameters, you must run your first simulation with some additional testbench code to display the simulation-derived values. For subsequent simulation runs with the same hardware design and simulator, you can force the values to these simulation-derived values to avoid the lengthy simulation required to achieve lane alignment. The process is particularly slow for this IP core because the IP core includes a soft processor that drives the lane alignment process. Forcing the parameters to the correct values for your design and simulator bypasses this process, increasing simulation efficiency.

The testbench file generated with the LL 100GbE IP core includes an initial block that displays the required force values. You can view the appropriate initial block for your IP core variation in the top-level testbench file you generate with your example design.

When you run simulation, this initial block prints values for the three parameters after lane alignment. Copy the values from standard output or from your log file and add the following lines to the testbench file, overwriting other forced values for these parameters if necessary:

```
defparam dut.top_inst.FASTSIM = 1;
defparam dut.<variation_name>_inst.FORCE_BPOS = <required BPOS value>;
defparam dut.<variation_name>_inst.FORC_WPOS  = <required WPOS value>;
defparam dut.<variation_name>_inst.FORCE_RO_SELS = <required RO_SELS value>;
```

Note: Whether you use the Intel-provided testbench or your own custom testbench, you must update the testbench file with these lines and the derived values. The testbench file generated with the LL 100GbE IP core does not include the correct values, which depend on your IP core variation and simulation tool.

The AM_CNT_BITS parameter specifies the interval between expected alignment markers. The default value of this parameter is 14; this value specifies that alignment markers are inserted every 2^{14} blocks, in compliance with the Ethernet specification. However, the testbench sets this parameter to the value of 6, to speed up simulation. In your own simulation environment, you must set this parameter to match the interval between incoming alignment markers to the IP core.

2.8.2.3. Simulating with the Modelsim Simulator

To run the simulation in the supported versions of the ModelSim simulation tool, follow these steps:

1. Change directory to the <example_design_install_dir>/example_testbench directory.
2. In the command line, type: `vsim -c -do run_vsim.do`

The example testbench will run and pass.

The ModelSim - Intel FPGA Edition simulator does not have the capacity to simulate this IP core.

2.8.2.4. Simulating with the NCSim Simulator

To run the simulation in the supported versions of the NCSim simulation tool, follow these steps:

1. Change directory to the <example_design_install_dir>/example_testbench directory.
2. In the command line, type: `sh run_ncsim.sh`

   The example testbench will run and pass.

2.8.2.5. Simulating with the VCS Simulator

To run the simulation in the supported versions of the VCS simulation tool, follow these steps:

1. Change directory to the <example_design_install_dir>/example_testbench directory.
2. In the command line, type: `sh run_vcs.sh`

The example testbench will run and pass.

2.8.2.6. Testbench Output Example

This section shows successful simulation using the LL 100GbE IP core testbench (*<variation_name>*_example_design/alt_eth_ultra/example_testbench/basic_avl_tb_top.v for Stratix V variations, or *<example_design_directory>*/example_testbench/basic_avl_tb_top.v for Arria 10 variations). The testbench connects the Ethernet TX lanes to the Ethernet RX lanes, so that the IP core is in an external loopback configuration. In simulation, the testbench resets the IP core and waits for lane alignment and deskew to complete successfully. The packet generator sends ten packets on the Ethernet TX lanes and the packet checker checks the packets when the IP core receives them on the Ethernet RX lanes.

The successful testbench run displays the following output:

```
# ****************************************
# ** Starting TX traffic...
# **
# **
# ** Sending Packet           1...
# ** Sending Packet           2...
# ** Sending Packet           3...
# ** Sending Packet           4...
# ** Sending Packet           5...
# ** Sending Packet           6...
# ** Sending Packet           7...
# ** Sending Packet           8...
# ** Sending Packet           9...
# ** Sending Packet          10...
# ** Received Packet          1...
# ** Received Packet          2...
# ** Received Packet          3...
# ** Received Packet          4...
# ** Received Packet          5...
# ** Received Packet          6...
# ** Received Packet          7...
# ** Received Packet          8...
# ** Received Packet          9...
# ** Received Packet         10...
# **
# ** Testbench complete.
# **
# ****************************************
```

2.9. 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 device with the Programmer and verify the design in hardware.

Note: The LL 100GbE IP core design example synthesis directories include Synopsys Constraint (.sdc) files that you can copy and modify for your own design.

Note: For additional .sdc file requirements for Intel Arria 10 designs, please refer to the KDB Answer at https://www.altera.com/support/support-resources/knowledge-base/tools/2017/fb470823.html.

Related Information

- Incremental Compilation for Hierarchical and Team-Based Design
- Programming Intel Devices

2.10. Initializing the IP Core

The testbench initializes the IP core. However, when you simulate or run your own design in hardware, you must implement the initialization steps yourself.

To initialize the LL 100GbE IP core in your own design, follow these steps:

1. Drive the clock ports.
2. Reset the IP core.

Related Information

- Clocks on page 84
  In step 1, drive the input clock ports as specified here.
- Resets on page 86
  In step 2, reset the IP core.

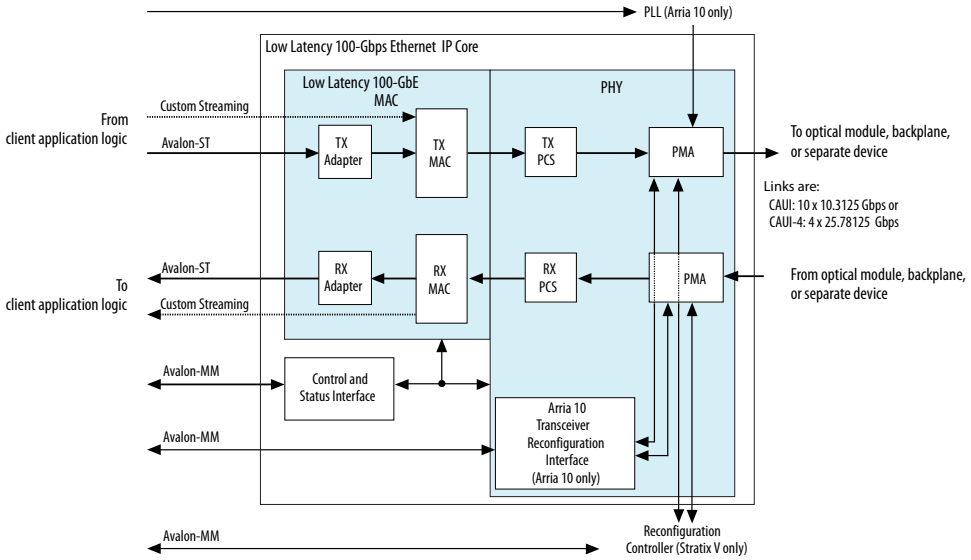
3. Functional Description

This chapter provides a detailed description of the LL 100GbE IP core. The chapter begins with a high-level overview of typical Ethernet systems and then provides detailed descriptions of the MAC, transmit (TX) and receive (RX) datapaths, signals, register descriptions, and an Ethernet glossary. This chapter includes the following sections:

3.1. High Level System Overview

Figure 9. LL 100GbE IP Core

Main blocks, internal connections, and external block requirements.



3.2. LL 100GbE IP Core Functional Description

The LL 100GbE IP core implements the 100GbE Ethernet MAC in accordance with the *IEEE 802.3ba 2010 High Speed Ethernet Standard.* This IP core handles the frame encapsulation and flow of data between a client logic and Ethernet network via a 100GbE Ethernet PCS and PMA (PHY).

In the transmit direction, the MAC accepts client frames, and inserts inter-packet gap (IPG), preamble, start of frame delimiter (SFD), padding, and CRC bits before passing them to the PHY. The PHY encodes the MAC frame as required for reliable transmission over the media to the remote end.

In the receive direction, the PHY passes frames to the MAC. The MAC accepts frames from the PHY, performs checks, updates statistics counters, strips out the CRC, preamble, and SFD, and passes the rest of the frame to the client. In RX 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 (bit 1 of the `CRC_CONFIG` register has the value of 1), the MAC passes on the CRC bytes to the client and asserts the EOP signal in the same clock cycle with the final CRC byte.

The LL 100GbE IP core includes the following interfaces:

- Datapath client-interface–The following options are available:
  - With adapters—Avalon-ST, 512 bits
  - Custom streaming, 256 bits
- Management interface—Avalon-MM host slave interface for MAC management. This interface has a data width of 32 bits and an address width of 16 bits.
- Datapath Ethernet interface–The following options are available:
  - CAUI: Ten 10.3125 Gbps serial links
  - CAUI–4: Four 25.78125 Gbps serial links
- In Arria 10 variations, an Arria 10 dynamic reconfiguration interface—an Avalon-MM interface to read and write the Arria 10 Native PHY IP core registers. This interface supports dynamic reconfiguration of the transceiver. LL 100GbE IP cores that target an Arria 10 device use the Arria 10 Native PHY IP core to configure the Ethernet link serial transceivers on the device. This interface has a data width of 32 bits. This interface has an address width of 12 bits for CAUI-4 variations, and an address width of 14 bits for standard variations.

3.2.1. LL 100GbE IP Core TX Datapath

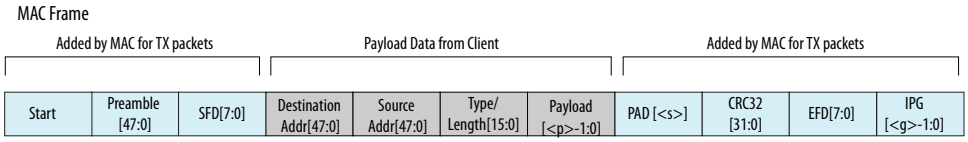
The TX MAC module receives the client payload data with the destination and source addresses and then adds, appends, or updates various header fields in accordance with the configuration specified. The MAC does not modify the destination address, the source address, or the payload received from the client. However, the TX MAC module adds a preamble (if the IP core is not configured to receive the preamble from user logic), pads the payload of frames greater than eight bytes to satisfy the minimum Ethernet frame payload of 46 bytes, and if you set **Enable TX CRC insertion** or turn on flow control, calculates the CRC over the entire MAC frame. (If padding is added, it is also included in the CRC calculation. If you turn off **Enable TX CRC insertion**, the client must provide the CRC bytes and must provide frames that have a minimum size of 64 bytes and therefore do not require padding). If you set **Average interpacket gap** to 8 or 12, the TX MAC module inserts IDLE bytes to maintain an average IPG. In addition, the TX MAC inserts an error in the Ethernet frame if the client requests to insert an error.

The LL 100GbE 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.

Figure 10. 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:

- <*p*> = payload size, which is arbitrarily large.
- <*s*> = padding bytes = 0–46 bytes.
- <*g*> = number of IPG bytes



The following sections describe the functions that the TX module performs:

Preamble, Start, and SFD Insertion on page 39
Address Insertion on page 39
Length/Type Field Processing on page 39
Frame Padding on page 40
Frame Check Sequence (CRC-32) Insertion on page 40
Inter-Packet Gap Generation and Insertion on page 40
TX RSFEC on page 40
Error Insertion Test and Debug Feature on page 41

3.2.1.1. Preamble, Start, and SFD Insertion

In the TX datapath the MAC appends an eight-byte preamble that begins with a Start byte (0xFB) to the client frame. If you turn on **Enable link fault generation**, this MAC module also incorporates the functions of the reconciliation sublayer.

The source of the preamble depends on whether you turn on the preamble pass-through feature by turning on **Enable preamble passthrough** in the LL 100GbE parameter editor.

If the preamble pass-through feature is turned on, the client provides the eight-byte preamble (including Start byte) on the data bus. The client is responsible for providing the correct Start byte.

3.2.1.2. Address Insertion

The client provides the destination MAC address and the source address of the local MAC.

3.2.1.3. Length/Type Field Processing

This two-byte header represents either the length of the payload or the type of MAC frame. When the value of this field is equal to or greater than 1536 (0x600) it indicates a type field. Otherwise, this field provides the length of the payload data that ranges from 0–1500 bytes. The TX MAC does not modify this field before forwarding it to the network.

3.2.1.4. Frame Padding

When the length of client frame is less than 64 bytes (meaning the payload is less than 46 bytes) and greater than eight 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.

Caution: The LL 100GbE IP core does not process incoming (egress) frames of less than nine bytes correctly. You must ensure such frames do not reach the TX client interface.

3.2.1.5. Frame Check Sequence (CRC-32) Insertion

The TX MAC computes and inserts a CRC32 checksum in the transmitted MAC frame. The frame check sequence (FCS) field contains a 32-bit CRC value. 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. The encoding is defined by the following generating polynomial:

```
FCS(X) = X32 +X26 +X23 +X22 +X16 +X12 +X11 +X10 +X8 +X7 +X5 +X4 +X2 +X1 +1
```

CRC bits are transmitted with MSB (X32) first.

If you configure your LL 100GbE IP core with no flow control, you can configure your IP core TX MAC to implement TX CRC insertion or not, by turning **Enable TX CRC insertion** on or off in the LL 100GbE parameter editor. By default, the CRC insertion feature is enabled. In variations with flow control, CRC insertion is enabled.

Related Information

Order of Transmission on page 46
Illustrations of the byte order and octet transmission order on the Avalon-ST client interface.

3.2.1.6. Inter-Packet Gap Generation and Insertion

If you set **Average interpacket gap** to **12** in the LL 100GbE parameter editor, the TX MAC maintains the minimum inter-packet gap (IPG) between transmitted frames required by the IEEE 802.3 Ethernet standard. The standard requires an average minimum IPG of 96 bit times (or 12 byte times). The deficit idle counter maintains the average IPG of 12 bytes.

If you set **Average interpacket gap** to **8**, the TX MAC maintains a minimum average IPG of 8 bytes. This option is provided as an intermediate option to allow you to enforce an IPG that does not conform to the Ethernet standard, but which increases the throughput of your IP core.

If you set **Average interpacket gap** to **Disable deficit idle counter**, the IP core transmits Ethernet packets as soon as the data is available, without maintaining the 12-byte IPG. In this case the IP core maintains only a minimum 1-byte IPG. If you select this parameter value, you optimize the IP core throughput.

3.2.1.7. TX RSFEC

If you turn on **Enable RS-FEC for CAUI4** in the LL 100GbE 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 91 of the IEEE Standard 802.3bj. The Reed-Solomon FEC algorithm includes the following modules:

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

The **Enable RS-FEC for CAUI4** parameter is available only in CAUI-4 variations of the IP core.

3.2.1.8. Error Insertion Test and Debug Feature

The client can specify the insertion of a TX error in a specific packet. If the client specifies the insertion of a TX error, the LL 100GbE IP core inserts an error in the frame it transmits on the Ethernet link. The error appears as a 66-bit error block that consists of eight /E/ characters (EBLOCK_T) in the Ethernet frame.

To direct the IP core to insert a TX error in a packet, the client should assert the TX error insertion signal as follows, depending on the TX client interface:

- On the Avalon-ST TX client interface, assert the `l8_tx_error` signal in the EOP cycle of the packet.
- On the custom streaming TX client interface, assert bit N of the `tx_error[3:0]` signal in the same cycle in which bit N of `din_eop[3:0]` is asserted.

The IP core overwrites Ethernet frame data with an EBLOCK_T error block when it transmits the Ethernet frame that corresponds to the packet EOP.

This feature supports test and debug of your IP core. In loopback mode, when the IP core receives a deliberately errored packet on the Ethernet link, the IP core recognizes it as a malformed packet.

Related Information

LL 100GbE IP Core Malformed Packet Handling on page 50

3.2.2. LL 100GbE IP Core TX Data Bus Interfaces

This section describes the TX data bus at the user interface and includes the following topics:

3.2.2.1. LL 100GbE IP Core User Interface Data Bus

The user interface width depends on your IP core variation. The LL 100GbE IP core provides two different client interfaces: the Avalon-ST interface and a custom interface.

- Custom streaming interface (no adapters): Data bus width is 256 bits.
- Avalon-ST interface: Data bus width is 512 bits.

Both interfaces operate at 390.625 MHz.

3.2.2.2. LL 100GbE IP Core TX Data Bus with Adapters (Avalon-ST Interface)

The LL 100GbE IP core TX datapath with adapters employs the Avalon-ST protocol. The Avalon-ST protocol is a synchronous point-to-point, unidirectional interface that connects the producer of a data stream (source) to a consumer of data (sink). The key properties of this interface include:

- Start of packet (SOP) and end of packet (EOP) signals delimit frame transfers.
- The SOP must always be in the MSB, simplifying the interpretation and processing of incoming data.
- A valid signal qualifies signals from source to sink.
- The sink applies backpressure to the source by using the ready signal. The source typically responds to the deassertion of the ready signal from the sink by driving the same data until the sink can accept it. The `readyLatency` defines the relationship between assertion and deassertion of the ready signal, and cycles which are considered to be `ready` for data transfer.The `readyLatency` on the TX client interface is zero cycles.

The client acts as a source and the TX MAC acts as a sink in the transmit direction.

Figure 11. TX Client to MAC Interface with Adapters (Avalon-ST)

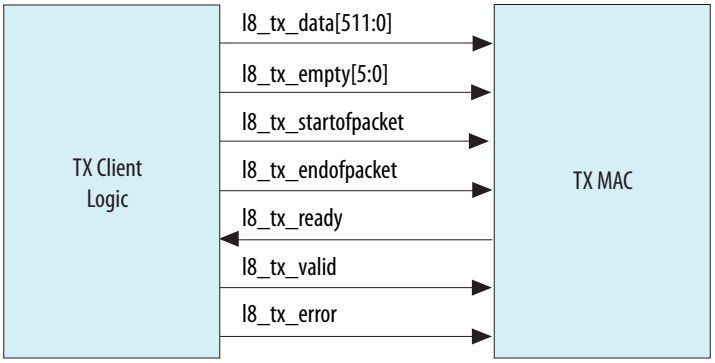


Table 16. Signals of the Avalon-ST TX Client Interface

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

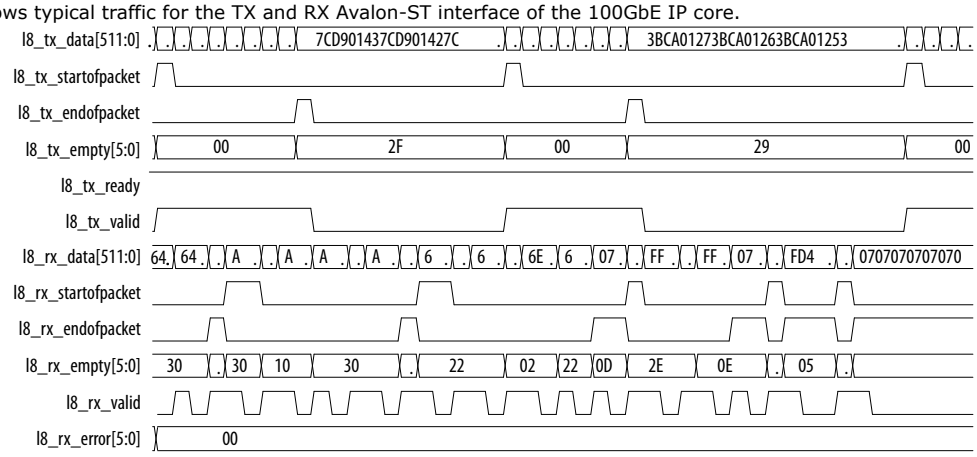
Signal Name	Direction	Description
`l8_tx_data[511:0]`	Input	TX data. If the preamble pass-through feature is enabled, data begins with the preamble.

continued...

Signal Name	Direction	Description
		The LL 100GbE 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. 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 `l8_tx_startofpacket` when you are assured the packet data to send on `l8_tx_data[511:0]` is available or will be available on time.
`l8_tx_empty[5:0]`	Input	Indicates the number of empty bytes on `l8_tx_data[511:0]` when `l8_tx_endofpacket` is asserted.
`l8_tx_startofpacket`	Input	When asserted, indicates the start of a packet. The packet starts on the MSB.
`l8_tx_endofpacket`	Input	When asserted, indicates the end of packet.
`l8_tx_ready`	Output	When asserted, the MAC is ready to receive data. The `l8_tx_ready` signal acts as an acknowledge. The source drives `l8_tx_valid` and `l8_tx_data[511:0]`, then waits for the sink to assert `l8_tx_ready`. The `readyLatency` is zero cycles, so that the IP core accepts valid data in the same cycle in which it asserts `l8_tx_ready`. The `l8_tx_ready` signal indicates the MAC is ready to receive data in normal operational model. However, the `l8_tx_ready` signal might not be an adequate indication following reset. To avoid sending packets before the Ethernet link is able to transmit them reliably, you should ensure that the application does not send packets on the TX client interface until after the `tx_lanes_stable` signal is asserted.
`l8_tx_valid`	Input	When asserted `l8_tx_data` is valid. This signal must be continuously asserted between the assertions of `l8_tx_startofpacket` and `l8_tx_endofpacket` for the same packet.
`l8_tx_error`	Input	When asserted in an EOP cycle (while `l8_tx_endofpacket` is asserted), directs the IP core to insert an error in the packet before sending it on the Ethernet link. This signal is a test and debug feature. In loopback mode, the IP core recognizes the packet upon return as a malformed packet.

Figure 12. Traffic on the TX and RX Avalon-ST Client Interface for Low Latency 100GbE IP Core

Shows typical traffic for the TX and RX Avalon-ST interface of the 100GbE IP core.



Related Information

Avalon Interface Specifications
For more information about the Avalon-ST interface.

3.2.2.3. LL 100GbE IP Core TX Data Bus Without Adapters (Custom Streaming Interface)

Figure 13. TX Client to MAC Interface Without Adapters

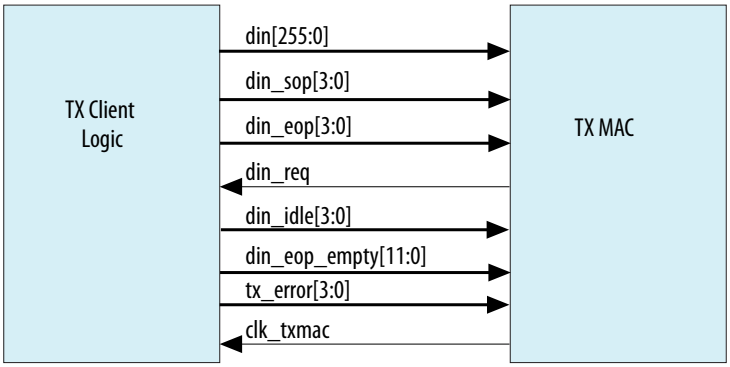


Table 17. Signals of the Custom Streaming TX Client Interface

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

Signal Name	Direction	Description
`din[255:0]`	Input	Data bytes to send in big-endian mode. Most significant 64-bit word is in the higher-order bits: bits[255:192] The LL 100GbE 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.
`din_sop[3:0]`	Input	Start of packet (SOP) location in the TX data bus. Only the most significant byte of each 64-bit word may be a start of packet. Bits 255, 191, 127, and 63 are allowed locations. Bit 0 of `din_sop` corresponds to the data word in `din[63:0]`.
`din_eop[3:0]`	Input	End of packet location in the TX data bus. Indicates the 64-bit word that holds the end-of-packet byte. Any byte may be the last byte in a packet. Bit 0 of `din_eop` corresponds to the data word in `din[63:0]`.
`din_eop_empty[11:0]`	Input	Indicates the number of empty (invalid) bytes in the end-of-packet 8-byte word indicated by `din_eop`. If `din_eop[z]` has the value of 0, then the value of `din_eop_empty[(z+2):z]` does not matter. However, if `din_eop[z]` has the value of 1, then you must set the value of `din_eop_empty[(z+2):z]` to the number of empty (invalid) bytes in the end-of-packet word `z`. For example, if you have a LL 100GbE IP core and want to indicate that in the current `clk_txmac` clock cycle, byte 6 in word 2 of `din` is an end-of-packet byte, and no other words hold an end-of-packet byte in the current clock cycle, you must set the value of `din_eop` to 4'b0100 and the value of `din_eop_empty` to 12'b000_110_000_000.
`din_idle[3:0]`	Input	Indicates the words in `din` that hold Idle bytes or control information rather than Ethernet data. One-hot encoded.

continued...

Signal Name	Direction	Description
din_req	Output	Indicates that input data was accepted by the IP core.
tx_error[3:0]	Input	When asserted in an EOP cycle (while din_eop is non-zero), directs the IP core to insert an error in the corresponding packet before sending it on the Ethernet link. This signal is a test and debug feature. In loopback mode, the IP core recognizes the packet upon return as a malformed packet.
clk_txmac	Output	TX MAC clock. The clock frequency should be 390.625 MHz. The clk_txmac clock and the clk_rxmac clock (which clocks the RX datapath) are assumed to have the same frequency.

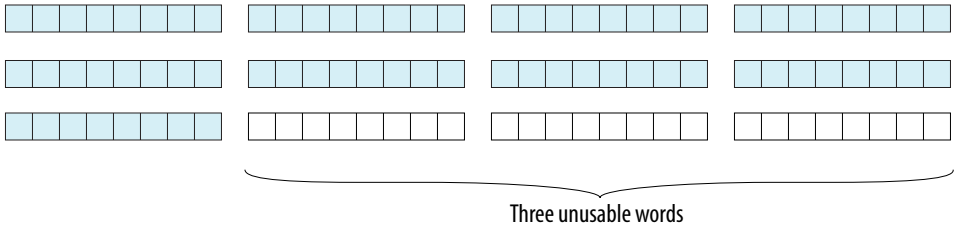
The IP core reads the bytes in big endian order. A packet may start in the most significant byte of any word. A packet may end on any byte.

3.2.2.4. Bus Quantization Effects With Adapters

The TX custom streaming interface allows a packet to start at any of four positions to maximize utilization of the link bandwidth. The TX Avalon-ST interface only allows start of packet (SOP) to be placed at the most significant position. If the SOP were restricted to the most significant position in the client logic data bus in the custom streaming interface, bus bandwidth would be reduced.

Figure 14. Reduced Bandwidth With Left-Aligned SOP Requirement

Illustrates the reduction of bandwidth that would be caused by left-aligning the SOP for the LL 100GbE IP core.



The example shows a nine-word packet, which is the worst case for bandwidth utilization. Assuming another packet is waiting for transmission, the effective ingress bandwidth is reduced by 25%. Running the MAC portion of the logic slightly faster than is required can mitigate this loss of bandwidth. Additional increases in the MAC frequency can provide further mitigation, although increases in frequency make timing closure more difficult. The wider data bus for the Avalon-ST interface also helps to compensate for the Avalon-ST left-aligned SOP requirement.

3.2.2.5. 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 Ethernet MAC and PHY transmit 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.

3.2.2.5.1. 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. Transmit frames the IP core receives on the client interface are big-endian. Frames the MAC sends to the PHY on the XGMII/CGMII between the MAC and the PHY are little-endian; the MAC TX transmits frames on this interface beginning with the least significant byte.

Figure 15. Byte Order on the Client Interface Lanes Without Preamble Pass-Through

Describes the byte order on the Avalon-ST interface when the preamble pass-through feature is turned off. 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/ 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 7 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 16. Octet Transmission on the Avalon-ST Signals Without Preamble Pass-Through

Illustrates how the octets of the client frame are transferred over the TX datapath when preamble pass-through is turned off.

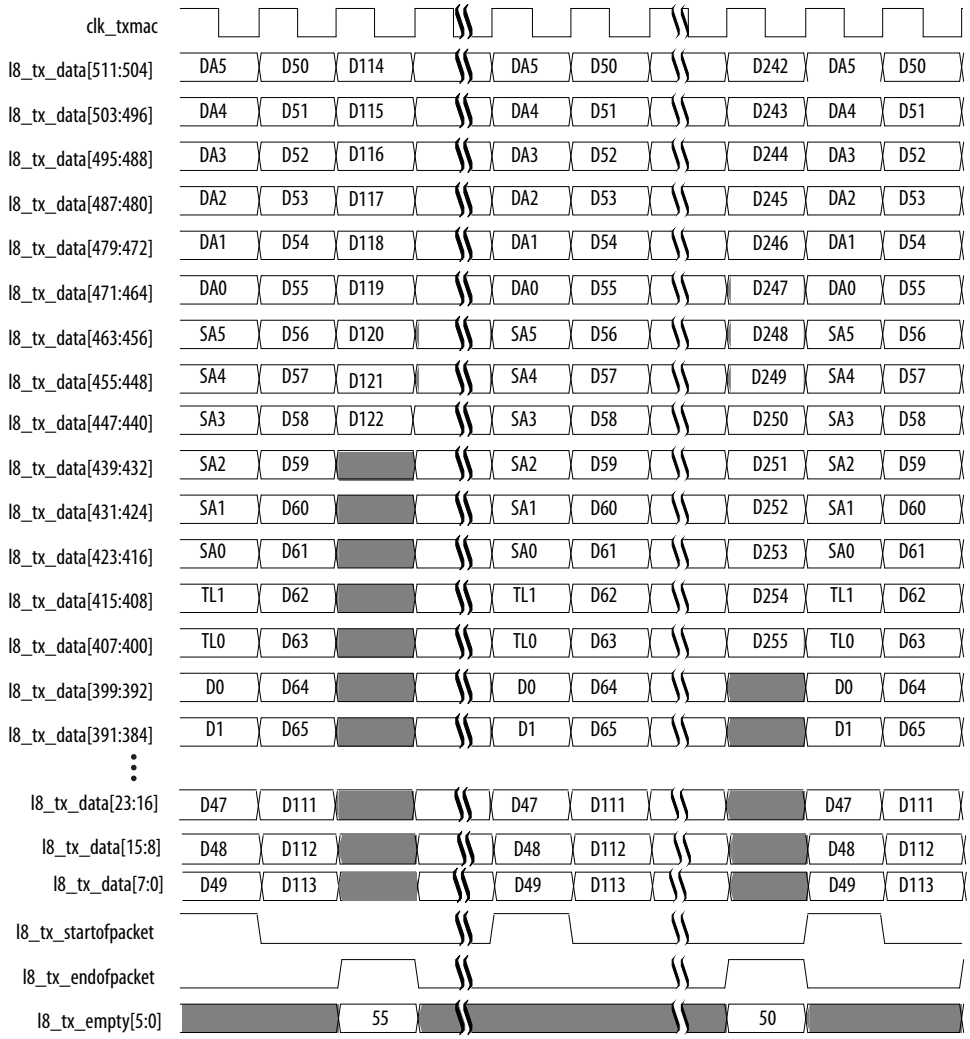


Figure 17. Byte Order on the Avalon-ST Interface Lanes With Preamble Pass-Through

Describes the byte order on the Avalon-ST interface when the preamble pass-through feature is turned on.

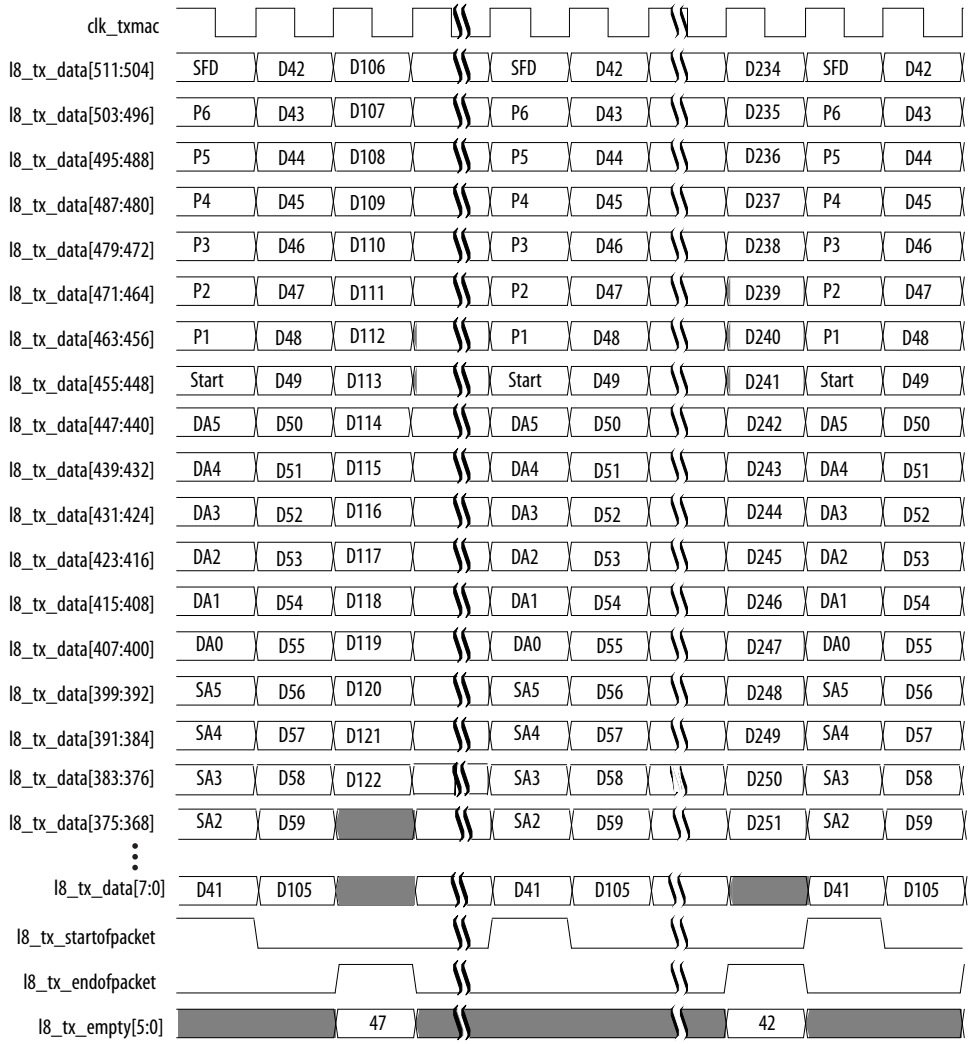
Destination Address[40] is the broadcast/multicast bit (a type bit), and Destination Address[41] is a locally administered address bit.

	SFD	Preamble						Start	Destination Address (DA)						Source Address (SA)						Type/ Length		Data (D)		
Octet	7	6	5	4	3	2	1	0	5	4	3	2	1	0	5	4	3	2	1	0	1	0	00	...	NN
Bit	[63:56]	[55:48]	[47:40]	[39:32]	[31:24]	[23:16]	[15:8]	[7:0]	[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]

Figure 18. Octet Transmission on the Avalon-ST Signals With Preamble Pass-Through

Illustrates how the octets of the client frame are transferred over the TX datapath when preamble pass-through is turned on. The eight preamble bytes precede the destination address bytes. The preamble bytes are reversed: the application must drive the Start byte on `l8_tx_data[455:448]` and the SFD byte on `l8_tx_data[511:504]`.

The destination address and source address bytes follow the preamble pass-through in the same order as in the case without preamble pass-through.



3.2.3. LL 100GbE IP Core RX Datapath

The LL 100GbE RX MAC receives Ethernet frames from the PHY and forwards the payload with relevant header bytes to the client after performing some MAC functions on header bytes.

Figure 19. Flow of Frame Through the MAC RX Without Preamble Pass-Through

Illustrates the typical flow of frame through the MAC RX when the preamble pass-through feature is turned off. In this figure, <*p*> is payload size, and <*s*> is the number of pad bytes (0–46 bytes).

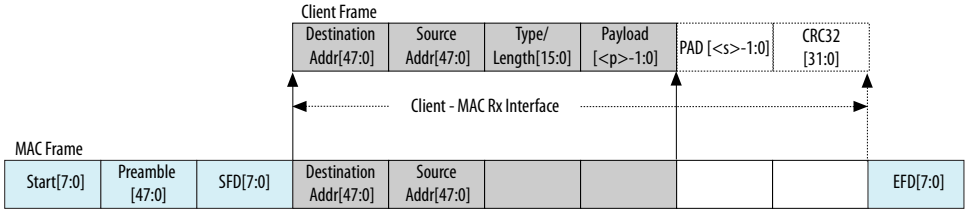
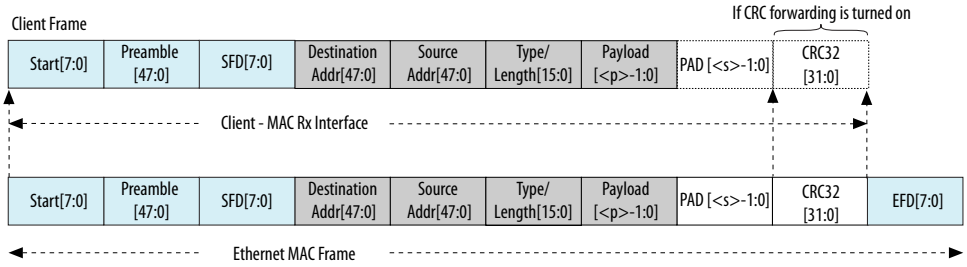


Figure 20. Flow of Frame Through the MAC RX With Preamble Pass-Through Turned On

Illustrates the typical flow of frame through the MAC RX when the preamble pass-through feature is turned on. In this figure, <*p*> is payload size, and <*s*> is the number of pad bytes (0–46 bytes).



The following sections describe the functions performed by the RX MAC:

3.2.3.1. LL 100GbE IP Core RX Filtering

The LL 100GbE IP core processes all incoming valid frames. However, the IP core does not forward pause frames to the Avalon-ST RX client interface by default.

If you set the `cfg_fwd_ctrl` bit of the RX_PAUSE_FWD register to the value of 1, the IP core forwards pause frames to the Avalon-ST RX client interface.

3.2.3.2. LL 100GbE IP Core Preamble Processing

The preamble sequence is Start, six preamble bytes, and SFD. If this sequence is incorrect the frame is ignored. The Start byte must be on receive lane 0 (most significant byte). The IP core uses the SFD byte (0xD5) to identify the last byte of the preamble. The MAC RX looks for the Start, six preamble bytes and SFD.

By default, the MAC RX removes all Start, SFD, preamble, and IPG bytes from accepted frames. However, if you turn on **Enable preamble passthrough** in the LL 100GbE parameter editor, the MAC RX does not remove the eight-byte preamble sequence.

3.2.3.3. LL 100GbE IP Core FCS (CRC-32) Removal

Independent user configuration register bits control FCS CRC removal at runtime. CRC removal supports both narrow and wide bus options. Bit 0 of the `MAC_CRC_CONFIG` register enables and disables CRC removal; by default, CRC removal is enabled.

In the user interface, the EOP signal (`l8_rx_endofpacket` or `dout_eop`) indicates the end of CRC bytes if CRC is not removed. When CRC is removed, the EOP signal indicates the final byte of payload.

The IP core signals an FCS error by asserting the FCS error output signal `l8_rx_fcs_error` and the `l8_rx_fcs_error` (or `rx_fcs_error` and the `rx_fcs_valid`) output signals in the same clock cycle. The `l8_rx_error[1]` or `rx_error[1]` also signals an FCS error.

If you turn on **Enable alignment EOP on FCS word** in the parameter editor, the IP core asserts `l8_rx_fcs_error` (or `rx_fcs_error`) and the EOP signal on the same clock cycle if the current frame has an FCS error. However, if you turn off **Enable alignment EOP on FCS word**, the IP core asserts `l8_rx_fcs_error` in a later clock cycle than the EOP signal.

3.2.3.4. LL 100GbE IP Core CRC Checking

The 32-bit CRC field is received in the order: X32, X30, . . . X1, and X0 , where X32 is the most significant bit of the FCS field and occupies the least significant bit position in the first FCS byte.

If a CRC32 error is detected, the RX MAC marks the frame invalid by asserting the `l8_rx_fcs_error` and `l8_rx_fcs_valid` (or `rx_fcs_error` and `rx_fcs_valid`) signals, as well as the `l8_rx_error[1]` (or `rx_error[1]`) signal.

3.2.3.5. LL 100GbE IP Core Malformed Packet Handling

While receiving an incoming packet from the Ethernet link, the LL 100GbE 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. The LL 100GbE IP core detects and handles the following forms of malformed packets:

- If the IP core detects an Error character, it generates an EOP, asserts a malformed packet error (`rx_error[0]` or `l8_rx_error[0]`), and asserts an FCS error (`rx_fcs_error` (`l8_rx_fcs_error`) and `rx_error[1]` (`l8_rx_error[1]`). If the IP core subsequently detects a terminate character, it does not generate another EOP indication.
- If the IP core detects any other control character when it is waiting for an EOP indication (terminate character), the IP core generates an EOP indication (for example, an IDLE or Start character), asserts a malformed packet error (`rx_error[0]` or `l8_rx_error[0]`), and asserts an FCS error (`rx_fcs_error` (`l8_rx_fcs_error`) and `rx_error[1]` (`l8_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 LL 100GbE TX error insertion feature, the IP core identifies it as a malformed packet.

Related Information

Error Insertion Test and Debug Feature on page 41

3.2.3.6. RX CRC Forwarding

The CRC-32 field is forwarded to the client interface after the final byte of data, if the CRC removal option is not enabled.

3.2.3.7. Inter-Packet Gap

The MAC RX removes all IPG octets received, and does not forward them to the Avalon-ST client interface. If you configure your IP core with a custom streaming client interface, the MAC RX does not remove IPG octets. The MAC RX forwards all received IPG octets to the custom client interface.

3.2.3.8. RX RSFEC

If you turn on **Enable RS-FEC for CAUI4** in the LL 100GbE 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 91 of the IEEE Standard 802.3bj. The Reed-Solomon FEC algorithm includes the following modules:

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

The **Enable RS-FEC for CAUI4** parameter is available only in CAUI-4 variations of the IP core.

3.2.3.9. Pause Ignore

If you turn on flow control by setting **Flow control mode** to the value of **Standard flow control** or **Priority-based flow control** in the LL 100GbE parameter editor, the IP core processes incoming pause frames by default. However, when the pause frame receive enable bit (for standard flow control) or bits (for priority-based flow control) is or are not set, the IP core does not process incoming pause frames. In this case, the MAC TX traffic is not affected by the valid pause frames.

If you turn off flow control by setting **Flow control mode** to the value of **No flow control** in the LL 100GbE parameter editor, the IP core does not process incoming pause frames.

Related Information

- Congestion and Flow Control Using Pause Frames on page 59
- Pause Control and Generation Interface on page 62
- Pause Registers on page 102

3.2.3.10. Control Frame Identification

The mechanisms to process flow control frames from the Ethernet link are specific to the flow control mode of the LL 100GbE IP core. Therefore, control frames that are inappropriate to the IP core mode, simply pass through the RX MAC to the RX client interface.

If you turn off flow control by setting **Flow control mode** to the value of **No flow control** in the LL 100GbE parameter editor, the IP core might nevertheless receive a flow control request on the Ethernet link. Similarly, if you set **Flow control mode** to **Standard flow control**, the IP core might nevertheless receive a priority-based flow control request on the Ethernet link, and if you set **Flow control mode** to **Priority-based flow control**, the IP core might nevertheless receive a standard flow control request.

The IP core provides a three-bit RX status flag that identifies the control frames that appear on the RX client interface. The flag is one-hot encoded to identify whether the control frame is a standard flow control frame, a priority-based flow control frame, or a different type of control frame. If the flag has the value of 3'b000, the current packet on the RX client interface is not a control frame. You can use this flag to identify control frames on the RX client interface that the client prefers to ignore.

Related Information

- LL 100GbE IP Core RX Data Bus with Adapters (Avalon-ST Interface) on page 53
- LL 100GbE IP Core RX Data Bus Without Adapters (Custom Streaming Interface) on page 56

3.2.4. LL 100GbE IP Core RX Data Bus Interfaces

This section describes the RX data bus at the user interface and includes the following topics:

LL 100GbE IP Core RX Data Bus Without Adapters (Custom Streaming Interface) on page 56

3.2.4.1. LL 100GbE IP Core User Interface Data Bus

The user interface width depends on your IP core variation. The LL 100GbE IP core provides two different client interfaces: the Avalon-ST interface and a custom interface.

- Custom streaming interface (no adapters): Data bus width is 256 bits.
- Avalon-ST interface: Data bus width is 512 bits.

Both interfaces operate at 390.625 MHz.

3.2.4.2. LL 100GbE IP Core RX Data Bus with Adapters (Avalon-ST Interface)

The LL 100GbE IP core RX datapath employs the Avalon-ST protocol. The Avalon-ST protocol is a synchronous point-to-point, unidirectional interface that connects the producer of a data stream (source) to a consumer of data (sink). The key properties of this interface include:

- Start of packet (SOP) and end of packet (EOP) signals delimit frame transfers.
- The SOP must always be in the MSB, simplifying the interpretation and processing of data you receive on this interface.
- A valid signal qualifies signals from source to sink.

The RX MAC acts as a source and the client acts as a sink in the receive direction.

Figure 21. RX MAC to Client Interface

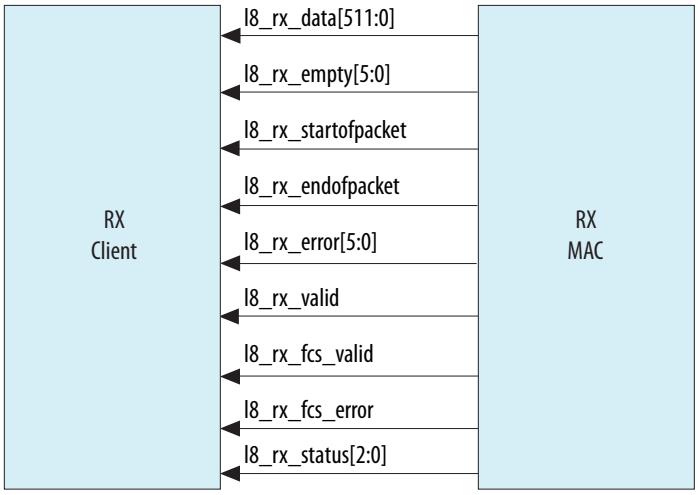


Table 18. Signals of the Avalon-ST RX Client Interface

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

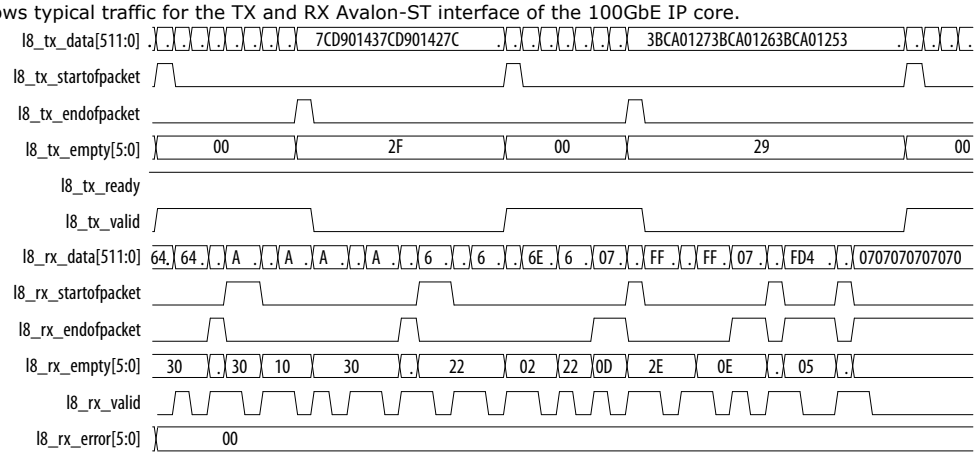
Name	Direction	Description
`l8_rx_data[511:0]`	Output	RX data.
`l8_rx_empty[5:0]`	Output	Indicates the number of empty bytes on `l8_rx_data[511:0]` when `l8_rx_endofpacket` is asserted, starting from the least significant byte (LSB).
`l8_rx_startofpacket`	Output	When asserted, indicates the start of a packet. The packet starts on the MSB.
`l8_rx_endofpacket`	Output	When asserted, indicates the end of packet. In the case of an undersized packet, or in the case of a packet that is exactly 64 bytes long, `l8_rx_startofpacket` and `l8_rx_endofpacket` are asserted in the same clock cycle.
`l8_rx_error[5:0]`	Output	Reports certain types of errors in the Ethernet frame whose contents are currently being transmitted on the client interface. This signal is valid in EOP cycles only. To ensure you can identify the corresponding packet, you must turn on **Enable alignment EOP on FCS word** in the LL 100GbE parameter editor. The individual bits report different types of errors: • Bit [0]: Malformed packet error. If this bit has the value of 1, the packet is malformed. The IP core identifies a malformed packet when it receives a control character that is not a terminate character, while receiving the packet. • Bit [1]: CRC error. If this bit has the value of 1, the IP core detected a CRC error in the frame. If you turn on **Enable alignment EOP on FCS word**, this bit and the `l8_rx_fcs_error` signal behave identically. • Bit [2]: undersized payload. If this bit has the value of 1, the frame size is between nine and 63 bytes, inclusive. The IP core does not recognize an incoming frame of size eight bytes or less as a frame, and those cases are not reported here. The `l8_rx_error[1]` bit also signals an FCS error. • Bit [3]: oversized payload. If this bit has the value of 1, the frame size is greater than the maximum frame size programmed in the `MAX_RX_SIZE_CONFIG` register at offset 0x506. • Bit [4]: payload length error. If this bit has the value of 1, the payload received in the frame did not match the length field value, and the value in the length field is less than 1536 bytes. This bit only reports errors if you set bit [0] of the `CFG_PLEN_CHECK` register at offset 0x50A to the value of 1. • Bit [5]: Reserved.
`l8_rx_valid`	Output	When asserted, indicates that RX data is valid. Only valid between the `l8_rx_startofpacket` and `l8_rx_endofpacket` signals.
`l8_rx_fcs_valid`	Output	When asserted, indicates that FCS is valid.
`l8_rx_fcs_error`	Output	When asserted, indicates an FCS error condition. The IP core asserts the `l8_rx_fcs_error` signal only when it asserts the `l8_rx_fcs_valid` signal. Runt frames always force an FCS error condition. However, 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 as a runt.
`l8_rx_status[2:0]`	Output	Indicates the IP core received a control frame on the Ethernet link. This signal identifies the type of control frame the IP core is passing through to the client interface.

continued...

Name	Direction	Description
		This signal is valid in EOP cycles only. To ensure you can identify the corresponding packet, you must turn on **Enable alignment EOP on FCS word** in the LL 100GbE parameter editor. The individual bits report different types of received control frames: • Bit [0]: Indicates the IP core received a standard flow control frame. If the IP core is in standard flow control mode and the `cfg_fwd_ctrl` bit of the RX_PAUSE_FWD register has the value of 0, this bit maintains the value of 0. • Bit [1]: Indicates the IP core received a priority flow control frame. If the IP core is in priority flow control mode and the `cfg_fwd_ctrl` bit of the RX_PAUSE_FWD register has the value of 0, this bit maintains the value of 0. • Bit [2]: Indicates the IP core received a control frame that is not a flow control frame.

Figure 22. Traffic on the TX and RX Avalon-ST Client Interface for Low Latency 100GbE IP Core

Shows typical traffic for the TX and RX Avalon-ST interface of the 100GbE IP core.



Related Information

- LL 100GbE IP Core MAC Configuration Registers on page 100
  Describes the MAX_RX_SIZE_CONFIG and CFG_PLEN_CHECK registers.
- Control Frame Identification on page 52
- Avalon Interface Specifications
  For more information about the Avalon-ST interface.

3.2.4.3. LL 100GbE IP Core RX Data Bus Without Adapters (Custom Streaming Interface)

Figure 23. RX MAC to Client Interface Without Adapters

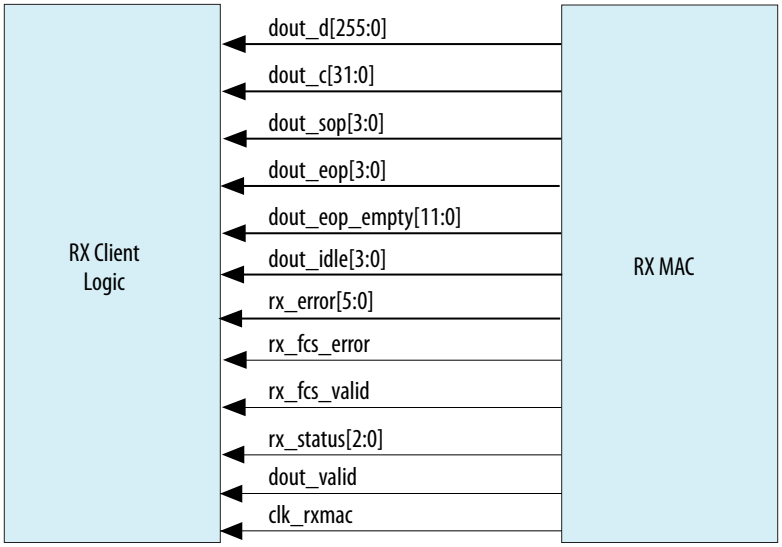


Table 19. Signals of the Custom Streaming RX Client Interface

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

Signal Name	Direction	Description
`dout_d[255:0]`	Output	Received data and Idle bytes. In RX preamble pass-through mode, this bus also carries the preamble.
`dout_c[31:0]`	Output	Indicates control bytes on the data bus. Each bit of dout_c indicates whether the corresponding byte of `dout_d` is a control byte. A bit is asserted high if the corresponding byte on `dout_d` is an Idle byte or the Start byte, and has the value of zero if the corresponding byte is a data byte or, in preamble pass-through mode, a preamble or SFD byte.
`dout_sop[3:0]`	Output	Indicates the first data word of a frame, in the current `clk_rxmac` cycle. In RX preamble pass-through mode, the first data word is the word that contains the preamble. When the RX preamble pass-through feature is turned off, the first data word is the first word of Ethernet data that follows the preamble. This signal is one-hot encoded.
`dout_eop[3:0]`	Output	Indicates the final word of a frame in the current `clk_rxmac` cycle. If CRC removal is disabled, this signal indicates the word with the final CRC byte. If CRC removal is enabled, this signal indicates the final word with data. This signal is one-hot encoded. In the case of an undersized packet, `dout_sop` and `dout_eop` might be non-zero in the same clock cycle.
`dout_eop_empty[11:0]`	Output	Indicates the number of empty (invalid) bytes in the end-of-packet 8-byte word indicated by `dout_eop`. If `dout_eop[z]` has the value of 0, then the IP core sets the value of `dout_eop_empty[(z+2):z]` to 0. However, if `dout_eop[z]` has the value of 1, then you must use the value of `dout_eop_empty[(z+2):z]` to determine the number of empty (invalid) bytes in the end-of-packet word (and therefore, the end-of-packet byte).

continued...

Signal Name	Direction	Description
		For example, if you have a LL 100GbE IP core and you observe that in the current `clk_rxmac` clock cycle, `dout_eop` has the value of 4'b0100 and `dout_eop_empty` has the value of 12'b000_110_000_000, you can conclude that byte 6 in word 2 of `dout_d` is an end-of-packet byte.
`dout_idle[3:0]`	Output	Indicates the words in `dout_d` that hold Idle bytes or control information rather than Ethernet data. This signal is one-hot encoded.
`rx_error[5:0]`	Output	Reports certain types of errors in the Ethernet frame whose contents are currently being transmitted on the client interface. This signal is valid in EOP cycles only. To ensure you can identify the corresponding packet, you must turn on **Enable alignment EOP on FCS word** in the LL 100GbE parameter editor. The individual bits report different types of errors: • Bit [0]: Malformed packet error. If this bit has the value of 1, the packet is malformed. The IP core identifies a malformed packet when it receives a control character that is not a terminate character, while receiving the packet. • Bit [1]: CRC error. If this bit has the value of 1, the IP core detected a CRC error in the frame. If you turn on **Enable alignment EOP on FCS word**, this bit and the `rx_fcs_error` signal behave identically. • Bit [2]: undersized payload. If this bit has the value of 1, the frame size is between nine and 63 bytes, inclusive. The IP core does not recognize an incoming frame of size eight bytes or less as a frame, and those cases are not reported here. • Bit [3]: oversized payload. If this bit has the value of 1, the frame size is greater than the maximum frame size programmed in the `MAX_RX_SIZE_CONFIG` register at offset 0x506. • Bit [4]: payload length error. If this bit has the value of 1, the payload received in the frame did not match the length field value, and the value in the length field is less than 1536 bytes. This bit only reports errors if you set bit [0] of the `CFG_PLEN_CHECK` register at offset 0x50A to the value of 1. • Bit [5]: Reserved.
`rx_fcs_error`	Output	The current or most recent EOP byte is part of a frame with an incorrect FCS (CRC-32) value. By default, the IP core asserts `rx_fcs_error` in the same cycle as the `dout_eop` signal. However, if you turn off **Enable alignment EOP on FCS word** in the LL 100GbE parameter editor, the `rx_fcs_error` signal might lag the `dout_eop` signal for the frame. Runt frames always force an FCS error condition. However, 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 as a runt.
`rx_fcs_valid`	Output	When set, indicates that `rx_fcs_error` has a valid value in the current clock cycle.

continued...

Signal Name	Direction	Description
rx_status[2:0]	Output	Indicates the IP core received a control frame on the Ethernet link. This signal identifies the type of control frame the IP core is passing through to the client interface. This signal is valid in EOP cycles only. To ensure you can identify the corresponding packet, you must turn on **Enable alignment EOP on FCS word** in the LL 100GbE parameter editor. The individual bits report different types of received control frames: • Bit [0]: Indicates the IP core received a standard flow control frame. If the IP core is in standard flow control mode and the cfg_fwd_ctrl bit of the RX_PAUSE_FWD register has the value of 0, this bit maintains the value of 0. • Bit [1]: Indicates the IP core received a priority flow control frame. If the IP core is in priority flow control mode and the cfg_fwd_ctrl bit of the RX_PAUSE_FWD register has the value of 0, this bit maintains the value of 0. • Bit [2]: Indicates the IP core received a control frame that is not a flow control frame.
dout_valid	Output	The dout_d bus contents are valid. This signal is occasionally deasserted due to clock crossing.
clk_rxmac	Output	RX MAC clock. The clock frequency should be390.625 MHz. The clk_rxmac clock is derived from the recovered CDR clock.

The data bytes use 100 Gigabit Media Independent Interface (CGMII-like) encoding. For packet payload bytes, the dout_c bit is set to 0 and the dout_d byte is the packet data. You can use this information to transmit out-of-spec data such as customized preambles when implementing non-standard variants of the *IEEE 802.3ba-2010 High Speed Ethernet Standard*.

In RX preamble pass-through mode, dout_c has the value of 1 while the start byte of the preamble is presented on the RX interface, and dout_c has the value of 0 while the remainder of the preamble sequence (six-byte preamble plus SFD byte) is presented on the RX interface.

Related Information

- LL 100GbE IP Core MAC Configuration Registers on page 100
  Describes the MAX_RX_SIZE_CONFIG and CFG_PLEN_CHECK registers.
- Control Frame Identification on page 52

3.2.5. Low Latency 100GbE CAUI–4 PHY

The LL 100GbE PHY IP core configured in an Arria 10 GT device supports CAUI-4 PCS and PMA at 4 x 25.78125 Gbps.

Related Information

LL 100GbE IP Core Device Speed Grade Support on page 7
Information about the device speed grades that support the LL 100GbE CAUI–4 IP core variation.

3.2.6. External Reconfiguration Controller

LL 100GbE IP cores that target a Stratix V device require an external reconfiguration controller.

Intel recommends that you configure a Transceiver Reconfiguration Controller for your Stratix V LL 100GbE IP core.

Related Information

External Transceiver Reconfiguration Controller Required in Stratix V Designs on page 26
Information about configuring and connecting the Transceiver Reconfiguration Controller. Includes signal descriptions.

3.2.7. External Transceiver PLL

LL 100GbE IP cores that target an Arria 10 device require an external transceiver PLL. The number and type of transceiver PLLs your design requires depends on the transceiver channels and available clock networks.

Related Information

- Transceiver PLL Required in Arria 10 Designs on page 26
  Information about configuring and connecting the external PLLs. Includes signal descriptions.
- Arria 10 Transceiver PHY User Guide
  Information about the correspondence between transceiver PLLs and transceiver channels, and information about how to configure an external transceiver PLL for your own design.

3.2.8. External TX MAC PLL

If you turn on **Use external TX MAC PLL** in the LL 100GbE parameter editor, the IP core has an extra input port, `clk_txmac_in`, which drives the TX MAC clock. You must connect this input port to a clock source, usually a PLL on the device.

The port is expected to receive the clock from the external TX MAC PLL and drives the internal clock `clk_txmac`. The required TX MAC clock frequency is 390.625 MHz. User logic must drive `clk_txmac_in` from a PLL whose input is the PHY reference clock, `clk_ref`.

3.2.9. Congestion and Flow Control Using Pause Frames

If you turn on flow control by setting **Flow control mode** to the value of **Standard flow control** or **Priority-based flow control** in the LL 100GbE parameter editor, the LL 100GbE IP core provides flow control to reduce congestion at the local or remote link partner. When either link partner experiences congestion, the respective transmit control sends pause frames. The pause frame instructs the remote transmitter to stop sending data for the duration that the congested receiver specified in an incoming XOFF frame. In the case of priority-based flow control, the pause frame applies to only the indicated priority class.

When the IP core receives the XOFF pause control frame, if the following conditions all hold, the IP core behavior depends on the flow control scheme.

- In priority-based flow control, the IP core informs the TX client of the received XOFF frame. The IP core continues to process packets it receives on the TX client interface. The IP core tracks the pause quanta countdown internally and informs the TX client when the pause period is ended.

  Note: In the priority-based flow control scheme, the client is responsible to throttle the flow of data to the TX client interface after the IP core informs the client of the received XOFF frame.

- In standard flow control, the IP core TX MAC does not process TX packets and in fact, prevents the client from sending additional data to the TX client interface, for the duration of the pause quanta of the incoming pause frame.

The IP core responds to an incoming XOFF pause control frame if the following conditions all hold:

- The relevant `cfg_enable` bit of the RX_PAUSE_ENABLE register has the value of 1.
- In the case of standard flow control, bit [0] of the TX_XOF_EN register also has the value of 1.
- Address matching is positive.

The pause quanta can be configured in the pause quanta register of the device sending XOFF frames. If the pause frame is received in the middle of a frame transmission, the transmitter finishes sending the current frame and then suspends transmission for a period specified by the pause quanta. Data transmission resumes when a pause frame with quanta of zero is received or when the timer has expired. The pause quanta received overrides any counter currently stored. When more than one pause quanta is sent, the value of the pause is set to the last quanta received.

XOFF pause frames stop the remote transmitter. XON pause frames let the remote transmitter resume data transmission.

One pause quanta fraction is equivalent to 512 bit times. The duration of a pause quantum is a function of the client interface datapath width (512 bits) and the system clock (`clk_txmac`) frequency (390.625 MHz).

Figure 24. The XOFF and XON Pause Frames for Standard Flow Control

XOFF Frame		XON Frame
START[7:0]		START[7:0]
PREAMBLE[47:0]		PREAMBLE[47:0]
SFD[7:0]		SFD[7:0]
DESTINATION ADDRESS[47:0] = 0x010000C28001 (1)		DESTINATION ADDRESS[47:0] = 0x010000C28001
SOURCE ADDRESS[47:0]		SOURCE ADDRESS[47:0]
TYPE[15:0] = 0x8808		TYPE[15:0] = 0x8808
OPCODE[15:0] = 0x001 (standard FC)		OPCODE[15:0] = 0x001 (standard FC)

continued...

(1) This is a multicast destination address.

PAUSE QUANTA[15:0] = 0xP1, 0xP2 [2]	PAUSE QUANTA[15:0] = 0x0000
PAD[335:0]	PAD[335:0]
CRC[31:0]	CRC[31:0]

Figure 25. The XOFF and XON Pause Frames for Priority Flow Control

XOFF Frame	XON Frame
START[7:0]	START[7:0]
PREAMBLE[47:0]	PREAMBLE[47:0]
SFD[7:0]	SFD[7:0]
DESTINATION ADDRESS[47:0] = 0x010000C28001 [1]	DESTINATION ADDRESS[47:0] = 0x010000C28001
SOURCE ADDRESS[47:0]	SOURCE ADDRESS[47:0]
TYPE[15:0] = 0x8808	TYPE[15:0] = 0x8808
OPCODE[15:0] = 0x0101 (PFC)	OPCODE[15:0] = 0x0101 (PFC)
PRIORITY_ENABLE[15:0][3]	PRIORITY_ENABLE[15:0] [4]
TIME0[15:0] [5]	TIME0[15:0] = 0x0000
...	...
TIME7[15:0] [5]	TIME7[15:0] = 0x0000
PAD[207:0]	PAD[207:0]
CRC[31:0]	CRC[31:0]

3.2.9.1. Conditions Triggering XOFF Frame Transmission

The LL 100GbE IP core supports retransmission. In retransmission mode, the IP core retransmits a XOFF frame periodically, extending the pause time, based on signal values.

(2) The bytes P1 and P2 are filled with the value configured in the TX_PAUSE_QUANTA register.

(3) Bit [n] has the value of 1 if the TIMEn field is valid.

(4) Bit [n] has the value of 1 if the XON request applies to priority queue n.

(5) The TIMEn field is filled with the value available in the TX_PAUSE_QUANTAregister when the TX_PAUSE_QNUMBER register holds the value of n.

The TX MAC transmits XOFF frames when one of the following conditions occurs:

- Client requests XOFF transmission—A client can explicitly request that XOFF frames be sent using the pause control interface signal. When `pause_insert_tx` is asserted, an XOFF frame is sent to the Ethernet network when the current frame transmission completes.
- Host (software) requests XOFF transmission—Setting the pause request register triggers a request that an XOFF frame be sent.
- Retransmission mode—If the retransmit hold-off enable bit has the value of 1, and the `pause_insert_tx` signal remains asserted or the pause request register value remains high, when the time duration specified in the hold-off quanta register has lapsed after the previous XOFF transmission, the TX MAC sends another XOFF frame to the Ethernet network. While the IP core is paused in retransmission mode, you cannot use either of the other two methods to trigger a new XOFF frame: the signal or register value is already high.

Note: Intel recommends that you use the `pause_insert_tx` signal to backpressure the remote Ethernet node.

Note: Intel recommends that you set and maintain the value of the retransmit hold-off enable bit at 1 to control the rate of XOFF pause frame transmission.

3.2.9.2. Conditions Triggering XON Frame Transmission

The TX MAC transmits XON frames when one of the following conditions occurs:

- Client requests XON transmission—A client can explicitly request that XON frames be sent using the pause control interface signal. When `pause_insert_tx` is de-asserted, an XON frame is sent to the Ethernet network when the current frame transmission completes.
- Host (software) requests XON transmission—Resetting the pause request register triggers a request that an XON frame be sent.

3.2.10. Pause Control and Generation Interface

The pause control interface implements flow control as specified by the *IEEE 802.3ba 2010 High Speed Ethernet Standard*. If you turn on priority-based flow control, the interface implements the *IEEE Standard 802.1Qbb*. The pause logic, upon receiving a pause packet, temporarily stops packet transmission, and can pass the pause packets through as normal traffic or drop the pause control frames in the RX direction.

Table 20. Pause Control and Generation Signals

Describes the signals that implement pause control. These signals are available only if you turn on flow control in the LL 100GbE parameter editor.

Signal Name	Direction	Description
`pause_insert_tx[N-1:0]` (6)	Input	Level signal which directs the IP core to insert a pause frame for priority traffic class [n] on the Ethernet link. If bit [n] of the TX_PAUSE_EN register has the value of 1, the IP core transmits an XOFF frame when this signal is

continued...

(6) N is the number of priority queues. If the IP core implements Ethernet standard flow control, N is 1.

Signal Name	Direction	Description
		first asserted. If you enable retransmission, the IP core continues to transmit XOFF frames periodically until the signal is de-asserted. When the signal is deasserted, the IP core inserts an XON frame.
pause_receive_rx[N-1:0](6)	Output	Asserted to indicate an RX pause signal match. The IP core asserts bit [n] of this signal when it receives a pause request with an address match, to signal the TX MAC to throttle its transmissions from priority queue [n] on the Ethernet link.

Related Information

- Pause Control Frame Filtering on page 63
  Information about enabling and disabling the pause packets pass-through.
- Pause Registers on page 102
  You can access the pause functionality using the pause registers for any LL 100GbE IP core variation that includes a MAC component. Values you program in the registers specify the pause quanta.

3.2.11. Pause Control Frame Filtering

The LL 100GbE IP core supports options to enable or disable the following features for incoming pause control frames. These options are available if you set the **Flow control mode** parameter to the value of **Standard flow control** or **Priority-based flow control**.

- Processing—You can enable or disable pause frame processing. If you disable pause frame processing, the IP core does not modify its behavior in response to incoming pause frames on the Ethernet link. You can enable or disable pause frame processing with the `cfg_enable` bit of the RX_PAUSE_ENABLE register. By default, RX pause frame processing is enabled.

  Note: IP core variations that target an Arria 10 device do not process standard pause frames that are runts or have FCS errors, for any setting of the `cfg_enable` bit of the RX_PAUSE_ENABLE register.

- Filtering—If pause frame processing is enabled, the IP core automatically performs address filtering on incoming pause control frames before processing them. You set the matching address value in these registers:

  — `RX_PAUSE_DADDRL[31:0]` at offset 0x707

  — `RX_PAUSE_DADDRH[15:0]` at offset 0x708

  The 48-bit address (`{RX_PAUSE_DADDRH[15:0],RX_PAUSE_DADDRL[31:0]}`) can be an individual MAC address, a multicast address, or a broadcast address.

- TX MAC filtering—For standard flow control only, Intel provides an additional level of filtering to enable or disable the TX MAC from responding to notification from the RX MAC that it received an incoming pause frame with an address match. Even if the RX MAC processes an incoming pause frame, you can separately set the TX MAC to ignore the RX MAC request to pause outgoing frames, by setting bit [0] of the TX_XOF_EN register to the value of 0. By default this register field has the value of 1.

- Pass-through—The LL 100GbE IP core can pass the matching pause packets through as normal traffic or drop these pause control frames in the RX direction. You can enable and disable pass-through with the `cfg_fwd_ctrl` bit of the RX_PAUSE_FWD register. By default, pass-through is disabled. All non-matching pause frames are passed through to the RX client interface irrespective of the `cfg_fwd_ctrl` setting.

The following rules define pause control frames filtering control:

1. If you have disabled pause frame processing, by setting the `cfg_enable` bit of the RX_PAUSE_ENABLE register to the value of 0, the IP core drops packets that enter the RX MAC and match the destination address, length, and type of 0x8808 with an opcode of 0x1 (pause packets).

   Note: IP core variations that target an Arria 10 device do not process standard Ethernet pause frames that are runts or have FCS errors.

2. If you have enabled pause frame processing, and the destination address in the pause frame is a match, when the RX MAC receives a pause packet it passes a pause request to the TX MAC. The RX MAC only processes pause packets with a valid packet multicast address or a destination address matching the destination address specified in the RX_PAUSE_DADDR1 and RX_PAUSE_DADDR0 registers, or in the RX_PAFC_DADDRH and RX_PFC_DADDRL registers, as appropriate for the flow control mode . If you have turned on pause frame pass-through, the RX MAC also forwards the pause frame to the RX client interface. If you have not turned on pause frame pass-through, the RX MAC does not forward the matching pause frame to the RX client interface.

3. In priority-based flow control, or in standard flow control if you have enabled TX MAC filtering, when the TX MAC receives a pause request from the RX MAC, it pauses transmission on the TX Ethernet link.

Pause packet pass-through does not affect the pause functionality in the TX or RX MAC. Pass-through applies equally to pause control frames that are runts or have FCS errors.

3.2.12. Link Fault Signaling Interface

If you turn on **Enable link fault generation** in the LL 100GbE parameter editor, the LL 100GbE IP core provides link fault signaling as defined in the *IEEE 802.3ba-2010 High Speed Ethernet Standard* and Clause 66 of the *IEEE 802.3-2012 Ethernet Standard*, based on the `LINK_FAULT_CONFIG` register settings. The LL 100GbE MAC includes a Reconciliation Sublayer (RS) located between the MAC and the XLGMII or CGMII to manage local and remote faults. Link fault signaling on the Ethernet link is disabled by default but can be enabled by bit [0] of the `LINK_FAULT_CONFIG` register. When the `LINK_FAULT_CONFIG` register bits [1:0] have the value of 2'b01, link fault signaling is enabled in normal bidirectional mode. In this mode, the local RS TX logic transmits remote fault sequences in case of a local fault and transmits IDLE control words in case of a remote fault.

If you turn on bit [1] of the `LINK_FAULT_CONFIG` register, the IP core conforms to Clause 66 of the *IEEE 802.3-2012 Ethernet Standard*. When `LINK_FAULT_CONFIG[1:0]` has the value of 2'b11, the IP core transmits the fault sequence ordered sets in the interpacket gaps according to the clause requirements.

The RS RX logic sets `remote_fault_status` or `local_fault_status` to 1 when the RS RX block receives remote fault or local fault sequence ordered sets. When valid data is received in more than 127 columns, the RS RX logic resets the relevant fault status (`remote_fault_status` or `local_fault_status`) to 0.

The IEEE standard specifies RS monitoring of RXC<7:0> and RXD<63:0> for Sequence `ordered_sets`. For more information, refer to *Figure 81–9—Link Fault Signaling state diagram* and *Table 81-5—Sequence* `ordered_sets` in the *IEEE 802.3ba 2010 High Speed Ethernet Standard*. The variable `link_fault` is set to indicate the value of an RX Sequence `ordered_set` when four `fault_sequences` containing the same fault value are received with fault sequences separated by less than 128 columns and with no intervening `fault_sequences` of different fault values. The variable `link_fault` is set to OK following any interval of 128 columns not containing a remote fault or local fault Sequence `ordered_set`.

Table 21. Signals of the Link Fault Signaling Interface

These signals are available only if you turn on **Enable link fault generation** in the LL 100GbE parameter editor.

Signal Name	Direction	Description
`remote_fault_status`	Output	Asserted by the IP core when it detects a real-time remote fault in the RX MAC. The IP core asserts this signal regardless of the settings in the `LINK_FAULT_CONFIG` register. This signal is clocked by `clk_rxmac`.
`local_fault_status`	Output	Asserted by the IP core when it detects a real-time local fault in the RX MAC. The IP core asserts this signal regardless of the settings in the `LINK_FAULT_CONFIG` register.

continued...

Signal Name	Direction	Description
		This signal is clocked by clk_rxmac.
unidirectional_en	Output	The IP core asserts this signal if it includes Clause 66 support for remote link fault reporting on the Ethernet link. Connects to the Unidir Enable field in bit [1] of the LINK_FAULT_CONFIG register at offset 0x405. This signal is clocked by clk_txmac.
link_fault_gen_en	Output	The IP core asserts this signal if the PCS is enabled to generate a remote fault sequence on the Ethernet link when appropriate. Connects to the Link Fault Reporting Enable field in bit [0] of the LINK_FAULT_CONFIG register at offset 0x405. This signal is clocked by clk_txmac.

Related Information

- Link Fault Signaling Registers on page 99
  Information about the Link Fault Reporting Enable register and the Unidir Enable register.
- IEEE website
  The *IEEE 802.3ba –2010 High Speed Ethernet Standard* and the *IEEE 802.3 – 2012 Ethernet Standard* are available on the IEEE website.

3.2.13. Statistics Counters Interface

The statistics counters modules are synthesis options that you select in the LL 100GbE parameter editor. However, the statistics status bit output vectors are provided whether you select the statistics counters module option or not.

The increment vectors are brought to the top level as output ports. The increment vectors also function internally as input ports to the control and status registers (CSR).

Table 22. Statistics Counters Increment Vectors

The TX statistics counter increment vectors are clocked by the clk_txmac clock, and the RX statistics counter increment vectors are clocked by the clk_rxmac clock.

Name	Signal Direction	Description
TX Statistics Counter Increment Vectors		
tx_inc_64	Output	Asserted for one cycle when a 64-byte TX frame is transmitted.
tx_inc_127	Output	Asserted for one cycle when a 65–127 byte TX frame is transmitted.
tx_inc_255	Output	Asserted for one cycle when a 128–255 byte TX frame is transmitted.
tx_inc_511	Output	Asserted for one cycle when a 256–511 byte TX frame is transmitted.
tx_inc_1023	Output	Asserted for one cycle when a 512–1023 byte TX frame is transmitted.
tx_inc_1518	Output	Asserted for one cycle when a 1024–1518 byte TX frame is transmitted.
tx_inc_max	Output	Asserted for one cycle when a maximum-size TX frame is transmitted. You program the maximum number of bytes in the MAX_TX_SIZE_CONFIG register.
tx_inc_over	Output	Asserted for one cycle when an oversized TX frame is transmitted.

continued...

Name	Signal Direction	Description
		You program the maximum number of bytes in the MAX_TX_SIZE_CONFIG register.
tx_inc_mcast_data_err	Output	Asserted for one cycle when an errored multicast TX frame, excluding control frames, is transmitted.
tx_inc_mcast_data_ok	Output	Asserted for one cycle when a valid multicast TX frame, excluding control frames, is transmitted.
tx_inc_bcast_data_err	Output	Asserted for one cycle when an errored broadcast TX frame, excluding control frames, is transmitted.
tx_inc_bcast_data_ok	Output	Asserted for one cycle when a valid broadcast TX frame, excluding control frames, is transmitted.
tx_inc_ucast_data_err	Output	Asserted for one cycle when an errored unicast TX frame, excluding control frames, is transmitted.
tx_inc_ucast_data_ok	Output	Asserted for one cycle when a valid unicast TX frame, excluding control frames, is transmitted.
tx_inc_mcast_ctrl	Output	Asserted for one cycle when a valid multicast TX frame is transmitted.
tx_inc_bcast_ctrl	Output	Asserted for one cycle when a valid broadcast TX frame is transmitted.
tx_inc_ucast_ctrl	Output	Asserted for one cycle when a valid unicast TX frame is transmitted.
tx_inc_pause	Output	Asserted for one cycle when a valid pause TX frame is transmitted.
tx_inc_fcs_err	Output	Asserted for one cycle when a TX packet with FCS errors is transmitted.
tx_inc_fragment	Output	Asserted for one cycle when a TX frame less than 64 bytes and reporting a CRC error is transmitted.
tx_inc_jabber	Output	Asserted for one cycle when an oversized TX frame reporting a CRC error is transmitted.
tx_inc_sizeok_fcserr	Output	Asserted for one cycle when a valid TX frame with FCS errors is transmitted.
RX Statistics Counter Increment Vectors		
rx_inc_runt	Output	Asserted for one cycle when an RX runt packet is received.
rx_inc_64	Output	Asserted for one cycle when a 64-byte RX frame is received.
rx_inc_127	Output	Asserted for one cycle when a 65–127 byte RX frame is received.
rx_inc_255	Output	Asserted for one cycle when a 128–255 byte RX frame is received.
rx_inc_511	Output	Asserted for one cycle when a 256–511 byte RX frame is received.
rx_inc_1023	Output	Asserted for one cycle when a 512–1023 byte RX frame is received.
rx_inc_1518	Output	Asserted for one cycle when a 1024–1518 byte RX frame is received.
rx_inc_max	Output	Asserted for one cycle when a maximum-size RX frame is received. You program the maximum number of bytes in the MAX_RX_SIZE_CONFIG register.
rx_inc_over	Output	Asserted for one cycle when an oversized RX frame is received. You program the maximum number of bytes in the MAX_RX_SIZE_CONFIG register.
rx_inc_mcast_data_err	Output	Asserted for one cycle when an errored multicast RX frame, excluding control frames, is received.

continued...

Name	Signal Direction	Description
rx_inc_mcast_data_ok	Output	Asserted for one cycle when valid a multicast RX frame, excluding control frames, is received.
rx_inc_bcast_data_err	Output	Asserted for one cycle when an errored broadcast RX frame, excluding control frames, is received.
rx_inc_bcast_data_ok	Output	Asserted for one cycle when a valid broadcast RX frame, excluding control frames, is received.
rx_inc_ucast_data_err	Output	Asserted for one cycle when an errored unicast RX frame, excluding control frames, is received.
rx_inc_ucast_data_ok	Output	Asserted for one cycle when a valid unicast RX frame, excluding control frames, is received.
rx_inc_mcast_ctrl	Output	Asserted for one cycle when a valid multicast RX frame is received.
rx_inc_bcast_ctrl	Output	Asserted for one cycle when a valid broadcast RX frame is received.
rx_inc_ucast_ctrl	Output	Asserted for one cycle when a valid unicast RX frame is received.
rx_inc_pause	Output	Asserted for one cycle when valid RX pause frames are received.
rx_inc_fcs_err	Output	Asserted for one cycle when a RX packet with FCS errors is received.
rx_inc_fragment	Output	Asserted for one cycle when a RX frame less than 64 bytes and reporting a CRC error is received.
rx_inc_jabber	Output	Asserted for one cycle when an oversized RX frame reporting a CRC error is received.
rx_inc_sizeok_fcserr	Output	Asserted for one cycle when a valid RX frame with FCS errors is received.
rx_inc_pause_ctrl_err	Output	Asserted for one cycle when an errored pause RX frame is received.
rx_inc_mcast_ctrl_err	Output	Asserted for one cycle when an errored multicast RX control frame is received.
rx_inc_bcast_ctrl_err	Output	Asserted for one cycle when an errored broadcast RX control frame is received.
rx_inc_ucast_ctrl_err	Output	Asserted for one cycle when an errored unicast RX control frame is received.

Related Information

- TX Statistics Registers on page 106
- RX Statistics Registers on page 110

3.2.13.1. OctetOK Count Interface

The statistics counters include two 64-bit counters, RxOctetsOK and TxOctetsOK, which count the payload bytes (octets) in frames with no FCS, undersized, oversized, or payload length errors. The RxOctetsOK register maintains a cumulative count of the payload bytes in the qualifying received frames, and complies with section 5.2.1.14 of the *IEEE Standard 802.3-2008*. The TxOctetsOK register maintains a cumulative count of the payload bytes in the qualifying transmitted frames, and complies with section 5.2.2.18 of the *IEEE Standard 802.3-2008*.

To support payload size checking per frame, the LL 100GbE IP core provides an octetOK count interface instead of standard increment vectors. For most purposes, the number of payload bytes per frame is of more interest than the cumulative count, and an increment vector that pulses when the counter increments would be difficult to track. For each of these two registers, the IP core maintains two signals. A 16-bit signal provides a count of the payload bytes in the current frame, and the other signal pulses to indicate when the first signal is valid. The per-frame count is valid only when the valid signal is asserted. All signals in this interface are functional even if you do not turn on the corresponding statistics module.

Table 23. OctetOK Count Interface Signals

The signals for received frames are clocked by the `clk_rxmac` clock. The signals for transmitted frames are clocked by the `clk_txmac` clock.

Name	Signal Direction	Description
`tx_inc_octetsOK[15:0]`	Output	When `tx_inc_octetsOK_valid` is asserted, `tx_inc_octetsOK[15:0]` holds the count of payload bytes in the current valid frame.
`tx_inc_octetsOK_valid`	Output	Pulses to indicate that `tx_inc_octetsOK[15:0]` currently holds the number of payload bytes for the current transmitted frame, and that the current frame is a qualifying frame. A qualifying frame has no FCS errors, no oversized error, no undersized error, and no payload length error.
`rx_inc_octetsOK[15:0]`	Output	When `rx_inc_octetsOK_valid` is asserted, `rx_inc_octetsOK[15:0]` holds the count of payload bytes in the current valid frame.
`rx_inc_octetsOK_valid`	Output	Pulses to indicate that `rx_inc_octetsOK[15:0]` currently holds the number of payload bytes for the current received frame, and that the current frame is a qualifying frame. A qualifying frame has no FCS errors, no oversized error, no undersized error, and no payload length error.

3.2.14. 1588 Precision Time Protocol Interfaces

If you turn on **Enable 1588 PTP**, the LL 100GbE 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 timestamp accuracy of ± 7 ns.

1588 PTP packets carry timestamp information. The LL 100GbE 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 LL 100GbE core, you can use the same module to provide the current time of day for multiple modules in your system.

Related Information

- External Time-of-Day Module for Variations with 1588 PTP Feature on page 28
- 1588 PTP Registers on page 114

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

3.2.14.1. Implementing a 1588 System That Includes a LL 100GbE 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 LL 100GbE 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 LL 100GbE 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 26. Example Ethernet System with Ordinary Clock Master and Ordinary Clock Slave

You can implement both master and slave clocks using the LL 100GbE core with 1588 PTP functionality.

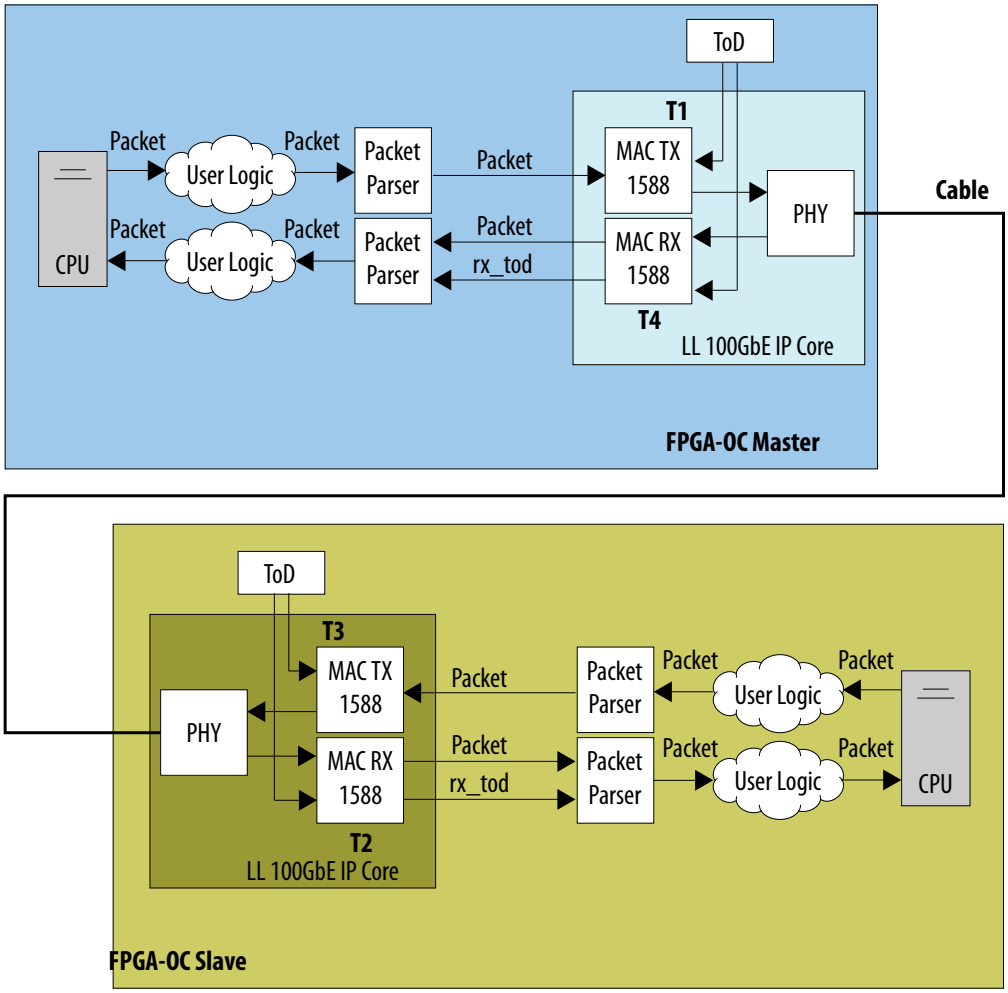


Figure 27. Hardware Configuration Example Using LL 100GbE core in a 1588 System in Transparent Clock Mode

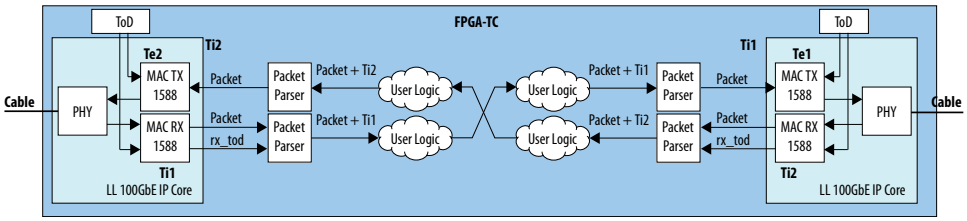


Figure 28. Software Flow Using Transparent Clock Mode System

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.

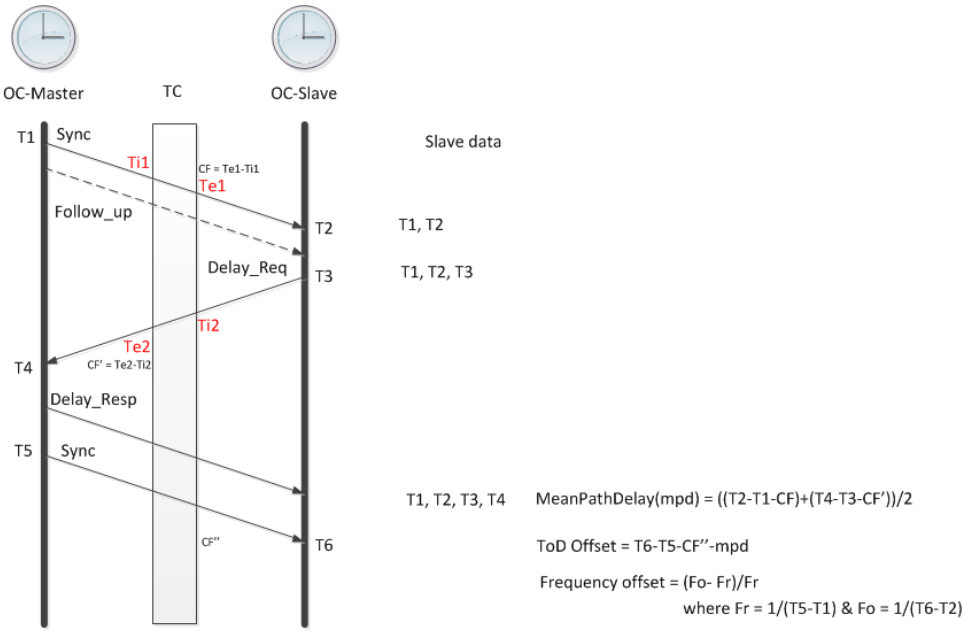
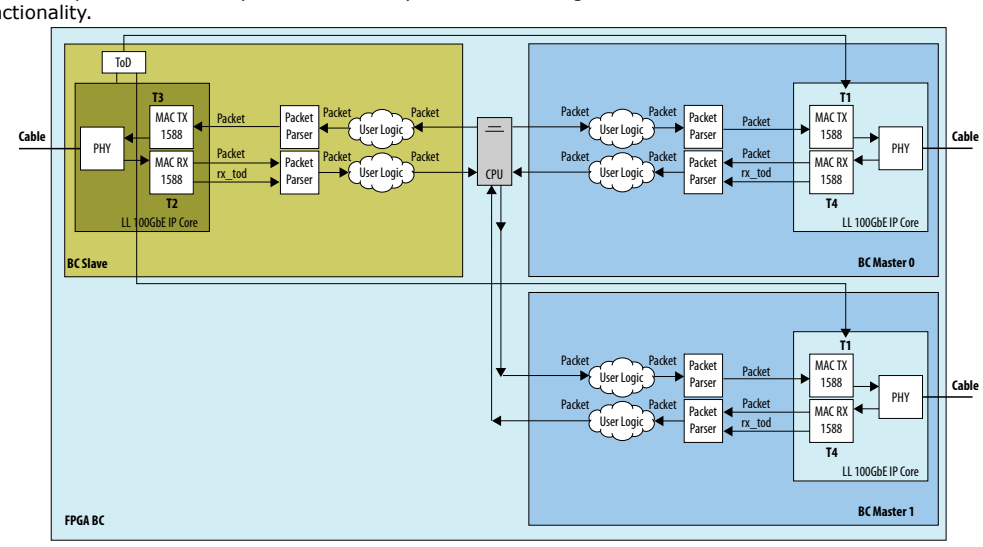


Figure 29. Example Boundary Clock with One Slave Port and Two Master Ports

You can implement a 1588 system in boundary clock mode using the LL 100GbE core with 1588 PTP functionality.



Related Information

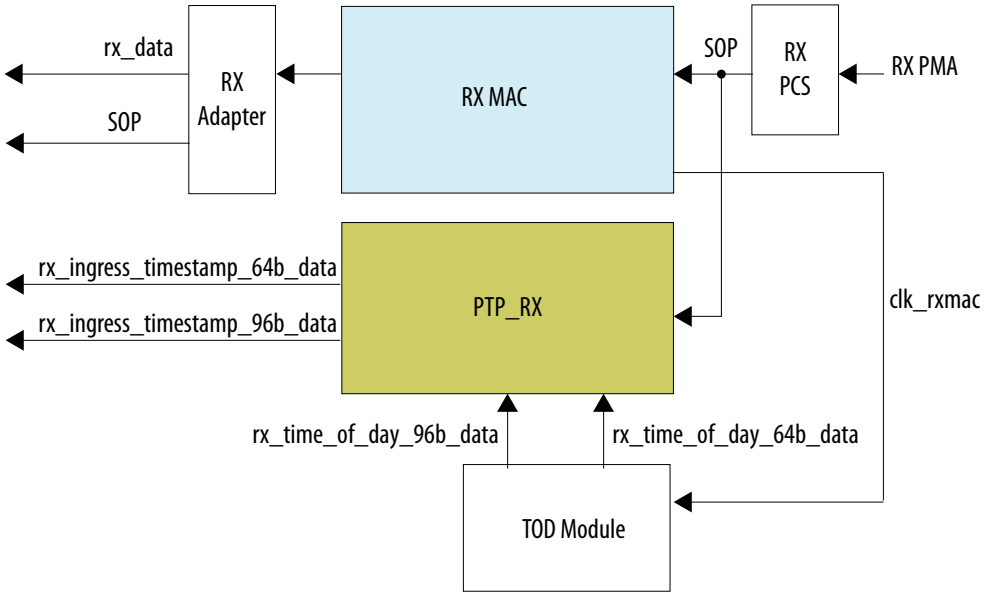
IEEE website

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

3.2.14.2. PTP Receive Functionality

If you turn on **Enable 1588 PTP** in the LL 100GbE 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 30. 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.

3.2.14.3. PTP Transmit Functionality

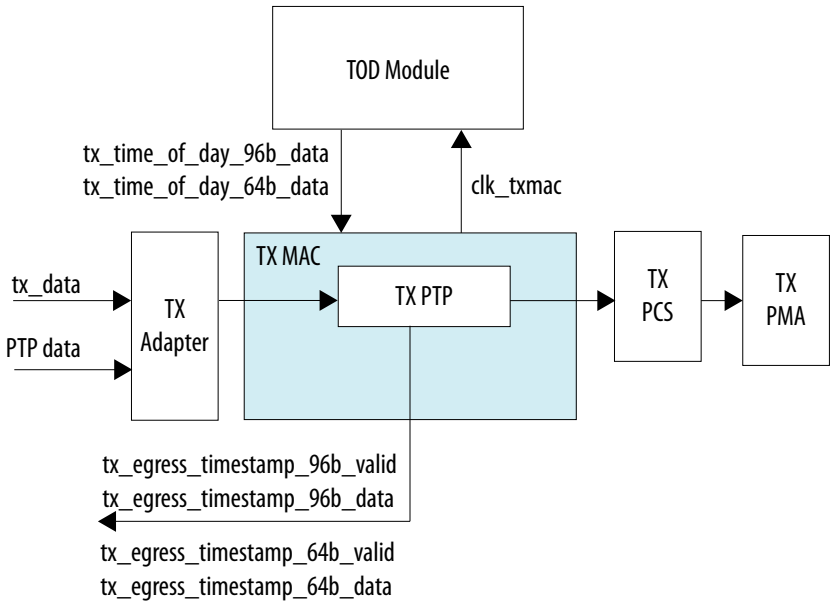
When you send a 1588 PTP packet to a LL 100GbE core with **Enable 1588 PTP** turned on in the parameter editor, you should assert the following respective input signals with the TX SOP signal to tell the IP core the PTP operations or processes that the IP core should perform to the packet:

- `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 31. 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.

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 **Timestamp 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-specified extra 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-specified extra latency from the TX_PTP_EXTRA_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 114
- IEEE website
  The *IEEE 1588-2008 Precision Clock Synchronization Protocol for Networked Measurement and Control Systems Standard* is available on the IEEE website.

3.2.14.4. External Time-of-Day Module for 1588 PTP Variations

LL 100GbE 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

External Time-of-Day Module for Variations with 1588 PTP Feature on page 28

3.2.14.5. PTP Timestamp and TOD Formats

The LL 100GbE 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 turn off the **Enable 64b Time of Day Format** parameter.

You control the format or formats the IP core supports with the **Enable 64b Time of Day Format** and **Enable 96b Time of Day Format** parameters. If you turn on **Enable 96b Time of Day Format**, 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 turn on **Enable 64b Time of Day Format** and turn off **Enable 96b Time of Day 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 settings for the **Enable 64b Time of Day Format** and **Enable 96b Time of Day Format** parameters.

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.

3.2.14.6. 1588 PTP Interface Signals

Table 24. 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
PTP Interface to TOD module		
`tx_time_of_day_96b_data[95:0]`	Input	Current V2-format (96-bit) TOD in `clk_txmac` clock domain. Connect this signal to the external TOD module. This signal is available only if you turn on the **Enable 96b Time of Day Format** parameter.
`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. This signal is available only if you turn on the **Enable 64b Time of Day Format** parameter.
`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. This signal is available only if you turn on the **Enable 96b Time of Day Format** parameter.
`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. This signal is available only if you turn on the **Enable 64b Time of Day Format** parameter.
PTP Interface to Client		
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. The TX client must assert and deassert this signal synchronously with the TX SOP signal for the 1588 PTP packet. If the TX client asserts this signal simultaneously with either of `tx_etstamp_ins_ctrl_residence_time_update` or `tx_egress_timestamp_request_valid`, 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. The TX client must assert and deassert this signal synchronously with the TX SOP signal for the 1588 PTP packet. 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. 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`. This signal is available only if you turn on the **Enable 96b Time of Day Format** parameter.

continued...

Signal Name	Direction	Description
`tx_etstamp_ins_ctrl_ingress_timestamp_64b[63:0]`	Input	Indicates the TOD (in Intel 64-bit format) when the packet entered the system. 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`. This signal is available only if you turn on the **Enable 64b Time of Day Format** parameter.
`tx_etstamp_ins_ctrl_timestamp_format`	Input	Specifies the timestamp format (V1 or V2 format) for the current packet if the TX client simultaneously asserts `tx_etstamp_ins_ctrl_timestamp_insert`. Values are: • 1'b0: 96-bit timestamp format (V2) • 1'b1: 64-bit timestamp format (V1) The TX client must maintain the desired value on this signal while the TX SOP signal is asserted. 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. *Note:* If you do not turn on the **Enable 96b Time of Day Format** parameter, the results of asserting `tx_etstamp_ins_ctrl_timestamp_insert` are undefined. Therefore, the timestamp in either case maps to the 96-bit signals.
`tx_etstamp_ins_ctrl_residence_time_calc_format`	Input	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 `tx_etstamp_ins_ctrl_residence_time_update`. Values are: • 1'b0: 96-bit TOD format (V2) • 1'b1: 64-bit TOD format (48-bit ns and 16-bit fns) The TX client must maintain the desired value on this signal while the TX SOP signal is asserted. If you set the **Time of day format** parameter to the value of **Enable 64-bit timestamp format** or **Enable both formats**, and the client specifies the 64-bit TOD format, the IP core uses the 64-bit TOD format for residence time calculation. If you set the **Time of day format** parameter to the value of **Enable 64-bit timestamp format** and the client specifies the 96-bit format (V2), the results are undefined.
`tx_etstamp_ins_ctrl_offset_timestamp[15:0]`	Input	Specifies the byte offset of the timestamp information in the current packet if the TX client simultaneously asserts `tx_etstamp_ins_ctrl_timestamp_insert`. 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. 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 `tx_etstamp_ins_ctrl_offset_correction_field`. The TX client must ensure that: • The offset supports inclusion of the entire timestamp in the packet. • If the packet is more than 256 bytes, the offset supports inclusion of the entire timestamp in the first 256 bytes of the packet. • 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).
`tx_etstamp_ins_ctrl_offset_correction_field[15:0]`	Input	If the TX client simultaneously asserts `tx_etstamp_ins_ctrl_residence_time_update`, this signal specifies the byte offset of the correction field in the current packet.

continued...

Signal Name	Direction	Description
		If the TX client simultaneously asserts `tx_etstamp_ins_ctrl_timestamp_insert` and deasserts (sets to the value of 0) the `tx_etstamp_ins_ctrl_timestamp_format` signal, this signal specifies the byte offset of bits [15:0]] of the timestamp. The TX client must maintain the desired value on this signal while the TX SOP signal is asserted. In addition, the TX client must ensure that: • The offset supports inclusion of the entire correction field or timestamp in the packet. • 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. • 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).
`tx_etstamp_ins_ctrl_checksum_zero`	Input	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. 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. 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.
`tx_etstamp_ins_ctrl_offset_checksum_field[15:0]`	Input	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. The TX client must ensure that: • The offset supports inclusion of the entire checksum in the packet. • 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).
`tx_etstamp_ins_ctrl_checksum_correct`	Input	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. 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. The application must assert this signal for correct processing of UDP IPv6 packets.
`tx_etstamp_ins_ctrl_offset_checksum_correction[15:0]`	Input	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_correct` signal. Holds the byte offset of the two bytes in the packet that the IP core should correct. In a PTP packet, two bytes before the CRC represent the valid byte offset for the checksum correction field. This signal is meaningful only in one-step clock mode. The TX client must ensure that: • The offset supports inclusion of the entire checksum in the packet. • 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).

continued...

Signal Name	Direction	Description
tx_egress_asymmetry_update	Input	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. This option is useful in one-step correction mode.
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. The TX client must assert and deassert this signal synchronously with the TX SOP signal for the 1588 PTP packet. If the TX client asserts this signal simultaneously with either of tx_etstamp_ins_ctrl_timestamp_insert or 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. This signal is available only if you turn on the **Enable 96b Time of Day Format** parameter.
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. This signal is available only if you turn on the **Enable 96b Time of Day Format** parameter.
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. This signal is available only if you turn on the **Enable 64b Time of Day Format** parameter.
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. This signal is available only if you turn on the **Enable 64b Time of Day Format** parameter.
tx_egress_timestamp_request_fingerprint[(W-1):0] where W is the value between 1 and 16, inclusive, that you specify for the **Timestamp fingerprint width** parameter	Input	Fingerprint of the current packet. 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] where W is the value between 1 and 16, inclusive, that you specify for the **Timestamp fingerprint width** 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. This signal is available only if you turn on the **Enable 96b Time of Day Format** parameter.
tx_egress_timestamp_64b_fingerprint[(W-1):0]	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.

continued...

Signal Name	Direction	Description
where W is the value between 1 and 16, inclusive, that you specify for the **Timestamp fingerprint width** parameter		This signal is available only if you turn on the **Enable 64b Time of Day Format** parameter.
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. This signal is available only if you turn on the **Enable 96b Time of Day Format** parameter.
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. This signal is available only if you turn on the **Enable 96b Time of Day Format** parameter.
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. This signal is available only if you turn on the **Enable 64b Time of Day Format** parameter.
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. This signal is available only if you turn on the **Enable 64b Time of Day Format** parameter.

3.2.15. PHY Status Interface

The rx_pcs_ready output signal is available to provide status information to user logic. This signal is asserted when the RX lanes are fully aligned and ready to receive data.

The tx_lanes_stable output signal is available to provide status information to user logic. This signal is asserted when the TX lanes are fully aligned and ready to transmit data.

3.2.16. Transceiver PHY Serial Data Interface

The core uses an <*n*>-lane digital interface to send data to the TX high-speed serial I/O pins operating at 10.3125 Gbps in the standard LL 100GbE variations and at 25.78125 Gbps in the CAUI–4 variations. The rx_serial and tx_serial ports connect to the 10.3125 Gbps or 25.78125 Gbps pins. Virtual lanes 0 and 1 transmit data on tx_serial[0].

3.2.17. Control and Status Interface

The control and status interface provides an Avalon-MM interface to the IP core control and status registers. The Avalon-MM interface implements a standard memory-mapped protocol. You can connect an embedded processor or JTAG Avalon master to this bus to access the control and status registers.

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

Table 25. Avalon-MM Control and Status Interface Signals

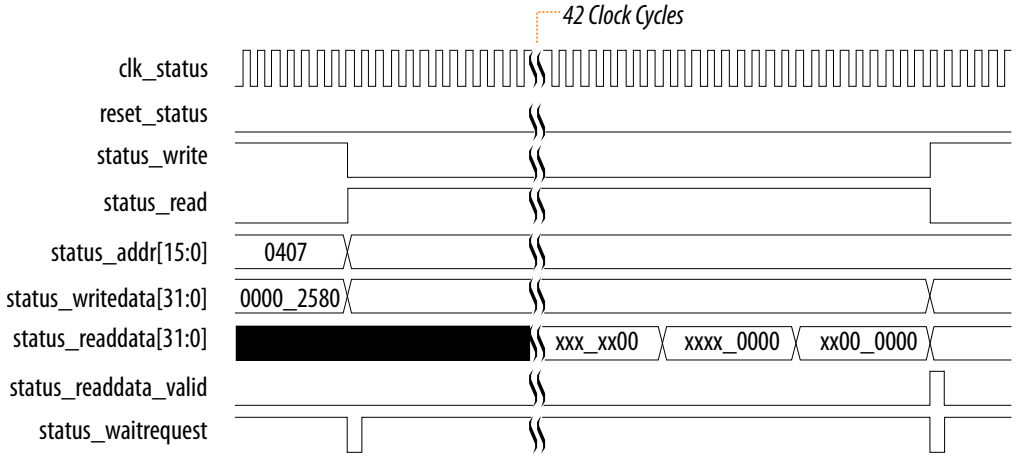
The `clk_status` clocks the signals on the LL 100GbE IP core control and status interface. The synchronous `reset_status` reset signal resets the interface.

Signal Name	Direction	Description
`status_addr [15:0]`	Input	Address for reads and writes
`status_read`	Input	Read command
`status_write`	Input	Write command
`status_writedata [31:0]`	Input	Data to be written
`status_readdata [31:0]`	Output	Read data
`status_readdata_valid`	Output	Read data is ready for use
`status_waitrequest`	Output	Busy signal indicating control and status interface cannot currently respond to requests
`status_read_timeout`	Output	Timeout signal indicating read data did not arrive when expected. Hardwired timeout counter is set so that this timeout should only occur in the presence of an error condition, such as `status_addr` with an undefined value. This signal is not an Avalon-MM defined signal

The status interface is designed to operate at a low frequencies, typically 100 MHz, so that control and status logic does not compete for resources with the surrounding high speed datapath.

Figure 32. Read and Write Register Access Example

The waveform demonstrates a write-read-write sequence of back-to-back register accesses. The delay from the time the application asserts `status_read` until the IP core asserts `status_readdata_valid` and deasserts `status_waitrequest` is approximately 80 `clk_status` cycles in this example.



Related Information

- Software Interface: Registers on page 94
- LL 100GbE IP Core Registers on page 96

- Avalon Interface Specifications
  For more information about the Avalon-MM protocol, including timing diagrams, refer to the *Avalon Memory-Mapped Interfaces* chapter.

3.2.18. Arria 10 Transceiver Reconfiguration Interface

Arria 10 variations provide a dedicated Avalon-MM interface, called the Arria 10 transceiver reconfiguration interface, to access the transceiver registers. You access the Arria 10 Native PHY IP core registers through this dedicated interface and not through the IP core general purpose control and status interface.

The Avalon-MM interface implements a standard memory-mapped protocol. You can connect an embedded processor or JTAG Avalon master to this bus to access the registers of the embedded Arria 10 Native PHY IP core.

Table 26. Avalon-MM Arria 10 Reconfiguration Interface Signals

The `reconfig_clk` clocks the signals on the LL 100GbE IP core Arria 10 transceiver reconfiguration interface. The synchronous `reconfig_reset` reset signal resets the interface.

Signal Name	Direction	Description
`reconfig_address [13:0]` `reconfig_address [11:0]` (CAUI-4)	Input	Address for reads and writes
`reconfig_read`	Input	Read command
`reconfig_write`	Input	Write command
`reconfig_writedata [31:0]`	Input	Data to be written
`reconfig_readdata [31:0]`	Output	Read data
`reconfig_waitrequest`	Output	Interface busy signal

The Arria 10 reconfiguration interface is designed to operate at a low frequency, 100 MHz to 125 MHz, so that the user's Arria 10 transceiver reconfiguration logic does not compete for resources with the surrounding high speed datapath.

Related Information

- Avalon Interface Specifications
  For more information about the Avalon-MM protocol, including timing diagrams, refer to the *Avalon Memory-Mapped Interfaces* chapter.
- Arria 10 Transceiver PHY User Guide
  Information about the Arria 10 Native PHY IP core hard PCS registers that you can program through the Arria 10 transceiver reconfiguration interface.
- Arria 10 Transceiver Registers
  Information about the Arria 10 transceiver registers.

3.2.19. Clocks

You must set the transceiver reference clock (`clk_ref`) frequency to a value that the IP core supports. The LL 100GbE IP core supports `clk_ref` frequencies of 644.53125 MHz ±100 ppm and 322.265625 MHz ± 100 ppm. The ±100ppm value is required for any clock source providing the transceiver reference clock.

Sync–E IP core variations are IP core variations for which you turn on **Enable SyncE** in the parameter editor. These variations provide the RX recovered clock as a top-level output signal. This option is available only for IP core variations that target an Arria 10 device.

The Synchronous Ethernet standard, described in the ITU-T G.8261, G.8262, and G.8264 recommendations, requires that the TX clock be filtered to maintain synchronization with the RX reference clock through a sequence of nodes. The expected usage is that user logic drives the TX PLL reference clock with a filtered version of the RX recovered clock signal, to ensure the receive and transmit functions remain synchronized. In this usage model, a design component outside the LL 100GbE IP core performs the filtering.

Table 27. Clock Inputs

Describes the input clocks that you must provide.

Signal Name	Description
`clk_status`	Clocks the control and status interface. The clock quality and pin chosen are not critical. `clk_status` is expected to be a 100–125 MHz clock.
`reconfig_clk`	Clocks the Arria 10 transceiver reconfiguration interface. The clock quality and pin chosen are not critical. `reconfig_clk` is expected to be a 100 MHz clock; the allowed frequency range depends on Arria 10 transceiver requirements and is not IP core specific.
`clk_ref`	In IP core variations that target an Arria 10 device, `clk_ref` is the reference clock for the transceiver RX CDR PLL. In other IP core variations, `clk_ref` is the reference clock for the transceiver TX PLL and the RX CDR PLL. The frequency of this input clock must match the value you specify for **PHY reference frequency** in the IP core parameter editor, with a ±100 ppm accuracy per the *IEEE 802.3ba-2010 High Speed Ethernet Standard.*. In addition, `clk_ref` must meet the jitter specification of the *IEEE 802.3ba-2010 High Speed Ethernet Standard*. The PLL and clock generation logic use this reference clock to derive the transceiver and PCS clocks. The input clock should be a high quality signal on the appropriate dedicated clock pin. Refer to the relevant device datasheet for transceiver reference clock phase noise specifications.
`clk_txmac_in`	If you turn on **Use external TX MAC PLL** in the LL 100GbE parameter editor, this clock drives the TX MAC. The port is expected to receive the clock from the external TX MAC PLL and drives the internal clock `clk_txmac`. The required TX MAC clock frequency is 390.625 MHz. User logic must drive `clk_txmac_in` from a PLL whose input is the PHY reference clock, `clk_ref`.
`tx_serial_clk[9:0]` `tx_serial_clk[3:0]` (CAUI-4)	These input clocks are present only in variations that target an Arria 10 device. They are part of the external PLL interface to these variations. Each clock targets a single transceiver PHY link. You must drive these clocks from one or more TX transceiver PLLs that you configure separately from the LL 100GbE IP core.

Table 28. Clock Outputs

Describes the output clocks that the IP core provides. In most cases these clocks participate in internal clocking of the IP core as well.

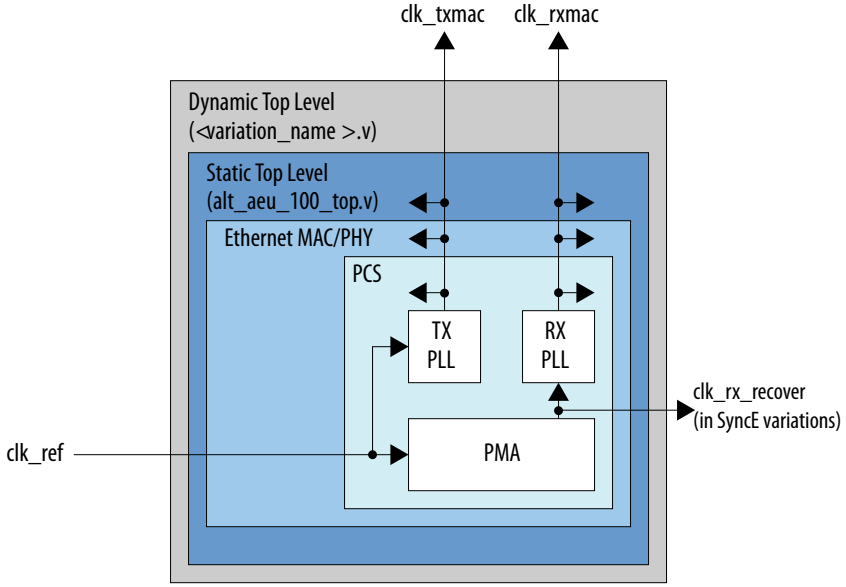
Signal Name	Description
`clk_txmac`	The TX clock for the IP core is `clk_txmac`. The TX MAC clock frequency is 390.625 MHz. If you turn on **Use external TX MAC PLL** in the LL 100GbE parameter editor, the `clk_txmac_in` input clock drives `clk_txmac`.
`clk_rxmac`	The RX clock for the IP core is `clk_rxmac`. The RX MAC clock frequency is 390.625 MHz.

continued...

Signal Name	Description
	This clock is only reliable when `rx_pcs_ready` has the value of 1. The IP core generates `clk_rxmac` from a recovered clock that relies on the presence of incoming RX data.
`clk_rx_recover`	RX recovered clock. This clock is available only if you turn on **Enable SyncE** in the LL 100GbE parameter editor. The RX recovered clock frequency is 322.265625 MHz in standard variations and 402.83203 MHz in CAUI-4 variations.. The expected usage is that you drive the TX transceiver PLL reference clock with a filtered version of `clk_rx_recover`, to ensure the receive and transmit functions remain synchronized in your Synchronous Ethernet system. To do so you must instantiate an additional component in your design. The IP core does not provide filtering.

Figure 33. Clock Generation Circuitry

Provides a high-level view of the clock generation circuitry and clock distribution to the transceiver. In Arria 10 variations, the TX transceiver PLL is configured outside the LL 100GbE IP core, and `clk_ref` drives the RX CDR PLL. In Arria 10 variations, you can connect a separate reference clock for the external TX transceiver PLL.



Related Information

- Transceiver PLL Required in Arria 10 Designs on page 26
  Information about configuring and connecting the external PLLs. Includes signal descriptions.
- Arria 10 Device Datasheet
  Provides transceiver reference clock phase noise specifications.
- Stratix V Device Datasheet
  Provides transceiver reference clock phase noise specifications.

3.2.20. Resets

The LL 100GbE IP core has a single asynchronous reset signal. Asserting this signal resets the full IP core. You must hold the reset signal asserted for ten `clk_status` clock cycles to ensure proper hold time.

You should not release the reset signal until after you observe that the reference clock is stable. If the reference clock is generated from an fPLL, wait until after the fPLL locks. Ten `clk_status` cycles should be sufficient for the fPLL to lock and the reference clock to stabilize.

Table 29. Asynchronous Reset Signal

Signal Name	Direction	Description
`reset_async`	Input	LL 100GbE IP core asynchronous reset signal

In addition, the LL 100GbE IP core has one or two of the following synchronous reset signals:

- `reset_status`—Resets the IP core control and status interface, an Avalon-MM interface. Associated clock is the `clk_status` clock, which clocks the control and status interface.
- `reconfig_reset`—Resets the IP core Arria 10 transceiver reconfiguration interface, an Avalon-MM interface. Associated clock is the `reconfig_clk`, which clocks the Arria 10 transceiver reconfiguration interface. This signal is available only in Arria 10 IP core variations.

3.3. Signals

This section lists the external signals of the different LL 100GbE IP core variations.

LL 100GbE IP Core Signals on page 87

3.3.1. LL 100GbE IP Core Signals

The signals of the LL 100GbE IP core are described in the following formats:

- The figure identifies the IP core interfaces.
- Tables list the signals and the interfaces to which they belong.
- Links guide you to descriptions for the individual signals, by interface. The links appear in Related Links.

Figure 34. Top-Level Signals of the LL 100GbE IP Core

In the figure, <*v*> is the number of transceiver PHY links (4 for CAUI-4 IP cores and 10 for standard 100GbE IP cores). <*N*> is the number of priority flow control queues. (<*N*> = 1 for IP cores configured with standard flow control or with no flow control, and <*N*> = the value of the relevant parameter for IP cores configured with priority-based flow control). In the 1588 PTP signals, <*W*> is the width of the fingerprint.

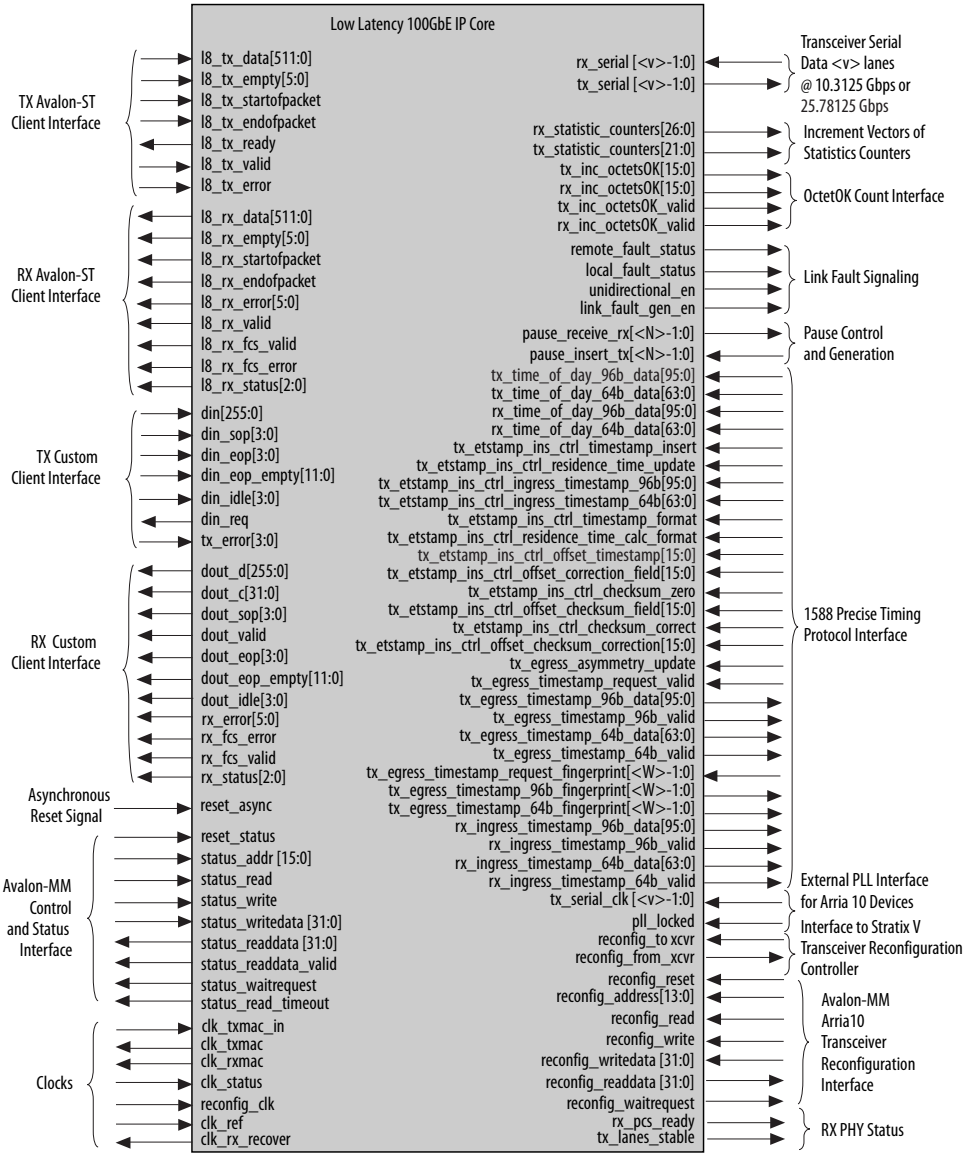


Table 30. LL 100GbE IP Core Signals

Signal Name	Direction	Interface
clk_ref	Input	Clocks
clk_rx_recover	Output	Clocks This signal is only available if you turn on **Enable SyncE** in the parameter editor.
reset_async	Input	Reset

continued...

Signal Name	Direction	Interface
tx_serial[9:0] tx_serial[3:0] (CAUI–4)	Output	Transceiver PHY serial data interface
rx_pcs_ready	Output	PHY status
tx_lanes_stable	Output	
clk_txmac_in	Input	Clocks Interface to external TX MAC PLL This signal is only available if you turn on **Enable external TX MAC PLL** in the parameter editor.
clk_txmac	Output	Clocks TX client interface
l8_tx_data[511:0]	Input	Avalon-ST TX client interface Each IP core instance has Avalon-ST TX and RX client interfaces, or custom streaming TX and RX client interfaces.
l8_tx_empty[5:0]	Input	
l8_tx_startofpacket	Input	
l8_tx_endofpacket	Input	
l8_tx_ready	Output	
l8_tx_valid	Input	
l8_tx_error	Input	
din[255:0]	Input	Custom streaming TX client interface Each IP core instance has Avalon-ST TX and RX client interfaces, or custom streaming TX and RX client interfaces.
din_sop[3:0]	Input	
din_eop[3:0]	Input	
din_eop_empty[11:0]	Input	
din_idle[3:0]	Input	
din_req	Output	
tx_error[3:0]	Input	
clk_rxmac	Output	Clocks RX client interface
l8_rx_data[511:0]	Output	Avalon-ST RX client interface Each IP core instance has Avalon-ST TX and RX client interfaces, or custom streaming TX and RX client interfaces.
l8_rx_empty[5:0]	Output	
l8_rx_startofpacket	Output	
l8_rx_endofpacket	Output	
l8_rx_error[5:0]	Output	
l8_rx_valid	Output	
l8_rx_fcs_valid	Output	
l8_rx_fcs_error	Output	
l8_rx_status[2:0]	Output	
dout_d[255:0]	Output	Custom streaming RX client interface Each IP core instance has Avalon-ST TX and RX client interfaces, or custom streaming TX and RX client interfaces.
dout_c[31:0]	Output	

continued...

Signal Name	Direction	Interface
dout_sop[3:0]	Output	
dout_eop[3:0]	Output	
dout_eop_empty[11:0]	Output	
dout_idle[3:0]	Output	
rx_error[5:0]	Output	
rx_fcs_error	Output	
rx_fcs_valid	Output	
rx_status[2:0]	Output	
dout_valid	Output	
pause_insert_tx[<N>-1:0]	Input	Pause control and generation interface
pause_receive_rx[<N>-1:0]	Output	
remote_fault_status	Output	Link fault signaling interface These signals are available only in IP core variations that include the link fault signaling module.
local_fault_status	Output	
unidirectional_en	Output	Link fault signaling interface, Clause 66 status These signals are available only in IP core variations that include the link fault signaling module.
link_fault_gen_en	Output	
tx_inc_64	Output	TX statistics counter increment vectors These signals are available whether or not your IP core includes TX statistics counters.
tx_inc_127	Output	
tx_inc_255	Output	
tx_inc_511	Output	
tx_inc_1023	Output	
tx_inc_1518	Output	
tx_inc_max	Output	
tx_inc_over	Output	
tx_inc_mcast_data_err	Output	
tx_inc_mcast_data_ok	Output	
tx_inc_bcast_data_err	Output	
tx_inc_bcast_data_ok	Output	
tx_inc_ucast_data_err	Output	
tx_inc_ucast_data_ok	Output	
tx_inc_mcast_ctrl	Output	
tx_inc_bcast_ctrl	Output	
tx_inc_ucast_ctrl	Output	
tx_inc_pause	Output	
tx_inc_fcs_err	Output	

continued...

Signal Name	Direction	Interface
tx_inc_fragment	Output	
tx_inc_jabber	Output	
tx_inc_sizeok_fcserr	Output	
rx_inc_runt	Output	RX statistics counter increment vectors These signals are available whether or not your IP core includes RX statistics counters.
rx_inc_64	Output	
rx_inc_127	Output	
rx_inc_255	Output	
rx_inc_511	Output	
rx_inc_1023	Output	
rx_inc_1518	Output	
rx_inc_max	Output	
rx_inc_over	Output	
rx_inc_mcast_data_err	Output	
rx_inc_mcast_data_ok	Output	
rx_inc_bcast_data_err	Output	
rx_inc_bcast_data_ok	Output	
rx_inc_ucast_data_err	Output	
rx_inc_ucast_data_ok	Output	
rx_inc_mcast_ctrl	Output	
rx_inc_bcast_ctrl	Output	
rx_inc_ucast_ctrl	Output	
rx_inc_pause	Output	
rx_inc_fcs_err	Output	
rx_inc_fragment	Output	
rx_inc_jabber	Output	
rx_inc_sizeok_fcserr	Output	
rx_inc_pause_ctrl_err	Output	
rx_inc_mcast_ctrl_err	Output	
rx_inc_bcast_ctrl_err	Output	
rx_inc_ucast_ctrl_err	Output	
tx_inc_octetsOK[15:0]	Output	OctetOK Count interface
tx_inc_octetsOK_valid	Output	
rx_inc_octetsOK[15:0]	Output	
rx_inc_octetsOK_valid	Output	

continued...

Signal Name	Direction	Interface
clk_status	Input	Clocks Control and status interface
reset_status	Input	Resets Control and status interface
status_addr[15:0]	Input	Control and status interface
status_read	Input	
status_write	Input	
status_writedata[31:0]	Input	
status_readdata[31:0]	Output	
status_readdata_valid	Output	
status_waitrequest	Output	
status_read_timeout	Output	
tx_time_of_day_96b_data[95:0]	Input	1588 PTP interface These signals are available only if 1588 PTP functionality is included in the IP core.
tx_time_of_day_64b_data[63:0]	Input	
rx_time_of_day_96b_data[95:0]	Input	
rx_time_of_day_64b_data[63:0]	Input	
tx_etstamp_ins_ctrl_timestamp_insert	Input	
tx_etstamp_ins_ctrl_residence_time_update	Input	
tx_etstamp_ins_ctrl_ingress_timestamp_96b[95:0]	Input	
tx_etstamp_ins_ctrl_ingress_timestamp_64b[63:0]	Input	
tx_etstamp_ins_ctrl_timestamp_format	Input	
tx_etstamp_ins_ctrl_residence_time_calc_format	Input	
tx_etstamp_ins_ctrl_offset_timestamp[15:0]	Input	
tx_etstamp_ins_ctrl_offset_correction_field[15:0]	Input	
tx_etstamp_ins_ctrl_checksum_zero	Input	
tx_etstamp_ins_ctrl_offset_checksum_field[15:0]	Input	
tx_etstamp_ins_ctrl_checksum_correct	Input	

continued...

Signal Name	Direction	Interface
tx_etstamp_ins_ctrl_offset_checksum_correction[15:0]	Input	
tx_egress_asymmetry_update	Input	
tx_egress_timestamp_request_valid	Input	
tx_egress_timestamp_96b_data[95:0]	Output	
tx_egress_timestamp_96b_valid	Output	
tx_egress_timestamp_64b_data[63:0]	Output	
tx_egress_timestamp_64b_valid	Output	
tx_egress_timestamp_request_fingerprint[(W–1):0]	Input	
tx_egress_timestamp_96b_fingerprint[(W–1):0]	Output	
tx_egress_timestamp_64b_fingerprint[(W–1):0]	Output	
rx_ingress_timestamp_96b_data[95:0]	Output	
rx_ingress_timestamp_96b_valid	Output	
rx_ingress_timestamp_64b_data[63:0]	Output	
rx_ingress_timestamp_64b_valid	Output	

reconfig_to_xcvr[1399:0]	Input	Interface to Stratix V reconfiguration controller These signals are available in Stratix V devices only.
reconfig_from_xcvr[919:0]	Output	
reconfig_busy	Input	
reconfig_clk	Input	Clocks Arria 10 Native PHY IP core reconfiguration interface This signal is available in Arria 10 devices only.
reconfig_reset	Input	Resets Arria 10 Native PHY IP core reconfiguration interface This signal is available in Arria 10 devices only.
reconfig_address [13:0] reconfig_address [11:0] (CAUI-4)	Input	Arria 10 Native PHY IP core reconfiguration interface These signals are available in Arria 10 devices only.
reconfig_read	Input	
reconfig_write	Input	

continued...

reconfig_writedata[31:0]	Input	
reconfig_readdata[31:0]	Output	
reconfig_waitrequest	Output	

pll_locked	Input	External transceiver PLL interface. These signals are available in Arria 10 devices only.
tx_serial_clk[9:0] tx_serial_clk[3:0] (CAUI-4)	Input	

Related Information

- Clocks on page 84
  Overview of IP core clocks. Includes list of clock signals and recommended and required frequencies.
- Resets on page 86
- Transceiver PHY Serial Data Interface on page 82
- PHY Status Interface on page 82
- LL 100GbE IP Core TX Data Bus with Adapters (Avalon-ST Interface) on page 42
  Describes the Avalon-ST TX client interface.
- LL 100GbE IP Core TX Data Bus Without Adapters (Custom Streaming Interface) on page 44
  Describes the custom streaming TX client interface.
- LL 100GbE IP Core RX Data Bus with Adapters (Avalon-ST Interface) on page 53
  Describes the Avalon-ST RX client interface.
- LL 100GbE IP Core RX Data Bus Without Adapters (Custom Streaming Interface) on page 56
  Describes the custom streaming RX client interface.
- Pause Control and Generation Interface on page 62
  Describes the pause signals available in the LL 100GbE IP core.
- Link Fault Signaling Interface on page 65
- Statistics Counters Interface on page 66
- Control and Status Interface on page 82
- 1588 Precision Time Protocol Interfaces on page 69
- External Reconfiguration Controller on page 58
- Arria 10 Transceiver Reconfiguration Interface on page 84
- External Transceiver PLL on page 59

3.4. Software Interface: Registers

This section provides information about the memory-mapped registers. You access these registers using the IP core 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 constant. Write operations to Reserved registers have no effect. Accesses to registers that do not exist in your IP core variation have an unspecified result.

Table 31. LL 100GbE IP Core Register Map Overview

Lists the main ranges of the memory mapped registers for the LL 100GbE IP core. Addresses are word addresses. Register address range 0x000–0x2FF is Reserved.

Word Offset	Register Category
0x300–0x3FF	PHY registers
0x400–0x4FF	TX MAC registers
0x500–0x5FF	RX MAC registers
0x600–0x6FF	TX flow control (pause functionality) registers
0x700–0x7FF	RX flow control (pause functionality) registers
0x800–0x8FF	TX statistics counters
0x900–0x9FF	RX statistics counters
0xA00–0xAFF	TX 1588 PTP registers
0xB00–0xBFF	RX 1588 PTP registers
0xC00–0xCFF	TX RS-FEC registers
0xD00–0xDFF	RX RS-FEC registers

Table 32. IP Core Address Map

Lists the memory mapped registers for the LL 100GbE IP core. Each register is 32 bits, and the addresses (word offsets) each address a full word. This table does not list the Revision ID, scratch, and name registers present in each of the address ranges at offsets 0x00, 0x01, and 0x02–0x04 respectively. However, those registers appear in the detailed listings with descriptions.

Word Offset	Register Description
0x310–0x344	PHY registers available in all IP core variations
0x405	Link fault signaling register `LINK_FAULT_CONFIG` Accessible only if you turn on **Enable link fault generation**
0x406	IPG column removal register `IPG_COL_REM` Accessible only if you turn on **Average interpacket gap**
0x407	TX maximum size Ethernet frame (in bytes) `MAX_TX_SIZE_CONFIG`. Value determines whether the IP core increments the `CNTR_TX_OVERSIZE` register
0x506	RX maximum size Ethernet frame (in bytes) `MAX_RX_SIZE_CONFIG`. Value determines whether the IP core increments the `CNTR_RX_OVERSIZE` register.
0x507	RX CRC forwarding configuration register `MAC_CRC_CONFIG`
0x508	Link fault status register Provides link fault status information if you turn on **Enable link fault generation**. Returns zeroes if you turn off **Enable link fault generation**.
0x50A	Enable RX payload length checking register Provides enable bit to determine whether the RX error signal flags payload lengths that do not match the length field..
0x605–0x60A	Transmit side pause registers Accessible only if you set **Flow control mode** to the value **Standard flow control** or **Priority-based flow control**.
0x700–0x703	Receive side pause registers Accessible only if you set **Flow control mode** to the value **Standard flow control** or **Priority-based flow control**.

continued...

Word Offset	Register Description
0x800–0x837, 0x860–0x861	Transmit side statistics registers Accessible only if you turn on **Enable TX statistics**
0x845	Transmit statistics counters configuration register CNTR_TX_CONFIG Accessible only if you turn on **Enable TX statistics**
0x846	Transmit statistics counters status register Accessible only if you turn on **Enable TX statistics**
0x900–0x937, 0x960–0x961	Receive side statistics registers Accessible only if you turn on **Enable RX statistics**
0x945	Receive statistics counters configuration register CNTR_RX_CONFIG Accessible only if you turn on **Enable RX statistics**
0x946	Receive statistics counters status register Accessible only if you turn on **Enable RX statistics**
0xA00–0xA0C	TX 1588 PTP registers Accessible only if you turn on **Enable 1588 PTP**
0xB00–0xB06	RX 1588 PTP registers Accessible only if you turn on **Enable 1588 PTP**
0xC00–0xC07	TX RS-FEC registers Accessible only if you turn on **Enable RS-FEC for CAUI4**
0xD00–0xD08	RX RS-FEC registers Accessible only if you turn on **Enable RS-FEC for CAUI4**

Table 33. LL 100GbE Hardware Design Example Registers

Lists the memory mapped register ranges for the LL 100GbE IP core hardware design example

Word Offset	Register Category
0x1000–0x1016	Packet client registers
0x2004–0x2023	Reserved
0x4000–0x4C00	Arria 10 dynamic reconfiguration register base addresses for four-lane variations. Register base address is 0x4000 for Lane 0, 0x4400 for Lane 1, 0x4800 for Lane 2, and 0x4C00 for Lane 3. (Bits [11:10] specify the lane).
0x4000–0x6400	Arria 10 dynamic reconfiguration register base addresses for ten-lane variations. Register base address is 0x4000 for Lane 0, 0x4400 for Lane 1, 0x4800 for Lane 2, and 0x4C00 for Lane 3, 0x5000 for Lane 4, ... 0x6400 for Lane 9. (Bits [13:10] specify the lane).

Related Information

Control and Status Interface on page 82

3.4.1. LL 100GbE IP Core Registers

The following sections describe the registers included in the LL 100GbE IP core.

RX Statistics Registers on page 110

1588 PTP Registers on page 114

TX Reed-Solomon FEC Registers on page 116

RX Reed-Solomon FEC Registers on page 117

Related Information

Control and Status Interface on page 82

3.4.1.1. PHY Registers

Table 34. PHY Registers

Addr	Name	Bit	Description	HW Reset Value	Access
0x300	PHY_REVID	[31:0]	IP core PHY module revision ID.	0x02062015	RO
0x301	PHY_SCRATCH	[31:0]	Scratch register available for testing.	32'b0	RW
0x302	PHY_NAME_0	[31:0]	First 4 characters of IP core variation identifier string "100GE pcs ".		RO
0x303	PHY_NAME_1	[31:0]	Next 4 characters of IP core variation identifier string "100GE pcs ".		RO
0x304	PHY_NAME_2	[31:0]	Final 4 characters of IP core variation identifier string "100GE pcs ".		RO
0x310	PHY_CONFIG	[5]	set_data_lock: Directs the PLL to lock to data.	6'b0	RW
		[4]	set_ref_lock: Directs the PLL to lock to the reference clock.		
		[3]	rxp_ignore_freq: Directs the IP core to proceed with the internal reset sequence (to reset the RX PLL) without waiting for the RX CDR PLL to lock.		
		[2]	soft_rxp_rst: RX PLL soft reset		
		[1]	soft_txp_rst: TX PLL soft reset		
		[0]	eio_sys_rst: PMA system reset. Set this bit to start the internal reset sequence.		
0x313	PHY_PMA_SLOOP	[3:0] or [9:0]	Serial PMA loopback. Each bit that is asserted directs the IP core to connect the corresponding TX–RX lane pair on the internal loopback path, by setting the corresponding transceiver in serial loopback mode. This option is not available for CAUI-4 variations.	0	RW
0x314	PHY_PCS_INDIRECT_ADDR	[2:0]	Supports indirect addressing of individual FIFO flags in the 10G 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 PHY_PCS_INDIRECT_DATA register. The value in the PHY_PCS_INDIRECT_ADDR register directs the IP core to make available the following FIFO flag: • 3'b111: RX partial empty • 3'b110: RX partial full • 3'b101: RX empty	3'b0	RW

continued...

Addr	Name	Bit	Description	HW Reset Value	Access
			• 3'b100: RX full • 3'b011: TX partial empty • 3'b010: TX partial full • 3'b001: TX empty • 3'b000: TX full		
0x315	PHY_PCS_INDIRECT_DATA	[3:0] or [9:0]	PCS indirect data. To read a FIFO flag, set the value in the PHY_PCS_INDIRECT_ADDR register to indicate the flag you want to read. After you set the specific flag indication in the PHY_PCS_INDIRECT_ADDR register, each bit [n] in the PHY_PCS_INDIRECT_DATA register has the value of that FIFO flag for the transceiver channel for lane [n]. This register has ten valid bits [9:0] in standard LL 100GbE IP core variations. This register is not valid for CAUI-4 variations.	TX full flags	RO
0x320	PHY_TX_PLL_LOCKED	[9:0]	Each bit that is asserted indicates that the corresponding lane TX transceiver PLL is locked.	10'b0	RO
0x321	PHY_EIOFREQ_LOCKED	[9:0]	Each bit that is asserted indicates that the corresponding lane RX CDR PLL is locked.	10'b0	RO RO
0x322	PHY_TX_CORE PLL_LOCKED	[2]	RX PLL is locked.	3'b0	RO
		[1]	TX MAC PLL is locked.		
		[0]	TX PCS is ready.		
0x323	PHY_FRAME_ERROR	[3:0] or [19:0]	Each bit that is asserted indicates that the corresponding virtual lane has a frame error. If you read a non-zero value in this register, the virtual lanes that correspond to non-zero bits have observed frame errors. You can clear these bits with the PHY_SCLR_FRAME_ERROR register. Intel recommends that you set bit [0] of the PHY_SCLR_FRAME_ERROR register and read the PHY_FRAME_ERROR register again to determine if the PHY frame error is generated continuously. These bits are not sticky: the IP core clears them more than once while attempting to achieve alignment.	0xF or 0xFFFFF	RO
0x324	PHY_SCLR_FRAME_ERROR	[0]	Synchronous clear for PHY_FRAME_ERROR register. Write the value of 1 to this register to clear the PHY_FRAME_ERROR register. This bit is not self-clearing. You should reset this register to the value of 0 within ten clk_status clock cycles. If you do not reset the PHY_SCLR_FRAME_ERROR register, the value in the PHY_FRAME_ERROR register is not useful.	1'b0	RW
0x325	PHY_EIO_SFT RESET	[1]	Set this bit to clear the RX FIFO. This bit is not self-clearing.	2'b00	RW
		[0]	RX PCS reset: set this bit to reset the RX PCS. This bit is not self-clearing.		
0x326	PHY_RXPCS_STATUS	[1:0]	Indicates the RX PCS is fully aligned and ready to accept traffic.	0	RO

continued...

Addr	Name	Bit	Description	HW Reset Value	Access
			• Bit [1]: HI BER status (bit error rate is high according to Ethernet standard definition). This bit maintains the value of 1'b0 unless you turn on **Enable link fault generation**. • Bit [0]: RX PCS fully aligned status.		
0x340	PHY_REFCLK_KHZ	[31:0]	Reference clock frequency in KHz, assuming the `clk_status` clock has the frequency of 100 MHz. The reference clock frequency is the value in the PHY_REFCLK_KHZ register times the frequency of the `clk_status` clock, divided by 100.		RO
0x341	PHY_RXCLK_KHZ	[31:0]	RX clock (`clk_rxmac`) frequency in KHz, assuming the `clk_status` clock has the frequency of 100 MHz. The RX clock frequency is the value in the PHY_RXCLK_KHZ register times the frequency of the `clk_status` clock, divided by 100.		RO
0x342	PHY_TXCLK_KHZ	[31:0]	TX clock (`clk_txmac`) frequency in KHz, assuming the `clk_status` clock has the frequency of 100 MHz. The TX clock frequency is the value in the PHY_TXCLK_KHZ register times the frequency of the `clk_status` clock, divided by 100.		RO
0x343	PHY_RECCLK_KHZ	[31:0]	RX recovered clock frequency in KHz, assuming the `clk_status` clock has the frequency of 100 MHz. The RX recovered clock frequency is the value in the PHY_RECCLK_KHZ register times the frequency of the `clk_status` clock, divided by 100.		RO
0x344	PHY_TXIOCLK_KHZ	[31:0]	TX PMA clock frequency in KHz, assuming the `clk_status` clock has the frequency of 100 MHz. The TX PMA clock frequency is the value in the PHY_TX10CLK_KHZ register times the frequency of the `clk_status` clock, divided by 100.		RO

3.4.1.2. Link Fault Signaling Registers

Table 35. LINK_FAULT_CONFIG Register—Offset 0x405

Name	Bit	Description	Reset Value	Access
Unidir Enable	[1]	When asserted, the IP core includes Clause 66 support for remote link fault reporting on the Ethernet link.	1'b0	RW
Link Fault Reporting Enable	[0]	• 1'b1: The PCS generates a proper fault sequence on the Ethernet link, if conditions are met. • 1'b0: The PCS does not generate any fault sequences. Whether this bit has the value of 0 or 1, the RX PCS always reports real-time link status on the `local_fault_status` and `remote_fault_status` output signals and in the `Local Fault Status` register at offset 0x508.	1’b1	RW

Table 36. Link Fault Status Register—Offset 0x508

Name	Bit	Description	HW Reset Value	Access
Remote Fault Status	[1]	The remote fault status register.The IP core sets this bit when it detects real-time remote faults in the RX MAC. The IP core sets this bit regardless of the settings in the LINK_FAULT_CONFIG register.	1'b0	RO
Local Fault Status	[0]	The local fault status register. The IP core sets this bit when it detects real-time local faults in the RX MAC. The IP core sets this bit regardless of the settings in the LINK_FAULT_CONFIG register.	1'b1	RO

Related Information

Link Fault Signaling Interface on page 65
Describes how the IP core uses the register values.

3.4.1.3. LL 100GbE IP Core MAC Configuration Registers

The MAC configuration registers control the following MAC features in the RX and TX datapaths:

- Fault link signaling on the Ethernet link (TX)
- Local and remote fault status signals (RX)
- CRC forwarding (RX)
- Inter-packet gap IDLE removal (TX)
- Maximum frame sizes for the CNTR_RX_OVERSIZE and CNTR_TX_OVERSIZE counters (RX and TX)

The fault link signaling and fault status signal registers are documented separately. Refer to Related Links below.

Table 37. TX MAC Configuration Registers

This table documents the non-fault link signaling registers in the TX MAC address space. The LINK_FAULT_CONFIG register at address 0x405 is documented with the link fault signaling registers. The LINK_FAULT_CONFIG register is available only if you turn on **Enable link fault generation** in the LL 100GbE parameter editor.

Address	Name	Bit	Description	HW Reset Value	Access
0x400	TXMAC_REVID	[31:0]	TX MAC revision ID.	0x02062015	RO
0x401	TXMAC_SCRATCH	[31:0]	Scratch register available for testing.	32'b0	RW
0x402	TXMAC_NAME_0	[31:0]	First 4 characters of IP core variation identifier string "100gMACTxCSR".		RO
0x403	TXMAC_NAME_1	[31:0]	Next 4 characters of IP core variation identifier string "100gMACTxCSR".		RO

continued...

Address	Name	Bit	Description	HW Reset Value	Access
0x404	TXMAC_NAME_2	[31:0]	Final 4 characters of IP core variation identifier string "100gMACTxCSR".		RO
0x406	IPG_COL_REM	[7:0]	Specifies the number of IDLE columns to be removed in every Alignment Marker period to compensate for alignment marker insertion. You can program this register to a larger value than the default value, for clock compensation. This register is not present if you set the value of the **Average interpacket gap** parameter to **Disable deficit idle counter** in the LL 100GbE parameter editor.	20 (decimal)	RW
0x407	MAX_TX_SIZE_CONFIG	[15:0]	Maximum size of Ethernet frames for CNTR_TX_OVERSIZE. If the IP core transmits an Ethernet frame of size greater than the number of bytes specified in this register, and the IP core includes TX statistics registers, the IP core increments the 64-bit CNTR_TX_OVERSIZE register.	9600 (decimal)	RW

Table 38. RX MAC Configuration Registers

This table documents the non-fault link signaling registers in the RX MAC address space. The local and remote fault status register at address 0x508 is documented with the link fault signaling registers. The local and remote fault status register is available only if you turn on **Enable link fault generation** in the LL 100GbE parameter editor.

Address	Name	Bit	Description	HW Reset Value	Access
0x500	RXMAC_REVID	[31:0]	RX MAC revision ID.	0x02062015	RO
0x501	RXMAC_SCRATCH	[31:0]	Scratch register available for testing.	32'b0	RW
0x502	RXMAC_NAME_0	[31:0]	First 4 characters of IP core variation identifier string "100gMACRxCSR".		RO
0x503	RXMAC_NAME_1	[31:0]	Next 4 characters of IP core variation identifier string "100gMACRxCSR".		RO
0x504	RXMAC_NAME_2	[31:0]	Final 4 characters of IP core variation identifier string "100gMACRxCSR".		RO
0x506	MAX_RX_SIZE_CONFIG	[15:0]	Maximum size of Ethernet frames for CNTR_RX_OVERSIZE and for rx_error[3] or l<n>_rx_error[3] and for .the RxOctetsOK register. 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 RX statistics registers, the IP core increments the 64-bit CNTR_RX_OVERSIZE register.	9600 (decimal)	RW

continued...

Address	Name	Bit	Description	HW Reset Value	Access
			An Ethernet frame of size greater than the number of bytes specified in this register is considered oversized and therefore does not contribute to the value in the RxOctetsOK register.and does cause the assertion of rx_error[3] or l<n>_rx_error[3].		
0x507	MAC_CRC_CONFIG	[0]	The RX CRC forwarding configuration register. Possible values are: • 1'b0: remove RX CRC— do not forward it to the RX client interface. • 1'b1: retain RX CRC—forward it to the RX client interface. In either case, the IP core checks the incoming RX CRC and flags errors.	1'b0	RW
0x50A	CFG_PLEN_CHECK	[0]	Enables payload length checking. If you set this bit to the value of 1, bit[4] of the rx_error signal flags any payload lengths that do not match the length field.	1'b1	RW

Related Information

Link Fault Signaling Registers on page 99
Describes the fault link signaling and fault status signal registers.

3.4.1.4. Pause Registers

The pause registers together with the pause signals implement the pause functionality defined in the *IEEE 802.3ba-2010 High Speed Ethernet Standard*. You can program the pause registers to control the insertion and decoding of pause frames, to help reduce traffic in congested networks.

Table 39. TX Ethernet Flow Control (Pause Functionality) Registers

Some registers are different depending on whether you select **Standard flow control** or **Priority-based flow control** in the LL 100GbE parameter editor. Where the difference is only whether the register refers to the single standard flow control priority class or whether distinct bits in the register refer to the individual priority queues, the two uses are documented together. In that case we understand that standard flow control effectively supports a single priority queue.

When your IP core implements priority-based flow control, the following registers provide an access window into an internal table of values, one per priority queue. The entry currently accessible in the registers is determined by the value you write to the TX_PAUSE_QNUMBER register.

- RETRANSMIT_XOFF_HOLDOFF_QUANTA at offset 0x608
- TX_PAUSE_QUANTA at offset 0x609

Addr	Name	Bit	Description	HW Reset Value	Access
0x600	TXSFC_REVID	[31:0]	TX standard flow control module revision ID.	0x0128_2014	RO
0x601	TXSFC_SCRATCH	[31:0]	Scratch register available for testing.	32'b0	RW
0x602	TXSFC_NAME_0	[31:0]	First 4 characters of IP core variation identifier string.	0x7843_5352	RO
0x603	TXSFC_NAME_1	[31:0]	Next 4 characters of IP core variation identifier string.	0x5054_5054	RO

continued...

Addr	Name	Bit	Description	HW Reset Value	Access
0x604	TXSFC_NAME_2	[31:0]	Final 4 characters of IP core variation identifier string.	0x3034_3067	RO
0x605	TX_PAUSE_EN	[N-1:0] (7)	Enable the IP core to transmit pause frames on the Ethernet link in response to a client request through the pause_insert_tx input signal or the TX_PAUSE_REQUEST register. If your IP core implements priority-based flow control, each bit of this field enables TX pause functionality for the corresponding priority queue. Intel recommends that you signal a pause request using the pause_insert_tx signal rather than using the TX_PAUSE_REQUEST register.	N'b1 ...1 (1'b1 in each defined bit)	RW
0x606	TX_PAUSE_REQUEST	[N-1:0](7)	Pause request. If bit [n] of the TX_PAUSE_EN register has the value of 1, setting bit [n] of the TX_PAUSE_REQUEST register field to the value of 1 triggers a XOFF pause packet insertion into the TX data stream on the Ethernet link. If the IP core implements priority-based flow control, the XOFF pause packet includes identity information for the corresponding priority queue. If RETRANSMIT_XOFF_HOLDOFF_EN is turned on for the associated priority queue, as long as the value in TX_PAUSE_REQUEST bit [n] remains high, the IP core retransmits the XOFF pause packet at intervals determined by the retransmit hold-off value associated with this priority queue. If bit [n] of the TX_PAUSE_EN register has the value of 1, resetting bit [n] of the TX_PAUSE_REQUEST register field to the value of 0 triggers an XON pause packet insertion into the TX data stream on the Ethernet link. If the IP core implements priority-based flow control, the XON pause packet includes identity information for the corresponding priority queue. Other pause registers, described in this table, specify the properties of the pause packets. Intel recommends that you signal a pause request using the pause_insert_tx signal rather than using the TX_PAUSE_REQUEST register.	0	RW
0x607	RETRANSMIT_XOFF_HOLDOFF_EN	[N-1:0] (7)	Enable XOFF pause frame retransmission hold-off functionality. If your IP core implements priority-based flow control with multiple priority queues, this register provides access to one bit for each priority queue. Intel recommends that you maintain this register at the value of all ones. If your IP core implements priority-based flow control, refer also to the description of the CFG_RETRANSMIT_HOLDOFF_EN and CFG_RETRANSMIT_HOLDOFF_QUANTA registers.	N'b1...1 (1'b1 in each defined bit)	RW
0x608	RETRANSMIT_XOFF_HOLDOFF_QUANTA	[15:0]	Specifies hold-off time from XOFF pause frame transmission until retransmission, if retransmission hold-off functionality is enabled and pause request remains at the value of 1. If your IP core implements priority-based flow control with	0xFFFF	RW

continued...

(7) N is the number of priority queues. If the IP core implements Ethernet standard flow control, N is 1.

Addr	Name	Bit	Description	HW Reset Value	Access
			multiple priority queues, this register provides access to an internal table of retransmit-hold-off times, one for each priority queue. Accesses are indexed by the value in the TX_PAUSE_QNUMBER register. Unit is quanta. One quanta is 512 bit times, which depends on the datapath width (512 bits) and the `clk_txmac` frequency. Note that in the case of 100GbE IP cores, one quanta is effectively one `clk_txmac` clock cycle. Pause request can be either of the TX_PAUSE_REQUEST register pause request bit and the `pause_insert_tx` signal.		
0x609	TX_PAUSE_QUANTA	[15:0]	Specifies the pause time to be included in XOFF frames. If your IP core implements priority-based flow control with multiple priority queues, this register provides access to an internal table of pause times, one for each priority queue. Accesses are indexed by the value in the TX_PAUSE_QNUMBER register. Unit is quanta. One quanta is 512 bit times, which depends on the datapath width (512 bits) and the `clk_txmac` frequency. In 100GbE IP cores, a quanta is effectively a single `clk_txmac` clock cycle.	0xFFFF	RW
0x60A if you set the value of **Flow control mode** to **Standard flow control** in the LL 100GbEparameter editor.	TX_XOF_EN	[0]	Enable the TX MAC to pause outgoing Ethernet traffic in response to a pause frame received on the RX Ethernet link and forwarded to the TX MAC by the RX MAC. If the `cfg_enable` bit of the RX_PAUSE_ENABLE register has the value of 1, the RX MAC processes incoming pause frames. When the RX MAC processes an incoming pause frame with an address match, it notifies the TX MAC to pause outgoing traffic on the TX Ethernet link. The TX MAC pauses outgoing traffic on the TX Ethernet link in response to this notification only if bit [0] of the TX_XOF_EN register has the value of 1. The value in the TX_XOF_EN register is only relevant when the `cfg_enable` bit of the RX_PAUSE_ENABLE register has the value of 1. If the `cfg_enable` bit of the RX_PAUSE_ENABLE register has the value of 0, pause frames received on the RX Ethernet link do not reach the TX MAC.	1'b0	RW
0x60A if you set the value of **Flow control mode** to **Priority-based flow control** in the LL 100GbEparameter editor.	TX_PAUSE_QNUMBER	[2:0]	Queue number (index to internal table) of queue whose relevant values are currently accessible (readable and writeable) in these registers: • RETRANSMIT_XOFF_HOLDOFF_QUANTA at offset 0x608 • TX_PAUSE_QUANTA at offset 0x609	0	RW
0x60B	`CFG_RETRANSMIT_HOLDOFF_EN`	[0]	The `CFG_RETRANSMIT_HOLDOFF_EN` and `CFG_RETRANSMIT_HOLDOFF_QUANTA` registers provide a mechanism to specify a uniform retransmission hold-off delay for all priority queues. If `CFG_RETRANSMIT_HOLDOFF_EN` has the value of 1, the IP core enforces a	1'b0	RW

continued...

Addr	Name	Bit	Description	HW Reset Value	Access
0x60C	CFG_RETRANSMIT_HOLDOFF_QUANTA	[15:0]	retransmission hold-off delay for each priority queue that is the longer of the queue-specific retransmission hold-off delay accessible in the RETRANSMIT_XOFF_HOLDOFF_QUANTA register (if enabled) and the retransmission hold-off delay specified in the CFG_RETRANSMIT_HOLDOFF_QUANTA register. CFG_RETRANSMIT_HOLDOFF_QUANTA unit is quanta. One quanta is 512 bit times, which depends on the datapath width (512 bits) and the clk_txmac frequency. In 100GbE IP cores, a quanta is effectively a single clk_txmac clock cycle. These registers are present only if you set the value of **Flow control mode** to **Priority-based flow control** in the LL 100GbE parameter editor.		
0x60D	TX_PFC_DADDRL	[31:0]	TX_PFC_DADDRH contains the 16 most significant bits of the destination address for PFC pause frames. TX_PFC_DADDRL contains the 32 least significant bits of the destination address for PFC pause frames. This feature allows you to program a destination address other than the standard multicast address for PFC frames, for debug or proprietary purposes. {TX_PFC_DADDRH[15:0],TX_PFC_DADDRL[31:0]} can be a broadcast, multicast, or unicast address. These registers are present only if you set the value of **Flow control mode** to **Priority-based flow control** in the LL 100GbE parameter editor.	0xC200_0001	RW
0x60E	TX_PFC_DADDRH	[15:0]		0x0180	RW
0x60F	TX_PFC_SADDRL	[31:0]	TX_PFC_SADDRH contains the 16 most significant bits of the source address for PFC pause frames. TX_PFC_SADDRL contains the 32 least significant bits of the source address for PFC pause frames. These registers are present only if you set the value of **Flow control mode** to **Priority-based flow control** in the LL 100GbE parameter editor.	0xCBFC_5ADD	RW
0x610	TX_PFC_SADDRH	[15:0]		0xE100	RW

Table 40. RX Ethernet Flow Control (Pause Functionality) Registers

Some registers are different depending on whether you select **Standard flow control** or **Priority-based flow control** in the LL 100GbE parameter editor. Where the difference is only whether the register refers to the single standard flow control priority class or whether distinct bits in the register refer to the individual priority queues, the two uses are documented together. In that case we understand that standard flow control effectively supports a single priority queue.

Addr	Name	Bit	Description	HW Reset Value	Access
0x700	RXSFC_REVID	[31:0]	RX standard flow control module revision ID.	0x0128_2014	RO
0x701	RXSFC_SCRATCH	[31:0]	Scratch register available for testing.	32'b0	RW
0x702	RXSFC_NAME_0	[31:0]	First 4 characters of IP core variation identifier string.	0x7843_5352	RO
0x703	RXSFC_NAME_1	[31:0]	Next 4 characters of IP core variation identifier string.	0x5054_5054	RO

continued...

Addr	Name	Bit	Description	HW Reset Value	Access
0x704	RXSFC_NAME_2	[31:0]	Final 4 characters of IP core variation identifier string.	0x3034_3067	RO
0x705	RX_PAUSE_ENABLE	[N-1:0]	cfg_enable bits. When bit [n] has the value of 1, the RX MAC processes the incoming pause frames for priority class n whose address matches {DADDR1[31:0],DADDR0[15:0]}. When bit [n] has the value of 0, the RX MAC does not process any incoming pause frames for priority class n. When the RX MAC processes an incoming pause frame with an address match, it notifies the TX MAC to pause outgoing traffic on the TX Ethernet link. If you implement priority-based flow control, the TX MAC pauses outgoing traffic from the indicated priority queue to the TX Ethernet link in response to this notification. If you implement standard flow control, the TX MAC pauses outgoing traffic on the TX Ethernet link in response to this notification only if bit [0] of the TX_XOF_EN register has the value of 1.	N'b1...1 (1'b1 in each defined bit)	RW
0x706	RX_PAUSE_FWD	[0]	cfg_fwd_ctrl bit. When this bit has the value of 1, the RX MAC forwards matching pause frames to the RX client interface. When this bit has the value of 0, the RX MAC does not forward matching pause frames to the RX client interface. In both cases, the RX MAC forwards non-matching pause frames to the RX client interface.	1'b0	RW
0x707	RX_PAUSE_DADDRL	[31:0]	RX_PAUSE_DADDRL contains the 32 least significant bits of the destination address for pause frame matching. RX_PAUSE_DADDRH contains the 16 most significant bits of the destination address for pause frame matching. When pause frame processing is turned on, if {RX_PAUSE_DADDRH[15:0],RX_PAUSE_DADDRL[31:0]} matches the incoming pause frame destination address, the IP core processes the pause frame. if there is no match, the IP core does not process the pause frame. {RX_PAUSE_DADDRH[15:0],RX_PAUSE_DADDRL[31:0]} can be a broadcast, multicast, or unicast address.	0xC200_0001	RW
0x708	RX_PAUSE_DADDRH	[15:0]		0x0180	RW

Related Information

Pause Control and Generation Interface on page 62
Describes the pause signals available in the LL 100GbE IP core.

3.4.1.5. TX Statistics Registers

The TX statistics registers count TX 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 TX statistics counters module is a synthesis option. The statistics registers are counters that are implemented inside the CSR. When you turn on the **Enable TX statistics** parameter in the LL 100GbE parameter editor, the counters are implemented in the CSR. When you turn off the **Enable TX statistics** parameter in the LL 100GbE parameter editor, the counters are not implemented in the CSR, and read access to the counters returns read data equal to 0.

Reading the value of a statistics register does not affect its value. A configuration register at offset 0x845 allows you to clear all of the TX statistics counters.

To ensure that the counters you read are consistent, you should issue a shadow request to create a snapshot of all of the TX statistics registers, by setting bit [2] of the configuration register at offset 0x845. Until you reset this bit, the counters continue to increment but the readable values remain constant.

Table 41. Transmit Side Statistics Registers

Address	Name-	Description	Access
0x800	CNTR_TX_FRAGMENTS_LO	Number of transmitted frames less than 64 bytes and reporting a CRC error (lower 32 bits)	RO
0x801	CNTR_TX_FRAGMENTS_HI	Number of transmitted frames less than 64 bytes and reporting a CRC error (upper 32 bits)	RO
0x802	CNTR_TX_JABBERS_LO	Number of transmitted oversized frames reporting a CRC error (lower 32 bits)	RO
0x803	CNTR_TX_JABBERS_HI	Number of transmitted oversized frames reporting a CRC error (upper 32 bits)	RO
0x804	CNTR_TX_FCS_LO	Number of transmitted packets with FCS errors. (lower 32 bits)	RO
0x805	CNTR_TX_FCS_HI	Number of transmitted packets with FCS errors. (upper 32 bits)	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
0x80D	CNTR_TX_UCAST_DATA_ERR_HI	Number of errored unicast frames transmitted, excluding control frames (upper 32 bits)	RO
0x80E	CNTR_TX_MCAST_CTRL_ERR_LO	Number of errored multicast control frames transmitted (lower 32 bits)	RO
0x80F	CNTR_TX_MCAST_CTRL_ERR_HI	Number of errored multicast control frames transmitted (upper 32 bits)	RO

continued...

Address	Name-	Description	Access
0x810	CNTR_TX_BCAST_CTRL_ERR_LO	Number of errored broadcast control frames transmitted (lower 32 bits)	RO
0x811	CNTR_TX_BCAST_CTRL_ERR_HI	Number of errored broadcast control frames transmitted (upper 32 bits)	RO
0x812	CNTR_TX_UCAST_CTRL_ERR_LO	Number of errored unicast control frames transmitted (lower 32 bits)	RO
0x813	CNTR_TX_UCAST_CTRL_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
0x823	CNTR_TX_1519toMAXB_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

continued...

Address	Name-	Description	Access
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_CTRL_LO	Number of valid multicast frames transmitted, excluding data frames (lower 32 bits)	RO
0x82D	CNTR_TX_MCAST_CTRL_HI	Number of valid multicast frames transmitted, excluding data frames (upper 32 bits)	RO
0x82E	CNTR_TX_BCAST_CTRL_LO	Number of valid broadcast frames transmitted, excluding data frames (lower 32 bits)	RO
0x82F	CNTR_TX_BCAST_CTRL_HI	Number of valid broadcast frames transmitted, excluding data frames (upper 32 bits)	RO
0x830	CNTR_TX_UCAST_CTRL_LO	Number of valid unicast frames transmitted, excluding data frames (lower 32 bits)	RO
0x831	CNTR_TX_UCAST_CTRL_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 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 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	CNTR_TX_ST_LO	Number of transmitted frame starts (lower 32 bits)	RO
0x837	CNTR_TX_ST_HI	Number of transmitted frame starts (upper 32 bits)	RO
0x838–0x83F	Reserved		
0x840	TXSTAT_REVID	TX statistics module revision ID.	RO
0x841	TXSTAT_SCRATCH	Scratch register available for testing. Default value is 0x08.	RW
0x842	TXSTAT_NAME_0	First 4 characters of IP core variation identifier string "100gMacStats"	RO

continued...

Address	Name-	Description	Access
0x843	TXSTAT_NAME_1	Next 4 characters of IP core variation identifier string "100gMacStats"	RO
0x844	TXSTAT_NAME_2	Final 4 characters of IP core variation identifier string "100gMacStats"	RO
0x845	CNTR_TX_CONFIG	Bits [2:0]: Configuration of TX statistics counters: • 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. • 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. • 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. Bits [31:3] are Reserved.	RW
0x846	CNTR_TX_STATUS	• Bit [1]: Indicates that the TX statistics registers are paused (while CNTR_TX_CONFIG[2] is asserted) . • Bit [0]: Indicates the presence of at least one parity error in the TX statistics counters. Bits [31:2] are Reserved.	RO
0x860	TxOctetsOK_LO	Number of transmitted payload bytes in frames with no FCS, undersized, oversized, or payload length errors. This register is compliant with section 5.2.2.18 of the *IEEE Standard 802.3-2008*. This register corresponds to the signals tx_inc_octetsOK[15:0] and tx_inc_octetsOK_valid.	RO
0x861	TxOctetsOK_HI		RO

Related Information

Statistics Counters Interface on page 66

3.4.1.6. RX Statistics Registers

The RX statistics registers count RX 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 RX statistics counters module is a synthesis option. The statistics registers are counters that are implemented inside the CSR. When you turn on the **Enable RX statistics** parameter in the LL 100GbE parameter editor, the counters are implemented in the CSR. When you turn off the **Enable RX statistics** parameter in the LL 100GbE parameter editor, the counters are not implemented in the CSR, and read access to the counters returns read data equal to 0.

Reading the value of a statistics register does not affect its value. A configuration register at offset 0x945 allows you to clear all of the RX statistics counters.

To ensure that the counters you read are consistent, you should issue a shadow request to create a snapshot of all of the RX statistics registers, by setting bit [2] of the configuration register at offset 0x945. Until you reset this bit, the counters continue to increment but the readable values remain constant.

Table 42. 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
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

continued...

Address	Name-	Description	Access
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
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

continued...

Address	Name-	Description	Access
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
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 valid pause frames received (lower 32 bits)	RO
0x933	CNTR_RX_PAUSE_HI	Number of valid pause frames received (upper 32 bits)	RO
0x934	CNTR_RX_RUNT_LO	Number of received runt packets (lower 32 bits) A run 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) A run 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	CNTR_RX_ST_LO	Number of received frame starts (lower 32 bits)	RO
0x937	CNTR_RX_ST_HI	Number of received frame starts (upper 32 bits)	RO
0x938–0x93F	Reserved		
0x940	RXSTAT_REVID	RX statistics module revision ID.	RO
0x941	RXSTAT_SCRATCH	Scratch register available for testing. Default value is 0x09.	RW
0x942	RXSTAT_NAME_0	First 4 characters of IP core variation identifier string "100gMacStats"	RO
0x943	RXSTAT_NAME_1	Next 4 characters of IP core variation identifier string "100gMacStats"	RO
0x944	RXSTAT_NAME_2	Final 4 characters of IP core variation identifier string "100gMacStats"	RO

continued...

Address	Name-	Description	Access
0x945	CNTR_RX_CONFIG	Bits [2:0]: Configuration of RX statistics counters: • 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. • 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. • 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. Bits [31:3] are Reserved.	RW
0x946	CNTR_RX_STATUS	• Bit [1]: Indicates that the RX statistics registers are paused (while CNTR_RX_CONFIG[2] is asserted) . • Bit [0]: Indicates the presence of at least one parity error in the RX statistics counters. Bits [31:2] are Reserved.	RO
0x960	RxOctetsOK_LO	Number of received payload bytes in frames with no FCS, undersized, oversized, or payload length errors. This register is compliant with section 5.2.1.14 of the *IEEE Standard 802.3-2008*. This register corresponds to the signals rx_inc_octetsOK[15:0] and rx_inc_octetsOK_valid.	RO
0x961	RxOctetsOK_HI		RO

Related Information

Statistics Counters Interface on page 66

3.4.1.7. 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 43. TX 1588 PTP Registers

Addr	Name	Bit	Description	HW Reset Value	Access
0xA00	TXPTP_REVID	[31:0]	IP core revision ID.	0x0916_2016	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.	0x7843_5352	RO
0xA03	TXPTP_NAME_1	[31:0]	Next 4 characters of IP core variation identifier string.	0x5054_5054	RO
0xA04	TXPTP_NAME_2	[31:0]	Final 4 characters of IP core variation identifier string.	0x3034_3067	RO
0xA05	TX_PTP_CLK_PERIOD	[19:0]	clk_txmac clock period. Bits [19:16]: nanoseconds Bits [15:0]: fraction of nanosecond	This value is set to the correct clock period for the	RW

continued...

Addr	Name	Bit	Description	HW Reset Value	Access
				required TX MAC clock frequency.	
0xA06–0xA09	Reserved		Reserved	96'b0	RO
0xA0A	TX_PTP_EXTRA_LATENCY	[31:0]	User-defined extra latency the IP core adds to outgoing timestamps. Bits [31:16]: Full nanoseconds Bits [15:0]: fraction of nanosecond	32'b0	RW
0xA0B	TX_PTP_ASYM_DELAY	[18:0]	Asymmetry adjustment as required for delay measurement. The IP core adds this value to the final delay. • Bit[18]: The value of 1 enables the feature and the value of 0 disables the feature. • 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. • Bits[16:0]: Asymmetry adjustment in nanoseconds.	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 pins. • Bits [31:16]: Full nanoseconds • Bits [15:0]: Fraction of a nanosecond In Arria 10 devices, the TX PMA latency is 187 UI. One UI is approximately 38.8 ps. Therefore, Intel recommends that you set this register to the value of 0x0007_428F. This is a device-dependent value that is sufficiently accurate in most cases. Intel recommends that you modify this value with extreme caution.	32'b0	RW

Table 44. RX 1588 PTP Registers

Addr	Name	Bit	Description	HW Reset Value	Access
0xB00	RXPTP_REVID	[31:0]	IP core revision ID.	0x0916_2016	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.	0x7843_5352	RO
0xB03	RXPTP_NAME_1	[31:0]	Next 4 characters of IP core variation identifier string	0x5054_5054	RO
0xB04	RXPTP_NAME_2	[31:0]	Final 4 characters of IP core variation identifier string.	0x3034_3067	RO
0xB05	RX_PTP_CLK_PERIOD	[19:0]	clk_rxmac clock period. Bits [19:16]: Full nanoseconds Bits [15:0]: Fraction of a nanosecond	This value is set to the correct clock period for the	RW

continued...

Addr	Name	Bit	Description	HW Reset Value	Access
				required RX MAC clock frequency.	
0xB06	RX_PTP_PMA_LATENCY	[31:0]	Latency through the RX PMA. This is the delay from the rx_serial pins to the RX PCS input. • Bits [31:16]: Full nanoseconds • Bits [15:0]: Fraction of a nanosecond In Arria 10 devices, the RX PMA latency is 102.5 UI. One UI is approximately 38.8 ps. Therefore, Intel recommends that you set this register to the value of 0x0003_FA1C. This is a device-dependent value that is sufficiently accurate in most cases. Intel recommends that you modify this value with extreme caution.	32'b0	RW

Related Information

- 1588 Precision Time Protocol Interfaces on page 69
- PTP Transmit Functionality on page 74

3.4.1.8. TX Reed-Solomon FEC Registers

Table 45. TX Reed-Solomon FEC Registers

Addr	Name	Bit	Description	Reset	Access
0xC00	REVID	[31:0]	Reed-Solomon FEC TX module revision ID.	0x05152015	RO
0xC01	TX_RSFEC_SCRATCH	[31:0]	Scratch register available for testing.	32'b0	RW
0xC02	TX_RSFEC_NAME_0	[31:0]	Final 4 characters of IP core variation identifier string, "100gRSFECoTX".	0x436F 5458	RO
0xC03	TX_RSFEC_NAME_1	[31:0]	Middle 4 characters of IP core variation identifier string, "100gRSFECoTX".	0x5253 4645	RO
0xC04	TX_RSFEC_NAME_2	[31:0]	Initial 4 characters of IP core variation identifier string, "100gRSFECoTX".	0x3130 3067	RO
0xC05	ERR_INS_EN	[4]	When 1'b1, enables error insertion for single FEC codeword. This bit self-clears after the Reed-Solomon FEC transmitter inserts the error.	0x00000000	RW
		[3:1]	Reserved.		
		[0]	When 1'b1, enables error insertion for every FEC codeword. Specifies that the Reed-Solomon FEC transmitter should insert the error in every FEC codeword.		
0xC06	ERR_MASK	[24]	SYM_32: Each FEC codeword consists of 16 groups of 33 symbols. This register field specifies whether the RS-FEC transmitter corrupts symbol 32 (of symbols 0-32) in each corrupted group.	0x00000000	RW
		[23:18]	Reserved.		

continued...

Addr	Name	Bit	Description	Reset	Access
		[17:8]	BIT_MASK: Specifies which of the ten bits the RS-FEC transmitter corrupts in each corrupted symbol.		
		[7:4]	Reserved.		
		[3:0]	GROUP_NUM: Each FEC codeword consists of 16 groups of 33 symbols. This register field specifies the single group of 33 symbols that the RS-FEC transmitter corrupts in the current FEC codeword. The following values are defined: • 4'b000: First group of 33 symbols (symbols 0-32) • 4'b0001: Second group of symbols (symbols 33-65) • ... • 4'b1110: Second-to-final group of symbols (symbols 462-494) • 4'b1111: Final group of symbols (symbols 495-527, or {chk[13:0],sym[514:495]})		
0xC07	SYMBOL_ERR_MASK	[31:0]	Each FEC codeword consists of 16 groups of 33 symbols. This register specifies which of the lower order 32 symbols in a group the RS-FEC transmitter corrupts.	32'b0	RW

3.4.1.9. RX Reed-Solomon FEC Registers

Table 46. RX Reed-Solomon FEC Registers

Addr	Name	Bit	Description	Reset	Access
0xD00	REVID	[31:0]	RSFEC RX module revision ID	0x0515 2015	RO
0xD01	RX_RSFEC_SCRATCH	[31:0]	Scratch register available for testing.	32'b0	RW
0xD02	RX_RSFEC_NAME0	[31:0]	Final 4 characters of IP core variation identifier string, "100gRSFECoRX".	0x436F 5258	RO
0xD03	RX_RSFEC_NAME1	[31:0]	Middle 4 characters of IP core variation identifier string, "100gRSFECoRX".	0x5253 4645	RO
0xD04	RX_RSFEC_NAME2	[31:0]	Initial 4 characters of IP core variation identifier string, "100gRSFECoRX".	0x3130 3067	RO
0xD05	BYPASS_RESTART	[4]	Restart state machines. When 1'b1, specifies the IP core restarts the FEC synchronization and alignment state machines. Bit self-clears after alignment marker synchronization is restarted. (Refer to Figure 91-8 and Figure 91-9 in *IEEE Standard 802.3bj-2014*).	0x0000 0000	RW
		[3:1]	Reserved.		

continued...

Addr	Name	Bit	Description	Reset	Access
		[0]	Bypass RS-FEC decoder. When 1'b1, specifies the IP core bypasses the RS-FEC decoder. When 1'b0, enables RS-FEC error correction.		
0xD06	RX_FEC_STATUS	[15:8]	fec_lane: Two bits per lane hold the FEC lane number when the corresponding amps_lock bit (in register bits [3:0]) has the value of 1. The following encodings are defined: • Bits[15:14]: fec_lane for lane 3 • Bits[13:12]: fec_lane for lane 2 • Bits[11:10]: fec_lane for lane 1 • Bits[9:8]: fec_lane for lane 0	0x0000 0000	RO
		[7:5]	Reserved.		
		[4]	fec_align_status: Alignment marker lock status. When 1'b1, indicates all lanes are synchronized and aligned. When 1'b0, indicates the deskew process is not yet complete. (Refer to Figure 91-9 in *IEEE Standard 802.3bj-2014*).		
		[3:0]	amps_lock: Each bit indicates that the receiver has detected the location of the alignment marker payload sequence for the corresponding FEC lane. (Refer to Figure 91-8 in *IEEE Standard 802.3bj-2014*).		
0xD07	CORRECTED_CW	[31:0]	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
0xD08	UNCORRECTED_CW	[31:0]	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

3.4.2. LL 100GbE Hardware Design Example Registers

The following sections describe the registers that are included in the LL 100GbE hardware design example and are not a part of the LL 100GbE IP core.

Related Information

Low Latency 100G Ethernet Design Example User Guide

3.4.2.1. Packet Client Registers

You can customize the LL 100GbE hardware design example by programming the packet client registers.

Table 47. Packet Client Registers

Addr	Name	Bit	Description	HW Reset Value	Access
0x1000	PKT_CL_SCRATCH	[31:0]	Scratch register available for testing.		RW
0x1001	PKT_CL_CLNT	[31:0]	Four characters of IP block identification string "CLNT"		RO
0x1002	PKT_CL_FEATURE	[9:0]	Feature vector to match DUT. Bits [8:3] have the value of 0 to indicate the DUT does not have the property or the value of 1 to indicate the DUT has the property. • Bit [0]: Has the value of 0 if the DUT targets a Stratix V device; has the value of 1 if the DUT targets an Arria 10 device. • Bit [1]: Has the value of 1 to indicate that the DUT is a LL 100GbE IP core. • Bit [2]: Reference clock frequency. Has the value 0 for 322 MHz; has the value of 1 for 644 MHz. • Bit [3]:Reserved. • Bit [4]: Indicates whether the DUT is a CAUI-4 IP core. • Bit [5]: Indicates whether the DUT includes PTP support. • Bit [6]: Indicates whether the DUT includes pause support. • Bit [7]: Indicates whether the DUT provides local fault signaling. • Bit [8]: Indicates whether the DUT has **Use external MAC TX PLL** turned on. Must have the value of 0. • Bit [9]: Value 0 if the DUT has a custom streaming client interface; value 1 if the DUT has an Avalon-ST client interface. Must have the value of 1.		RO
0x1006	PKT_CL_TSD	[7:0]	Arria 10 device temperature sensor diode readout in Fahrenheit.		RO
0x1010	PKT_GEN_TX_CTRL	[3:0]	• Bit [0]: Reserved. • Bit [1]: Packet generator disable bit. set this bit to the value of 1 to turn off the packet generator, and reset it to the value of 0 to turn on the packet generator. • Bit [2]: Reserved. • Bit [3]: Has the value of 1 if the IP core is in MAC loopback mode; has the value of 0 if the packet client uses the packet generator.	4'b0101	RW
0x1015	PKT_CL_LOOPBACK_FIFO_ERR_CLR	[2:0]	Reports MAC loopback errors. • Bit [0]: FIFO underflow. Has the value of 1 if the FIFO has underflowed. This bit is sticky. Has the value of 0 if the FIFO has not underflowed. • Bit [1]: FIFO overflow. Has the value of if the FIFO has overflowed. This bit is sticky. Has the value of 0 if the FIFO has not overflowed. • Bit [2]: Assert this bit to clear bits [0] and [1].	3'b0	RO
0x1016	PKT_CL_LOOPBACK_RESET	[0]	MAC loopback reset. Set to the value of 1 to reset the design example MAC loopback.	1'b0	RW

3.5. Ethernet Glossary

Table 48. Ethernet Glossary

Provides definitions for some terms associated with the Ethernet protocol.

Term	Definition
BIP	Bit Interleaved Parity. A diagonal parity field which is carried in the periodic alignment markers on each virtual lane, allowing isolation of a bit error to a physical channel.
CAUI	100 gigabit attachment unit interface. (C is the symbol in Roman Numerals for 100). This is an electrical interface that which is based on a 10-lane interface with a bandwidth of 10 Gbps per lane. (In this implementation, the PMA multiplexes the 20 PCS lanes into 10 physical lanes.
CGMII	100 gigabit media independent interface. (C is the symbol in Roman Numerals for 100). This is the byte-oriented interface protocol that connects the PCS and MAC.
DIC	Deficit Idle Counter. A rule for inserting and deleting idles as necessary to maintain the average IPG. The alternative is to always insert idles which could lead to reduced bandwidth.
FCS	Frame Check Sequence. A CRC-32 with bit reordering and inversion.
Frame	Ethernet formatted packet. A frame consists of a start delimiter byte, a 7 byte preamble, variable length data, 4-byte FCS, and an end delimiter byte.
IPG	Inter Packet Gap. Includes the end of frame delimiter and subsequent IDLE bytes up to, but not including the next start of frame delimiter. The protocol requires an average gap of 12 bytes.
MAC	Media Access Control. Formats a user packet stream into proper Ethernet frames for delivery to the PCS. The MAC generates the FCS and checks and maintains the IPG.
MII	Media Independent Interface. The byte-oriented protocol used by the PCS. Sometimes distinguished with roman numerals, XGMII (10), XLGMII (40), CGMII (100).
Octet	Byte. Note that Ethernet specifications primarily use least significant bit first ordering which is opposite from the default behavior of most contemporary CAD tools.
PCS	Physical Coding Sublayer. Presents the underlying hardware as a byte-oriented communication channel.
XLAUI	40 gigabit attachment unit interface. (XL is the symbol in Roman Numerals for 40). This is an electrical interface that which is based on a 4-lane interface with a bandwidth of 10 Gbps per lane.
XLGMII	40 gigabit media independent interface. (XL is the symbol in Roman Numerals for 40). This is the byte-oriented interface protocol that connects the PCS and MAC.

4. 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 LL 100GbE 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 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:
   - A bit error rate (BER)
   - 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 analog parameters to improve the bit error rate. IDLE traffic is representative for analog purposes.

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 triggers the RX digital reset process.

Related Information

Arria 10 Transceiver PHY User Guide
For information about the analog parameters for Arria 10 devices.

4.1. Creating a Signal Tap Debug File to Match Your Design Hierarchy

For Intel Arria 10 and Intel Cyclone® 10 GX devices, the Intel Quartus Prime software generates two files, `build_stp.tcl` and `<ip_core_name>.xml`. You can use these files to generate a Signal Tap file with probe points matching your design hierarchy.

The Intel Quartus Prime software stores these files in the `<IP core directory>/synth/debug/stp/` directory.

Synthesize your design using the Intel Quartus Prime software.

1. To open the Tcl console, click **View ➤ Utility Windows ➤ Tcl Console**.
2. Type the following command in the Tcl console:
   `source <IP core directory>/synth/debug/stp/build_stp.tcl`
3. To generate the STP file, type the following command:
   `main -stp_file <output stp file name>.stp -xml_file <input xml_file name>.xml -mode build`
4. To add this Signal Tap file (**.stp**) to your project, select **Project ➤ Add/Remove Files in Project**. Then, compile your design.
5. To program the FPGA, click **Tools ➤ Programmer**.
6. To start the Signal Tap Logic Analyzer, click **Quartus Prime ➤ Tools ➤ Signal Tap Logic Analyzer**.

   The software generation script may not assign the Signal Tap acquisition clock in `<output stp file name>.stp`. Consequently, the Intel Quartus Prime software automatically creates a clock pin called `auto_stp_external_clock`. You may need to manually substitute the appropriate clock signal as the Signal Tap sampling clock for each STP instance.
7. Recompile your design.
8. To observe the state of your IP core, click **Run Analysis**.

You may see signals or Signal Tap instances that are red, indicating they are not available in your design. In most cases, you can safely ignore these signals and instances. They are present because software generates wider buses and some instances that your design does not include.

A. Additional Information

A.1. LL 100GbE IP Core User Guide Archives

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

IP Core Version	User Guide
16.0	Low Latency 40- and 100-Gbps Ethernet MAC and PHY MegaCore Function User Guide 16.0
15.1	Low Latency 40- and 100-Gbps Ethernet MAC and PHY MegaCore Function User Guide 15.1
15.0	Low Latency 40- and 100-Gbps Ethernet MAC and PHY MegaCore Function User Guide 15.0
14.1	Low Latency 40- and 100-Gbps Ethernet MAC and PHY MegaCore Function User Guide 14.1

A.2. LL 100GbE IP Core User Guide Revision History

Date	Compatible Intel Quartus Prime Version	Changes
2021.04.15	16.1	• Updated the descriptions for the following signals in Table: *Signals of the 1588 Precision Time Protocol Interface*: — `tx_etstamp_ins_ctrl_residence_time_calc_format` — `tx_egress_timestamp_64b_data[63:0]` — `tx_egress_timestamp_96b_fingerprint[(W-1):0]` — `tx_egress_timestamp_64b_fingerprint[(W-1):0]` • Clarified the description of PHY_PMA_SLOOP register.
2020.09.04	16.1	Added clarifying text for the `tx_etstamp_ins_ctrl_offset_checksum_correction[15:0]` signal in the *1588 PTP Interface Signals* section: *In a PTP packet, two bytes before the CRC field represent the valid offset for the checksum correction field.*
2020.02.11	16.1	• Updated for latest Intel branding standards. • Renamed Altera Debug Master Endpoint (ADME) to Native PHY Debug Master Endpoint (NPDME). • Added clarifying note for the `tx_etstamp_ins_ctrl_residence_time_update` signal in the *PTP Transmit Functionality* section. • Renamed OpenCore Plus IP Evaluation to Intel FPGA IP Evaluation Mode and updated its content. • Fixed minor typos.
2018.01.03	16.1	Clarified that this IP core is not supported in Platform Designer. Refer to Specifying the IP Core Parameters and Options on page 17. Fixed assorted errors and minor typos.
2017.11.06	16.1	Added link to KDB Answer that provides workaround for potential jitter on Arria 10 devices due to cascading ATX PLLs in the IP core. Refer to Handling Potential Jitter in Intel Arria 10 Devices on page 28.

continued...

Date	Compatible Intel Quartus Prime Version	Changes
		Clarified that despite `.vhd` files being generated with the IP core, the IP core does not support VHDL. Refer to Files Generated for Arria 10 Variations on page 22. Added missing information: values of RX pause, TX pause, RX PTP, and TX PTP registers at offsets 0x02, 0x03, 0x04. Refer to Pause Registers on page 102 and 1588 PTP Registers on page 114. Clarified that software must reset the PHY_SCLR_FRAME_ERROR register to the value of 0 within ten `clk_status` clock cycles of setting it to the value of 1. If you do not reset the PHY_SCLR_FRAME_ERROR register, the value in the PHY_FRAME_ERROR register is not useful. Refer to PHY Registers on page 97. Clarified that the design example includes SDC files that you can modify for your own design. Refer to Compiling the Full Design and Programming the FPGA on page 35.
2016.11.23	16.1	Initial version of *Low Latency 100G Ethernet IP Core User Guide*. Changes from the 16.0 version of the *Low Latency 40- and 100-Gbps Ethernet MAC and PHY MegaCore Function User Guide*: • Updated for the Intel Quartus Prime software v16.1. • Removed information about the Low Latency 40GbE IP core. For the 16.1 release, the LL 40GbE IP core is documented in the 16.0 version of the *Low Latency 40- and 100-Gbps Ethernet MAC and PHY MegaCore Function User Guide* and changes are noted in the *Intel FPGA IP Release Notes*. • Added option for Reed-Solomon forward error correction (RS-FEC) in CAUI-4 variations. This new feature in the 16.1 software release adds a parameter and new registers to the IP core. Refer to IP Core Parameters on page 18, TX RSFEC on page 40, RX RSFEC on page 51, TX Reed-Solomon FEC Registers on page 116, and RX Reed-Solomon FEC Registers on page 117. • Updated for new handling of runt pause frames and pause frames with FCS errors. The IP core no longer processes these standard flow control pause frames. This feature is new in the 16.1 release and is relevant only for Arria 10 variations. Refer to Pause Control Frame Filtering on page 63. • Removed erroneous statement that the IP core ignores a Start control character it receives on any lane other than Lane 0. Refer to LL 100GbE IP Core Malformed Packet Handling on page 50. • Clarified that the IP core does not support a correct VHDL variation. You must generate a Verilog HDL variation of this IP core. Refer to Specifying the IP Core Parameters and Options on page 17. • Clarified that the control and status interface is a non-pipelined Avalon-MM interface with variable latency, and therefore cannot process multiple pending read transfers correctly. Refer to Control and Status Interface on page 82. • Clarified that the RX recovered clock frequency in CAUI-4 variations is 402.83203 MHz. Refer to Clocks on page 84. • Clarified that the client is responsible to ensure that the PTP offset input values guarantee the full timestamp or checksum does not overflow the packet. The warning applies to `tx_etstamp_ins_ctrl_offset_timestamp`, `tx_etstamp_ins_ctrl_offset_correction_field`, `tx_etstamp_ins_ctrl_offset_checksum_field`, and `tx_etstamp_ins_ctrl_offset_checksum_correction`. Refer to 1588 PTP Interface Signals on page 78. • Clarified that the PHY_SCLR_FRAME_ERROR (at offset 0x324) and PHY_EIO_SFTRESET (at offset 0x325) registers are not self-clearing. Refer to PHY Registers on page 97. • Clarified the specific PLLs referred to in the descriptions of the PHY_TX_PLL_LOCKED register and the PHY_TX_COREPLL_LOCKED[1] register field. Refer to PHY Registers on page 97. • Clarified that SOP and EOP can occur on the RX client interface on the same clock cycle. Refer to LL 100GbE IP Core RX Data Bus with Adapters (Avalon-ST Interface) on page 53 and LL 100GbE IP Core RX Data Bus Without Adapters (Custom Streaming Interface) on page 56.

Date	Compatible Intel Quartus Prime Version	Changes
		• Reorganized the Getting Started chapter for clarity. Refer to Getting Started on page 12. • Relocated information about the PTP timestamp accuracy. Refer to 1588 Precision Time Protocol Interfaces on page 69. • Fixed assorted typos and minor errors.