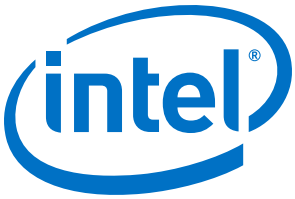


Low Latency 100G Ethernet Intel® Stratix® 10 FPGA IP Core User Guide

Updated for Intel® Quartus® Prime Design Suite: **18.0**

UG-20085 | 2018.07.18
Latest document on the web: **PDF** | **HTML**

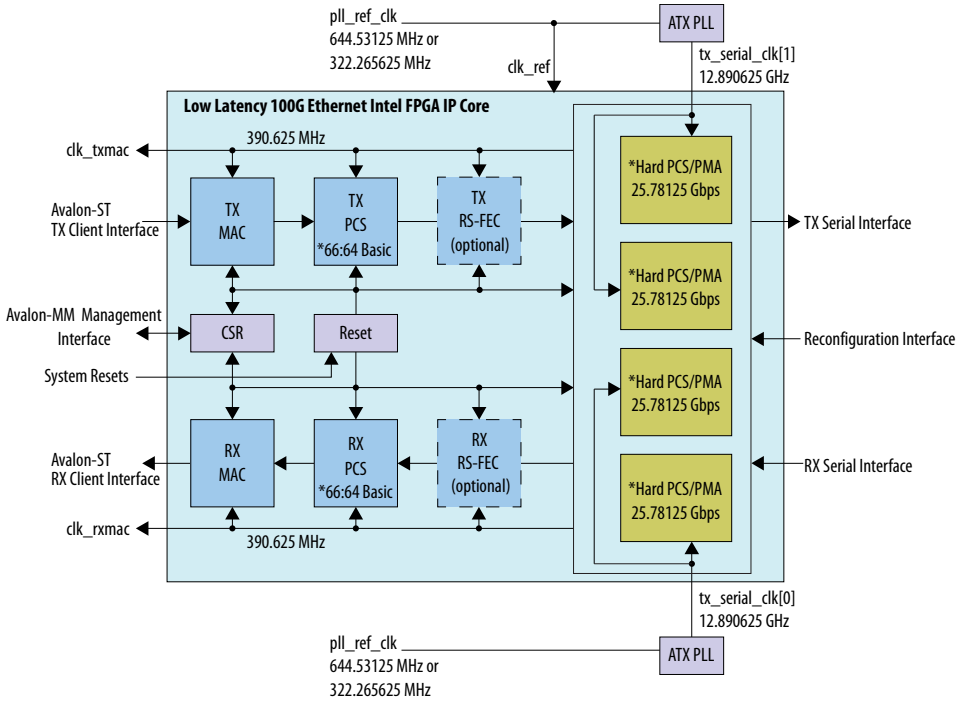
Contents

1. About the Low Latency 100G Ethernet Intel FPGA IP Core

The Intel® Stratix® 10 Low Latency 100G Ethernet Intel FPGA IP core offers low round-trip latency and small size to implement the *IEEE 802.3ba High Speed Ethernet Standard*.

Figure 1. Low Latency 100G Ethernet Intel FPGA IP Core

Main blocks, internal connections, and external block requirements.



*The IP core uses TX PCS and RX PCS to do 66:64 bit encoding/decoding. Hard PCS isused for other link related functions.

The MAC client side Avalon Streaming (Avalon-ST) interface data bus is 512 bits wide. The client-side data maps to four 25.78125 Gbps transceiver PHY links.

The FPGA serial transceivers are compliant with the IEEE 802.3ba standard CAUI-4 specification. You can connect the transceiver interfaces directly to an external physical medium dependent (PMD) optical module or to another device.

Related Information

- Functional Description on page 21
  Provides detailed descriptions of Low Latency 100G Ethernet Intel FPGA IP core operation and functions.

- Low Latency 100G Ethernet Intel Stratix 10 FPGA IP Design Example User Guide

1.1. Low Latency 100G Ethernet Intel FPGA IP Core Supported Features

The IP core is designed to the *IEEE 802.3ba-2010 High Speed Ethernet Standard* available on the IEEE website (www.ieee.org). The MAC provides cut-through frame processing to optimize latency, and supports full wire line speed with a 64-byte frame length and back-to-back or mixed length traffic with no dropped packets. All Low Latency 100G Ethernet Intel FPGA 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:

- PHY features:
  - Soft PCS logic that interfaces seamlessly to Intel Stratix 10 FPGA 25.78125 Gbps serial transceivers.
  - CAUI-4 external interface consisting of four FPGA hard serial transceiver lanes operating at 25.78125 Gbps.
  - Optional Reed-Solomon forward error correction RS-FEC(528,514).
- Frame structure control features:
  - Support for jumbo packets.
  - TX and RX CRC pass-through control.
  - Optional TX CRC generation and insertion.
  - 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 Low Latency 100G Ethernet Intel FPGA IP Ethernet connection.
- Frame monitoring and statistics:
  - RX CRC checking and error reporting.
  - Optional RX strict SFD checking per IEEE specification.
  - RX malformed packet checking per IEEE specification.
  - Received control frame type indication.
  - Optional statistics counters.
  - Optional fault signaling: reports local fault and generates remote fault, with *IEEE 802.3ba-2012 Ethernet Standard* Clause 66 support.
- Flow control:
  - 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.
  - Pause frame filtering control.

- Debug and testability features:
  - Optional serial PMA loopback (TX to RX) at the serial transceiver for self-diagnostic testing.
  - TX error insertion capability supports test and debug.
  - Optional access to Altera Debug Master Endpoint (ADME) for debugging or monitoring PHY signal integrity.
- User system interfaces:
  - Avalon Memory-Mapped (Avalon-MM) management interface to access the IP core control and status registers.
  - Avalon-ST data path interface connects to client logic with the start of frame in the most significant byte (MSB). Interface has data width 512 bits, to ensure the data rate despite this RX client interface SOP alignment and RX and TX preamble passthrough option.
  - Hardware and software reset control.

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 is 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 100G Ethernet Intel FPGA IP core:

Low Latency 100G Ethernet Intel FPGA IP Core Device Family Support on page 6

Low Latency 100G Ethernet Intel FPGA IP Core Device Speed Grade Support on page 7

1.2.1. Low Latency 100G Ethernet Intel FPGA IP Core Device Family Support

Table 1. Intel FPGA IP Core Device Support Levels

Device Support Level	Definition
Advance	The IP core is available for simulation and compilation for this device family. Timing models include initial engineering estimates of delays based on early post-layout information. The timing models are subject to change as silicon testing improves the correlation between the actual silicon and the timing models. You can use this IP core for system architecture and resource utilization studies, simulation, pinout, system latency assessments, basic timing assessments (pipeline budgeting), and I/O transfer strategy (datapath width, burst depth, I/O standards tradeoffs).
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. Low Latency 100G Ethernet Intel FPGA IP Core Device Family Support

Shows the level of support offered by the Low Latency 100G Ethernet Intel FPGA IP core for each Intel FPGA device family.

Device Family	Support
Intel Stratix 10	Advance
Other device families	No support

Related Information

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

1.2.2. Low Latency 100G Ethernet Intel FPGA IP Core Device Speed Grade Support

Table 3. Slowest Supported Device Speed Grades

Lists the slowest supported device speed grades for the Low Latency 100G Ethernet Intel FPGA IP core.

Device Family	Supported Speed Grades
Intel Stratix 10	E2

1.3. IP Core Verification

To ensure functional correctness of the Low Latency 100G Ethernet Intel FPGA IP core, Intel performs extensive validation through both simulation and hardware testing. Before releasing a version of the Low Latency 100G Ethernet Intel FPGA 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 100G Ethernet Intel FPGA IP core
  Exceptions to functional correctness that first manifest in software releases 17.1 and later are documented in the Intel Stratix 10 Low Latency 100GbE IP core errata.
- Intel FPGA IP Release Notes:Intel Stratix 10 Low Latency 100-Gbps Ethernet IP Core Release Notes
  Changes to the Low Latency 100G Ethernet Intel FPGA IP core in software releases 17.1 and later are noted in the Intel FPGA IP Release Notes.

1.3.1. Simulation Environment

Intel performs the following tests on the Low Latency 100G Ethernet Intel FPGA IP core in the simulation environment using internal and third party standard bus functional models (BFM):

- Constrained random tests that cover randomized frame size and contents
- 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 Low Latency 100G Ethernet Intel FPGA 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 Low Latency 100G Ethernet Intel FPGA IP core using standard 100Gbps Ethernet network test equipment and optical modules. The Intel hardware tests of the Low Latency 100G Ethernet Intel FPGA 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

Resource utilization changes depending on the parameter settings you specify in the Low Latency 100G Ethernet Intel FPGA IP parameter editor. For example, if you turn on RS-FEC in the Low Latency 100G Ethernet Intel FPGA parameter editor, the IP core requires additional resources to implement the additional functionality.

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	E
Enable RS-FEC	—	—	On	—	On
Enable TX CRC insertion	—	On	On	On	On
Enable preamble passthrough	—	—	—	On	On
Enable RX/TX statistics counters	—	On	On	On	On

Table 5. IP Core FPGA Resource Utilization

Lists the resources and expected performance for selected variations of the Low Latency 100G Ethernet Intel FPGA IP core, from one compilation of each IP core variation. Your results may vary depending on your overall design.

These results were obtained using the Intel Quartus Prime Pro Edition v17.1 software.

Note: Resource utilization numbers for variations with RS-FEC enabled, reflect preliminary results for the RS-FEC feature. The resource utilization for this block might vary by up to 5% in the final implementation of this feature.

- 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
A	24200	61400	40
B	29100	74800	40
C	55200	132500	101
D	29100	74500	40
E	55200	141700	101

1.5. Release Information

Table 6. Low Latency 100G Ethernet Intel FPGA IP Core Current Release Information

Item	Description
Version	Intel Quartus Prime Pro Edition v18.0
Release Date	2018.05.04
Ordering Codes	Low Latency 100G Ethernet MAC and PHY: IP-100GEUMACPHY

2. Getting Started

The following sections explain how to install, parameterize, simulate, and initialize the Low Latency 100G Ethernet Intel FPGA 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.
- Generating a Combined Simulator Setup Script
  Create simulation scripts that do not require manual updates for software or IP version upgrades.
- Project Management Best Practices
  Guidelines for efficient management and portability of your project and IP files.

2.1. Installing and Licensing Intel FPGA IP Cores

The Intel Quartus Prime 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 2. IP Core Installation Path

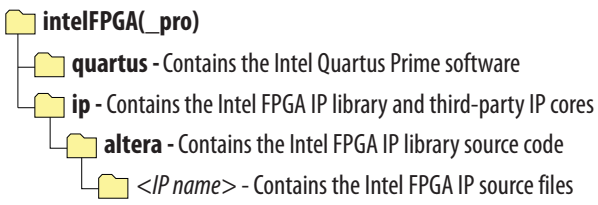


Table 7. IP Core Installation Locations

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

2.2. Specifying the IP Core Parameters and Options

The Low Latency 100G Ethernet Intel FPGA parameter editor allows you to quickly configure your custom IP variation. Use the following steps to specify IP core options and parameters in the Intel Quartus Prime Pro Edition software.

1. If you do not already have an Intel Quartus Prime Pro Edition project in which to integrate your Low Latency 100G Ethernet Intel FPGA IP core, you must create one.
   a. In the Intel Quartus Prime Pro Edition, click **File ➤ New Project Wizard** to create a new Quartus Prime project, or **File ➤ Open Project** to open an existing Quartus Prime project. The wizard prompts you to specify a device.
   b. Specify the device family **Intel Stratix 10** and select a device that meets all of these requirements:
      i. Transceiver tile is L-tile or H-tile (any transceiver tile)
      ii. Transceiver speed grade is 1 or 2
      iii. Core speed grade is 1 or 2
      iv. Device is not an 1SG280L ES1 device (part name 1SG280L...VGS1)
   c. Click **Finish**.
2. In the IP Catalog, locate and select **Low Latency 100G Ethernet**. The **New IP Variation** window appears.
3. Specify a top-level name for your new custom IP variation. The parameter editor saves the IP variation settings in a file named `<your_ip>.ip`.
4. Click **OK**. The parameter editor appears.
5. Specify the parameters for your IP core variation. Refer to IP Core Parameters on page 19 for information about specific IP core parameters.
6. Optionally, to generate a simulation testbench or compilation and hardware design example, follow the instructions in the *Intel Stratix 10 Low Latency 100G Ethernet Design Example User Guide*.
7. Click **Generate HDL**. The **Generation** dialog box appears.
8. Specify output file generation options, and then click **Generate**. The IP variation files generate according to your specifications.

   Note: A functional VHDL IP core is not available. Specify Verilog HDL only, for your IP core variation.
9. Click **Finish**. The parameter editor adds the top-level `.ip` file to the current project automatically. If you are prompted to manually add the `.ip` file to the project, click **Project ➤ Add/Remove Files in Project** to add the file.
10. After generating and instantiating your IP variation, make appropriate pin assignments to connect ports.

Related Information

Intel Stratix 10 Low Latency 100G Ethernet Design Example User Guide
Information about generating the Low Latency 100G Ethernet Intel FPGA design example.

2.3. Generated File Structure

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

For information about the file structure of the design example, refer to the *Low Latency 100G Ethernet Intel Stratix 10 FPGA IP Design Example User Guide*.

Figure 3. Low Latency 100G Ethernet Intel FPGA IP Core Generated Files

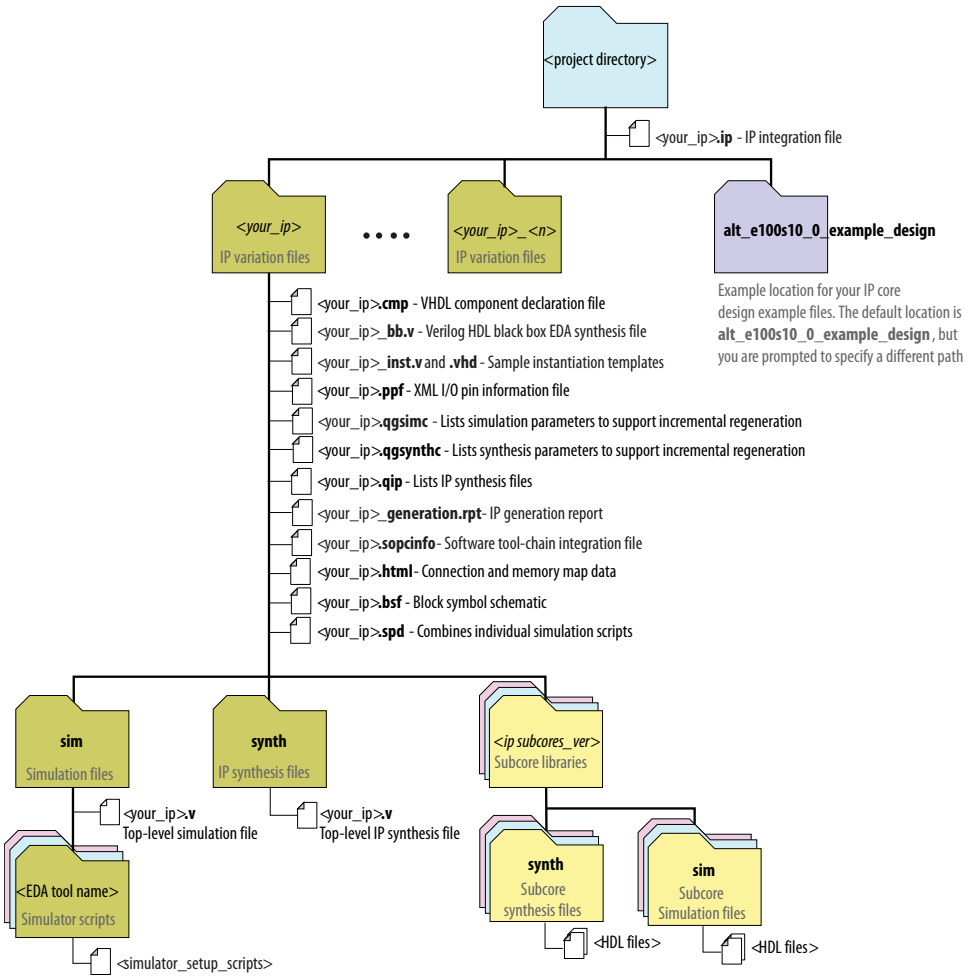


Table 8. IP Core Generated Files

File Name	Description
<*your_ip*>.ip	The Platform Designer system or top-level IP variation file. <*your_ip*> is the name that you give your IP variation.
<*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. This IP core does not support VHDL. However, the Intel Quartus Prime Pro Edition 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 generation log file. A summary of the messages during IP generation.
<*your_ip*>.qgsimc	Lists simulation parameters to support incremental regeneration.
<*your_ip*>.qgsynthc	Lists synthesis parameters to support incremental regeneration.
<*your_ip*>.qip	Contains all the required information about the IP component to integrate and compile the IP component in the Intel Quartus Prime software.
<*your_ip*>.sopcinfo	Describes the connections and IP component parameterizations in your Platform Designer system. You can parse its contents to get requirements when you develop software drivers for IP components. 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*>.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 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*>_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 Pro Edition software generates the `_inst.vhd` 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 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 can query for register map information. For system slaves, Platform Designer can access the registers by name.
<*your_ip*>.v	HDL files that instantiate each submodule or child IP core for synthesis or simulation.

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 submodules.
<child IP cores>/	For each generated child IP core directory, Platform Designer generates synth/ and sim/ sub-directories.

Related Information

Low Latency 100G Ethernet Intel Stratix 10 FPGA IP Design Example User Guide
Information about the Low Latency 100G Ethernet Intel FPGA design example file structure.

2.4. 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 14

Adding the Transceiver PLLs on page 15

Placement Settings for the Low Latency 100G Ethernet Intel FPGA IP Core on page 17

Related Information

Low Latency 100G Ethernet Intel Stratix 10 FPGA IP Design Example User Guide

2.4.1. Pin Assignments

When you integrate your Low Latency 100G Ethernet Intel FPGA 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.

Related Information

- Adding the Transceiver PLLs on page 15
- Quartus Prime Help
  For information about the Quartus Prime software, including virtual pins and the IP Catalog.
- Intel Stratix 10 L- and H-Tile Transceiver PHY User Guide
  Information about constraints on transceiver configuration in Intel Stratix 10 devices.
- Intel Stratix 10 GX 2800 L-Tile ES-1Transceiver PHY User Guide
  Information about transceiver configuration in L-tile ES1 devices.

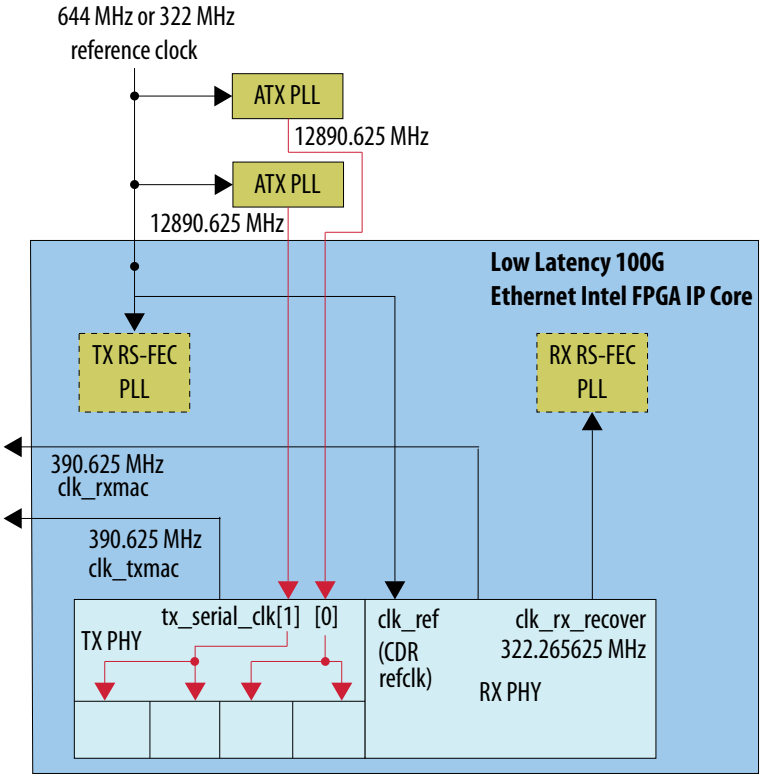
2.4.2. Adding the Transceiver PLLs

The Low Latency 100G Ethernet Intel FPGA IP core requires two external TX transceiver PLLs to compile and to function correctly in hardware. On Intel Stratix 10 devices, only the ATX PLL supports the required data rate.

The transceiver PLLs you configure are physically present on the device, but the Low Latency 100G Ethernet Intel FPGA IP core does not configure and connect them. The required number of ATX PLLs is two. Each ATX PLL drives the clocks for two transceiver channels.

Figure 4. PLL Configuration Example

The TX transceiver PLLs are instantiated with two Intel Stratix 10 ATX PLL IP cores. The TX transceiver PLLs must always be instantiated outside the Low Latency 100G Ethernet Intel FPGA IP core.



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

- Select **Stratix 10 L-Tile/H-Tile Transceiver ATX PLL**.
- In the parameter editor, set the following parameter values:
  - Set **VCCR_GXB and VCCT_GXB supply voltage for the Transceiver** to **1_1V**.
  - Set **Primary PLL clock output buffer** to **GXT clock output buffer**.
  - Turn on **Enable GXT local clock output port (tx_serial_clk_gxt)**.
  - Set **GXT output clock source** to **Local ATX PLL**.
  - **PLL output frequency** to **12890.625 MHz**. The transceiver performs dual edge clocking, using both the rising and falling edges of the input clock from the PLL. Therefore, this PLL output frequency setting supports a 25.78125 Gbps data rate through the transceiver.
  - Set **PLL auto mode reference clock frequency** to the value you specified for the **PHY reference frequency** parameter. .

When you generate a Low Latency 100G Ethernet Intel FPGA IP core, the software also generates the HDL code for an ATX PLL, in the file *<variation_name>*/`atx_pll_s100.v`. However, the HDL code for the Low Latency 100G Ethernet Intel FPGA IP core does not instantiate the ATX PLL. If you choose to use the ATX PLL provided with the Low Latency 100G Ethernet Intel FPGA IP core, you must instantiate and connect the instances of the ATX PLL with the Low Latency 100G Ethernet Intel FPGA IP core in user logic.

Note: If your design includes multiple instances of the Low Latency 100G Ethernet Intel FPGA 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.

You must drive the reference clock input ports of the two PLLs with the same clock to minimize PMM differences. This clock can be but need not be the same as the clock that drives the Low Latency 100G Ethernet Intel FPGA IP core reference clock.

Each PLL drives the `tx_serial_clk` input of two of the Low Latency 100G Ethernet Intel FPGA IP core PHY links. You must connect the PLLs to the Low Latency 100G Ethernet Intel FPGA IP core as follows:

PLL	PLL Signal	Low Latency 100G Ethernet Intel FPGA IP Core Signal
A	`tx_serial_clk`	`tx_serial_clk[0]`
A	`pll_locked`	`tx_pll_locked[0]`
B	`tx_serial_clk`	`tx_serial_clk[1]`
B	`pll_locked`	`tx_pll_locked[1]`

Refer to the example compilation project or design example for working user logic that demonstrates one correct method to instantiate and connect the external PLLs.

Related Information

- Intel Stratix 10 L- and H-Tile Transceiver PHY User Guide
  Information about the correspondence between PLLs and transceiver channels in Intel Stratix 10 devices, and information about how to configure an external transceiver PLL for your own design. Refer to the sections about the GXT clock network and about using the ATX PLL for GXT channels.
- Intel Stratix 10 GX 2800 L-Tile ES-1Transceiver PHY User Guide
  Information about the correspondence between PLLs and transceiver channels in L-tile ES1 devices, and information about how to configure an external transceiver PLL for your own design. Refer to the sections about the GXT clock network and about using the ATX PLL for GXT channels.
- Low Latency 100G Ethernet Intel Stratix 10 FPGA IP Design Example User Guide
  Information about the Low Latency 100G Ethernet Intel FPGA design example, which connects two external PLLs to the IP core PHY links.

2.4.3. Placement Settings for the Low Latency 100G Ethernet Intel FPGA IP Core

The Intel Quartus Prime Pro Edition software provides the options to specify design partitions and Logic Lock regions for block-based design, 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 full design.

Related Information

- Intel Quartus Prime Pro Edition Handbook Volume 2: Design Implementation and Optimization
  Describes design constraints and LogicLock Plus regions.
- Block-Based Design Flows

2.5. IP Core Testbenches

Intel provides a compilation-only design example and a testbench with certain variations of the Low Latency 100G Ethernet Intel FPGA IP core.

To generate the testbench, in the Low Latency 100G Ethernet Intel FPGA IP 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 parameter editor provides warnings and the design example generation process creates a testbench that does not function correctly.

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 Intel Stratix 10 FPGA IP Design Example User Guide
Information about generating and running the design example and testbench files for the Low Latency 100G Ethernet Intel FPGA IP core. This testbench

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

2.6. Compiling the Full Design and Programming the FPGA

You can use the **Start Compilation** command on the Processing menu in the Intel Quartus Prime Pro Edition 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 Low Latency 100G Ethernet Intel FPGA IP core design example synthesis directories include Synopsys Constraint (`.sdc`) files that you can copy and modify for your own design.

Related Information

- Block-Based Design Flows
- Programming Intel FPGA Devices
- Low Latency 100G Ethernet Intel Stratix 10 FPGA IP Design Example User Guide

3. IP Core Parameters

The Low Latency 100G Ethernet Intel FPGA IP parameter editor provides the parameters you can set to configure the Low Latency 100G Ethernet Intel FPGA IP core and simulation and hardware design examples.

Low Latency 100G Ethernet Intel FPGA IP parameter editor includes an **Example Design** tab. For information about that tab, refer to the *Low Latency 100G Ethernet Intel Stratix 10 FPGA IP Design Example User Guide*.

Table 9. Low Latency 100G Ethernet Intel FPGA IP Parameters: Main Tab

Describes the parameters for customizing the Low Latency 100G Ethernet Intel FPGA IP core on the Main tab of the Low Latency 100G Ethernet Intel FPGA IP parameter editor.

Parameter	Type	Range	Default Setting	Parameter Description
General Options				
Device family	String	Stratix 10	Stratix 10	Selects the device family.
Target transceiver tile	String	• H-Tile • L-Tile	Default is set according to your Quartus project target device.	Selects the Intel Stratix 10 target transceiver tile. The value is set automatically according to your Quartus project target device.
PCS/PMA Options				
Enable RS-FEC	Boolean	• True • False	False	If this parameter is turned on, the IP core implements Reed-Solomon forward error correction (FEC) RS-FEC(528, 514).
PHY reference frequency	Integer (encoding)	• 644.53125 MHz • 322.265625 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).
Flow Control Options				
Enable MAC Flow Control	Boolean	• True • False	False	If turned on, the IP core enables the flow control mechanism and generates the `pause_insert_tx [1:0]` and `pause_receive_rx` signals. If turned off, the IP core disables the flow control mechanism.
Number of queues in priority flow control	Integer	1–8	1	Number of distinct priority queues for priority-based flow control.
MAC Options				
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

continued...

Parameter	Type	Range	Default Setting	Parameter Description
				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 RX/TX statistics counters	Boolean	• True • False	True	If turned on, the IP core includes built–in TX and RX statistics counters. If turned off, the IP core is configured without statistics counters. In any case, the IP core outputs frame status flags for the current input or output data.
Enable Strict SFD check	Boolean	• True • False	False	If turned on, the IP core can implement strict SFD checking, depending on register settings.
Configuration, Debug and Extension Options				
Enable Altera Debug Master Endpoint (ADME)	Boolean	• True • False	False	If turned on, the IP core turns on the following features in the Intel Stratix 10 Native PHY IP core that is included in the Low Latency 100G Ethernet Intel FPGA IP core: • **Enable Altera Debug Master Endpoint (ADME)** • **Enable capability registers** If turned off, the IP core is configured without these features.

Related Information

- Clocks on page 45
  The **PHY reference frequency** value is the required frequency of the transceiver reference clock.
- Intel Stratix 10 L- and H-Tile Transceiver PHY User Guide
  Information about the Stratix 10 L-tile Native PHY IP core features, including ADME.
- Low Latency 100G Ethernet Intel Stratix 10 FPGA IP Design Example User Guide
  Information about the **Example Design** tab in the Low Latency 100G Ethernet Intel FPGA IP parameter editor.

4. Functional Description

The Low Latency 100G Ethernet Intel FPGA IP core implements an Ethernet MAC in accordance with the *IEEE 802.3 Ethernet Standard*. The IP core handles the frame encapsulation and flow of data between client logic and an Ethernet network through a 100-Gbps Ethernet PCS and PMA (PHY) .

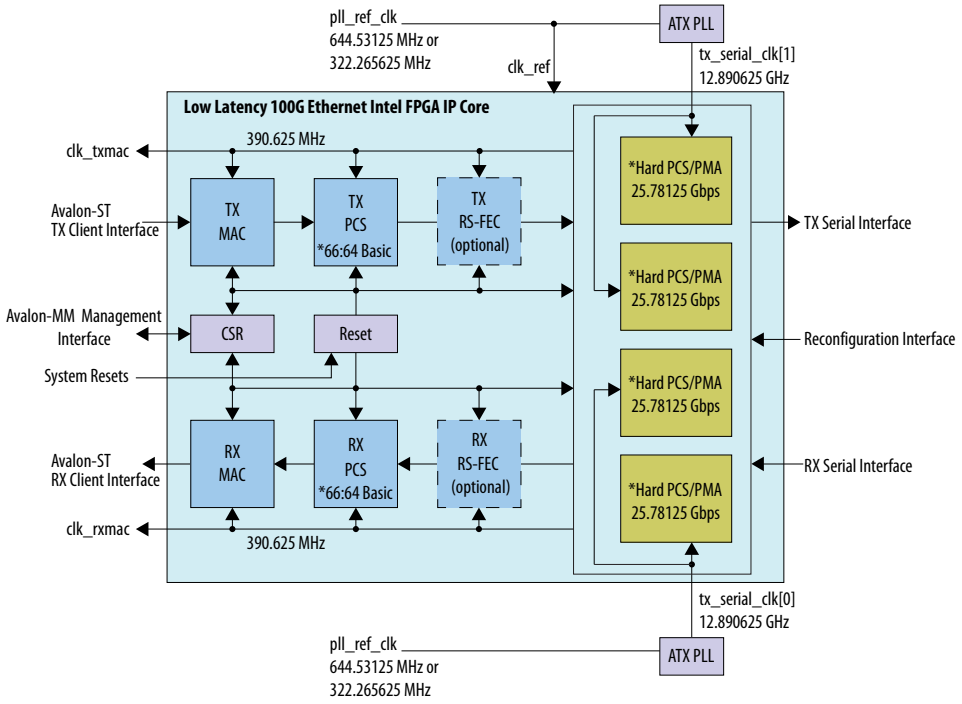
In the transmit direction, the MAC accepts client frames, and inserts inter-packet gap (IPG), preamble, start of frame delimiter (SFD), padding, and CRC bits before passing them to the PHY.The MAC also updates the TX statistics counters if they are present. The PHY encodes the MAC frame as required for 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 if they are present, 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.

4.1. High Level System Overview

Figure 5. Low Latency 100G Ethernet Intel FPGA IP Core

Main blocks, internal connections, and external block requirements.



*The IP core uses TX PCS and RX PCS to do 66:64 bit encoding/decoding. Hard PCS isused for other link related functions.

4.1.1. Low Latency 100G Ethernet Intel FPGA IP Core TX Datapath

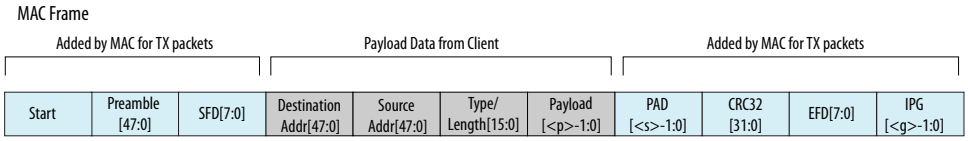
The TX MAC module receives the client payload data with the destination and source addresses and then adds, appends, or updates various header fields in accordance with the configuration specified. The MAC does not modify the destination address, the source address, or the payload received from the client. 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). The TX MAC module always inserts IDLE bytes to maintain an average IPG.

The Low Latency 100G Ethernet Intel FPGA IP core does not process incoming frames of less than nine bytes correctly. You must ensure such frames do not reach the TX client interface.

Figure 6. Typical Client Frame at the Transmit Interface

Illustrates the changes that the TX MAC makes to the client frame when **Enable preamble passthrough** is turned off. This figure uses the following notational conventions:

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



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

Preamble, Start, and SFD Insertion on page 23
Length/Type Field Processing on page 23
Frame Padding on page 23
Frame Check Sequence (CRC-32) Insertion on page 24
Inter-Packet Gap Adjustment on page 24
TX PCS on page 24
Error Insertion Test and Debug Feature on page 25

4.1.1.1. 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 Low Latency 100G Ethernet Intel FPGA IP 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.

4.1.1.2. 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.

4.1.1.3. 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 Low Latency 100G Ethernet Intel FPGA 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.

4.1.1.4. 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 Low Latency 100G Ethernet Intel FPGA 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 Low Latency 100G Ethernet Intel FPGA parameter editor. By default, the CRC insertion feature is enabled.

Related Information

Order of Transmission on page 32
Illustrations of the byte order and octet transmission order on the Avalon-ST client interface.

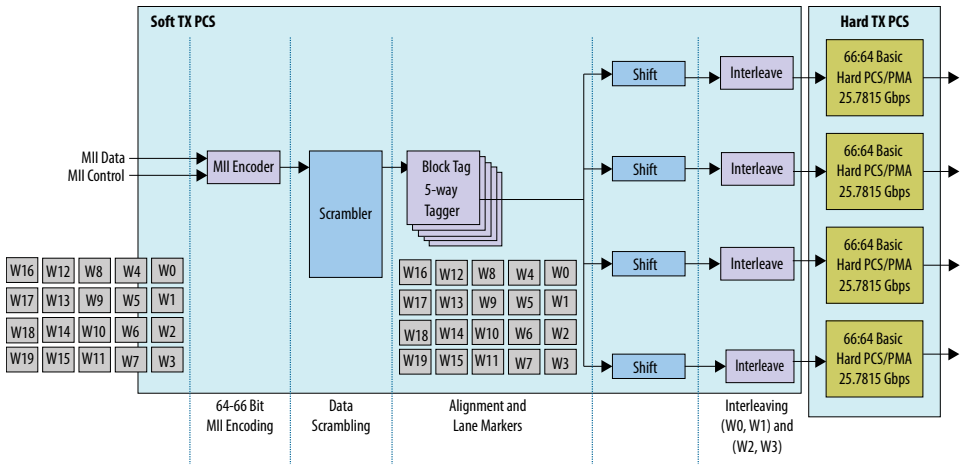
4.1.1.5. Inter-Packet Gap Adjustment

You can program the IPG adjustment to compensate for Alignment Marker insertion by the PHY by setting the number IDLE columns to be removed in the `IPG_COL_REM` register at offsets 0x406. By default, the IP core removes 20 IDLE columns in every Alignment Marker period (for 20 virtual lanes). You may set the this register to a larger value for clock compensation.

4.1.1.6. TX PCS

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

Figure 7. High Level Block Diagram of the Soft TX PCS



The Hard PCS and PMA blocks are configured in 66:64 bit basic generic 25G PCS mode. These blocks use FIFOs in elastic-buffer mode. The PMA operates at 25.78125 Gbps.

4.1.1.6.1. TX RSFEC

If you turn on **Enable RS-FEC** in the Low Latency 100G Ethernet Intel FPGA IP parameter editor, the IP core includes Reed-Solomon forward error correction (FEC) in both the receive and transmit datapaths.

The IP core implements Reed-Solomon FEC per Clause 91 of the IEEE Standard 802.3bj. The Reed-Solomon FEC algorithm includes the following modules:

- 64B/66B to 256B/257B Transcoding
- High-Speed RS-FEC(528,514) Reed-Solomon Encoder

When RS-FEC feature is enabled, the IP core instantiates an IOPLL to provide clock to the RS-FEC logic.

4.1.1.7. 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 Low Latency 100G Ethernet Intel FPGA 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 `l8_tx_error` signal in the EOP cycle of the packet.

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.

4.2. Low Latency 100G Ethernet Intel FPGA IP Core RX Datapath

The Low Latency 100G Ethernet Intel FPGA IP 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 8. 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).

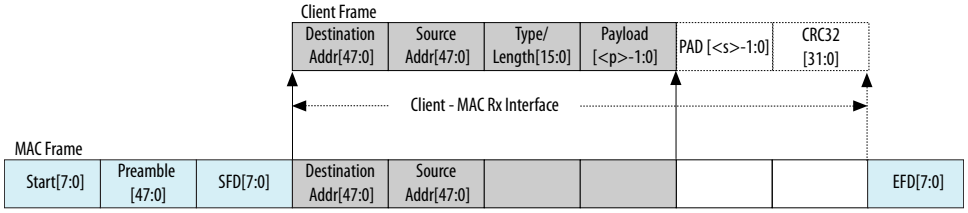
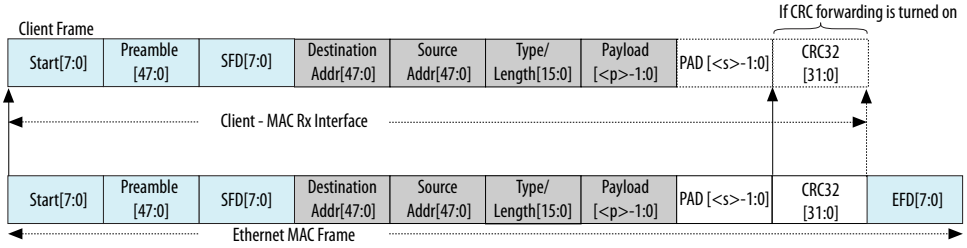


Figure 9. Flow of Frame Through the MAC RX With Preamble Pass-Through Turned On

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:

Low Latency 100G Ethernet Intel FPGA IP Core Preamble Processing on page 26
IP Core Strict SFD Checking on page 27
Low Latency 100G Ethernet Intel FPGA IP Core FCS (CRC-32) Removal on page 27
Low Latency 100G Ethernet Intel FPGA IP Core CRC Checking on page 27
Low Latency 100G Ethernet Intel FPGA IP Core Malformed Packet Handling on page 28
RX CRC Forwarding on page 28
Inter-Packet Gap on page 28
RX PCS on page 28

4.2.1. Low Latency 100G Ethernet Intel FPGA IP Core Preamble Processing

The preamble sequence is Start, six preamble bytes, and SFD. The Start byte must be on receive lane 0 (most significant byte). The IP core uses the Start byte (0xFB) to identify the preamble. The MAC RX looks for the Start, six preamble bytes and SFD, depending on the strict SFD checking settings of the IP core.

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 Low Latency 100G Ethernet Intel FPGA parameter editor, the MAC RX does not remove the eight-byte preamble sequence.

4.2.2. IP Core Strict SFD Checking

The Low Latency 100G Ethernet Intel FPGA IP core RX MAC checks all incoming packets for a correct Start byte (0xFB). If you turn on **Enable Strict SFD check** in the Low Latency 100G Ethernet Intel FPGA parameter editor, you enable the RX MAC to check the incoming preamble and SFD for the following values:

- SFD = 0xD5
- Preamble = 0x555555555555

The RX MAC checks one or both of these values depending on the values in bits [4:3] of the RXMAC_CONTROL register at offset 0x50A.

Table 10. Strict SFD Checking Configuration

Enable Strict SFD check	0x50A[4]: Preamble Check	0x50A[3]: SFD Check	Fields Checked	Behavior if Check Fails
Off	Don't Care	Don't Care	Start byte	IP core does not recognize a malformed Start byte as a Start byte
On	0	0	Start byte	
	0	1	Start byte and SFD	IP Core drops the packet
	1	0	Start byte and preamble	
	1	1	Start byte and preamble and SFD	

4.2.3. Low Latency 100G Ethernet Intel FPGA IP Core FCS (CRC-32) Removal

Independent user configuration register bits control FCS CRC removal at runtime. 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`) indicates the end of CRC bytes if CRC is not removed. When CRC is removed, the EOP signal indicates the final byte of payload.

4.2.4. Low Latency 100G Ethernet Intel FPGA 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 the RX MAC detects a CRC32 error, it marks the frame invalid by asserting `l8_rx_error[1]`.

4.2.5. Low Latency 100G Ethernet Intel FPGA IP Core Malformed Packet Handling

The malformed packet handling feature ensures the client receives the expected SOP-EOP sequences on the RX client interface. While receiving an incoming packet from the Ethernet link, the Low Latency 100G Ethernet Intel FPGA IP core expects to detect a terminate character at the end of the packet. When it detects an expected terminate character, the IP core generates an EOP on the client interface. However, sometimes the IP core detects an unexpected control character when it expects a terminate character. The Low Latency 100G Ethernet Intel FPGA IP core detects and handles the following forms of malformed packets:

- If the IP core detects an Error character, it generates an EOP and asserts a malformed packet error (`l8_rx_error[0]`). 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 (for example, an IDLE or Start character) when it is waiting for an EOP indication (terminate character), the IP core generates an EOP indication, asserts a malformed packet error (`l8_rx_error[0]`), and asserts a CRC error (`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 Low Latency 100G Ethernet Intel FPGA TX error insertion feature, the IP core identifies it as a malformed packet.

4.2.6. 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.

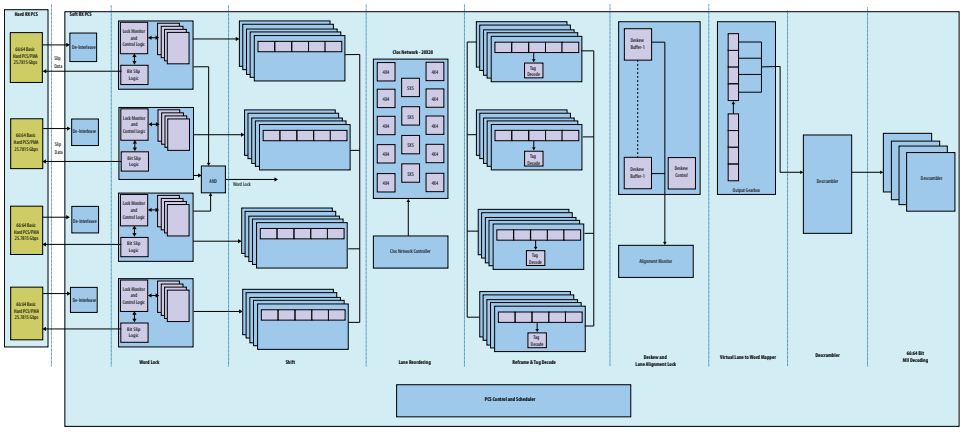
4.2.7. Inter-Packet Gap

The MAC RX removes all IPG octets received, and does not forward them to the client interface.

4.2.8. RX PCS

The soft RX PCS interfaces to the hard PCS and PMA blocks configured in 66:64 25G PCS Basic Generic Mode, with bitslip enabled. The hard PCS drives two, 66-bit output streams containing four virtual lanes to the soft RX PCS. The soft RX PCS implements word lock, lane reordering, descrambling, and MII decoding.

Figure 10. High Level Block Diagram of the Soft RX PCS



PCS Compliance

The Low Latency 100G Ethernet Intel FPGA IP RX PCS individual lock stages are designed to offer maximum compliance while reducing design resources. Hence, the design is not fully compliant to IEEE 802.3 Clause 82 specification. The non-compliance lock and unlock processes are listed in the following table.

Table 11. RX PCS Non-Compliance List

Lock and unlock processes which are not listed in this table are compliance to the IEEE 802.3 specification.

Process	Description
Virtual lane re-ordering lock	IEEE specification: The virtual lanes re-ordering is initiated when alignment lock gets acquired. Low Latency 100G Ethernet Intel FPGA IP core: The virtual lane reordering is initiated by block lock.
PCS lane deskew lock	IEEE specification: Deskew lock is acquired when the following conditions are met: • Alignment lock is acquired • Virtual lanes re-ordering completed • Lanes deskew completed Low Latency 100G Ethernet Intel FPGA IP core: The PCS lane deskew lock is initiated after virtual lanes reordering is complete.
PCS alignment lock	IEEE specification: Alignment lock is acquired when the following conditions are met: • Block lock is acquired • All virtual lanes alignment markers are received at proper alignment interval of 2^{14} words per virtual lanes for two consecutive cycles Low Latency 100G Ethernet Intel FPGA IP core: The alignment lock happens after PCS lane deskew lock is completed and interval frequency checking of individual markers for each virtual lane is complete.

4.2.8.1. RX RSFEC

If you turn on **Enable RS-FEC** in the Low Latency 100G Ethernet Intel FPGA 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

When RS-FEC feature is enabled, the IP core instantiates an IOPLL to provide clock to the RS-FEC logic.

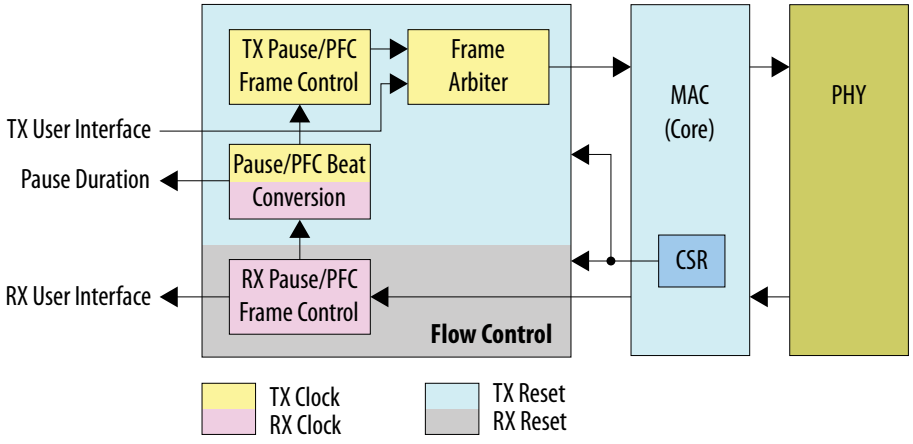
4.3. Flow Control

Flow control reduces congestion at the local or remote link partner. When either link partner experiences congestion, the respective transmit control sends pause frames. XOFF Pause frames stop the remote transmitter. XON Pause frames let the remote transmitter resume data transmission. Flow control supports both Pause and Priority Flow Control (PFC) control frames:

- IEEE 802.3 flow control—implements the IEEE 802.3 Annex 31B standard to manage congestion. This flow control is a mechanism to manage congestion at the local or remote partner. When the receiving device experiences congestion, it sends an XOFF pause frame to the emitting device to instruct the emitting device to stop sending data for a duration specified by the congested receiver. Data transmission resumes when the emitting device receives an XON pause frame (pause quanta = zero) or when the timer expires.
- Priority-based flow control (PFC)—implements the IEEE 802.1Qbb standard. PFC manages congestion based on priority levels. It supports up to 8 priority queues. When the receiving device experiences congestion on a priority queue, it sends a PFC frame requesting the emitting device to stop transmission on the priority queue for a duration specified by the congested receiver. When the receiving device is ready to receive transmission on the priority queue again, it sends a PFC frame instructing the emitting device to resume transmission on the priority queue.

Figure 11. Flow Control Module Conceptual Overview

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



Flow Control includes the following features:

- Pause or PFC frame generation and transmission:
  - Configurable selection of standard or priority-based flow control
  - Programmable 1- or 2-bit XON/XOFF request mode
  - In 2-bit request mode, programmable selection of register or signal-based control
  - Programmable per-queue XOFF frame separation
  - Programmable destination and source addresses in outgoing pause and PFC frames
  - Programmable pause and PFC quanta
- Client versus Pause or PFC frame transmission based on a priority-based arbitration scheme with frame-type indication for external downstream logic
- Stopping the next client frame transmission on the reception of a valid Pause frame
- Stopping the per queue client frame transmission on the reception of a valid PFC frame from the client. Includes per-queue PFC Pause quanta duration indicator
- Pause or PFC frame reception and decode:
  - Programmable destination address for filtering incoming pause and PFC frames
  - Configurable Pause or PFC per-queue enable, directing the IP core to ignore incoming pause frames on disabled queues
  - Per-queue client frame transmission pause duration indicator

4.3.1. TX Pause/PFC Flow Control Transmission

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

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

4.3.2. XON/XOFF Pause Frames

The sender transmits a PFC frame with the specified PFC pause quanta value when it receives an XOFF request. If an enabled priority queue is in the XOFF condition, a new PFC frame is transmitted after the minimum time gap. You specify the minimum time gap in the per priority queue `TX Flow Control Signal XOFF Request Hold Quanta` register. The minimum time gap between two consecutive PFC frames is 1 pause quanta or 512-bit times. PFC frame transmission ends when none of the PFC interfaces of all enabled priority queues is requesting PFC frames.

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

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

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

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

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

4.4. 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.

4.4.1. Order of Transmission

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

Figure 12. 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 6 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 13. 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.

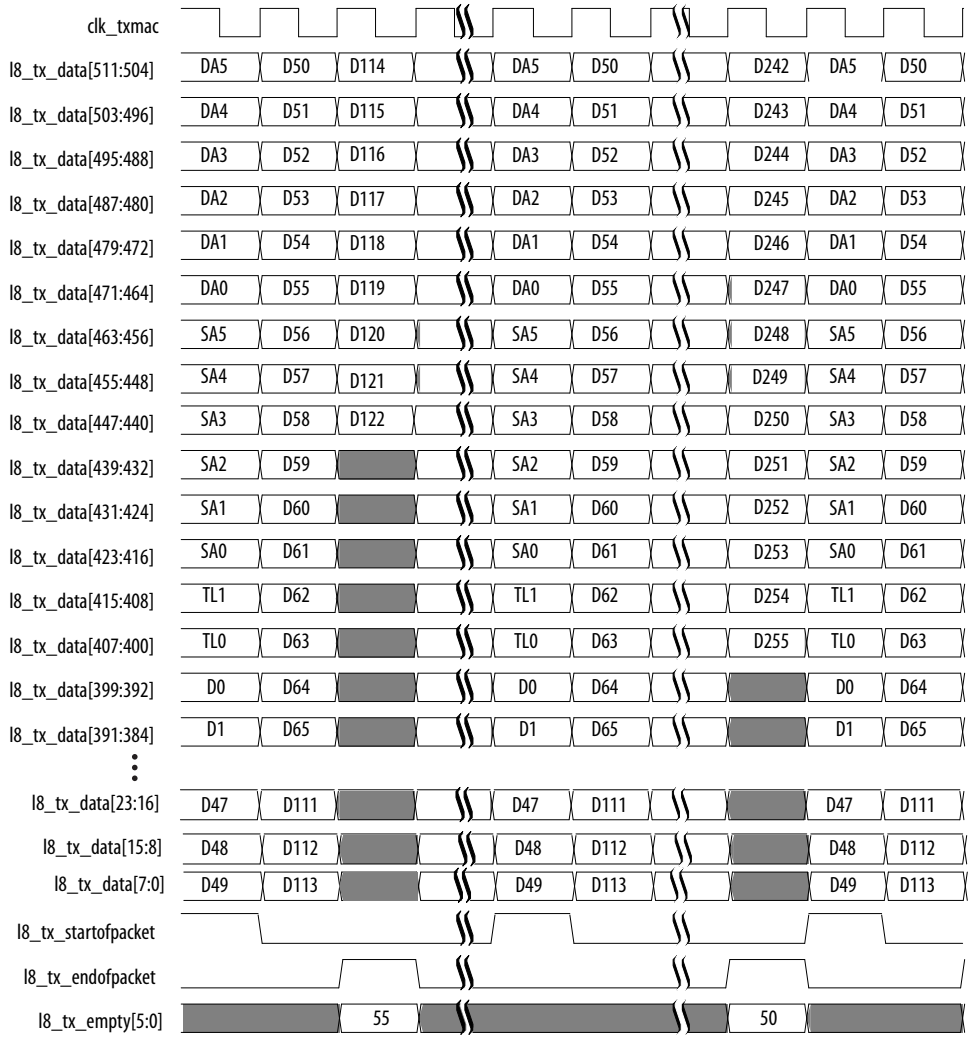


Figure 14. Byte Order on the Avalon-ST Interface Lanes With Preamble Pass-Through

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

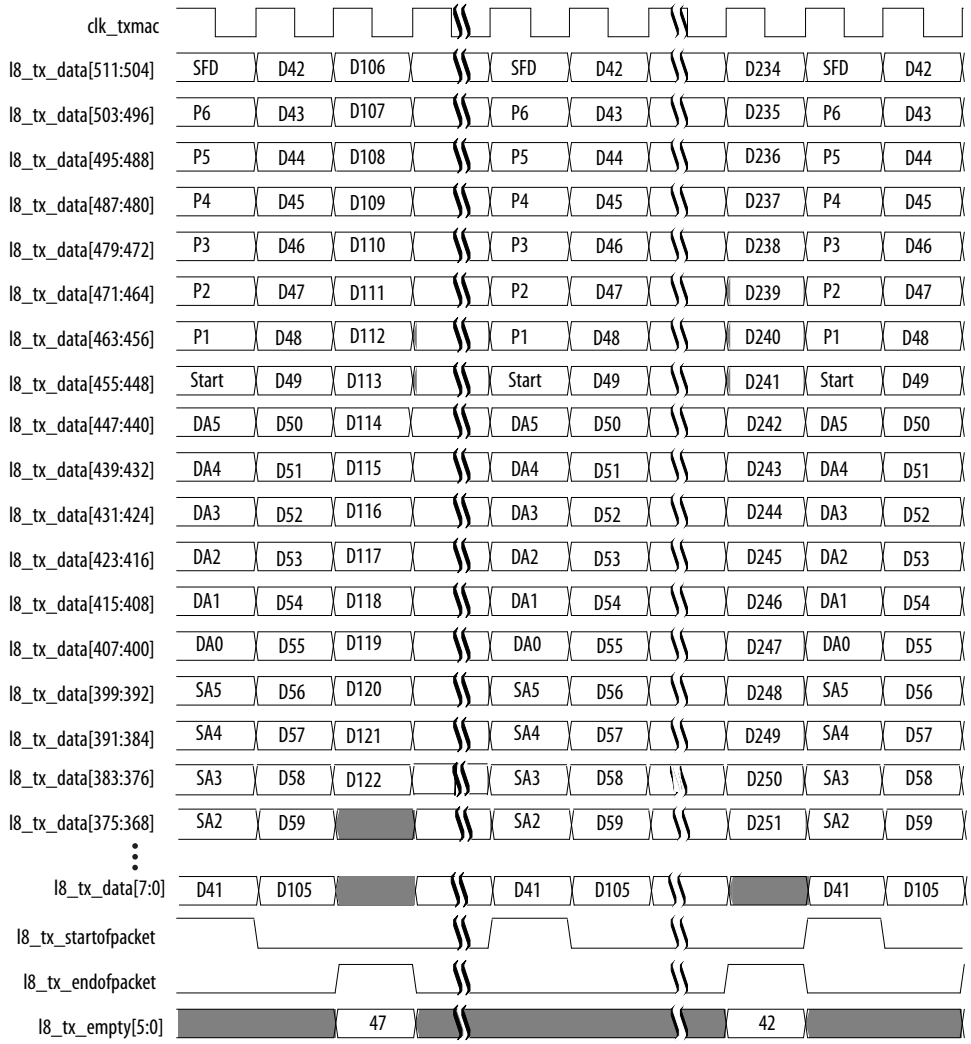
Destination Address[40] is the broadcast/multicast bit (a type bit), and Destination Address[41] is a locally administered address bit.

	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 15. 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.



4.4.2. Bit Order For TX and RX Datapaths

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

Related Information

IEEE website

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

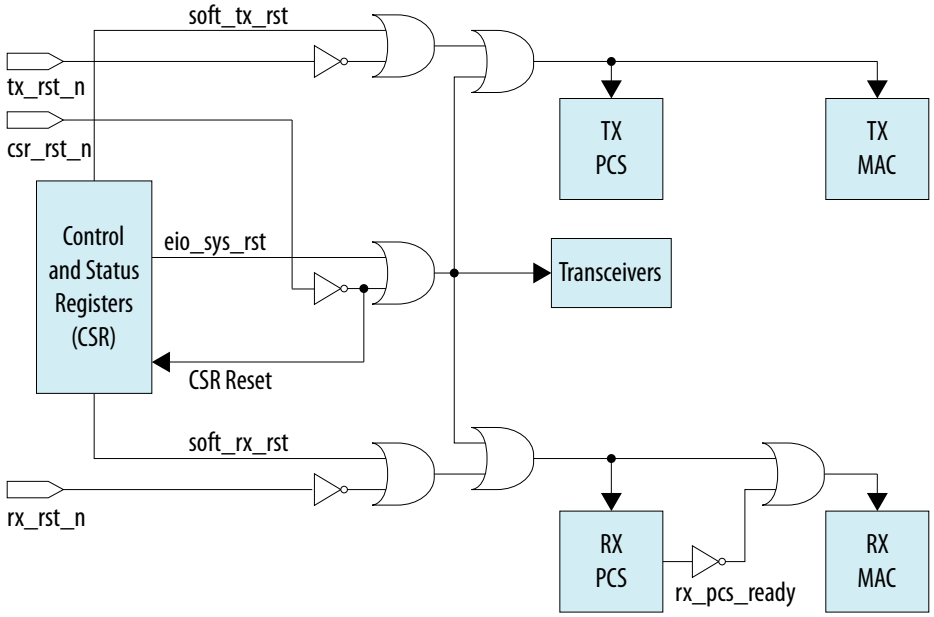
5. Reset

Control and Status registers control three parallel soft resets. These soft resets are not self-clearing. Software clears them by writing to the appropriate register. In addition, the IP core has three hard reset signals.

Asserting the external hard reset `csr_rst_n` returns all Control and Status registers to their original values, including the statistics counters. An additional dedicated reset signal resets the transceiver reconfiguration interface.

Figure 16. Conceptual Overview of General IP Core Reset Logic

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



The general reset signals reset the following functions:

- `soft_tx_rst`, `tx_rst_n`: Resets the IP core in the TX direction. Resets the TX PCS, TX MAC, and digital portion of the transceiver. This reset leads to deassertion of the `tx_lanes_stable` output signal.
- `soft_rx_rst`, `rx_rst_n`: Resets the IP core in the RX direction. Resets the RX PCS and RX MAC. This reset leads to deassertion of the `rx_pcs_ready` output signal.
- `sys_rst`, `csr_rst_n`: Resets the IP core. Resets the TX and RX MACs, PCS, and transceivers.

  Note: `csr_rst_n` resets the Control and Status registers, including the statistics counters. `sys_rst` does not reset any Control and Status registers.

  This reset leads to deassertion of the `tx_lanes_stable` and `rx_pcs_ready` output signals.

In addition, the synchronous `reconfig_reset` signal resets the IP core transceiver reconfiguration interface, an Avalon-MM interface. Associated clock is the `reconfig_clk`, which clocks the transceiver reconfiguration interface.

System Considerations

You should perform a system reset before beginning IP core operation, preferably by asserting the `csr_rst_n` signal.

If you assert the transmit reset when the downstream receiver is already aligned, the receiver loses alignment. Before the downstream receiver loses lock, it might receive some malformed frames.

If you assert the receive reset while the upstream transmitter is sending packets, the packets in transit are corrupted.

If the ATX PLL loses lock, the IP core forces a transmit side reset. After the ATX PLL acquires lock, the IP core deasserts the transmit reset.

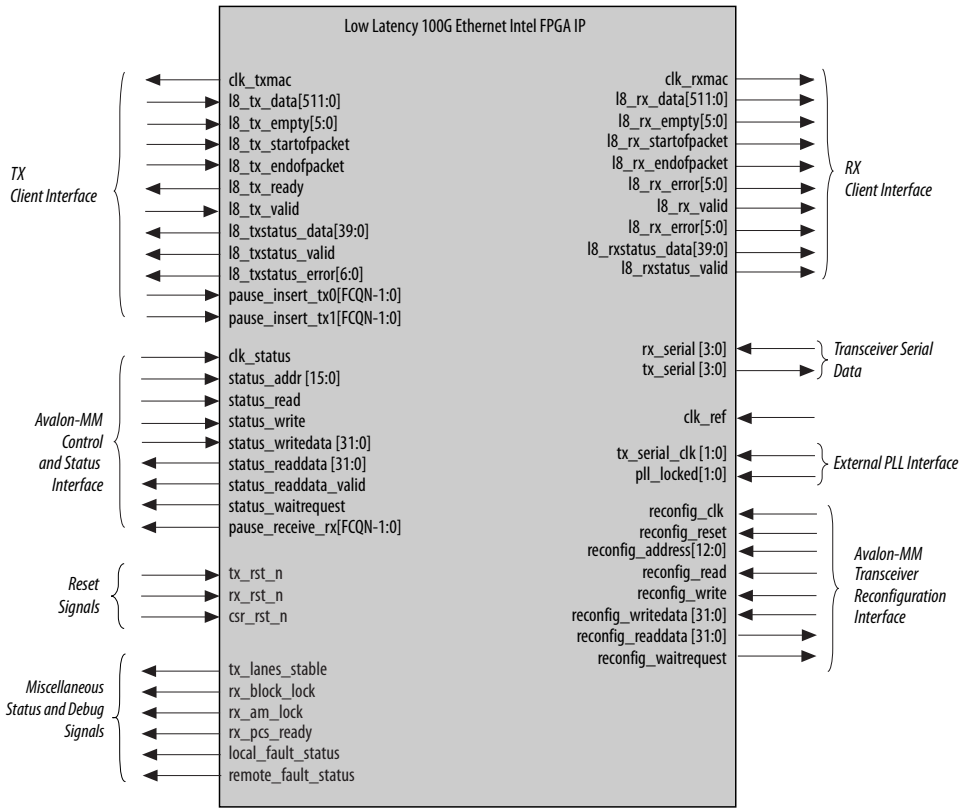
If the IP core suffers loss of signal on the serial links, it asserts the receive reset.

Related Information

- Reset Signals on page 45
  Information about the required behavior of the hard reset signals. To trigger a reset, you must assert the desired reset signal at least ten clock cycles.
- PHY Registers on page 47
  Information about the reset fields in the PHY_CONFIG register at offset 0x310.

6. Interfaces and Signal Descriptions

Figure 17. Low Latency 100G Ethernet Intel FPGA Signals and Interfaces



6.1. TX MAC Interface to User Logic

The Low Latency 100G Ethernet Intel FPGA IP core TX client interface 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 cycle.

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

Table 12. Signals of the Avalon-ST TX Client Interface

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

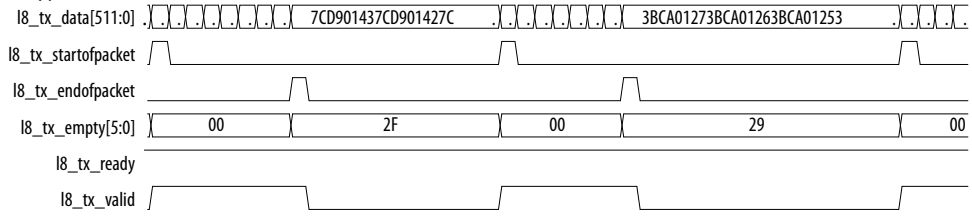
Signal Name	Direction	Description
`clk_txmac`	Output	The TX clock for the IP core is `clk_txmac`. The frequency of this clock is 390.625 MHz.
`l8_tx_data[511:0]`	Input	TX data. If the preamble pass-through feature is enabled, data begins with the preamble. The Low Latency 100G Ethernet Intel FPGA IP core does not process incoming frames of less than nine bytes correctly. You must ensure such frames do not reach the TX client interface. 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 cycle, 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 mode. 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

continued...

Signal Name	Direction	Description
		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.
`l8_txstatus_valid`	Output	When asserted, indicates that `l8_txstatus_data` and `l8_txstatus_error[6:0]` are driving valid data.
`l8_txstatus_data[39:0]`	Output	Specifies information about the transmit frame. The following fields are defined: • Bit[39]: When asserted, indicates a PFC frame • Bit[38]: When asserted, indicates a unicast frame • Bit[37]: When asserted, indicates a multicast frame • Bit[36]: When asserted, indicates a broadcast frame • Bit[35]: When asserted, indicates a pause frame • Bit[34]: When asserted, indicates a control frame • Bit[33]: When asserted, indicates a VLAN frame • Bit[32]: When asserted, indicates a stacked VLAN frame • Bits[31:16]: Specifies the frame length from the first byte of the destination address to the last bye of the FCS • Bits[15:0]: Specifies the payload length
`l8_txstatus_error[6:0]`	Output	Specifies the error type in the transmit frame. The following fields are defined: • Bits[6:3]: Reserved • Bit[2]: Payload length error • Bit[1]: Oversized frame • Bit[0]: Reserved. This signal is valid when `l8_txstatus_valid` is asserted.

Figure 18. Traffic on the TX Avalon-ST Client Interface forLow Latency 100G Ethernet Intel FPGA IP

Shows typical traffic for the TX client interface of the IP core.



Related Information

Avalon Interface Specifications

For more information about the Avalon-ST interface.

6.2. RX MAC Interface to User Logic

The Low Latency 100G Ethernet Intel FPGA 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.

Table 13. Signals of the Avalon-ST RX Client Interface

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

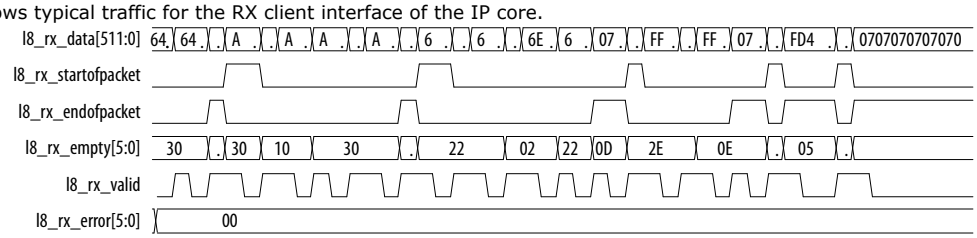
Name	Direction	Description
`clk_rxmac`	Output	The RX clock for the IP core is `clk_rxmac`. The IP core recovers this clock from the incoming data. This clock is guaranteed stable when `rx_pcs_ready` is asserted. The frequency of this clock is 390.625 MHz.
`l8_rx_data[511:0]`	Output	RX data. Bit 511 is the MSB and bit 0 is the LSB. Bytes are read in the usual left to right order. The IP core reverses the byte order to meet the requirements of the Ethernet standard.
`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. 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. • 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 RXMAC_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. • Bit [5]: Reserved.

continued...

Name	Direction	Description
`l8_rx_valid`	Output	When asserted, indicates that RX data is valid. Only valid between the `l8_rx_startofpacket` and `l8_rx_endofpacket` signals. This signal might be deasserted between the assertion of `l8_rx_startofpacket` and `l8_rx_endofpacket`.
`l8_rxstatus_valid`	Output	When asserted, indicates that `l8_rxstatus_data` is driving valid data. This signal behaves identically to the l8_rx_endofpacket signal.
`l8_rxstatus_data[39:0]`	Output	Specifies information about the received frame. The following fields are defined: • Bit[39]: When asserted, indicates a PFC frame • Bit[38]: When asserted, indicates a unicast frame • Bit[37]: When asserted, indicates a multicast frame • Bit[36]: When asserted, indicates a broadcast frame • Bit[35]: When asserted, indicates a pause frame • Bit[34]: When asserted, indicates a control frame • Bit[33]: When asserted, indicates a VLAN frame • Bit[32]: When asserted, indicates a stacked VLAN frame • Bits[31:16]: Specifies the frame length from the first byte of the destination address to the last bye of the FCS • Bits[15:0]: Specifies the payload length

Figure 19. Traffic on the RX Avalon-ST Client Interface for Low Latency 100G Ethernet Intel FPGA IP

Shows typical traffic for the RX client interface of the IP core.



Related Information

- RX MAC Registers on page 53
- Avalon Interface Specifications
  For more information about the Avalon-ST interface.

6.3. Transceivers

The transceivers implement four CAUI-4 physical lanes at 25.78125 MHz and require two separately instantiated advanced transmit (ATX) PLLs to generate the high speed serial clocks. On Intel Stratix 10 devices, only the ATX PLL supports the required data rate.

Table 14. Transceiver Signals

Signal	Direction	Description
`tx_serial[3:0]`	Output	TX transceiver data. Each `tx_serial` bit becomes two physical pins that form a differential pair.
`rx_serial[3:0]`	Input	RX transceiver data. Each `rx_serial` bit becomes two physical pins that form a differential pair.

continued...

Signal	Direction	Description
clk_ref	Input	The input clock clk_ref is the reference clock for the transceiver RX CDR PLL and the RS-FEC PLLs. This clock must have a frequency of 644.53125 or 322.265625 MHz with a ±100 ppm accuracy per the *IEEE 802.3ba-2010 Ethernet Standard.* In addition, clk_ref must meet the jitter specification of the *IEEE 802.3ba-2010 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.
tx_serial_clk[1:0]	Input	High speed serial clocks driven by the two ATX PLLs. The frequency of these clocks is 12.890625 GHz. You must drive these clocks from the two ATX PLLs that you configure separately from the Low Latency 100G Ethernet Intel FPGA IP core.
tx_pll_locked[1:0]	Input	Lock signals from the two ATX PLLs. Each bit indicates the corresponding ATX PLL is locked.

Related Information

- Intel Stratix 10 L- and H-Tile Transceiver PHY User Guide
  Information about the Intel Stratix 10 ATX PLL and Native PHY IP core.
- Intel Stratix 10 GX 2800 L-Tile ES-1Transceiver PHY User Guide
  Information about the Stratix 10 L-tile ES1 ATX PLL and Native PHY IP Core.

6.4. Transceiver Reconfiguration Signals

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

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

Table 15. Reconfiguration Interface Ports with Shared Native PHY Reconfiguration Interface

All interface signals are clocked by the reconfig_clk clock. These signals are provided by a four-channel Intel Stratix 10 Native PHY IP core embedded in the Low Latency 100G Ethernet Intel FPGA IP core.

Port Name	Direction	Description
reconfig_clk	Input	Avalon® clock. The clock frequency is 100-162 MHz. All signals transceiver reconfiguration interface signals are synchronous to reconfig_clk.
reconfig_reset	Input	Resets the Avalon-MM interface and all of the registers to which it provides access.
reconfig_write	Input	Write request signal. Signal is active high.
reconfig_read	Input	Read request signal. Signal is active high.
reconfig_address[13:0]	Input	Address bus. Refer to the Refer to the *Intel Stratix 10 L- and H-Tile Transceiver PHY User Guide* for information about the address fields for the Native PHY four-channel configuration.

continued...

Port Name	Direction	Description
`reconfig_writedata[31:0]`	Input	A 32-bit data write bus. `reconfig_address` specifies the address.
`reconfig_readdata[31:0]`	Output	A 32-bit data read bus. Drives read data from the specified address. Signal is valid after `reconfig_waitrequest` is deasserted.
`reconfig_waitrequest`	Output	Indicates the Avalon-MM interface is busy. Keep the `reconfig_write` or `reconfig_read` asserted until `reconfig_waitrequest` is deasserted.

Related Information

- Intel Stratix 10 L- and H-Tile Transceiver PHY User Guide
  Provides more information about the transceiver reconfiguration interface in H-tile devices, including timing diagrams for reads and writes.
- Intel Stratix 10 GX 2800 L-Tile ES-1Transceiver PHY User Guide
  Information about the Stratix 10 L-tile ES1 ATX PLL and Native PHY IP Core.

6.5. Avalon-MM Management Interface

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

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

Table 16. Avalon-MM Management Interface

Signal	Direction	Description
`clk_status`	Input	The clock that drives the control and status registers. The frequency of this clock is 100-162 MHz.
`status_addr[15:0]`	Input	Drives the Avalon-MM register address.
`status_read`	Input	When asserted, specifies a read request.
`status_write`	Input	When asserted, specifies a write request.
`status_readdata[31:0]`	Output	Drives read data. Valid when `status_readdata_valid` is asserted.
`status_readdata_valid`	Output	When asserted, indicates that `status_read_data[31:0]` is valid.
`status_writedata[31:0]`	Input	Drives the write data.
`status_waitrequest`	Output	Indicates that the control and status interface is not ready to complete the transaction. `status_waitrequest` is only used for read transactions.

Related Information

- Registers on page 47
- Typical Read and Write Transfers section in the *Avalon Interface Specifications*
  Describes typical Avalon-MM read and write transfers with a slave-controlled waitrequest signal.

6.6. Miscellaneous Status and Debug Signals

The miscellaneous status and debug signals are asynchronous.

Table 17. Miscellaneous Status and Debug Signals

Signal	Direction	Description
`tx_lanes_stable`	Output	Asserted when all four physical TX lanes are stable and ready to transmit data.
`rx_block_lock`	Output	Asserted when all 20 virtual lanes have identified 66-bit block boundaries in the serial data stream.
`rx_am_lock`	Output	Asserted when all 20 incoming virtual lanes have been ordered.
`rx_pcs_ready`	Output	Asserted when the RX lanes are fully aligned and ready to receive data.

6.7. Reset Signals

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

Table 18. Reset Signals

Signal	Direction	Description
`tx_rst_n`	Input	Active low hard reset signal. Resets the TX interface, including the TX PCS and TX MAC. This reset leads to the deassertion of the `tx_lanes_stable` output signal.
`rx_rst_n`	Input	Active low hard reset signal. Resets the RX interface, including the RX PCS and RX MAC. This reset leads to the deassertion of the `rx_pcs_ready` output signal.
`csr_rst_n`	Input	Active low hard global reset. Resets the full IP core. Resets the TX MAC, RX MAC, TX PCS, RX PCS,transceivers, and control and status registers. This reset leads to the deassertion of the `tx_lanes_stable` and `rx_pcs_ready` output signals.

Related Information

- Reset on page 35
- PHY Registers on page 47
  Information about the reset fields in the `PHY_CONFIG` register at offset 0x310.

6.8. Clocks

You must set the transceiver reference clock (`clk_ref`) frequency to a value that the IP core supports. The Low Latency 100G Ethernet Intel FPGA IP core supports a `clk_ref` frequency of 644.53125 MHz or 322.265625 MHz ±100 ppm. The ±100ppm value is required for any clock source providing the transceiver reference clock.

Table 19. Clock Inputs

Describes the input clocks that you must provide.

Signal Name	Description
`clk_ref`	The input clock `clk_ref` is the reference clock for the transceiver RX CDR PLL and the RS-FEC PLLs. This clock must have a frequency of 644.53125 MHz with a ±100 ppm accuracy per the *IEEE 802.3ba-2010 Ethernet Standard.* In addition, `clk_ref` must meet the jitter specification of the *IEEE 802.3ba-2010 Ethernet Standard*.

continued...

Signal Name	Description
	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.
`tx_serial_clk[1:0]`	These two input clocks are part of the external PLL interface. The IP core fans out each clock to target two of the four transceiver PHY links. You must drive these clocks from two ATX PLLs that you configure separately from the Low Latency 100G Ethernet Intel FPGA IP core. The required frequency is 12.890625 GHz.
`clk_status`	Clocks the control and status interface. `clk_status` is expected to be a 100–162 MHz clock.
`reconfig_clk`	Clocks the transceiver reconfiguration interface.`reconfig_clk` is expected to be a 100–162 MHz clock.

Table 20. 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. This clock is guaranteed stable when `tx_lanes_stable` is high.
`clk_rxmac`	The RX clock for the IP core is `clk_rxmac`. The RX MAC clock frequency is 390.625 MHz. 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.

Related Information

- Adding the Transceiver PLLs on page 15
  Information about configuring and connecting the external PLLs. Includes signal descriptions.
- Intel Stratix 10 Device Datasheet
  Provides transceiver reference clock phase noise specifications.

7. Registers

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

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

Table 21. Low Latency 100G Ethernet Intel FPGA IP Core Register Map Overview

Lists the main ranges of the memory mapped registers for the Low Latency 100G Ethernet Intel FPGA IP core. Addresses are word addresses.

Word Offset	Register Category
0x300–0x3FF	PHY registers
0x400–0x4FF	TX MAC registers
0x500–0x5FF	RX MAC registers
0x800–0x8FF	TX statistics counters
0x900–0x9FF	RX statistics counters
0xC00–0xCFF	TX RS-FEC registers
0xD00–0xDFF	RX RS-FEC registers

Related Information

Avalon-MM Management Interface on page 44

7.1. PHY Registers

Table 22. PHY Registers

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

Addr	Name	Description	Reset	Access
0x300	`REVID`	IP core PHY module revision ID	0x0809 2017	RO
0x301	`SCRATCH`	Scratch register available for testing	0x0000 0000	RW
0x302	`PHY_NAME_0`	First characters of IP core variation identifier string, "100".	0x0031 3030	RO

continued...

Addr	Name	Description	Reset	Access
0x303	PHY_NAME_1	Next characters of IP core variation identifier string, "GE".	0x0000 4745	RO
0x304	PHY_NAME_2	Final characters of IP core variation identifier string, "pcs".	0x0070 6373	RO
0x310	PHY_CONFIG	PHY configuration registers. The following bit fields are defined: • Bit[0]: : eio_sys_rst. Full system reset (except registers). Set this bit to initiate the internal reset sequence. • Bit[1]: soft_txp_rst. TX soft reset. • Bit[2]: soft_rxp_rst. RX soft reset. • Bit[3]: Reserved. • Bit[4]: set_ref_lock. Directs the RX CDR PLL to lock to the reference clock. • Bit[5]: set_data_lock. Directs the RX CDR PLL to lock to data. • Bits[31:6]: Reserved. The reset bits are not self-clearing. To force a reset, you can set and reset a reset bit in back-to-back register write operations.	26'hX_2'b0_1'bX_3'b0 (X= don't care)	RW
0x312	WORD_LOCK	Each of the 20 lower order bits, when asserted, indicates that the corresponding virtual channel has identified 66 bit block boundaries in the serial data stream. If **Enable RS-FEC** is turned on, the value is always zero.	0xXXX0 0000 (X= don't care)	RO
0x313	EIO_SLOOP	Serial PMA loopback. Setting a bit puts the corresponding transceiver in serial loopback mode. In serial loopback mode, the TX lane loops back to the RX lane on an internal loopback path.	0xXXXX XXX0	RW
0x314	EIO_FLAG_SEL	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 EIO_FLAGS register. The value in the EIO_FLAG_SEL register directs the IP core to make available the following FIFO flag: • 3'b000: TX FIFO full • 3'b001: TX FIFO empty • 3'b010: TX FIFO partially full • 3'b011: TX FIFO partially empty • 3'b100: RX FIFO full • 3'b101: RX FIFO empty • 3'b110: RX FIFO partially full • 3'b111: RX FIFO partially empty	29'bX_3'b000	RW
0x315	EIO_FLAGS	PCS indirect data. To read a FIFO flag, set the value in the EIO_FLAG_SEL register to indicate the flag you want to read. After you specify the flag in the EIO_FLAG_SEL register, each bit [n] in the EIO_FLAGS register has the value of that FIFO flag for the transceiver channel for lane [n].	0xXXXX XXX0	RO

continued...

Addr	Name	Description	Reset	Access
0x321	EIO_FREQ_LOCK	Each of the lower order four bits, when asserted, indicates that the corresponding lane RX clock data recovery (CDR) phase-locked loop (PLL) is locked.	0xXXXX XXX0	RO
0x322	PHY_CLK	The following encodings are defined: • Bit[0]: If set to 1, indicates the TX transceivers have completed reset. • **Enable RS-FEC** is turned on, that the FEC TX PLL has acquired frequency lock. • Bit[2] : If set to 1, indicates the RX core clock is stable and if Bit [1] : If set to 1, indicates the TX core clock is stable and if **Enable RS-FEC** is turned on, that the FEC RX PLL has acquired frequency lock.	29'bX_3'b000	RO
0x323	FRM_ERR	Each of the 20 lower order bits, when asserted, indicates that the corresponding virtual lane has a frame error. You can read this register to determine if the IP core sustains a low number of frame errors, below the threshold to lose word lock. These bits are sticky, unless the virtual lane loses word lock. Write 1'b1 to the SCLR_FRM_ERR register to clear. If a virtual lane loses word lock, it clears the corresponding register bit. Each bit in this register has a valid value only if the corresponding bit in the WORD_LOCK register at offset 0x312 has the value of 1. If **Enable RS-FEC** is turned on, the value is always zero.	0xXXX0 0000	RO
0x324	SCLR_FRM_ERR	Synchronous clear for FRM_ERR register. Write 1'b1 to this register to clear the FRM_ERR register and bit [1] of the LANE_DESKEWED register. A single bit clears all sticky framing errors. This bit does not auto-clear. Write a 1'b0 to continue logging frame errors.	0x0000 0000	RW
0x325	EIO_RX_SOFT_PURGE_S	Set bit [0] to clear the RX FIFO for all four physical lanes.	31'bX_1'b0	RW
0x326	RX_PCS_FULLY_ALIGNED_S	Indicates the RX PCS is fully aligned and ready to accept traffic. • Bit[0]: RX PCS fully aligned status. • Bit[1]: RX PCS HI BER status. If **Enable RS-FEC** is turned on, the value is always zero.	30'bX_2'b00	RO
0x327	ERR_INJ	When set to 1, injects an error in the corresponding lane. The register is rising-edge triggered. Write a 0 to clear.	0xXXXX XXX0	RW
0x328	AM_LOCK	When bit [0] is asserted, indicates that the IP core has identified virtual lane alignment markers in the data stream of all 20 virtual lanes, and has ordered the virtual lanes. If **Enable RS-FEC** is turned on, the value is always zero.	0xXXXX XXX0	RO
0x329	LANE_DESKEWED	The following encodings are defined:	30'bX_2'b00	RO

continued...

Addr	Name	Description	Reset	Access
		• Bit [0]: Indicates all lanes are deskewed. • Bit [1]: When asserted indicates a change in lanes deskewed status. To clear this sticky bit, write 1'b1 to the corresponding bit of the SCLR_FRM_ERR register. This is a latched signal. If **Enable RS-FEC** is turned on, the value is always zero.		
0x330	PCS_VLANE0	PCS virtual lane mapping. Identifies the five virtual lanes detected on physical lane 0. Virtual lanes are encoded with the five-bit binary virtual lane number. One virtual lane index is encoded in register bits [4:0], another in register bits [9:5], another in register bits [14:10], another in register bits [19:15], and another in register bits [24:20]. For example, if the value of the register is 25'b00001_00101_00011_00000_01000, virtual lanes 0, 1, 3, 5, and 8 map to physical lane 0. The value 0x1F in any of these fields indicates no virtual lane is recorded yet. Before the IP core asserts `rx_pcs_ready`, transitional values can appear in the register fields. Therefore, you should read the register three to four times to ensure you read the correct virtual lane indicators. If **Enable RS-FEC** is turned on, the value remains at the reset value.	0x01FF FFFF	RO
0x331	PCS_VLANE1	PCS virtual lane mapping for physical lane 1. The value 0x1F in any of these fields indicates no virtual lane is recorded yet. Before the IP core asserts `rx_pcs_ready`, transitional values can appear in the register fields. Therefore, you should read the register three to four times to ensure you read the correct virtual lane indicators. If **Enable RS-FEC** is turned on, the value remains at the reset value.	0x01FF FFFF	RO
0x332	PCS_VLANE2	PCS virtual lane mapping for physical lane 2. The value 0x1F in any of these fields indicates no virtual lane is recorded yet. Before the IP core asserts `rx_pcs_ready`, transitional values can appear in the register fields. Therefore, you should read the register three to four times to ensure you read the correct virtual lane indicators. If **Enable RS-FEC** is turned on, the value remains at the reset value.	0x01FF FFFF	RO
0x333	PCS_VLANE3	PCS virtual lane mapping for physical lane 3. The value 0x1F in any of these fields indicates no virtual lane is recorded yet. Before the IP core asserts `rx_pcs_ready`, transitional values can appear in the register fields. Therefore, you should read the register three to four times to ensure you read the correct virtual lane indicators. If **Enable RS-FEC** is turned on, the value remains at the reset value.	0x01FF FFFF	RO

continued...

Addr	Name	Description	Reset	Access
0x340	KHZ_REF	Reference clock frequency in KHz, assuming the `clk_status` clock is 100 MHz. The reference clock frequency is the value in this register times the frequency of the `clk_status` clock, divided by 100.	0x0000 0000	RO
0x341	KHZ_RX	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 this register times the frequency of the `clk_status` clock, divided by 100.	0x0000 0000	RO
0x342	KHZ_TX	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 this register times the frequency of the `clk_status` clock, divided by 100.	0x0000 0000	RO
0x343	KHZ_TX_RS	FEC TX clock (`clk_tx_rs`) frequency in KHz, assuming the `clk_status` clock has the frequency of 100 MHz. The TX FEC clock frequency is the value in this register times the frequency of the `clk_status` clock, divided by 100. This register is available only if **Enable RS-FEC** is turned on.	0x0000 0000	RO
0x344	KHZ_RX_RS	FEC RX clock (`clk_rx_rs`) frequency in KHz, assuming the `clk_status` clock has the frequency of 100 MHz. The RX FEC clock frequency is the value in this register times the frequency of the `clk_status` clock, divided by 100. This register is available only if **Enable RS-FEC** is turned on.	0x0000 0000	RO

7.2. TX MAC Registers

Table 23. TX MAC Registers

Addr	Name	Description	Reset	Access
0x400	TXMAC_REVID	TX MAC revision ID.	0x0809 2017	RO
0x401	TXMAC_SCRATCH	Scratch register available for testing.	0x0000 0000	RW
0x402	TXMAC_NAME_0	First 4 characters of IP core variation identifier string, "100g".	0x3130 3067	RO
0x403	TXMAC_NAME_1	Next 4 characters of IP core variation identifier string, "MACT".	0x4D41 4354	RO
0x404	TXMAC_NAME_2	Final 4 characters of IP core variation identifier string, "xCSR".	0x7843 5352	RO

continued...

Addr	Name	Description	Reset	Access
0x405	LINK_FAULT	Link Fault Configuration Register. The following bits are defined: • `Force Remote Fault` bit[3]: When link fault generation is enabled, stops data transmission and forces transmission of a remote fault. • `Disable Remote Fault` bit[2]: When both link fault reporting and unidirectional transport are enabled, the core transmits data and does not transmit remote faults (RF). This bit takes effect when the value of this register is 28'hX4'b0111. • `Unidir Enable` bit[1]: When asserted, the core includes Clause 66 support for the remote link fault reporting on the Ethernet link. • `Link Fault Reporting Enable` bit[0]: The following encodings are defined: — 1'b1: The PCS generates the proper fault sequence on Ethernet link, when conditions are met. — 1'b0: The PCS does not generate the fault sequence.	28'hX_4'b0001	RW
0x406	IPG_COL_REM	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. The default value is 20 (decimal).. Bits [31:8] of this register are Reserved.	0xXXXX XX14	RW
0x407	MAX_TX_SIZE_CONFIG	Specifies the maximum TX frame length. Frames that are longer are considered oversized. However, the IP core does transmit them. 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 statistics registers, the IP core increments the 64-bit CNTR_TX_OVERSIZE counter. The minimum value of this register is 64 (decimal). Bits [31:16] of this register are Reserved.	0xXXXX 2580	RW
0x40A	TX_MAC_CONTROL	TX MAC Control Register. A single bit is defined: • Bit [1] – VLAN detection disabled. This bit is deasserted by default, implying VLAN detection is enabled.	30'hX2'b0X	RW

7.3. RX MAC Registers

Table 24. RX MAC Registers

Addr	Name	Description	Reset	Access
0x500	RXMAC_REVID	RX MAC revision ID.	0x0809 2017	RO
0x501	RXMAC_SCRATCH	Scratch register available for testing.	0x0000 0000	RW
0x502	RXMAC_NAME_0	First 4 characters of IP core variation identifier string, "100g".	0x3130 3067	RO
0x503	RXMAC_NAME_1	Next 4 characters of IP core variation identifier string, "MACR".	0x4D41 4352	RO
0x504	RXMAC_NAME_2	Final 4 characters of IP core variation identifier string, "xCSR".	0x7843 5352	RO
0x506	RXMAC_SIZE_CONFIG	Specifies the maximum frame length available. The MAC asserts `l8_rx_error[3]` when the length of the received frame exceeds the value of this 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 statistics registers, the IP core increments the 64-bit `CNTR_RX_OVERSIZE` counter. The minimum value of this register is 64 (decimal).	0xXXXX 2580	RW
0x507	MAC_CRC_CONFIG	The RX CRC forwarding configuration register. The following encodings are defined: • 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.	31'hX1'b0	RW
0x508	LINK_FAULT	Link Fault Status Register. For unidirectional Link Fault, implements *IEEE 802.3 Ethernet Clause 66.*	30'hX2'b00	RO
0x50A	RXMAC_CONTROL	RX MAC Control Register. The following bits are defined: • Bit[4] – Preamble check. Strict SFD checking option to compare each packet preamble to 0x555555555555. This field is available only if you turn on **Enable Strict SFD check**. • Bit[3] – SFD check. Strict SFD checking option to compare each SFD byte to 0x5D. This field is available only if you turn on **Enable Strict SFD check**. • Bit [1] – VLAN detection disabled. This bit is deasserted by default implying VLAN detection is enabled.	27'h0_5'b11X0X	RW

7.4. Pause/PFC Flow Control Registers

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

Table 25. TX Flow Control Registers

Addr	Bit	Name	Description	Reset	Access
0x600	31:0	TX Flow Control Revision ID	Specifies the revision ID, "100GFCTx CSR"	0x0809_20017	RO
0x601	31:0	TX Flow Control Scratch Pad	Scratch register for testing.	0	RW
0x602	31:0	TX Flow Control IP Core Variant 0	Specifies first 4 characters of IP core variation identifier ASCII string, "100G".	0x3130_3047	RO
0x603	31:0	TX Flow Control IP Core Variant 1	Next 4 characters of IP core variation identifier ASCII string, "FCTx"	0x4643_5478	RO
0x604	31:0	TX Flow Control IP Core Variant 2	Final 4 characters of IP core variation identifier ASCII string, "xCSR".	0x0043_5352	RO
0x605	7:0	TX Flow Control Enable One bit per queue	Enables the IP core to generate XON and XOFF Pause/PFC flow control frames to the remote partner. The following encodings are defined: • 1'b0: XON or XOFF Pause/PFC flow control is disabled • 1'b1: XON or XOFF Pause/PFC flow control is enabled. You can change this field dynamically.	0xFF	RW
	31:8	Reserved	Reserved	0	RO
0x606	7:0	TX Flow Control CSR XON/XOFF Request 0	XON/XOF flow control frame request bit 0. Interpretation depends on whether the IP core is in 1-bit FC request mode or in 2-bit FC request mode. This register affects a flow control queue only if the corresponding bit of the TX Flow Control Enable register has the value of 1. In 2-bit mode, in addition, this register is active for a specific flow control queue only if the corresponding bit in the TX 2-bit Flow Control Request Mode register field (bits 7:0 of the register at offset 0x641) specifies that the flow control logic accepts input from this register. The following encodings are defined for 1-bit mode. The IP core reads the 1-bit mode value in TX Flow Control CSR XON/XOFF Request 0. • 0 = No request • 0 to 1 = Generate XOFF request • 1 = Continue to generate XOFF request • 1 to 0 = Generate XON request	0	RW

continued...

Addr	Bit	Name	Description	Reset	Access
			The following encodings are defined for 2-bit mode. The IP core reads the 2-bit mode value in {`TX Flow Control CSR XON/XOFF Request 1`, `TX Flow Control CSR XON/XOFF Request 1`}. • 00 = No request • 01 = XON request • 10 = XOFF request • 11 = Invalid You can modify the value of this field dynamically.		
	15:8	Reserved	Reserved	0	RO
	23:16	`TX Flow Control CSR XON/XOFF Request 1`	In conjunction with Flow Control XON/XOFF Request 0 specifies a 2-bit request for XON/XOFF flow control frame transmission. This bit is the upper bit of the 2-bit control field. You can change the value of this field dynamically.	0	RW
	31:24	Reserved	Reserved	0	RO
0x60A	0	`TX Pause Enable` 1-bit	Determines whether receiving a valid Pause frame stops TX user data transmission. 1'b0: Transmission is not stopped 1b1: Transmission stops You cannot change the value of this field dynamically.	0	RW
	31:1	Reserved	Reserved	0	RO
0x60D	31:0	`TX Flow Control Destination Address Lower`	Specifies the 48-bit `Destination Address` of the flow control frame. Contains the 32 LSB of the address field. You cannot modify the value of this field dynamically.	0xC2000001	RW
0x60E	15:0	`TX Flow Control Destination Address Upper`	Specifies the 48-bit `Destination Address` of flow control frame. Contains the 16 MSB of the address field. You cannot modify the value of this field dynamically.	0x0180	RW
	31:16	Reserved	Reserved	0	RO
0x60F	31:0	`TX Flow Control Source Address Lower`	Specifies the 48-bit `Source Address` of flow control frame. Contains the 32 LSB of the address field.	0xCBFC5ADD	RW
0x610	15:0	`TX Flow Control Source Address Upper`	Specifies the 48-bit `Source Address` of flow control frame. Contains the 16 MSB of the address field. You cannot modify the value of this field dynamically.	0xE100	RW
	31:16	Reserved	Reserved	0	RO
0x620, 0x621, ..., 0x620	15:0	`TX Flow Control Quanta` 16-bit per FCQN	Specifies the pause quanta of Pause/PFC flow control frames to be sent to remote partner.	0xFFFF	RW

continued...

Addr	Bit	Name	Description	Reset	Access
+ (FCQN-1 where FCQN = 0 to 7)			You cannot modify the value of this field dynamically.		
	31:16	Reserved	Reserved	0	RO
0x628, 0x629, …, 0x628 + (FCQN-1 where FCQN = 0 to 7)	15:0	`TX Flow Control Signal XOFF Request Hold Quanta` 16-bit per FCQN	Specifies the separation between 2 consecutive XOFF flow control frames. You cannot modify the value of this field dynamically.	0xFFFF	RW
	31:16	Reserved	Reserved	0	RO
0x640	0	`TX Flow Control Select` 1-bit	Specifies whether the TX hardware generates Pause or PFC frames. Affects only PFC Queue 0. Usage example: You can synthesize a single PFC queue and use it for both Pause and PFC purpose. 1'b0: Pause 1'b1: PFC You cannot modify the value of this field dynamically.	1	RW
	31:1	Reserved.	Reserved.	0	RO
0x641	(FCQN-1):0	`TX 2-bit Flow Control Request Mode` 1-bit per FCQN	Determines whether the `TX Flow Control CSR XON/XOFF Request` register or the `pause_insert_tx0` and `pause_insert_tx1` signals control XON/XOFF mode in 2-bit control mode. 1'b0: The `pause_insert_tx0` and `pause_insert_tx1` signals control requests 1'b1: The `TX Flow Control CSR XON/XOFF Request` register fields control requests You cannot modify the value of this field dynamically.	0	RW
	16	`TX Flow Control Request Mode FCQN`	Determines whether the IP core is in TX flow control 1-bit mode or 2-bit mode. 1'b0: Use 1-bit mode to make TX flow control requests 1'b1: Use 2-bit mode to make TX flow control requests	0	RW
	31:17	Reserved	Reserved	0	RO

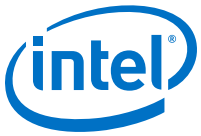


Table 26. RX Flow Control Registers

Addr	Bit	Name	Description	Reset	Access
0x700	31:0	RX Flow Control Revision ID	Specifies the revision ID, "100GFCTx CSR"	0x0809_20017	RO
0x701	31:0	RX Flow Control Scratch Pad	Provides a register for debug.	0	RW
0x702	31:0	RX Flow Control IP Core Variant 0	First 4 characters of IP core variation identifier ASCII string, Specifies first 4 characters of IP core variation identifier ASCII string, "100G".	0x3130_3047	RO
0x703	31:0	RX Flow Control IP Core Variant 1	Next 4 characters of IP core variation identifier ASCII string, "FCRx".	0x4643_5278	RO
0x704	31:0	RX Flow Control IP Core Variant 2	Final 4 characters of IP core variation identifier ASCII string, "0CSR". The "0" is unprintable.	0x0043_5352	RO
0x705	7:0	RX PFC Enable 1 bit per queue	Determines whether receiving a valid PFC frame causes the PFC duration user interface to indicate a valid pause quanta duration to the user logic. 1'b0: Disable 1'b1: Enable You cannot modify the value of this field dynamically.	0xFF	RW
	31:8	Reserved	Reserved	0	RO
0x707	31:0	RX Flow Control Destination Address Lower	Specifies the 48-bit Destination Address of the flow control frame. Contains the 32 LSB of the address field. The flow control frame is sent with the destination address matching the address specified in this register or the multicast address. If the address is not a match, the Flow Control block does not respond to the incoming frame; the IP core just passes it through. You cannot modify the value of this field dynamically.	0xC2000001	RW
0x708	15:0	RX Flow Control Destination Address Upper	Specifies the 48-bit Destination Address of flow control frame. Contains the 16 MSB of the address field. The flow control frame is sent with the destination address matching the address specified in this register or the multicast address. If the address is not a match, the Flow Control block does not respond to the incoming frame; the IP core just passes it through. You cannot modify the value of this field dynamically.	0x0180	RW
	31:16	Reserved	Reserved	0	RO

7.5. Statistics Registers

The Low Latency 100G Ethernet Intel FPGA statistics registers count Ethernet traffic and errors. The 64-bit statistics registers are designed to roll over, to ensure timing closure on the FPGA. However, these registers should never roll over if the link is functioning properly. The statistics registers check the size of frames, which includes the following fields:

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

The statistics counters module is a synthesis option. The statistics registers are counters that are implemented inside the CSR. When you turn on the **Enable RX/TX statistics counters** parameter in the Low Latency 100G Ethernet Intel FPGA parameter editor, the counters are implemented in the CSR. When you turn off the **Enable RX/TX statistics counters** parameter in the Low Latency 100G Ethernet Intel FPGA parameter editor, the counters are not implemented in the CSR, and read access to the counters returns undefined results.

After system power-up, the statistics counters have random values. You must clear these registers and confirm the system is stable before using their values. You can clear the registers with the `csr_rst_n` input signal, or with the configuration registers at offsets 0x845 and 0x945.

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

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

To ensure that the counters you read are consistent, you should issue a shadow request to create a snapshot of all of the TX or RX statistics registers, by setting bit [2] of the configuration register at offset 0x845 or 0x945, respectively. Until you reset this bit, the counters continue to increment but the readable values remain constant. You can read bit [1] of the status register at offset 0x846 or 0x946, respectively, to confirm your shadow request has been granted or released.

7.5.1. TX Statistics Registers

Table 27. Transmit Side Statistics Registers

The TX MAC does not check outgoing frames for FCS errors, Therefore, the TX statistics registers do not collect statistics for related categories: all FCS error-related registers should maintain the value of 0. In addition, the TX MAC does not check for undersized frames. Therefore, the `CNTR_TX_FRAGMENTS` and `CNTR_TX_RUNT` registers should maintain the value of 0.

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

continued...

Address	Name-	Description	Access
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
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

continued...

Address	Name-	Description	Access
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 siz between 1519 bytes and the number of bytes specified in the MAX_TX_SIZE_CONFIG register (upper 32 bits)	RO
0x824	CNTR_TX_OVERSIZE_LO	Number of oversized frames (frames with more bytes than the number specified in the MAX_TX_SIZE_CONFIG register) transmitted (lower 32 bits)	RO
0x825	CNTR_TX_OVERSIZE_HI	Number of oversized frames (frames with more bytes than the number specified in the MAX_TX_SIZE_CONFIG register) transmitted (upper 32 bits)	RO
0x826	CNTR_TX_MCAST_DATA_OK_LO	Number of valid multicast frames transmitted, excluding control frames (lower 32 bits)	RO
0x827	CNTR_TX_MCAST_DATA_OK_HI	Number of valid multicast frames transmitted, excluding control frames (upper 32 bits)	RO
0x828	CNTR_TX_BCAST_DATA_OK_LO	Number of valid broadcast frames transmitted, excluding control frames (lower 32 bits)	RO
0x829	CNTR_TX_BCAST_DATA_OK_HI	Number of valid broadcast frames transmitted, excluding control frames (upper 32 bits)	RO
0x82A	CNTR_TX_UCAST_DATA_OK_LO	Number of valid unicast frames transmitted, excluding control frames (lower 32 bits)	RO
0x82B	CNTR_TX_UCAST_DATA_OK_HI	Number of valid unicast frames transmitted, excluding control frames (upper 32 bits)	RO
0x82C	CNTR_TX_MCAST_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

continued...

Address	Name-	Description	Access
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. Therefore, this counter does not increment in normal operating conditions.	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. Therefore, this counter does not increment in normal operating conditions.	RO
0x836–0x844	Reserved		
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
0x847–0x85F	Reserved		
0x860	TxPayloadOctetsOK_LO	Number of transmitted payload bytes in frames with no FCS, undersized, oversized, or payload length errors. If VLAN detection is turned off for the TX MAC (bit[1] of the TX_MAC_CONTROL register at offset 0x40A has the value of 1), the IP core counts the VLAN header bytes (4 bytes for VLAN and 8 bytes for stacked VLAN) as payload bytes. This register is compliant with the requirements for aOctetsTransmittedOK in section 5.2.2.1.8 of the *IEEE Standard 802.3-2008*.	RO
0x861	TxPayloadOctetsOK_HI		RO
0x862	TxFrameOctetsOK_LO	Number of transmitted bytes in frames with no FCS, undersized, oversized, or payload length errors. This register is compliant with the requirements for ifOutOctets in RFC3635 (Managed Objects for Ethernet-like Interface Types) and TX etherStatsOctets in RFC2819(Remote Network Monitoring Management Information Base (RMON)).	RO
0x863	TxFrameOctetsOK_HI		RO

7.5.2. RX Statistics Registers

Table 28. 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
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

continued...

Address	Name	Description	Access
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 RXMAC_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 RXMAC_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 RXMAC_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 RXMAC_SIZE_CONFIG register) received (upper 32 bits)	RO
0x926	CNTR_RX_MCAST_DATA_OK_LO	Number of valid multicast frames received, excluding control frames (lower 32 bits)	RO
0x927	CNTR_RX_MCAST_DATA_OK_HI	Number of valid multicast frames received, excluding control frames (upper 32 bits)	RO
0x928	CNTR_RX_BCAST_DATA_OK_LO	Number of valid broadcast frames received, excluding control frames (lower 32 bits)	RO

continued...

Address	Name	Description	Access
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 received pause frames, with or without error (lower 32 bits)	RO
0x933	CNTR_RX_PAUSE_HI	Number of received pause frames, with or without error (upper 32 bits)	RO
0x934	CNTR_RX_RUNT_LO	Number of received runt packets (lower 32 bits) 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
0x936–0x944	Reserved		
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

continued...

Address	Name	Description	Access
0x947–0x95F	Reserved		
0x960	RxPayloadOctetsOK_LO	Number of received payload bytes in frames with no FCS, undersized, oversized, or payload length errors. If VLAN detection is turned off for the RX MAC (bit [1] of the RXMAC_CONTROL register at offset 0x50A has the value of 1), the IP core counts the VLAN header bytes (4 bytes for VLAN and 8 bytes for stacked VLAN) as payload bytes. This register is compliant with the requirements for aOctetsReceivedOK in section 5.2.2.1.14 of the *IEEE Standard 802.3-2008*.	RO
0x961	RxPayloadOctetsOK_HI		RO
0x962	RxFrameOctetsOK_LO	Number of received bytes in frames with no FCS, undersized, oversized, or payload length errors. This register is compliant with the requirements for ifInOctets in RFC3635 (Managed Objects for Ethernet-like Interface Types) and RX etherStatsOctets in RFC2819 (Remote Network Monitoring Management Information Base (RMON)).	RO
0x963	RxFrameOctetsOK_HI		RO

7.6. TX Reed-Solomon FEC Registers

Table 29. TX Reed-Solomon FEC Registers

Addr	Name	Bit	Description	Reset	Access
0xC00	REVID	[31:0]	Reed-Solomon FEC TX module revision ID.	0x0809 2017	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	[31:5]	Reserved.	0x00000000	RW
		[4]	When 1'b1, enables error insertion for single FEC codeword. This bit self-clears after the Reed-Solomon FEC transmitter inserts the error.		
		[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	[31:25]	Reserved.	0x00000000	RW
		[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. Specifically, the value of 1 directs the IP core to corrupt symbol 32 according to BIT_MASK.		
		[23:18]	Reserved.		
		[17:8]	BIT_MASK: Specifies which of the ten bits the RS-FEC transmitter corrupts in each corrupted symbol. Specifically, the value of 1 in bit [n+8] directs the IP core to corrupt bit [n] 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.		

continued...

Addr	Name	Bit	Description	Reset	Access
			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]}) Continuous corruption of groups 0-3 might lead to strange behavior as a result of alignment marker corruption.		
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. Specifically the value of 1 in bit [n] directs the IP core to corrupt symbol n.	32'b0	RW

7.7. RX Reed-Solomon FEC Registers

Table 30. RX Reed-Solomon FEC Registers

Addr	Name	Bit	Description	Reset	Access
0xD00	REVID	[31:0]	RSFEC RX module revision ID	0x0809 2017	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.		
		[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 FF00	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	RC

continued...

Addr	Name	Bit	Description	Reset	Access
			This register gets updated based on the error correction logic even when BYPASS_RESTART bit [0] is 1.		
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. This register gets updated based on the error correction logic even when BYPASS_RESTART bit [0] is 1.	0x0000 0000	RC

8. Debugging the Link

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

The following steps should help you identify and resolve common problems that occur when bringing up a Low Latency 100G Ethernet Intel FPGA IP core link:

1. Establish word lock—The RX lanes should be able to achieve word lock even in the presence of extreme bit error rates. If the IP core is unable to achieve word lock, check the transceiver clocking and data rate configuration. Check for cabling errors such as the reversal of the TX and RX lanes. Check the clock frequency monitors 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 high 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 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. If the IP core acquires word lock but the link is still not established, check the AM_LOCK register at offset 0x328 by reading it repeatedly. If it is deasserted or toggling, check the cables and connections.

6. If the IP core acquires alignment marker lock on all virtual lanes (bit [0] of the AM_LOCK has the consistent value of 1), but the link is still not established, check the LANE_DESKEWED register at offset 0x329. If this register remains at the value of 0, the skew is greater than the deskew limit.
7. 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.
8. 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 asserts the `Local Fault Status` bit (bit [0]) of the `Link_Fault` register at offset 0x508. This change is linked to the change in value of the `local_fault_status` signal.
- The IP core triggers the RX digital reset process by asserting `soft_rxp_rst`.

Related Information

- Stratix 10 L- and H-Tile Transceiver PHY User Guide
  Information about the analog parameters for Stratix 10 production devices.
- Stratix 10 GX 2800 L-Tile ES-1 Transceiver PHY User Guide
  Information about the analog parameters for Stratix 10 L-tile ES1 devices.

9. Low Latency 100G Ethernet Intel Stratix 10 FPGA 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
17.1	Intel Stratix 10 Low Latency 100-Gbps Ethernet IP Core User Guide 17.1

10. Document Revision History for the Low Latency 100G Ethernet Intel Stratix 10 FPGA IP Core User Guide

Document Version	Intel Quartus Prime Version	Changes
2018.07.18	18.0	• Added flow control, TX error insertion, and RX control frame indication features in supported features list. • Updated release information for the IP core. • Added 322.265625 MHz clock option to the **PHY reference frequency** parameter. • Added **Enable MAC Flow Control**, **Number of queues in priority flow control**, and **Enable link fault generation** parameters in the IP core parameter table. • Added PCS compliance table *Functional Description* section. • Added `pause_insert_tx0`, `pause_insert_tx1`, and `pause_receive_rx` signals in the *Low Latency 100G Ethernet Intel FPGA Signals and Interfaces* diagram. • Updated `PHY_CONFIG` and RX_FEC_STATUS registers' default value. • Added `ERR_INJ`, `LINK_FAULT`, Pause/PFC Flow Control registers in the *Registers* section. • Added *Inter-Packet Gap Adjustment* topic. • Updated the `clk_status` and `reconfig_clk` frequency to 100 - 162 MHz.
2017.11.06	17.1	Initial public release.