eCPRI Intel® FPGA IP User Guide

Updated for Intel® Quartus® Prime Design Suite: **22.4**

IP Version: **2.0.2**

Online Version

Send Feedback

UG-20277

ID: **683685**

Version: **2023.02.24**

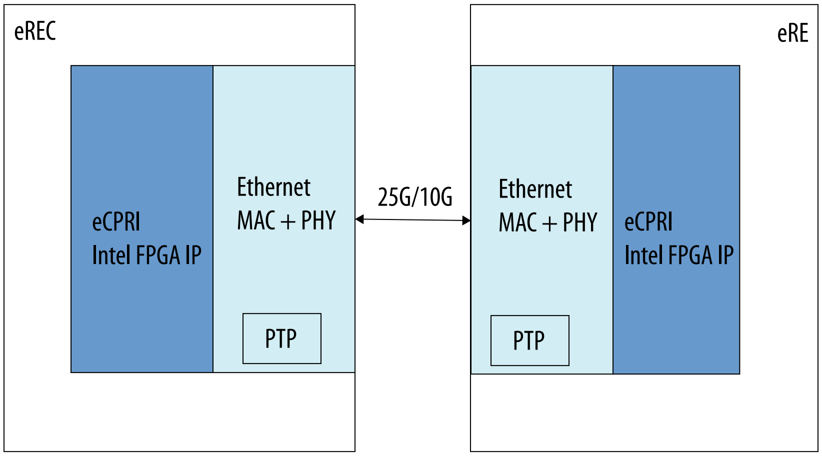
Contents

Send Feedback

1. Introduction

The enhanced Common Public Radio Interface (eCPRI) Intel® FPGA IP core implements the *eCPRI specification version 2.0*. The eCPRI IP is a front-haul interface protocol for radio base station aimed at connecting the eCPRI Radio Equipment Control (eREC) and the eCPRI Radio Equipment (eRE) via front-haul transport network.

Figure 1. Typical eCPRI Application on Intel FPGA Devices



Related Information

- eCPRI Specification V2.0
- eCPRI Intel FPGA IP Design Example User Guide

1.1. Supported Features

The eCPRI Intel FPGA IP core offers the following features:

- Compliant with the *eCPRI Specification V2.0 (2018-06-25)* available on the CPRI Industry Initiative (CII) website.
- Supports eCPRI radio equipment controller (eREC) and eCPRI radio equipment (eRE) module configurations.
- Supports Ethernet headers in a variety of formats, including VLAN tag, source/destination MAC address, IPv4, UDP extraction and encapsulation.
- Supports eCPRI one-way delay measurement based on IEEE Standard 1588 Precision Time Protocol (1588 PTP) hardware timestamp. Full hardware support, and required 1588 PTP software stack.
- Supports 25 Gbps and 10 Gbps Ethernet ports.
- Supports pairing of eCPRI Intel FPGA IP with O-RAN Intel FPGA IP.

- Supports interworking function (IWF) type 0 between eCPRI node and one CPRI node.
- Capable of streaming Ethernet frame size up to 9,000 bytes as defined by Ethernet jumbo frames standard.
- Packet classifier responsible to classify eCPRI packet and send packets to eCPRI IP. All other packets are redirected to external port for user processing.
- Programmable packet queue (maximum 16 entries) to hold incoming packets when eCPRI packets transmission in progress.
- Arbitration between eCPRI packet and external incoming Ethernet frames, e.g., Control & Management (C&M) and synchronization packets.
- Offers mapping logic between eCPRI message physical channel ID to VLAN/MAC address CSR.
- Supports single distributed unit (DU) and up to eight radio unit (RU) configurations using source/destination MAC address CSR.
- Support all eCPRI message types compliant to eCPRI specification v2.0
- Input/output ports compliant with Avalon® streaming interface .

Table 1. eCPRI Intel FPGA IP Feature Matrix

Device Support	Data Rate
Intel Agilex™	25G
	10G
Intel Stratix® 10	25G
	10G
Intel Arria® 10	10G

Related Information

- CPRI Industry Initiative website
- IEEE website
- Supported Ethernet Variants on page 23
  Refer to this section for information on supported Ethernet variants from Intel.
- O-RAN Intel FPGA IP User Guide

1.2. Device Family Support

Table 2. 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

continued...

Device Support Level	Definition
	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 3. eCPRI Intel FPGA IP Core Device Family Support

Shows the level of support offered by the eCPRI Intel FPGA IP for each Intel FPGA device family.

Device Family	Support
Intel Agilex (F-tile devices)	Advance
Intel Agilex (E-tile devices)	Advance
Intel Stratix 10 (E-tile devices)	Final
Intel Stratix 10 (H-tile devices)	Final
Intel Arria 10	Final
Other device families	No support

1.3. eCPRI Intel FPGA IP Device Speed Grade Support

The eCPRI Intel FPGA IP core supports the following speed grades.

Table 4. Device Speed Grade Support

Device	Transceiver Speed Grade	Core Speed Grade
Intel Agilex with F-tile	–3, –2 and –1	–3, –2 and –1
Intel Agilex with E-tile	–2 and –1	–2 and –1
Intel Stratix 10 with E-tile	–2 and –1	–2 and –1
Intel Stratix 10 with H-tile	–2 and –1	–2 and –1
Intel Arria 10	–2 and –1	–2 and –1

1.4. Resource Utilization

The resources for the eCPRI Intel FPGA IP core were obtained form the Intel Quartus® Prime Pro Edition software version 22.3 with advance mapping enabled:

Table 5. Resource Utilization

Device	Mode	ALMs	Dedicated Logic Registers	Memory 20K
Intel Agilex	Non-streaming	9576	21913	53
	Streaming	9335	22127	52
	O-RAN Fixed	7538	17383	43

continued...

Send Feedback

Device	Mode	ALMs	Dedicated Logic Registers	Memory 20K
	O-RAN L2COS	11978	20305	44
	IWF (with streaming)	11250	24365	72
Intel Stratix 10	Non-streaming	9535	22641	53
	Streaming	9429	22542	52
	O-RAN Fixed	7650	17273	43
	O-RAN L2COS	11886	22746	44
	IWF (with streaming)	11176	23940	74
Intel Arria 10	Non-streaming	8876	20827	53
	Streaming	8653	20251	52
	O-RAN Fixed	6729	15779	43
	O-RAN L2COS	10844	18842	44
	IWF (with streaming)	14026	25126	69

1.5. Intel FPGA IP Core Verification

To ensure functional correctness of the eCPRI Intel FPGA IP core, Intel performs validation through both simulation and hardware testing. Before releasing a version of the eCPRI Intel FPGA IP core, Intel runs regression tests in the associated version of the Intel Quartus Prime software.

1.6. Release Information

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

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

Table 6. eCPRI Intel FPGA IP Core Release Information

Item	Description
IP Version	2.0.2
Intel Quartus Prime Version	22.4
Release Date	2023.02.24
Ordering Code	IP-eCPRI

Related Information

eCPRI Intel FPGA IP Release Notes

The IP Release Notes describes changes to the IP in a particular release.

2. Getting Started

The following sections explain how to install, parameterize, simulate, and initialize the eCPRI Intel FPGA IP IP core:

2.1. Installing and Licensing

The eCPRI Intel FPGA IP core is an extended FPGA IP core which is not included with the Intel Quartus Prime release. This section provides a general overview of the Intel extended FPGA IP core installation process to help you quickly get started with any Intel extended FPGA IP core.

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

Figure 2. eCPRI Intel FPGA IP Core Installation Directory Structure

Directory structure after you install the eCPRI IP core.

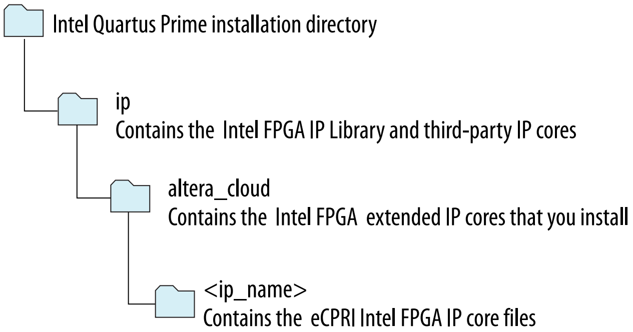

Intel Quartus Prime installation directory
ip
Contains the Intel FPGA IP Library and third-party IP cores
altera_cloud
Contains the Intel FPGA extended IP cores that you install
<ip_name>
Contains the eCPRI Intel FPGA IP core files

Table 7. Intel FPGA IPCore Installation Locations

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

Related Information

Self-Service Licensing Center (SSLC)

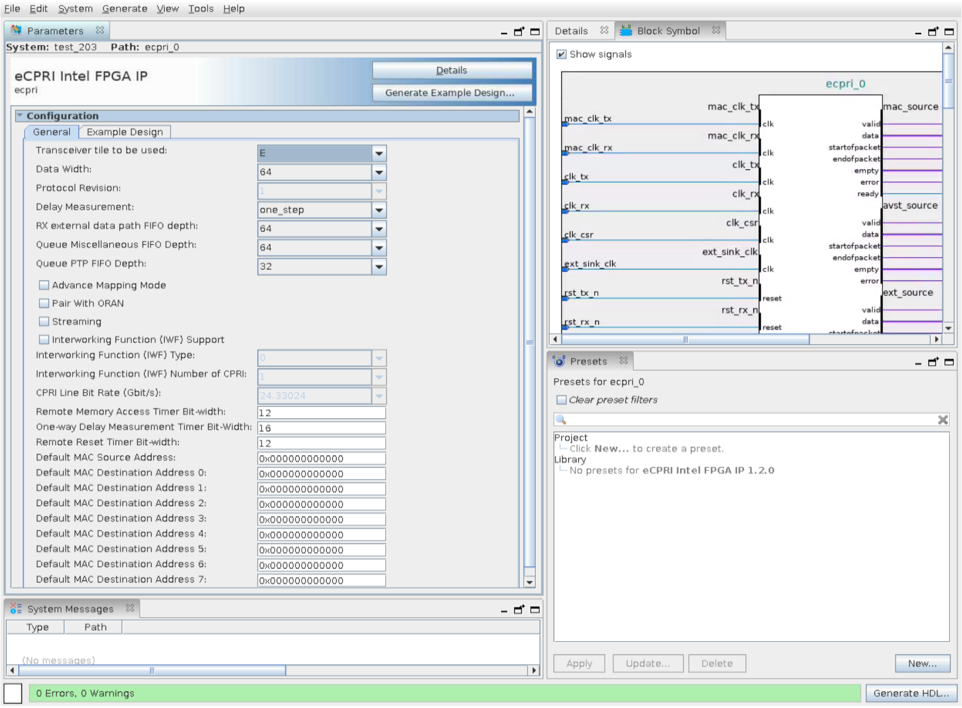
After you purchase the eCPRI Intel FPGA IP core, the IP core is available for download from the SSLC page in your My Intel account. You must create a My Intel account if you do not have one already, and log in to access the SSLC. On the SSLC page, click Run for this IP core. The SSLC provides an installation dialog box to guide your installation of the IP core.

2.2. Specifying the IP Core Parameters and Options

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

Prerequisite: Once you receive the eCPRI web-core IP, save the web-core installer to the local area. Run the installer with Windows/Linux. When prompt, install to the same location as Intel Quartus Prime folder. The eCPRI Intel FPGA IP now appears in the IP Catalog.

Figure 3. eCPRI IP Parameter Editor



System: test_203 Path: ecpri_0

eCPRI Intel FPGA IP
ecpri

Transceiver tile to be used:	E
Data Width:	64
Protocol Revision:	1
Delay Measurement:	one_step
RX external data path FIFO depth:	64
Queue Miscellaneous FIFO Depth:	64
Queue PTP FIFO Depth:	32
☐ Advance Mapping Mode	
☐ Pair With ORAN	
☐ Streaming	
☐ Interworking Function (IWF) Support	
Interworking Function (IWF) Type:	0
Interworking Function (IWF) Number of CPRI:	1
CPRI Line Bit Rate (Gbit/s):	24.33024
Remote Memory Access Timer Bit-width:	12
One-way Delay Measurement Timer Bit-Width:	16
Remote Reset Timer Bit-width:	12
Default MAC Source Address:	0x000000000000
Default MAC Destination Address 0:	0x000000000000
Default MAC Destination Address 1:	0x000000000000
Default MAC Destination Address 2:	0x000000000000
Default MAC Destination Address 3:	0x000000000000
Default MAC Destination Address 4:	0x000000000000
Default MAC Destination Address 5:	0x000000000000
Default MAC Destination Address 6:	0x000000000000
Default MAC Destination Address 7:	0x000000000000

Presets for ecpri_0
☐ Clear preset filters
Project
└ Click **New...** to create a preset.
Library
└ No presets for **eCPRI Intel FPGA IP 1.2.0**

(No messages)

0 Errors, 0 Warnings

1. If you do not already have an Intel Quartus Prime Pro Edition project in which to integrate your eCPRI 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 that meets the speed grade requirements for the IP core.
   c. Click **Finish**.
2. In the IP Catalog, locate and select **eCPRI Intel FPGA IP**. 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 Parameters on page 15 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 *Design Example User Guide*.
7. Click **Generate HDL**. The **Generation** dialog box appears.
8. Specify output file generation options, and then click **Generate**. The IP variation files generate according to your specifications.
9. Click **Finish**. The parameter editor adds the top-level `.ip` file to the current project automatically. If you are prompted to manually add the `.ip` file to the project, click **Project ➤ Add/Remove Files in Project** to add the file.
10. After generating and instantiating your IP variation, make appropriate pin assignments to connect ports and set any appropriate per-instance RTL parameters.

Related Information

eCPRI Intel FPGA Design Example User Guide

2.2.1. Reference and System PLL Clock for your IP Design

Each F-tile system must instantiate one F-Tile Reference and System PLL Clocks Intel FPGA IP. The F-Tile Reference and System PLL Clocks Intel FPGA IP performs three main functions:

1. Configure reference clock for FHT PMA:
   - Enable the FHT common PLLs and select the reference clock source for FHT common PLL
   - Specify the FHT reference clock source frequency
2. Configure reference clock for FGT PMA:
   - Enable FGT reference clocks and specify the reference clock frequency
   - Specify FGT CDR output
3. Configure system PLL:
   - Enable system PLL and specify its mode
   - Specify the reference clock source and frequency for system PLL

Note: In your IP design, you must include an F-Tile Reference and System PLL Clocks Intel FPGA IP core to pass logic generation flow.

The F-Tile Reference and System PLL Clocks Intel FPGA IP must always connect to a protocol based Intel FPGA IP. The F-Tile Reference and System PLL Clocks Intel FPGA IP cannot be compiled or simulated as a standalone IP. For more information on parameters and port list for F-Tile Reference and System PLL Clocks Intel FPGA IP core, refer to the *F-tile Architecture and PMA/FEC Direct PHY IP User Guide*.

When you design multiple interfaces or protocol-based IP cores within a single F-tile, you must use only one instance of the F-Tile Reference and System PLL Clocks Intel FPGA IP core to configure:

- All required reference clocks for FGT PMA (up to 10) and FHT PMA (up to 2) to implement multiple interfaces within a single F-tile.
- All required FHT common PLLs (up to 2) to implement multiple interfaces within a single F-tile.
- All required System PLLs (up to 3) to implement multiple interfaces within a single F-tile.
- All required reference clocks for system PLLs (up to 8 – shared with FGT PMA) to implement multiple interfaces within a single F-tile.

When you design multiple interfaces or protocol-based IP cores within a single F-tile, you can only use three System PLLs. For example, you can use one System PLL for PCIe and two for Ethernet and other protocols. However, there are other use cases where you can use all three for various interfaces within the Ethernet and PMA-Direct digital blocks. As there are only three System PLLs, multiple interfaces or protocol-based IP cores with different line rates may have to share a System PLL. While sharing a System PLL, the interface with the highest line rate determines the system PLL frequency, and the interfaces with the lower line rates must be overclocked. For more information, refer to the *F-tile Architecture and PMA/FEC Direct PHY IP User Guide*.

Related Information

F-tile Architecture and PMA/FEC Direct PHY IP User Guide

2.3. Generated File Structure

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

Figure 4. eCPRI IP Core Generated Files

For more information about the file structure of the design example, refer to the *eCPRI Intel FPGA Design Example User Guide*.

Figure 5. eCPRI IP Core Generated Files

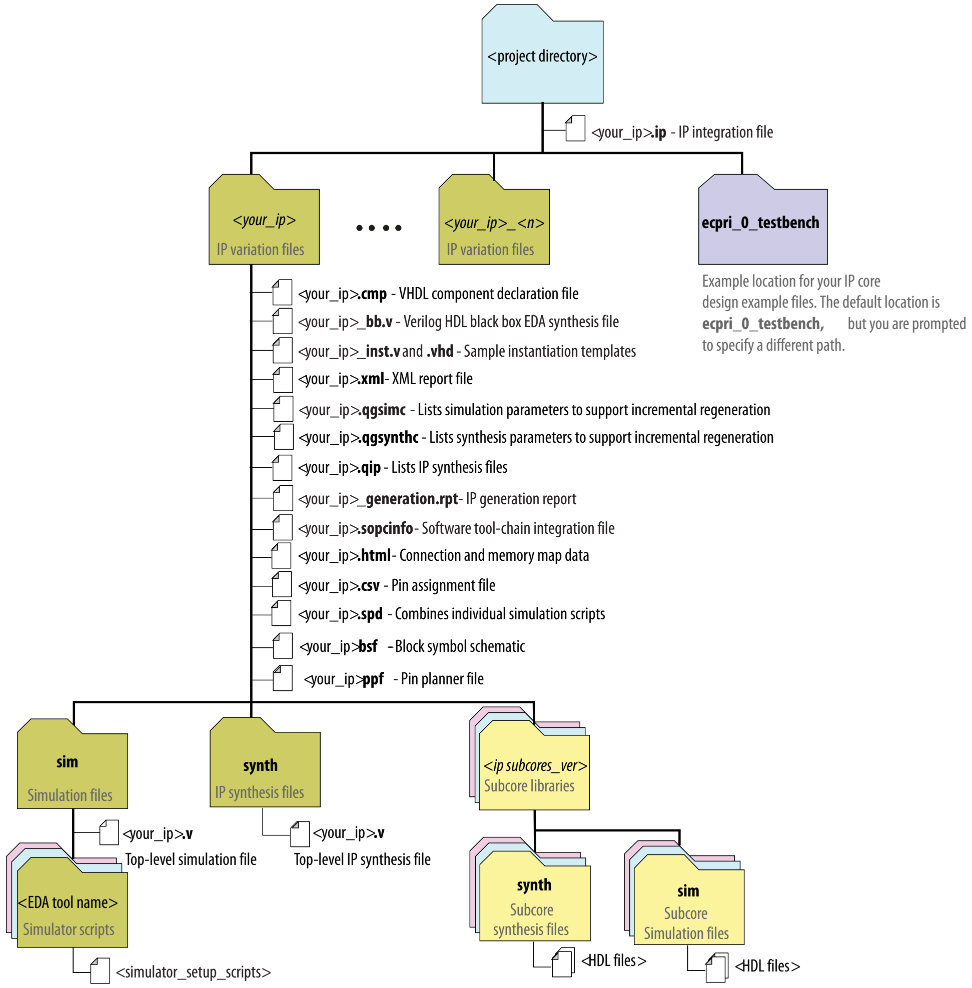

<project directory>
<your_ip>.ip - IP integration file
<your_ip>
IP variation files
• • • •
<your_ip>_<n>
IP variation files
ecpri_0_testbench
Example location for your IP core design example files. The default location is ecpri_0_testbench, but you are prompted to specify a different path.
<your_ip>.cmp - VHDL component declaration file
<your_ip>_bb.v - Verilog HDL black box EDA synthesis file
<your_ip>_inst.v and .vhd - Sample instantiation templates
<your_ip>.xml- XML report file
<your_ip>.qgsimc - Lists simulation parameters to support incremental regeneration
<your_ip>.qgsynthc - Lists synthesis parameters to support incremental regeneration
<your_ip>.qip - Lists IP synthesis files
<your_ip>_generation.rpt- IP generation report
<your_ip>.sopcinfo- Software tool-chain integration file
<your_ip>.html- Connection and memory map data
<your_ip>.csv - Pin assignment file
<your_ip>.spd - Combines individual simulation scripts
<your_ip>bsf - Block symbol schematic
<your_ip>ppf - Pin planner file
sim
Simulation files
synth
IP synthesis files
<ip subcores_ver>
Subcore libraries
<your_ip>.v
Top-level simulation file
<your_ip>.v
Top-level IP synthesis file
<EDA tool name>
Simulator scripts
<simulator_setup_scripts>
synth
Subcore
synthesis files
sim
Subcore
Simulation files
<HDL files>
<HDL files>

Table 8. eCPRI 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.
<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.

continued...

File Name	Description
<*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 Intel Quartus Prime Block Diagram Files (.bdf).
<*your_ip*>.spd	Required input file for ip-make-simscript to generate simulation scripts for supported simulators. The .spd file contains a list of files generated for simulation, along with information about memories that you can initialize.
<*your_ip*>.ppf	The Pin Planner File (.ppf) stores the port and node assignments for IP components created for use with the Pin Planner
<*your_ip*>_bb.v	You can use the Verilog black-box (_bb.v) file as an empty module declaration for use as a black box.
<*your_ip*>_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.
<*your_ip*>.v or <*your_ip*>.vhd	HDL files that instantiate each submodule or child IP core for synthesis or simulation.
mentor/	Contains a QuestaSim* script msim_setup.tcl to set up 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.
aldec/	Contains a shell script rivierapro_setup.sh to setup and run a Riviera-PRO* simulation.
xcelium/	Contains a shell script xcelium_setup.sh and other setup files to set up and run an Xcelium* simulation.
submodules/	Contains HDL files for the IP core submodules.
<*child IP cores*>/	For each generated child IP core directory, Platform Designer generates synth/ andsim/ sub-directories.

Related Information

eCPRI Intel FPGA Design Example User Guide

2.4. Simulating the IP Core

You can simulate your eCPRI IP variation using any of the vendor-specific IEEE encrypted functional simulation models which are available in the <instance_name>/sim subdirectory of your project directory.

The eCPRI IP core supports the Synopsys* VCS, Synopsys VCS MX, Siemens* EDA QuestaSim, Aldec* Riviera-PRO and Xcelium Parallel simulators. The eCPRI IP core generates a Verilog HDL and VHDL simulation model. The IP core parameter editor offers you the option of generating a Verilog HDL or VHDL simulation model for the IP core. The IP core design example also supports Verilog HDL/VHDL simulation model or testbench.

For more information about functional simulation models for Intel FPGA IP cores, refer to the *Simulating Intel FPGA Designs chapter in Quartus Prime Pro Edition User Guide: Third-party Simulation*.

Related Information

- Simulating Intel FPGA Designs
- eCPRI Intel Stratix 10 FPGA Design Example User Guide

2.5. Compiling the Full Design and Programming the FPGA

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

3. IP Parameters

You customize the IP core by specifying parameters in the IP parameter editor.

Table 9. Parameters: Configuration Tab

Parameter	Supported Values	Default Setting	Description
Transceiver Tile to be used	E F H	E	You can choose: • H-tile or E-tile for your Intel Stratix 10 device • E-tile or F-tile for your Intel Agilex device *Note:* This parameter is not present in the Intel Arria 10 IP variations.
Data Width	64	64	Primary data bus width.
Protocol Revision	1	1	Specifies eCPRI protocol revision used in eCPRI common header. This option is grayed out in the current version of the Intel Quartus Prime software.
Delay Measurement	off one_step two_step	one_step	Indicates option to support and the operation mode of delay measurement for eCPRI message type 5 delay measurement. When set to **off**, the IP does not include the delay measurement logic.
RX external data path FIFO depth	64 128 256	64	Indicates the depth of the RX external data path FIFO. The actual depth is $\log_2$ of the FIFO depth.
Queue Miscellaneous FIFO depth	32 64 128 256	64	Indicates the depth of the Queue miscellaneous FIFO. The actual depth is $\log_2$ of the FIFO depth.
Queue PTP FIFO Depth	32 64 128 256	32	Indicates the depth of the Queue PTP FIFO. The actual depth is $\log_2$ of the FIFO depth.
Advance Mapping Mode	On Off	On	When you turn on this parameter, it allows the mapping of the destination MAC address and VLAN tag CSE to eCPRI message PC_ID field.
Pair with ORAN	On Off	Off	Turn on this option to pair your eCPRI Intel FPGA IP with Intel O-RAN FPGA IP. You can also pair your eCPRI Intel FPGA IP with any external vendor O-RAN IP. *Note:* When you turn on this parameter, the eCPRI Intel FPGA IP only supports message type 0, 2, and 5.
Streaming	On Off	Off	Indicates Ethernet frame size.

continued...

Parameter	Supported Values	Default Setting	Description
			When you turn off this parameter, the maximum Ethernet frame size can be 1500 bytes, and when you turn on, the maximum Ethernet frame size can be 9,000 bytes. *Note:* When you turn on this parameter, the `sink_pkt_size` port is available at the Sink Interface of the eCPRI Intel FPGA IP.
Interworking Function (IWF) Support	On Off	Off	Turn on this option to connect your eCPRI IP with one CPRI IP node. The eCPRI Intel FPGA IP currently support IWF Type 0 only. It does not support IWF type 1 and type 2 in current release of the IP. *Note:* When you turn on this parameter, the eCPRI Intel FPGA IP supports message type 0, 2, 5, 6 and 7 with IWF function.
Interworking Function (IWF) Type	0	0	Specifies eCPRI IP IWF type configuration. Currently the IP support IWF Type 0 configuration.
Interworking Function (IWF) Number of CPRI	1	1	Specifies the number of CPRI MAC that can connect to IWF.
Remote Memory Access Timer Bit-width	12	12	Specifies bit-width of the request-response sequence timer for the eCPRI message type 4. This parameter triggers the timeout no memory access response.
One-way Delay Measurement Time Bit-width	16	16	Specifies bit-width of the request-response sequence timer for the eCPRI message type 5,One-way delay measurement. This parameter triggers the timeout no memory access response.
Remote Reset Timer Bit-width	12	12	Specifies bit-width of the request-response sequence timer for the eCPRI message type 6. This parameter triggers the timeout no memory access response.
Default MAC Source Address	-	0x000000000000	Default MAC source address after cold and soft reset.
Default MAC Destination Address 0	-	0x000000000000	Default MAC destination address 0 after cold and soft reset.
Default MAC Destination Address 1	-	0x000000000000	Default MAC destination address 1 after cold and soft reset.
Default MAC Destination Address 2	-	0x000000000000	Default MAC destination address 2 after cold and soft reset.
Default MAC Destination Address 3	-	0x000000000000	Default MAC destination address 3 after cold and soft reset.
Default MAC Destination Address 4	-	0x000000000000	Default MAC destination address 4 after cold and soft reset.
Default MAC Destination Address 5	-	0x000000000000	Default MAC destination address 5 after cold and soft reset.
Default MAC Destination Address 6	-	0x000000000000	Default MAC destination address 6 after cold and soft reset.
Default MAC Destination Address 7	-	0x000000000000	Default MAC destination address 7 after cold and soft reset.
Default VLAN ID	-	0x000	Default VLAN ID after cold and soft reset.

continued...

Parameter	Supported Values	Default Setting	Description
Data Flow Identification	MACADDR VLANID	MACADDR	Use MAC Address or VLAN ID for Data Identification.
Packets Arbitration Scheme	L2COS Fixed	Fixed	Specifies the TX packets arbitration scheme.
TX Packets Default Priority	0 to 7	7	Indicates the default priority for S/M/other plane packets that doesn't contain VLAN ID within L2 header.
TX Arbitration Queue 0 Depth	0 32 64 128 256	128	Indicates the TX arbitration queue 0 depth. FIFO width is 8 Bytes.
TX Arbitration Queue 1 Depth	0 32 64 128 256	128	Indicates the TX arbitration queue 1 depth. FIFO width is 8 bytes.
TX Arbitration Queue 2 Depth	0 32 64 128 256	128	Indicates the TX arbitration queue 2 depth. FIFO width is 8 bytes.
TX Arbitration Queue 3 Depth	0 32 64 128 256	128	Indicates the TX arbitration queue 3 depth. FIFO width is 8 bytes.
TX Arbitration Queue 4 Depth	0 32 64 128 256	128	Indicates the TX arbitration queue 4 depth. FIFO width is 8 bytes.
TX Arbitration Queue 5 Depth	0 32 64 128 256	128	Indicates the TX arbitration queue 5 depth. FIFO width is 8 bytes.
TX Arbitration Queue 6 Depth	0 32 64 128 256	128	Indicates the TX arbitration queue 6 depth. FIFO width is 8 bytes.
TX Arbitration Queue 7 Depth	32 64 128 256	128	Indicates the TX arbitration queue 7 depth. FIFO width is 8 bytes.

For parameters in the **Example Design** tab, refer to the *eCPRI Intel Stratix 10 FPGA Design Example User Guide*.

Related Information

- O-RAN Intel FPGA IP User Guide
- eCPRI Intel FPGA Design Example User Guide
- 25G Ethernet Intel Stratix 10 FPGA IP User Guide
- E-tile Hard IP User Guide

4. Functional Description

The eCPRI Intel FPGA IP core provides the functionality described in the *eCPRI specification version 2.0*.

4.1. Interfaces

The eCPRI Intel FPGA IP supports the following interfaces:

- **Clock and Reset Interface**

  The main interface for the clock and reset signals in the eCPRI IP.

- **Configuration Avalon Memory-Mapped Interface**

  This interface provides access to the internal control and status registers of the eCPRI IP. This interface complies with Avalon memory-mapped interface specification as defined in the *Avalon Interface Specifications*.

- **External MAC Source Interface**

  This interface provides datapath from eCPRI IP to 25G Ethernet MAC IP. This interface complies with Avalon streaming interface specification as defined in the *Avalon Interface Specifications*.

- **External MAC Sink Interface**

  This interface provides datapath from 25G Ethernet MAC IP to eCPRI IP. This interface complies with Avalon streaming interface specification as defined in the *Avalon Interface Specifications*.

- **eCPRI IP Source Interface**

  This interface provides datapath from eCPRI IP to client logic. This interface includes a number of sideband signals which align with the Avalon streaming interface clock. This interface complies with Avalon streaming interface specification as defined in the *Avalon Interface Specifications*.

- **eCPRI IP Sink Interface**

  This interface provides datapath from client logic to eCPRI IP. This interface includes a number of sideband signals which align with the Avalon streaming interface clock. This interface complies with Avalon streaming interface specification as defined in the *Avalon Interface Specifications*.

- **IWF Type 0 eCPRI Source Interface**

  This interface provides datapath from eCPRI IP to IWF logic. This interface includes a number of sideband signals which align with the eCPRI IP source interface.

- **IWF Type 0 eCPRI Sink Interface**

  This interface provides datapath from IWF logic to eCPRI IP. This interface includes a number of sideband signals which align with the eCPRI IP source interface.

- **IWF Type 0 CPRI MAC Interface**

  This interface provides datapath from eCPRI IWF type 0 function to CPRI MAC. This interface consists of the following interfaces:

  - CPRI 32-bit IQ Data
  - CPRI 64-bit IQ Data
  - CPRI 32-bit Ctrl_AxC
  - CPRI 64-bit Ctrl_AxC
  - CPRI 32-bit Vendor Specific
  - CPRI 64-bit Vendor Specific
  - CPRI 32-bit Real-Time Vendor Specific
  - CPRI 64-bit Real-Time Vendor Specific
  - CPRI Gigabit Media Independent Interface (GMII)

- **External ST Source Interface**

  This interface provides datapath from eCPRI IP to client logic. This interface is a primary output interface for PTP and C&M messages. This interface complies with Avalon streaming interface specification as defined in the *Avalon Interface Specifications*. This interface includes a number of sideband signals which align with the Avalon streaming interface clock.

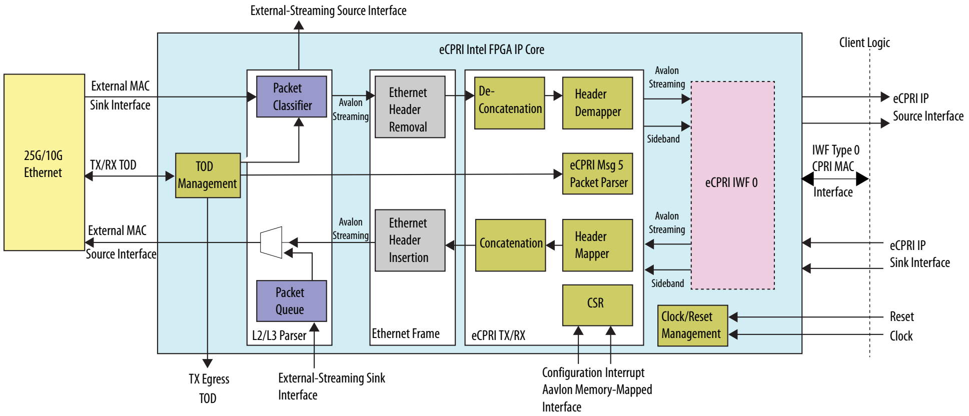
- **External ST Sink Interface**

  This interface provides datapath from client logic to eCPRI IP. This interface is a primary input for PTP and C&M messages. This interface complies with Avalon streaming interface specification as defined in the *Avalon Interface Specifications*.

- **TX and RX Time-of-Day (TOD) Interface**:

  This interface provides 96-bit timestamp from PTP module to eCPRI IP core and to client logic.

Figure 6. eCPRI Intel FPGA IP High-Level System Overview



Related Information

- eCPRI Specification V2.0
- Avalon Interface Specifications

- 1588 Precision Time Protocol Interfaces
  For 25G Ethernet Intel Stratix 10 Intel FPGA IP
- 1588 Precision Time Protocol Interfaces
  For E-tile Hard IP for Ethernet Intel FPGA IP

4.2. High Level Data Path Flow

The eCPRI IP core consists of two paths:

- Transmit TX path
- Receive RX Path

4.2.1. Transmit TX Path

There are two sets of Avalon streaming interface source and sink signals available to the incoming packets on the transmit TX path. Avalon streaming interface source/sink connects to the eCPRI IP and external source/sink interface connects to external user logic. The incoming eCPRI packets passes through Ethernet header insertion block to insert Ethernet header, optionally with different VLAN tags, IPv4, and UDP headers configured during configuration time.

You can send different types of packets through the external source/sink interface signal (For example, C&M and synchronization packets) which arbitrates with eCPRI packets and the IP sends packets with the higher priority to Ethernet MAC for transmission. The incoming external user packets are expected to arrive with Ethernet MAC header inserted on the packets.

The eCPRI IP supports two types of packets arbitration:

- Fixed priority arbitration
- L2 CoS priority arbitration based on the *O-RAN Control, User and Synchronization Plane Specification 7.01 (ORAN-WG4.CUS.0-v07.01), Section 5.3 Quality of Service*.

Both of the arbitration features are mutually exclusive to each other and you can enable one of them using the **Packets Arbitration Scheme** parameter.

For fixed priority arbitration, the priority of the packets send to Ethernet MAC is listed as below, with highest priority order from top to bottom:

- PTP synchronization packet
- eCPRI packet
- C&M packets and remaining type of packets

The C&M and PTP synchronization packets are send/receive through external source/sink interface signal. The C&M and PTP synchronization packets are generally low bandwidth traffic. When there is collision between external PTP synchronization packets and eCPRI packets, backpressure to the eCPRI IP occurs to stop eCPRI packets from transmitting. The eCPRI IP implements a counter to track the number of eCPRI packets and PTP packets granted and raise the priority of the C&M packet when the counter reaches a programmable threshold to allow the C&M packet transmission to Ethernet MAC and avoid starvation.

Ensure the bandwidth of external source/sink interface signal won't starve the overall bandwidth and cause interruption on eCPRI traffics. The grant ratio between C&M packets versus eCPRI/PTP packets is 10:1. The bandwidth allocated to C&M packets is 2.5G or 1G. The maximum C&M/PTP FIFO depth is 256 or 2048 bytes for eCPRI IP. The C&M/PTP FIFO should not be kept full beyond 2062.5 * (`mac_clk_tx`) clock period, to allow for enough read margin prior to the arrival of the new packets.

When you set the Ethernet frame size to 9000 bytes, data from Avalon streaming interface sink directly pass through and does not required buffering. You must assert `avst_sink_valid` continuously between the assertions of `avst_sink_sop` and `avst_sink_eop`. The only exception is when `avst_sink_ready` signal deasserts, and you are required to deassert `avst_sink_valid` for three cycles of READY_LATENCY.

The second L2 CoS priority arbitration is based on the *ORAN-WG4.CUS.0-v07.01 specification, Section 5.3 Quality of Service*, where the L2 CoS Priority of the packet determines the arbitration priority of the packet. You can enable this arbitration with Advance Mapping Mode. There are eight queues available to queue the packets for arbitration. Each queue is assigned with fixed priority 0 to 7, where queue 7 has the highest arbitration priority, and queue 0 has the lowest arbitration priority. Packets are stored into each queue according to the Priority Code Point (PCP) tag value extracted from the packet; for example, a packet with the PCP tag value of 7 is queued into queue 7.

The queue size for each priority must be configurable independently and use M20K to reduce ALM count. Queue depth must be configurable with a depth of 0/32/64/128/256 based on IP parameter selection per queue. The unused queue shall be configurable with the size of 0 to ensure no resource wastage. For queue 7, the minimum size is 32, and value 0 is not allowed to avoid a case where all queues are 0 sizes. When a particular queue size is configured to 0 (other than queue 7) and there is a request decoded with the PCP tag value of the queue, the request shall be routed to queue 7.

The ORAN C/U Plane packet uses the eCRPI IP Advance Mapping Mode feature to get the PCP value for the packet. When ORAN C/U Plane packet is sent into eCPRI IP, eCPRI IP uses the PCID sent as sideband and the packet to map to the VLAN tag register. The PCP values extracted from the VLAN tag register determine the queue to store the ORAN C/U Plane packet. The PCP field is a user input at packet SOP for S Plane, M-plane, and other traffic. Packets are stored in the queue according to the PCP field.

The PCP checker arbitrates the traffic (C/U/S/M/Other) based on the PCP tag value in the round-robin and stores it into the respective queue. Since the PTP software stack might not always generate PTP packets with the VLAN ID, the following statements explain the PCP values for handling each ITU-T profilethe the adetermine:

- The first two profiles (8264 and 8275.1 profiles) – No VLAN ID (IP assigns the priority based on the TX Packets Default Priority parameter, default to highest 7).
- The third profile (8275.2 profile) – parse PTP packet to extract PCP from L2/L3 header.
- The primary use case is based on the L2 header and not the L3 header (8275.2 profile) – parser can parse the L2 header only to determine PTP packets and no L3 header parsing.

When the arbitration queue is full, backpressure applies to both eCPRI IP and S/M packets DCFIFO. Backpressure shall only apply on the same priority request. For example, when queue 7 is full, back pressure happens on traffic with PCP = 7 only (can be S/U/C/M plane traffic). Due to one input from ORAN IP for the U plane, subsequent packets with different priorities are blocked when the head of traffic is blocked. If the queue full happens in the middle of the packet, you must submit the remaining packet to the queue before switching to other traffic.

Run time priority change on traffic: You can switch PCP field values at packet boundary only, priority should remain the same for whole packet.

There is no starvation handling in hardware. For example, low-priority traffic always gives way to high-priority traffic and waits for its turn. Therefore, you are expected to handle the traffic flow between different queues to ensure no starvation scenarios which causes low-priority traffic continuously to get stuck in the queue without getting the arbitration grant.

Each queue generates FIFO full port and sends it to the eCPRI IP interface.

4.2.2. Receive RX Path

The receiving Ethernet frames from the Ethernet MAC first enters packet classifier block. Packet classifier block classifies the packet into eCPRI packets and non-eCPRI packets. The packet classifier sends eCPRI packets with matching MAC address to the Ethernet header removal block of the eCPRI IP, while sends all other non-eCPRI packets or eCPRI packets with non-matching MAC address to external user logic for processing.

The **Data Flow Identification** parameter determines the reset default value of `match_macaddr_vlanid` register bit of `cu_vlanid_match_address` register. This register bit can be overridden during run time. You shall handle the Run time Data Flow Identification change gracefully (for example, empty the traffic before switching) to avoid indeterministic behavior or decision on the packet routing.

For detailed information on conditions when the IP classifies packet as eCPRI packet, refer to section *Packet Classifier*.

Related Information

Packet Classifier on page 24

4.2.3. Supported Ethernet Variants

The eCPRI Intel FPGA IP pairs together with the 25G/10G Ethernet. The eCPRI IP is validated together with the 25G Ethernet for Intel Stratix 10 designs.

For your Intel Stratix 10 designs, you can select **25G Ethernet Intel FPGA IP** for H-tile variants and **E-tile Hard IP for Ethernet Intel FPGA IP** for E-tile variants. When you use these IPs, you must set the following parameter values in the IP parameter editor:

- **Select Ethernet Rate** if you use **E-tile Hard IP for Ethernet Intel FPGA IP**.

  Note: This option is not available with **25G Ethernet Intel FPGA IP**. Use **Enable 10G/25G dynamic rate switching** for 10G data rate.
- Enable **Enable IEEE 1588** parameter to support client PTP message and eCPRI one-way delay measurement. The 25G Ethernet MAC only supports 96-bit (V2) timestamp format.
- Disable **Enable preamble pass-through** and **Enable TX CRC pass-through** parameters.
- Turn on **Enable 10G/25G dynamic rate switching** option to switch between 10G and 25G data rates.

For your Intel Arria 10 designs, you can use **Low Latency Ethernet 10G MAC Intel FPGA IP** and **1G/10GbE and 10GBASE-KR PHY Intel FPGA IP** to implement MAC and PHY respectively for your Ethernet.

Related Information

- 25G Ethernet Intel Stratix 10 FPGA IP User Guide
- E-tile Hard IP User Guide
- Low Latency Ethernet 10G MAC Intel FPGA IP User Guide

4.3. Operation of the eCPRI IP Blocks

The following section explains the operation of the eCPRI IP blocks.

4.3.1. Packet Classifier

The packet classifier parses the incoming Ethernet frame to identify the types of incoming packets. The incoming packets could be eCPRI packet, PTP packet, or C&M packet with different types of frames (e.g., standard Ethernet frame, IPv4, and etc.)

Packet classifier redirects eCPRI packets to next component for further processing and classifies a packet as eCPRI packet if all the condition listed in the table below met. The packet classifier sends all non-eCPRI packets and eCPRI packets with non-matching MAC address fields to external Avalon streaming interface.

Table 10. Ethernet Frame Format (User Data over Ethernet)

Number of Bits	RX Frame	Condition
48	MAC Destination address	Destination MAC address matches receiver source MAC address.
48	MAC Source address	Do not check.
32 (Optional)	VLAN tag	Packet parser checks for VLAN tag and adjust the offset accordingly.
32 (Optional)	Stack VLAN tag	Packet parser checks for SVLAN tag and adjusts the offset accordingly.
16	Ethertype (2 Bytes)= IP	Ethertype is equal to 0xAEFE

If eCPRI message transmitted over IP/UDP, the IP supports only IPv4 with UDP.

Table 11. Ethernet Frame Format with IPv4 (User Data over IP)

No. of Bits	IPv4 Header	Condition
48	MAC destination address	Destination MAC address matches receiver source MAC address.
48	MAC source address	Do not check.
32 (Optional)	VLAN Tag	Packet parser checks for VLAN tag and adjusts the offset accordingly.
32 (Optional)	Stack VLAN Tag	Packet parser checks for SVLAN tag and adjusts the offset accordingly.
16	Ethertype (2B) = IP	Ethertype must be 0x0800 for IPv4.
4	Version	Version must be 4'h4.
4	Internet Header Length	The value must be 4'h5. The IP does not support IPv4 "Options" field.
6	Differentiated Services Code Point (DSCP)	Do not check.
2	Explicit Congestion Notification (ECN)	Do not check.
16	Total length	Do not check.
16	Identification	Do not check.
3	Flags	Do not check.
13	Fragment offset	Do not check.
8	Time To live	Do not check.
8	Protocol	The protocol value must be equal to 0x11 for IPv4 with UDP.
16	Header checksum	Do not check.
32	Source address	Do not check.
32	Destination address	Destination IP address matches receiver source IP address.
16	Source port	Do not check.
16	Destination port	Destination port number matches receiver UDP port number.
16	Length	Do not check.
16	Checksum	Do not check.

4.3.2. Ethernet Header Insertion/Removal

The Ethernet header insertion block inserts Ethernet header to incoming eCPRI packet on TX path. Optionally it can insert IPv4/UDP headers to the packet based on the configuration. The Ethernet header removal block removes Ethernet header to incoming eCPRI packet on RX path. Optionally it can remove IPv4/UDP headers to the packet based on the configuration selected. The Ethernet header encapsulated the incoming eCPRI packets as shown in the figure below. The table listed the source of each fields within the Ethernet header.

Figure 7. Ethernet Header Field

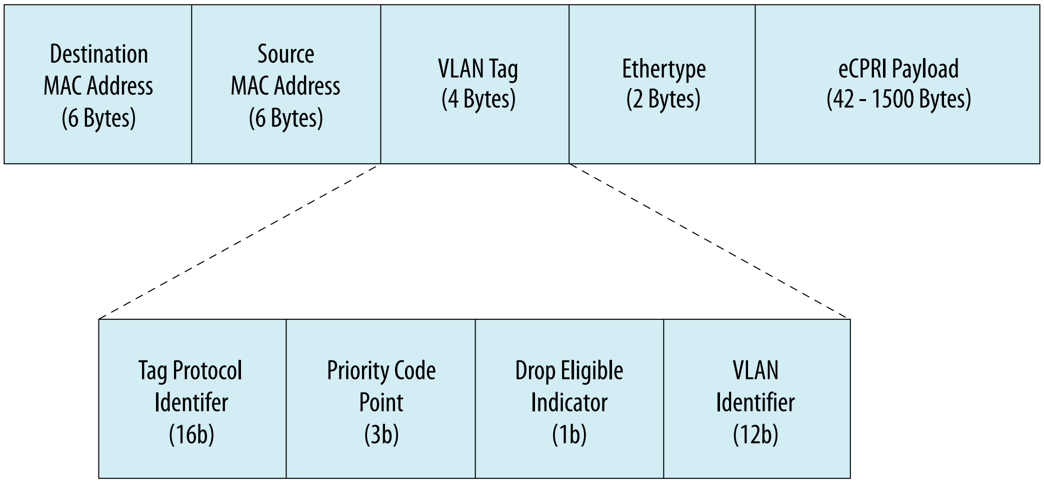

Destination MAC Address (6 Bytes)
Source MAC Address (6 Bytes)
VLAN Tag (4 Bytes)
Ethertype (2 Bytes)
eCPRI Payload (42 - 1500 Bytes)
Tag Protocol Identifer (16b)
Priority Code Point (3b)
Drop Eligible Indicator (1b)
VLAN Identifier (12b)

Table 12. Ethernet Header Field and CSR

Ethernet Header Field	CSR
Destination MAC address	Destination MAC address <N=0,1,2,3,4,5,6,7> Register 0,1 With enabled Advance mapping mode: N – eCPRI message PCID [2:0] Default mapping mode: N = 0 for all eCPRI message
Source MAC address	Source MAC address register 0, and 1
VLAN tag	VLAN Tag Register <N=0,1,2,3,4,5,6,7> With Enabled advance mapping mode: N – eCPRI message PCID [2:0] Default mapping mode: N = 0 for all eCPRI message
Ethertype	0xAEFE
eCPRI payload	Incoming eCPRI packet from the eCPRI IP

If you select IPv4 header as encapsulation to eCPRI payload, the following table lists the CSR to fill the IPv4 header fields:

Table 13. IPv4 Field and CSR

Number of Bits	IPv4 Header	CSR
48 (6 Bytes)	MAC destination Address	Refer to the *Table: Ethernet Header Field and CSR* above.
48 (6 Bytes)	MAC source address	Refer to the *Table: Ethernet Header Field* above.
16 (2 Bytes)	Ethertype (2 Bytes)= IP	0x0800
4	Version	`ipv4_dw0_address`
4	Internet header length	`ipv4_dw0_address`
6	Differentiated Services Code Point (DSCP)	`ipv4_dw0_address`

continued...

Number of Bits	IPv4 Header	CSR
2	Explicit Congestion Notification (ECN)	`ipv4_dw0_address`
16	Total length	eCPRI IP calculates incoming IP packet header length
16	Identification	`ipv4_dw1_address`
3	Flags	`ipv4_dw1_address`
13	Fragment offset	`ipv4_dw1_address`
8	Time to live	`ipv4_dw2_address`
8	Protocol	`ipv4_dw2_address`
16	Header checksum	eCPRI IP calculates incoming IP packet header checksum
32	Source address	`ipv4_src_address_0`
32	Destination address	`ipv4_dst_address_0`

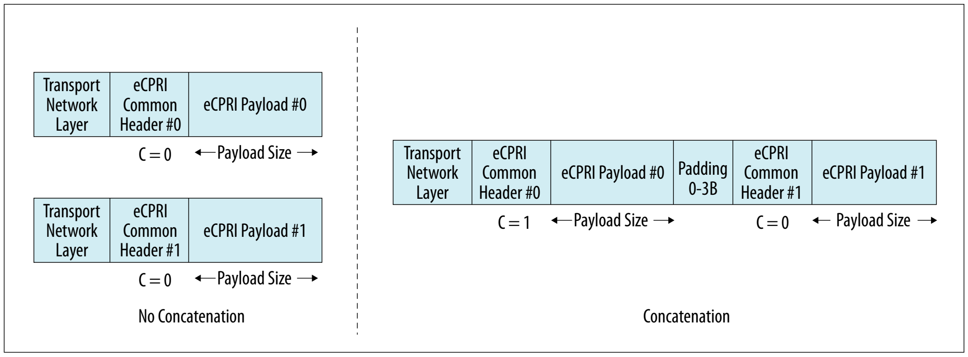
Table 14. UDP Field and CSR

Number of Bytes	IPv4 Header	CSR
14	Ethernet header	Refer to the *Table: Ethernet Header Field and CSR* above.
20	IPv4 header	Refer to the *Table: IPv4 Field and CSR Header* above.
2	Source port	`mudp_dw0_address`
2	Destination port	`mudp_dw0_address`
2	Length	eCPRI IP calculates incoming payload length
2	Checksum	eCPRI IP calculates incoming payload checksum

4.3.3. Concatenation/De-concatenation

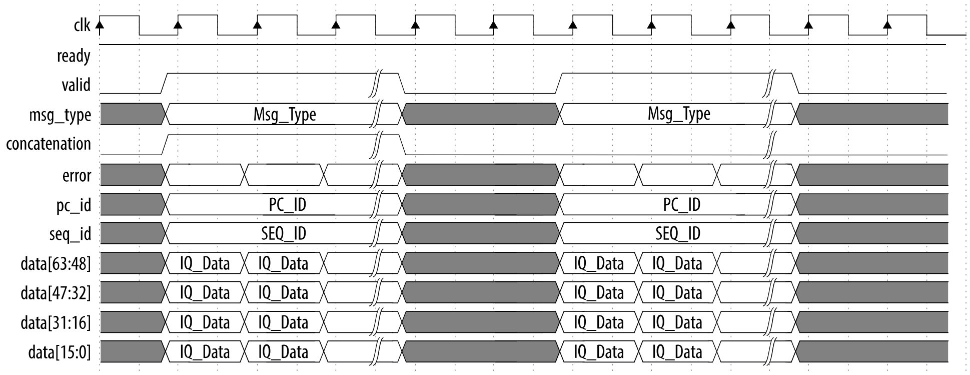
The Concatenation/De-concatenation blocks of the eCPRI IP implements concatenation logic of the eCPRI messages into single Ethernet frame or single IP/UDP packet. The `sink_concatenation` sideband signal identifies packets that required concatenation. The below diagrams illustrates the eCPRI messages with and without concatenation.

Figure 8. eCPRI Message Concatenation



When multiple eCPRI messages are concatenated together, 0 to 3 "zero" padding bytes are added if the following message does not start at a 4 byte boundary. The payload size specified in the eCPRI common header does not include this extra zero padding bytes.

Figure 9. Concatenation/De-concatenation Example Waveform



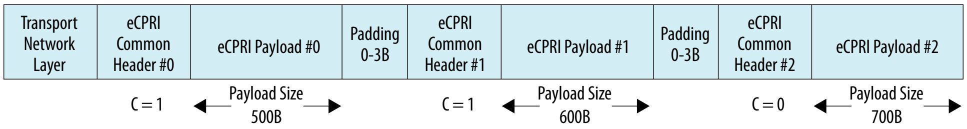
The above waveform shows two incoming eCPRI messages entering to eCPRI IP, first eCPRI message with concatenation sideband interface signal = 1 and the second eCPRI message with concatenation sideband interface signal = 0. These two eCPRI messages are combined and send through single transport network layer protocol. On the receiving end, the combined eCPRI message will then de-concatenate into 2 eCPRI messages and output to Avalon streaming interface.

There is a timeout counter used to detect the end of the concatenation message. If the counter overflows and no message with C=0 is detected, an error will be logged and the message with C=1 will be converted to message with C=0 and send to MAC.

The message type allowed for concatenation is restricted to message type 0,1,2,3 and 6. The de-concatenation is supported on all message type except message type 5.

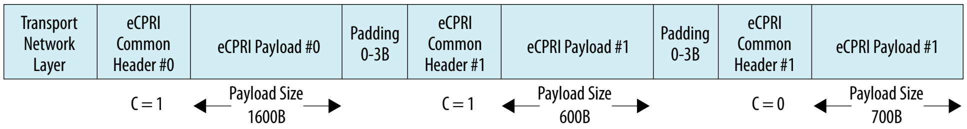
There are 2 different eCPRI packet concatenation scenarios which trigger error and it is shown in below diagram.

Figure 10. eCPRI Packet Concatenation Scenario 1



In the first scenario, there are three incoming Avalon streaming interface packets payload size of 500 bytes, 600 bytes and 700 bytes. The total payload size after concatenation is 1800 bytes which is bigger than maximum eCPRI IP supported maximum transmission unit (MTU) size of 1500 bytes. In this case, error will be logged in the eCPRI TX error message register and payload 0 and 1 will be sent as concatenated packets while payload 2 will be sent by itself.

Figure 11. eCPRI Packet Concatenation Scenario 2



In the second scenario, the first packet payload size is more than 1500 bytes. In this case, all the packets drop and error logged in eCPRI TX error message register.

4.3.4. Header Mapper/De-Mapper

The Header mapper/De-mapper block append or remove the eCPRI common header from the eCPRI message. The Mapper block calculates the payload size of the incoming Avalon streaming interface packet and append it into the packet as part of the eCPRI common header field. The table below shows the eCPRI common header format. The eCPRI protocol version is a read only field and the **Protocol Revision** parameter determines the value of this field. The concatenation and message type are determined from the Avalon streaming interface sink sideband interface signals which come along with eCPRI message. The payload size is calculated when the eCPRI message enter eCPRI IP at Avalon streaming interface interface.

Table 15. eCPRI Common Header Format

0 (MSB)	1	2	3	4	5	6	7 (LSB)	No. of Bytes
eCPRI Protocol Version= 0001b				Reserved			Concatenation	1
eCPRI Message Type								1
eCPRI Payload Size								2

4.3.5. eCPRI IWF Type 0

The eCPRI IWF type 0 converts eCPRI message type to CPRI protocol. This block allows the interface between eCPRI transport network with CPRI node(s).

Note: Only eCPRI message type 0, 2, 6, and 7 can be converted to CPRI.

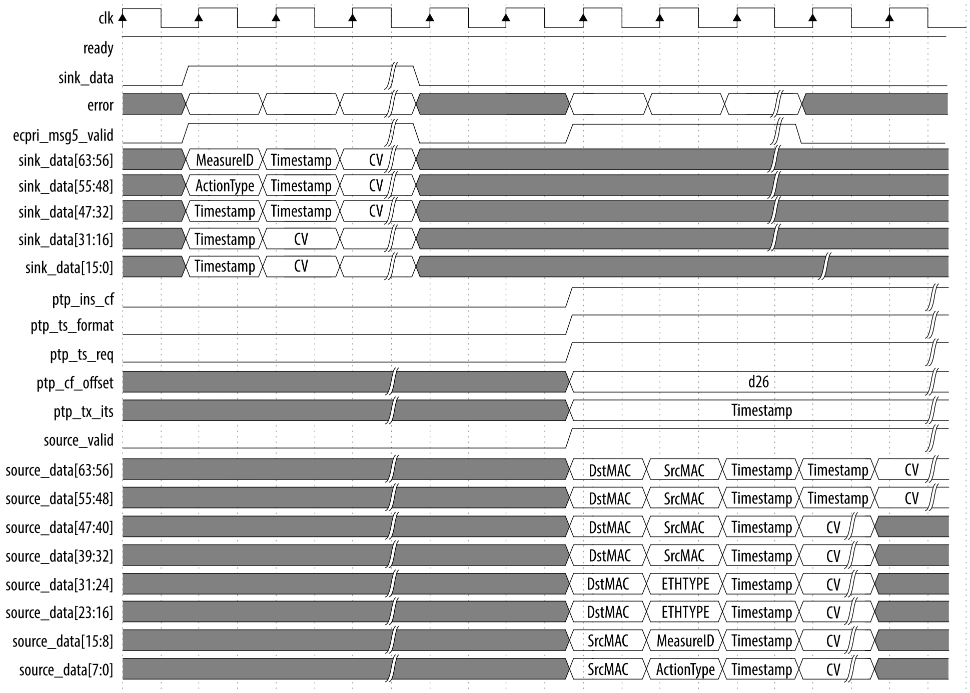
4.3.6. eCPRI Message 5 Packet Parser

This block is responsible to initiate and calculate the eCPRI one-way delay measurement on the transport link. The eCPRI one-way delay measurement can be performed without (one-step) or with a follow-up message (two-step). The process is initiated when a CSR is written to eCPRI Message 5 Control Register. The IP transmits eCPRI message 5 with one-step first before sending eCPRI message 5 with timestamp t1 and cv1 for two-step delay measurement. The packet parser assembles an eCPRI message 5 with timestamp t1 taken from Time-of-Day (TOD) module. Then, this eCPRI message 5 is sent through the Ethernet MAC with compensation value cv1 filled using 1588 PTP hardware.

On the receiving end, the eCPRI IP responses the message 5 with t2 and cv2. Upon receiving the response packet, this calculates the transport delay using the formula:$t_{D12}= (t_2-+t_{CV2}) - (t_1 + t_{CV1})$

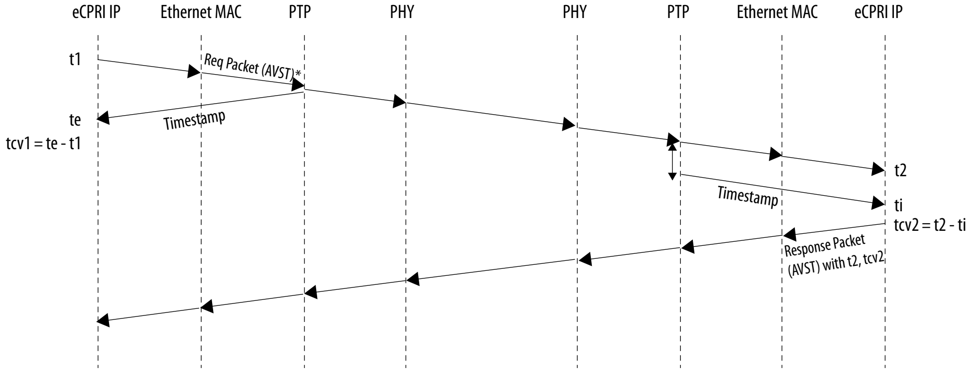
The waveform below shows an example of the Avalon streaming interface source and sink data through L2/L3 parser. The example in this section uses E-tile Ethernet Hard IP with 1588 PTP feature enabled.

Figure 12. Timing Diagram of One-Way Delay Measurement Example



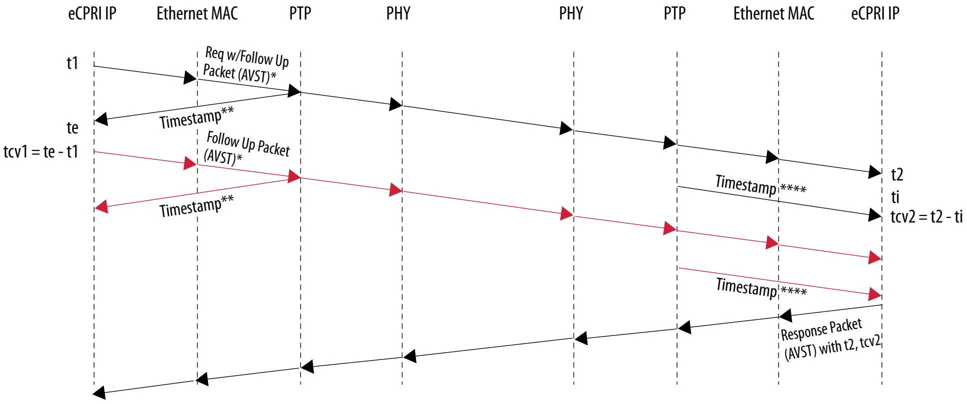
The timing diagram below illustrates the eCPRI message 5 in one-step one way delay measurement.

Figure 13. Timing Diagram of eCPRI Message Type 5 in one-step



The timing diagram below illustrates the eCPRI message 5 in two-step one way delay measurement.

Figure 14. Timing Diagram of eCPRI Message Type 5 in two-step



The one and two- steps one way delay measurement sequences uses same remote request type. The only difference is destination eCPRI IP measures t1 and tcv1 while the source eCPRI IP measures t2 and tcv2.

4.3.7. Packet Queue

This block is responsible to stage user incoming Ethernet frames (e.g., Control and Management packets, synchronization packets & etc) and arbitrate with eCPRI packets. These user Ethernet frames share the same Ethernet link with eCPRI packets. eCPRI IP does not encapsulate Ethernet header to these frames.

4.3.8. eCPRI Message Type

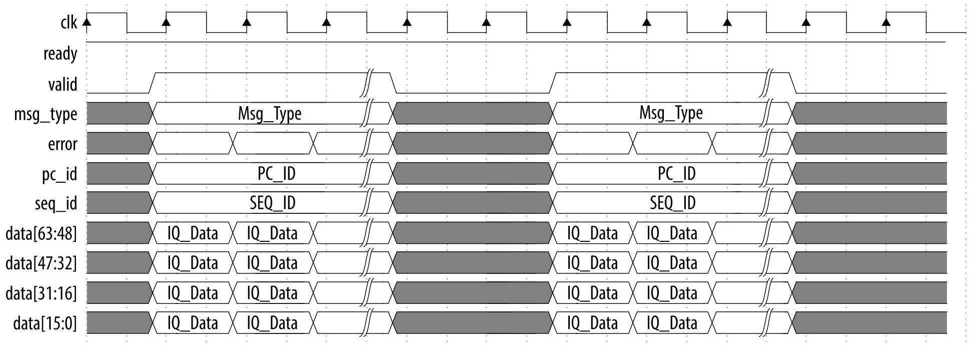
This section covers information about different types of eCPRI messages supported by eCPRI Intel FPGA IP.

4.3.8.1. eCPRI Message Type 0- IQ Data Transfer

Table 16. eCPRI Message Type 0- IQ Data Transfer Message Format

0 (MSB)	1	2	3	4	5	6	7 (LSB)	No. of Bytes
PC_ID								2
SEQ_ID								2
IQ_DATA								L

Figure 15. eCPRI Message Type 0- IQ Data Transfer Message Timing Diagram



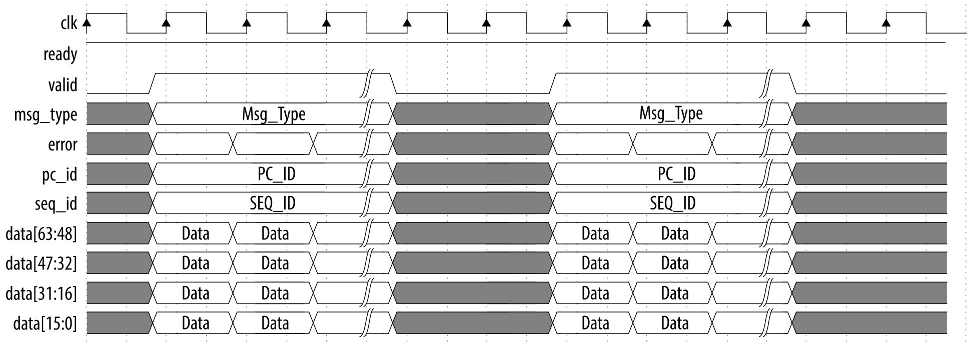
Note: The PC_ID, and SEQ_ID fields are 2 bytes wide. The PC_ID and PC_ID sideband interfaces are 4 bytes wide, so the MSB 2 bytes are set to zero.

4.3.8.2. eCPRI Message Type 1- Bit Sequence Transfer

Table 17. eCPRI Message Type 1- Bit Sequence Transfer Message Format

0 (MSB)	1	2	3	4	5	6	7 (LSB)	No. of Bytes
PC_ID								2
SEQ_ID								2
Bit Sequence of User Data								L

Figure 16. eCPRI Message Type 1 – Bit Sequence Transfer Message Timing Diagram

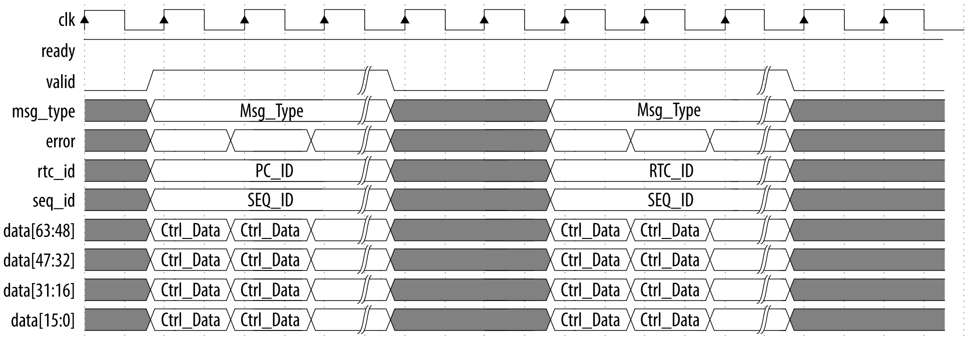


4.3.8.3. eCPRI Message Type 2- Real Time Control Data

Table 18. eCPRI Message Type 1- Real Time Control Data Message Format

0 (MSB)	1	2	3	4	5	6	7 (LSB)	No. of Bytes
PC_ID								2
SEQ_ID								2
Real time control data								L

Figure 17. eCPRI Message Type 2 – Real Time Control Data Message Timing Diagram

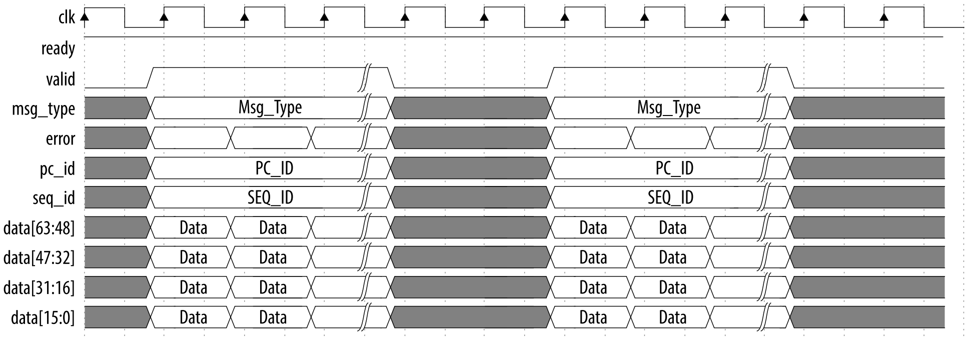


4.3.8.4. eCPRI Message Type 3- Generic Data Transfer

Table 19. eCPRI Message Type 3- Generic Data Transfer Message Format

0 (MSB)	1	2	3	4	5	6	7 (LSB)	No. of Bytes
PC_ID								2
SEQ_ID								2
Data transferred								L

Figure 18. eCPRI Message Type 3 – Generic Data Transfer Message Timing Diagram



4.3.8.5. eCPRI Message Type 4- Remote Memory Access

Table 20. eCPRI Message Type 4- Remote Memory Access Message Format

0 (MSB)	1	2	3	4	5	6	7 (LSB)	No. of Bytes
Remote Memory Access ID								1
Read/Write				Req/Resp				1
Element ID								2
Address								6
Length								2
Data								L

The eCPRI IP core supports two different modes for remote memory access message type: basic mode and buffer mode. The basic mode provides direct tunneling on the memory access, with all the necessary information for the memory access output to user logic. User logic is responsible to send the same ID/Element ID upon responding to the original request.

The buffer mode keeps the receiving request ID/Element ID/Address/read/write operations per table below. User logic carries out the operation and response to the original request with number of read/write bytes and read data. eCPRI IP appends necessary fields and then send the eCPRI message back to sender.

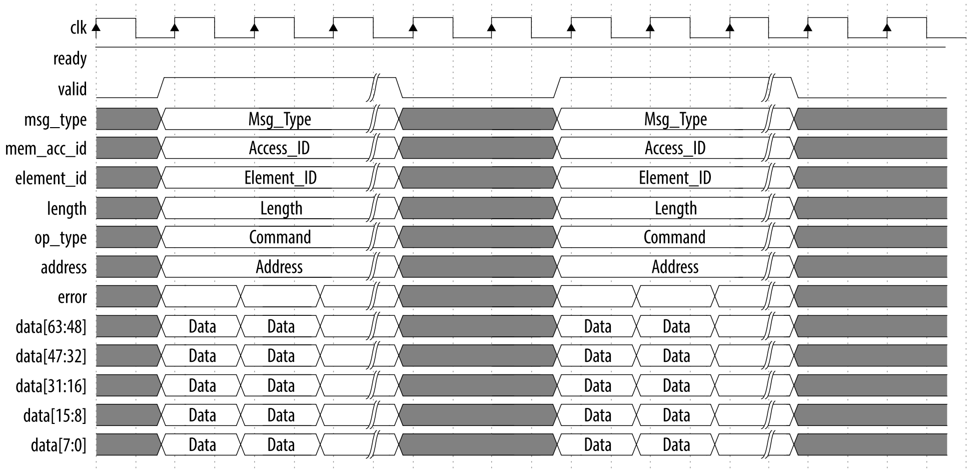
eCPRI IP expects the request and response to be in-order for this buffer mode. eCPRI IP can hold up to a maximum of eight pending requests. If there are eight pending requests in the queue and there is additional request received, the IP drops the additional request and logs error in RX error register.

eCPRI IP operates in basic mode by default.

Table 21. Parameter Handling

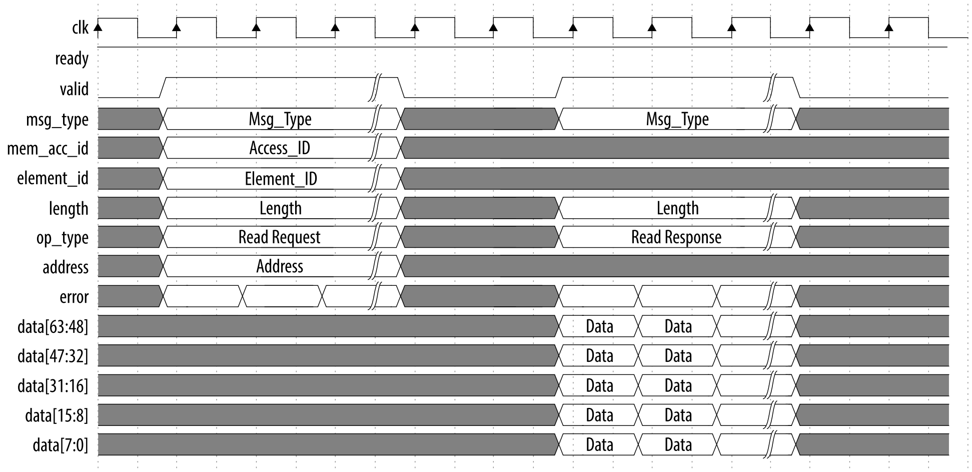
Action	ID	Read/Write	Request/ Response	Element ID	Address	Length	Data
Read request	Set	Set to read	Set to request	Set	Set	Set	No data
Read response	Copied	Copied	Set to response	Copied	Copied	No. of read bytes	Read data
Write request	Set	Set to write	Set to request	Set	Set	Set	The data to be written
Write response	Copied	Copied	Set to response	Copied	Copied	No. of written bytes	Vendor specific
Write no response	Set	Set to write no response	Set to request	Set	Set	Set	The data to be written
Failure response	Copied	Copied	Set to Failure	Copied	Copied	Vendor specific	Vendor specific

Figure 19. eCPRI Message Type 4 – Remote Memory Access in Basic Mode Message Timing Diagram



The waveform below illustrates the buffer mode where read request is sent to user logic. User logic response with read data and the actual length of the operation and message type 4. eCPRI IP extracts the memory access ID/element ID and address from internal buffer and combine with read data to send back to sender.

Figure 20. eCPRI Message Type 4 – Remote Memory Access Message in Buffer Mode Timing Diagram



4.3.8.6. eCPRI Message Type 5- One-Way Delay Measurement

Table 22. eCPRI Message Type 5- One Way Delay Measurement Message Format

0 (MSB)	1	2	3	4	5	6	7 (LSB)	No. of Bytes
Measurement ID								1
Action Type								1
Timestamp								10
Compensation Value								8
Dummy Bytes								L

Related Information

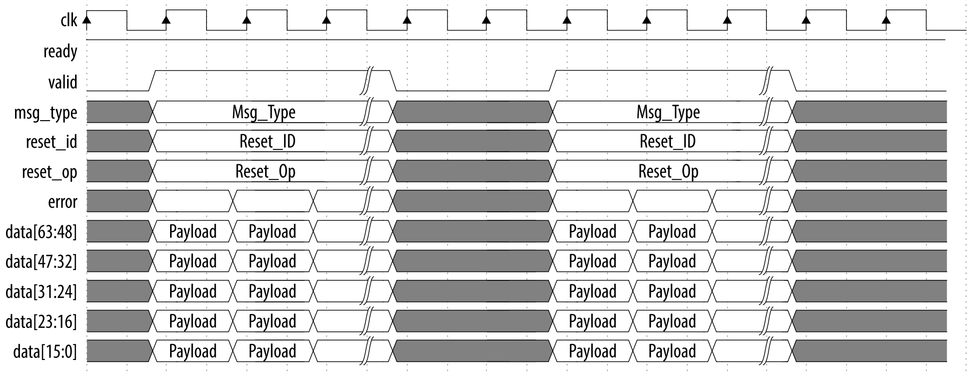
eCPRI Message 5 Packet Parser on page 29
Refer to this section for more information on one-way delay measurement.

4.3.8.7. eCPRI Message Type 6- Remote Reset

Table 23. eCPRI Message Type 6- Remote Reset Message Format

0 (MSB)	1	2	3	4	5	6	7 (LSB)	No. of Bytes
Reset ID								2
Reset Code Op								1
Payload								L

Figure 21. eCPRI Message Type 6 – Remote Reset Message Timing Diagram



Related Information

eCPRI Specification V2.0

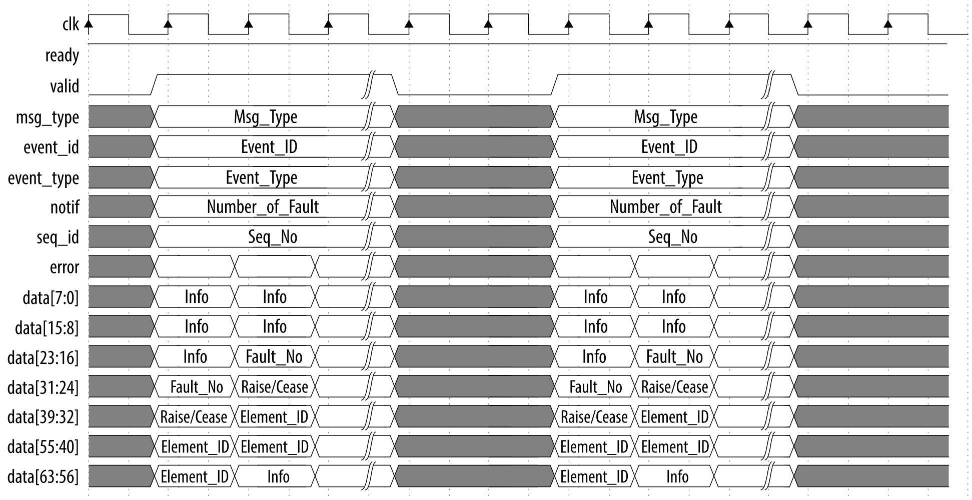
Refer to this specification for information on Op code.

4.3.8.8. eCPRI Message Type 7- Event Indication

Table 24. eCPRI Message Type 7- Event Indication Message Format

0 (MSB)	1	2	3	4	5	6	7 (LSB)	No. of Bytes
Event ID								1
Event Type								1
Sequence Number								1
Number Of Faults/Notif = N								1
Element ID # N								2
Raise/Cease #N				Fault/Notif #N MSB				1
Fault/Notif #N LSB								1
Additional Information								4

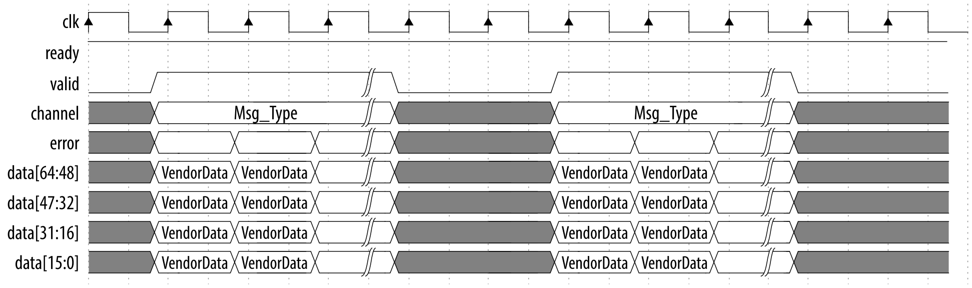
Figure 22. eCPRI Message Type 7 – Event Indication Message Timing Diagram



4.3.8.9. eCPRI Message Type 64- 255 Vendor Specific

Vendor specific eCPRI message types are not defined in the eCPRI specification. The eCPRI IP allows a direct pass through for vendor specific data. The output of the eCPRI IP for this message type is eCPRI common header and vendor specific data.

Figure 23. eCPRI Message Type 64 – 255 Vendor Specific Message Timing Diagram



4.3.9. Error Handling

Table 25. Error Condition Behavior

Events	Hardware Logging	Mitigations
Invalid measurement ID received on eCPRI message type 5.	Log last error measurement ID and action type in eCPRI RX error message register.	None
Timeout no response for eCPRI message type 5.	Log pending measurement ID and action type in eCPRI RX error message register.	None
Timeout no end of concatenation message received.	eCPRI TX error message register.	Convert last message to C=0 and send out the messages.
Invalid eCPRI message types. The invalid message types are 8 to 63.	Log last error message type in eCPRI TX error message register	eCPRI message drop.

continued...

Events	Hardware Logging	Mitigations
When you enable **Pair with ORAN** parameter, the IP only supports message type 0, 2 and 5. All other message types are invalid.		
Invalid message type 5 action types.	eCPRI RX error message register	None
Multiple message concatenation size is greater than MTU.	eCPRI TX error message register.	Split the messages into 2 or more PDU and send out the messages.
Single message concatenation size is greater than MTU.	eCPRI RX error message register.	eCPRI message drop.
Timeout no reset access response.	eCPRI RX error message register.	None
Timeout no memory access response	Log last memory access ID and op code in eCPRI RX error message register.	None
Missing SOP	eCPRI TX/RX error message register.	Incoming data drop.
Missing EOP	eCPRI TX/RX error message register.	Incoming data drop. When the Ethernet frame size set to 9000 bytes, there is no message drop but the eCPRI IP sets Avalon streaming interface source error.
Buffer overflow	eCPRI TX/RX error message register.	None
M20K ECC	eCPRI TX/RX error message register.	None
RX eCPRI payload length not match payload size.	eCPRI RX error message register	eCPRI error asserted only on the last packet of the concatenated packet and Avalon streaming interface error asserted at EOP. That means earlier packet(s) have integrity issues.
RX eCPRI invalid concatenation bit.	eCPRI Rx Error Message Register	eCPRI error asserted only on the last packet of the concatenated packet and Avalon streaming interface error asserted at EOP. That means earlier packet(s) have integrity issues.
TX Avalon streaming interface does not follow the requirement. The eCPRI IP performs this check only in streaming mode.	User Avalon streaming interface error register	eCPRI IP asserts `mac_source_error` at EOP.
TX Avalon streaming interface packet size does not match with the user supply packet size The eCPRI IP performs this check only in streaming mode.	Invalid eCPRI sink packet size register	eCPRI IP asserts `mac_source_error` at EOP.
Receiving an error packet from the MAC for message type 5	RX message 5 error register	Incoming packets drop.

The eCPRI IP behaves as follows upon observing timeout error due to multiple no memory access responses in receiver:

- When first request timeout due to no response, first timeout counter stops counting and error interrupt triggered.
- Second request timeout due to no response happen, second timeout counter stops counting as well. Now there are two errors pending in IP.
- Software service the interrupt routine and determine the error is due to timeout no response. The software clears the error.
- Interrupt is deasserted and then asserted again due to second error and software handling of error is repeated.

It is software responsibility to handle the timeout error to avoid software hang due to pending memory access response.

Related Information

eCPRI Specification V2.0
Refer to this specification for information on Op code.

4.3.10. RX Throttling

The eCPRI Intel FPGA IP doesn't support throttling on the RX side (from 25G Ethernet MAC to eCPRI). Packets from Ethernet MAC are continuously streamed out either to external-ST source interface or eCPRI IP source interface and you should allocate enough buffer to hold the packets.

5. Interface Overview

The eCPRI IP core communicates with the surrounding design though multiple external signals.

5.1. Clock Signals

Table 26. eCPRI IP Input Clocks

<table>
<tr><th>Signal Name</th><th>Width (Bits)</th><th>I/O Direction</th><th>Description</th></tr>
<tr><td><code>clk_tx</code></td><td>1</td><td>Input</td><td>eCPRI IP TX clock.
For 25G eCPRI data rate variations, the default frequency value is 390.625 MHz.
For 10G eCPRI data rate variations, the default frequency value is 156.25 MHz.</td></tr>
<tr><td><code>clk_rx</code></td><td>1</td><td>Input</td><td>eCPRI IP RX clock.
For 25G eCPRI data rate variations, the default frequency value is 390.625 MHz.
For 10G eCPRI data rate variations, the default frequency value is 156.25 MHz.</td></tr>
<tr><td><code>mac_clk_tx</code></td><td>1</td><td>Input</td><td>Ethernet MAC TX clock.
The frequency of <code>mac_clk_tx</code> depends on device and data rate:
<table>
<tr><th>Device</th><th>Data Rate</th><th><code>mac_clk_tx</code> Frequency(in)</th></tr>
<tr><td>Intel Agilex (E-tile)</td><td>25G</td><td>402.835 MHz</td></tr>
<tr><td></td><td>10G</td><td>161.32 MHz</td></tr>
<tr><td>Intel Agilex (F-tile)</td><td>25G</td><td>402.835 MHz</td></tr>
<tr><td></td><td>10G</td><td>161.32 MHz</td></tr>
<tr><td>Intel Stratix 10 (H-tile)</td><td>25G</td><td>390.625 MHz</td></tr>
<tr><td></td><td>10G</td><td>156.25 MHz</td></tr>
<tr><td>Intel Stratix 10 (E-tile)</td><td>25G</td><td>402.835 MHz</td></tr>
<tr><td></td><td>10G</td><td>161.32 MHz</td></tr>
<tr><td>Intel Arria 10</td><td>10G</td><td>156.25 MHz</td></tr>
</table></td></tr>
<tr><td><code>mac_clk_rx</code></td><td>1</td><td>Input</td><td>Ethernet MAC RX clock.
• For the Intel Stratix 10 H-tile IP variations, the default frequency value is 390.625 MHz.
• For the Intel Stratix 10 E-tile IP variations, the default frequency value is 402.835 MHz.
• For the Intel Agilex E-tile and F-tile IP variations, the default frequency value is 402.835 MHz.</td></tr>
</table>

continued...

<table>
<tr><th>Signal Name</th><th>Width (Bits)</th><th>I/O Direction</th><th>Description</th></tr>
<tr><td>clk_csr</td><td>1</td><td>Input</td><td>CSR clock. The default frequency value can be 100 MHz to 162 MHz.</td></tr>
<tr><td>ext_sink_clk</td><td>1</td><td>Input</td><td>External user interface clock. The frequency value is greater than or equal to 390.625 MHz.</td></tr>
<tr><td>cpri_clkout[N]</td><td>1</td><td>Input</td><td>Master clock for the CPRI IP core.
The frequency of cpri_clkout[N] depends on the CPRI line bit rate:
<table>
<tr><th>CPRI Line Bit Rate</th><th>cpri_clkout[N] Frequency</th></tr>
<tr><td>0.6144 Gbps</td><td>15.36 MHz</td></tr>
<tr><td>1.2288 Gbps</td><td>30.72 MHz</td></tr>
<tr><td>2.4576 Gbps</td><td>61.44 MHz</td></tr>
<tr><td>3.0720 Gbps</td><td>76.80 MHz</td></tr>
<tr><td>4.9152 Gbps</td><td>122.88 MHz</td></tr>
<tr><td>6.1440 Gbps</td><td>153.6 MHz</td></tr>
<tr><td>8.11008 Gbps</td><td>245.76 MHz</td></tr>
<tr><td>9.8304 Gbps</td><td>245.76 MHz</td></tr>
<tr><td rowspan="2">10.1376 Gbps</td><td>153.60 MHz[1]</td></tr>
<tr><td>307.20 MHz[2]</td></tr>
<tr><td>12.16512 Gbps</td><td>184.32 MHz</td></tr>
<tr><td>24.33024 Gbps</td><td>368.64 MHz</td></tr>
</table></td></tr>
<tr><td>iwf_gmii_rxclk[N]</td><td>1</td><td>Input</td><td rowspan="2">iwf_gmii_txclk clocks the GMII transmitter interface and iwf_gmii_rxclk clocks the GMII receiver interface. You must drive these clocks at the frequency of 125 MHz to achieve the 1000 Mbps bandwidth required for this interface.
These clocks are present only if you set the value of <b>Ethernet PCS</b> interface to the value of <b>GMII</b> in the CPRI parameter editor.</td></tr>
<tr><td>iwf_gmii_txclk[N]</td><td>1</td><td>Input</td></tr>
<tr><td>gmii_rxclk[N]</td><td>1</td><td>Output</td><td rowspan="2">gmii_txclk clocks the GMII transmitter interface and gmii_rxclk clocks the GMII receiver interface. You must drive these clocks at the frequency of 125 MHz to achieve the 1000 Mbps bandwidth required for this interface.
These clocks are present only if you set the value of <b>Ethernet PCS</b> interface to the value of <b>GMII</b> in the CPRI parameter editor.</td></tr>
<tr><td>gmii_txclk[N]</td><td>1</td><td>Output</td></tr>
</table>

[1] For Intel Agilex E-tile and F-tile and Intel Stratix 10 E-tile device variations.

[2] For all other device variations.

5.2. Power, Reset, and Firewalls Signals

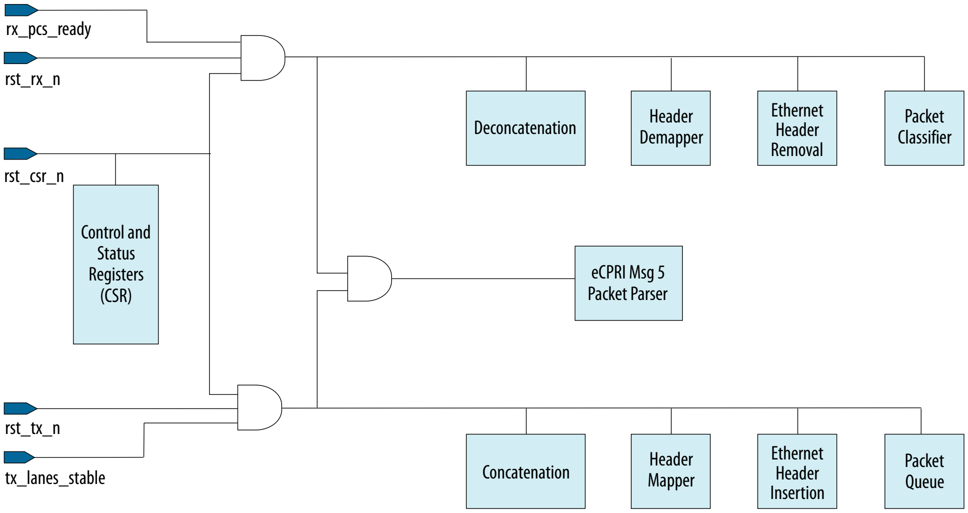
Table 27. eCPRI IP Reset, Power, and Firewalls Signals

These signals are asynchronous.

Signal Name	Width (Bits)	I/O Direction	Description
rst_tx_n	1	Input	Reset signal from Ethernet MAC TX. Resets the eCPRI IP in RX direction. Resets the De-concatenation, Header mapper/De-mapper, Ethernet header removal, eCPRI message 5 packet parser and, Packet classifier.
rst_rx_n	1	Input	Reset signal from Ethernet MAC RX. Resets the eCPRI IP in TX direction. Resets the Concatenation, Header mapper/De-mapper, Ethernet header insertion, eCPRI message 5 packet parser, and Packet queue.
rst_csr_n	1	Input	Reset signal for CSR logic. Resets the eCPRI IP control and status registers. When asserted, resets the eCPRI IP.
tx_lanes_stable	1	Input	Signal that indicates the clk_tx signal from MAC is stable and ready for operation.
rx_pcs_ready	1	Input	Signal that indicates the clk_rx signal from MAC is stable and ready for operation.
iwf_rst_tx_n	1	Input	Reset signal for the IWF TX path.
iwf_rst_rx_n	1	Input	Reset signal for the IWF RX path.
rst_tx_n_sync	1	Output	Reset output from IWF. This signal is synchronous to clk_tx. Intel recommends you to connect this signal to iwf_rst_tx_n.
rst_rx_n_sync	1	Output	Reset output from IWF. This signal is synchronous to clk_rx. Intel recommends you to connect this signal to iwf_rst_rx_n.
iwf_gmii_rxreset_n[N]	1	Input	Resets the GMII receiver interface and FIFO read logic.
iwf_gmii_txreset_n[N]	1	Input	Resets the GMII transmitter interface and FIFO write logic.
gmii_rxreset_n[N]	1	Output	Resets the GMII receiver interface and FIFO read logic.
gmii_txreset_n[N]	1	Output	Resets the GMII transmitter interface and FIFO write logic.

5.2.1. Reset Control and Initialization Flows

Figure 24. eCPRI IP Core Reset Logic



Three reset ports of the eCPRI IP assert together to fully reset the eCPRI IP. The deassertion of these three signals can happen together or the IP can just deassert `rst_csr_n` signal follows by `rst_tx_n` and `rst_rx_n` signals depending on use case.

You should perform reset before beginning IP core operation. Alternatively, you can trigger reset after you reconfigure the eCPRI IP during run time.

Reset Length Requirement

You need to assert reset signals for additional ten cycles after `tx_lanes_stable` and `rx_pcs_ready` signals are asserted to ensure the Ethernet MAC clocks are stable and run at designated speed. The `avst_sink_ready`, and `mac_sink_ready` signals are asserted when the IP core exists from reset successfully and ready to accept client data.

5.3. TX Time of Day Interface

Table 28. Signals of the TX Time of Day Interface

All signals are synchronous to `clk_tx` clock.

Signal Name	Width (Bits)	I/O Direction	Description
`tx_tod_time_of_day_96b_data`	96	Input	Current V2-format (96-bit) TOD in `clk_txmac` clock domain.
`tx_egress_timestamp_96b_data`	96	Input	Provides the V2-format timestamp when a 1588 PTP frame begins transmission on the Ethernet link. Value is valid when the `tx_egress_timestamp_96b_valid` signal is asserted.

continued...

Signal Name	Width (Bits)	I/O Direction	Description
			This signal is present only in two-step clock mode.
tx_egress_timestamp_96b_valid	1	Input	Indicates that the tx_egress_timestamp_96b_data and tx_egress_timestamp_96b_fingerprint signals are valid in the current clk_txmac clock cycle. This signal is present only in two-step clock mode.
tx_egress_timestamp_96b_fingerprint	8	Input	Provides the fingerprint of the V2-format 1588 PTP frame currently beginning transmission on the Ethernet link. Value is valid when the tx_egress_timestamp_96b_valid signal is asserted. The encoding format is: • Bit [6:0]: PTP Fingerprint ID • Bit [7]: PTP/eCPRI
ext_tx_egress_timestamp_96b_data	96	Output	Provides the V2-format timestamp when a 1588 PTP frame begins transmission on the Ethernet link. Value is valid when the tx_egress_timestamp_96b_valid signal is asserted.
ext_tx_egress_timestamp_96b_valid	1	Output	Indicates that the ext_tx_egress_timestamp_96b_data signals are valid in the current ext_sink_clk clock cycle. This signal is meaningful only in two step clock mode.

Related Information

- 1588 PTP Interface Signals
  For more information on 1588 PTP signals for the 25G Ethernet Intel Stratix 10 IP.
- 1588 PTP Interface
  For more information on 1588 PTP signals for the E-tile Hard IP for Ethernet.

5.4. RX Time of Day Interface

Table 29. Signals of the RX Time of Day Interface

All signals are synchronous to clk_rx clock.

Signal Name	Width (Bits)	I/O Direction	Description
rx_tod_time_of_day_96b_data	96	Input	Current V2-format (96-bit) TOD in clk_rxmac clock domain.
rx_ingress_timestamp_96b_data	96	Input	Whether or not the current packet on the RX client interface is a 1588 PTP packet, indicates the V2-format timestamp when the IP core received the packet on the Ethernet link. The IP

continued...

Signal Name	Width (Bits)	I/O Direction	Description
			core provides a valid value on this signal in the same cycle it asserts the RX SOP signal for 1588 PTP packets.
`rx_ingress_timestamp_96b_valid`	1	Input	Indicates that the `rx_ingress_timestamp_96b_data` signal is valid in the current cycle. This signal is redundant with the RX SOP signal for 1588 PTP packets.
`ext_rx_ingress_timestamp_96b_data`	96	Output	Indicates V2-format timestamp when the IP core receives the RX packet on the Ethernet link. The IP core provides a valid value on this signal in the same cycle it asserts the RX SOP signal for 1588 PTP packets.

Related Information

- 1588 PTP Interface Signals
  For more information on 1588 PTP signals for the 25G Ethernet Intel Stratix 10 IP.
- 1588 PTP Interface
  For more information on 1588 PTP signals for the E-tile Hard IP for Ethernet.

5.5. Interrupt

Table 30. Interrupt Signals

This signal is synchronous to `clk_csr` signal.

Signal Name	Width (Bits)	I/O Direction	Description
`err_interrupt`	1	Output	Error interrupt signal. Indicates errors occur in the eCPRI IP. Software can poll eCPRI error message register to determine the error info.

5.6. Configuration Avalon Memory-Mapped Interface

Table 31. Signals of the Configuration Avalon Memory-Mapped Interface

This section lists ports that provides access to internal control and status registers of the eCPRI IP. All signals are synchronous to `clk_csr`.

Signal Name	Width (Bits)	I/O Direction	Description
`csr_address`	16	Input	Configuration register address.
`csr_write`	1	Input	Configuration register write enable.
`csr_writedata`	32	Input	Configuration register write data.
`csr_read`	1	Input	Configuration register read enable.
`csr_readdata`	32	Output	Configuration register read data.
`csr_writerequest`	1	Output	Configuration register write request.
`csr_readdatavalid`	1	Output	Configuration register read data valid.

5.7. Ethernet MAC Source Interface

Table 32. Signals of the 25G Ethernet MAC Avalon streaming interface Source Interface

This section lists port from eCPRI IP to 25G Ethernet MAC. All signals are synchronous to `mac_clk_tx`.

Signal Name	Width (Bits)	I/O Direction	Description
`mac_source_valid`	1	Output	Indicates Avalon source valid from eCPRI to Ethernet MAC.
`mac_source_data`	DATA_WIDTH[3]	Output	Indicates Avalon source write data from eCPRI to Ethernet MAC.
`mac_source_sop`	1	Output	Indicates Avalon source start of packet (SOP) from eCPRI to Ethernet MAC. Indicate the beginning of packet.
`mac_source_eop`	1	Output	Avalon source end of packet (EOP) from eCPRI to Ethernet MAC. Indicate the end of packet.
`mac_source_empty`	LOG2(DATA_WIDTH[3]/8)	Output	Avalon source empty from eCPRI to Ethernet MAC. Indicates the number of symbols that are empty, that is, do not represent valid data.
`mac_source_ready`[4]	1	Input	Avalon source ready driven from Ethernet MAC. Indicate Ethernet MAC can accept data.
`mac_source_error`	1	Output	Avalon source error from eCPRI to Ethernet MAC. A bit mask to mark errors affecting the data being transferred in the current cycle.

5.7.1. E-tile Hard IP for Ethernet 1588 PTP Signals

Table 33. Signals of the E-tile Hard IP for Ethernet 1588 PTP Interface

All signals are synchronous to `clk_tx` clock.

Signal Name	Width	Direction	Description
`ptp_timestamp_insert`	1	Output	Inserts an egress timestamp into the current TX Packet on the respective channel. Valid only when the TX valid and TX SOP signals are asserted.
`ptp_tx_etstamp_ins_ctrl_residence_time_update`	1	Output	When asserted, inserts a residence time timestamp into the correction field in the current TX packet on the respective channel. Valid only when the TX valid and TX SOP signals are asserted.
`i_ptp_zero_csum`	1	Output	When asserted, overwrites the checksum in a UDP packet carried inside the current TX packet with zeros during IPv4. Valid only when the TX valid and TX SOP signals are asserted.
`i_ptp_update_eb`	1	Output	When asserted, overwrites the extended bytes field in an IPv6 packet carried inside the current TX packet with a value that cancels out changes to the checksum due to changes to the UDP packet. Valid only when the TX valid and TX SOP signals are asserted.

continued...

[3] This is set to 64. This parameter is hidden from user and you can't change it.

[4] This signal has READY_LATENCY of 3 clock cycles.

Signal Name	Width	Direction	Description
i_ptp_ts_format	1	Output	When asserted, selects the format of the PTP 1-step operation on the respective channel. Tie to 1 to indicate the use of IEEE 1588v2 timestamp and correction field formats (96 bits) Valid only when either the egress time timestamp signal (i_ptp_ins_ets) or the residence time timestamp signal (i_ptp_ins_cf), and the TX valid signal, and SOP signal are asserted.
ptp_offset_timestamp	16	Output	When asserted, indicates the position of the PTP timestamp field in the current TX packet. Valid only when the TX valid and TX SOP signals are asserted.
ptp_offset_correction_field	16	Output	When asserted, indicates the position of the PTP correction field in the current TX packet. Valid only when the TX valid and TX SOP signals are asserted.
i_ptp_csum_offset	16	Output	When asserted, indicates the position of the first byte of a UDP checksum field in the current TX packet. Valid only when the checksum overwrite in a UDP packet (e.g. i_ptp_zero_csum), TX valid, TX SOP signals are asserted.
i_ptp_eb_offset	16	Output	When asserted, indicates the position of the first byte of extended bytes field in the current TX packet. Valid only when the extended bytes overwrite in an IPv6 packet (e.g. i_ptp_update_eb), TX valid, TX SOP signals are asserted.
ptp_timestamp_request_valid	1	Output	Request a 2-step timestamp signal for the current TX packet. When asserted, generates a TX timestamp for the current packet. Valid only when the TX valid and TX SOP signals are asserted.
ptp_timestamp_request_fingerprint	8	Output	Fingerprint signal for current TX packet. Assigns an 8-bit fingerprint to a TX packet that is being transmitted, so that the 2-step or 1-step PTP/eCPRI one way delay measurement timestamp associated with the TX packet can be identified. The timestamp returns with the same fingerprint. The encoding format is: • Bit [6:0]: PTP Fingerprint ID • Bit [7]: PTP/eCPRI Valid only when the TX valid and TX SOP signals are asserted.
o_tx_ptp_ready	1	Input	TX PTP ready signal. When asserted, the core to ready to request for TX PTP functions on the respective channel.
o_rx_ptp_ready	1	Input	RX PTP ready signal. When asserted, indicates the RX PTP logic ready for use on the respective channel.

Related Information

1588 PTP Interface
For more information on 1588 PTP signals for the E-tile Hard IP for Ethernet.

5.7.2. 25G Ethernet MAC 1588 PTP Signals

Table 34. Signals of the 25G Ethernet MAC 1588 PTP

All signals are synchronous to `clk_tx` clock.

Signal Name	Width	Direction	Description
`ptp_timestamp_insert`	1	Output	Indicates the current packet on the TX client interface is a 1588 PTP packet and directs the IP core to process the packet in one-step processing insertion mode. In this mode, the IP core overwrites the timestamp of the packet with the timestamp when the packet appears on the TX Ethernet link. The TX client must assert and deassert this signal synchronously with the TX SOP signal for the 1588 PTP packet.
`ptp_tx_etstamp_ins_ctrl_residence_time_update`	1	Output	Indicates the current packet on the TX client interface is a 1588 PTP packet and directs the IP core to process the packet in one-step processing correction mode. In this mode, the IP core adds the latency through the IP core (residence time) to the current contents of the timestamp field. The TX client must assert and deassert this signal synchronously with the TX SOP signal for the 1588 PTP packet.
`tx_etstamp_ins_ctrl_times tamp_format`	1	Output	Specifies the timestamp format (V1 or V2 format) for the current packet if the TX client simultaneously asserts `tx_etstamp_ins_ctrl_timestamp_insert`. Valid value is: • Tie to 0 to indicate 96-bit timestamp format (V2). The TX client must maintain the desired value on this signal while the TX SOP signal is asserted.
`tx_etstamp_ins_ctrl_residence_time_calc_format`	1	Output	Specifies the TOD format (Intel 64-bit TOD format or the V2 96-bit format) for the current packet if the TX client simultaneously asserts `tx_etstamp_ins_ctrl_residence_time_update`. Value is: • Tie to 0 to indicate 96-bit TOD format (V2) The TX client must maintain the desired value on this signal while the TX SOP signal is asserted.
`ptp_offset_timestamp`	16	Output	Specifies the byte offset of the timestamp information in the current packet if the TX client simultaneously asserts `tx_etstamp_ins_ctrl_timestamp_insert`. The IP core overwrites the value at this offset. The TX client must maintain the desired value on this signal while the TX SOP signal is asserted. The timestamp has 96 bits. In this case, the IP core inserts ten bytes (bits [95:16]) of the timestamp at this offset and the remaining two bytes (bits [15:0]) of the timestamp at the offset specified in `tx_etstamp_ins_ctrl_offset_correction_field`.
`ptp_offset_correction_field`	16	Output	If the TX client simultaneously asserts `tx_etstamp_ins_ctrl_residence_time_update`, this signal specifies the byte offset of the correction field in the current packet. If the TX client simultaneously asserts `tx_etstamp_ins_ctrl_timestamp_insert` and deasserts (sets to the value of 0) the `tx_etstamp_ins_ctrl_timestamp_format` signal, this signal specifies the byte offset of bits [15:0]] of the timestamp. The TX client must maintain the desired value on this signal while the TX SOP signal is asserted.

continued...

Signal Name	Width	Direction	Description
tx_etstamp_ins_ctrl_checksum_zero	1	Output	The TX client asserts this signal during a TX SOP cycle to tell the IP core to zero the UDP checksum in the current packet. A zeroed UDP checksum indicates the checksum value is not necessarily correct. This information is useful to tell the application to skip checksum checking of UDP IPv4 packets. This function is illegal for UDP IPv6 packets.
tx_etstamp_ins_ctrl_offset_checksum_field	16	Output	Indicates the byte offset of the UDP checksum in the current packet. The TX client must ensure this signal has a valid value during each TX SOP cycle when it also asserts the `tx_etstamp_ins_ctrl_checksum_zero` signal. Holds the byte offset of the two bytes in the packet that the IP core should reset.
tx_etstamp_ins_ctrl_checksum_correct	1	Output	The TX client asserts this signal during a TX SOP cycle to tell the IP core to update (correct) the UDP checksum in the current packet. This signal is asserted for correct processing of UDP IPv6 packets.
tx_etstamp_ins_ctrl_offset_checksum_correction	16	Output	Indicates the byte offset of the UDP checksum in the current packet. The TX client must ensure this signal has a valid value during each TX SOP cycle when it also asserts the `tx_etstamp_ins_ctrl_checksum_correct` signal. Holds the byte offset of the two bytes in the packet that the IP core should correct. This signal is meaningful only in one-step clock mode.
ptp_timestamp_request_valid	1	Output	Indicates the current packet on the TX client interface is a 1588 PTP packet and directs the IP core to process the packet in two-step processing mode. In this mode, the IP core outputs the timestamp of the packet when it exits the IP core, and does not modify the packet timestamp information. The TX client must assert and deassert this signal synchronously with the TX SOP signal for the 1588 PTP packet.
ptp_timestamp_request_fingerprint	8	Output	Fingerprint of the current packet. The TX client must assert and deassert this signal synchronously with the TX SOP signal for the 1588 PTP packet and eCPRI one way delay measurement packet.

Related Information

1588 PTP Interface Signals
For more information on 1588 PTP signals for the 25G Ethernet Intel Stratix 10 IP.

5.7.3. 10G Ethernet MAC 1588 PTP Signals

Table 35. Signals of the 10G Ethernet MAC 1588 PTP

All signals are synchronous to `clk_tx` clock.

Signal Name	Width	Direction	Description
ptp_timestamp_insert	1	Output	Indicates the current packet on the TX client interface is a 1588 PTP packet and directs the IP core to process the packet in one-step processing insertion mode. In this mode, the IP core overwrites the timestamp of the packet with the timestamp when the packet appears on the TX Ethernet link.

continued...

Signal Name	Width	Direction	Description
			The TX client must assert and deassert this signal synchronously with the TX SOP signal for the 1588 PTP packet.
`ptp_tx_etstamp_ins_ctrl_residence_time_update`	1	Output	Indicates the current packet on the TX client interface is a 1588 PTP packet and directs the IP core to process the packet in one-step processing correction mode. In this mode, the IP core adds the latency through the IP core (residence time) to the current contents of the timestamp field. The TX client must assert and deassert this signal synchronously with the TX SOP signal for the 1588 PTP packet.
`tx_etstamp_ins_ctrl_times tamp_format`	1	Output	Specifies the timestamp format (V1 or V2 format) for the current packet if the TX client simultaneously asserts `tx_etstamp_ins_ctrl_timestamp_insert`. Valid value is: • Tie to 0 to indicate 96-bit timestamp format (V2). The TX client must maintain the desired value on this signal while the TX SOP signal is asserted.
`tx_etstamp_ins_ctrl_residence_time_calc_format`	1	Output	Specifies the TOD format (Intel 64-bit TOD format or the V2 96-bit format) for the current packet if the TX client simultaneously asserts `tx_etstamp_ins_ctrl_residence_time_update`. Value is: • Tie to 0 to indicate 96-bit TOD format (V2) The TX client must maintain the desired value on this signal while the TX SOP signal is asserted.
`ptp_offset_timestamp`	16	Output	Specifies the byte offset of the timestamp information in the current packet if the TX client simultaneously asserts `tx_etstamp_ins_ctrl_timestamp_insert`. The IP core overwrites the value at this offset. The TX client must maintain the desired value on this signal while the TX SOP signal is asserted. The timestamp has 96 bits. In this case, the IP core inserts ten bytes (bits [95:16]) of the timestamp at this offset and the remaining two bytes (bits [15:0]) of the timestamp at the offset specified in `tx_etstamp_ins_ctrl_offset_correction_field`.
`ptp_offset_correction_field`	16	Output	If the TX client simultaneously asserts `tx_etstamp_ins_ctrl_residence_time_update`, this signal specifies the byte offset of the correction field in the current packet. If the TX client simultaneously asserts `tx_etstamp_ins_ctrl_timestamp_insert` and deasserts (sets to the value of 0) the `tx_etstamp_ins_ctrl_timestamp_format` signal, this signal specifies the byte offset of bits [15:0]] of the timestamp. The TX client must maintain the desired value on this signal while the TX SOP signal is asserted.
`tx_etstamp_ins_ctrl_checksum_zero`	1	Output	The TX client asserts this signal during a TX SOP cycle to tell the IP core to zero the UDP checksum in the current packet. A zeroed UDP checksum indicates the checksum value is not necessarily correct. This information is useful to tell the application to skip checksum checking of UDP IPv4 packets. This function is illegal for UDP IPv6 packets.
`tx_etstamp_ins_ctrl_offset_checksum_field`	16	Output	Indicates the byte offset of the UDP checksum in the current packet. The TX client must ensure this signal has a valid value during each TX SOP cycle when it also asserts the `tx_etstamp_ins_ctrl_checksum_zero` signal.

continued...

Signal Name	Width	Direction	Description
			Holds the byte offset of the two bytes in the packet that the IP core should reset.
tx_etstamp_ins_ctrl_checksum_correct	1	Output	The TX client asserts this signal during a TX SOP cycle to tell the IP core to update (correct) the UDP checksum in the current packet. This signal is asserted for correct processing of UDP IPv6 packets.
tx_etstamp_ins_ctrl_offset_checksum_correction	16	Output	Indicates the byte offset of the UDP checksum in the current packet. The TX client must ensure this signal has a valid value during each TX SOP cycle when it also asserts the tx_etstamp_ins_ctrl_checksum_correct signal. Holds the byte offset of the two bytes in the packet that the IP core should correct. This signal is meaningful only in one-step clock mode.
tx_path_delay_10g_data	Output	16 or 24	Connect this to the Intel FPGA PHY IP. This bus carries the path delay, which is measured between the physical network and the PHY side of the MAC IP Core (XGMII). The MAC IP core uses this value while generating the egress timestamp to account for the delay. The path delay is in the following format: • Bit [9:0]: Fractional number of clock cycle • Bits [23/15:10]: Number of clock cycle
tx_egress_p2p_update	Output	1	Assert this signal when the correction factor is added with <meanPathDelay> given by tx_egress_p2p_val for a transmit frame, as part of peer-to-peer mechanism. Assert this signal in the same clock cycle as the start of packet (avalon_st_tx_startofpacket).
tx_egress_p2p_val	Output	46	This represents <meanPathDelay> for peer to peer operations. • Bits [45:16]: Link delay in nanoseconds field • Bits [15:0]: Link delay in fractional nanoseconds field
ptp_timestamp_request_valid	1	Output	Indicates the current packet on the TX client interface is a 1588 PTP packet and directs the IP core to process the packet in two-step processing mode. In this mode, the IP core outputs the timestamp of the packet when it exits the IP core, and does not modify the packet timestamp information. The TX client must assert and deassert this signal synchronously with the TX SOP signal for the 1588 PTP packet.
ptp_timestamp_request_fingerprint	8	Output	Fingerprint of the current packet. The TX client must assert and deassert this signal synchronously with the TX SOP signal for the 1588 PTP packet and eCPRI one way delay measurement packet.

Related Information

IEEE 1588v2 Interfaces

For more information on 1588 PTP signals for the Low Latency Ethernet 10G MAC Intel FPGA IP.

5.8. Ethernet MAC Sink Interface

Table 36. Signals of the 25G Ethernet MAC Sink Interface

This section lists port from 25G Ethernet MAC to eCPRI IP . All signals are synchronous to mac_clk_rx.

Signal Name	Width (Bits)	I/O Direction	Description
mac_sink_valid	1	Input	Indicates Avalon source valid from Ethernet to MAC eCPRI.
mac_sink_data	DATA_WIDTH[5]	Input	Indicates Avalon source write data from Ethernet MAC to eCPRI.
mac_sink_sop	1	Input	Indicates Avalon sink start of packet (SOP) from Ethernet MAC to eCPRI. Indicate the beginning of packet.
mac_sink_eop	1	Input	Avalon source end of packet (EOP) from Ethernet MAC to eCPRI. Indicate the end of packet.
mac_sink_empty	LOG2(DATA_WIDTH[5]/8)	Input	Avalon source empty from Ethernet MAC to eCPRI. Indicates the number of symbols that are empty, that is, do not represent valid data.
mac_sink_ready	1	Output	Avalon sink ready driven from Ethernet MAC. Indicate eCPRI can accept data.
mac_sink_error	6	Input	Avalon sink error from Ethernet MAC to eCPRI. A bit mask to mark errors affecting the data being transferred in the current cycle.

5.9. External ST Source Interface

Table 37. Signals of the External ST Source Interface

All signals are synchronous to ext_sink_clk.

Signal Name	Width (Bits)	I/O Direction	Description
ext_source_valid	1	Output	Avalon source valid from L2/L3 parser to external user logic. This signal is synchronous to ext_sink_clk signal.
ext_source_data	DATA_WIDTH[6]	Output	Avalon source write data from L2/L3 parser to external user logic.
ext_source_sop	1	Output	Avalon source start of packet from L2/L3 parser to external user logic. Indicate the beginning of packet.
ext_source_eop	1	Output	Avalon source end of packet from L2/L3 parser to external user logic. Indicates the end of packet.

continued...

[5] This is set to 64. This parameter is hidden from user and you can't change it.

[6] This is set to 64. This parameter is hidden from user and you can't change it.

Signal Name	Width (Bits)	I/O Direction	Description
ext_source_empty	LOG2(DATA_WIDTH[6]/8)	Output	Avalon source empty from L2/L3 parser to external user logic. Indicates the number of symbols that are empty, that is, do not represent valid data.
ext_source_error	1	Output	Avalon source error from L2/L3 parser to external user logic. A bit mask to mark errors affecting the data being transferred in the current cycle.
ext_source_pkt_type	3	Output	Indicate frame type of the packet output from L2/L3 parser to external user logic. • 000b: eCPRI • 001b: PTP • 010b: Misc This signal is synchronous with ext_source_valid signal.

5.10. External ST Sink Interface

Table 38. Signals of the External ST Sink Interface

This table lists the ports from external user logic to L2/L3 parser. All signals are synchronous to ext_sink_clk.

Signal Name	Width (Bits)	I/O Direction	Description
ext_sink_valid	1	Input	Avalon sink valid from external user logic to L2/L3 parser.
ext_sink_data	DATA_WIDTH[7]	Input	Avalon sink write data from external user logic to L2/L3 parser.
ext_sink_sop	1	Input	Avalon sink start of packet from external user logic to L2/L3 parser. Indicates the beginning of packet.
ext_sink_eop	1	Input	Avalon sink end of packet from external user logic to L2/L3 parser. Indicates the end of packet.
ext_sink_empty	LOG2(DATA_WIDTH[7]/8)	Input	Avalon sink empty from external user logic to L2/L3 parser. Indicates the number of symbols that are empty, that is, do not represent valid data.
ext_sink_ready	1	Output	Avalon sink ready driven from L2/L3 parser. Indicate L2/L3 parser can accept data.
ext_sink_error	1	Input	Avalon sink error from external user logic to L2/L3 parser. A bit mask to mark errors affecting the data being transferred in the current cycle.
ext_ptp_timestamp_request_fingerprint	7	Input	Provides the fingerprint of the V2-format 1588 PTP frame currently beginning transmission on the Ethernet link. Value is valid when the ext_sink_sop signal is asserted. The encoding format is:

continued...

[7] This is set to 64. This parameter is hidden from user and you can't change it.

Signal Name	Width (Bits)	I/O Direction	Description
			• Bit [6:0]: PTP Fingerprint ID
ext_tx_egress_timestamp_96b_data	96	Output	Provides the V2-format timestamp when a 1588 PTP frame begins transmission on the Ethernet link. Value is valid when the tx_egress_timestamp_96b_valid signal is asserted.
ext_tx_egress_timestamp_96b_valid	1	Output	Indicates that the ext_tx_egress_timestamp_96b_data signals are valid in the current ext_sink_clk clock cycle. This signal is valid only in two-step clock mode.
ext_tx_egress_timestamp_96b_fingerprint	7	Output	Fingerprint signal for current TX packet. Assigns an PTP_TS_FP_WIDTH fingerprint to a TX packet that is being transmitted, so that the two-step or one-step PTP/ eCPRI one-way delay measurement timestamp associated with the TX packet can be identified. The timestamp returns with the same fingerprint. Valid only when the TX valid and TX SOP signals are asserted. The encoding format is: • Bit [6:0]: PTP Fingerprint ID
ext_tx_ingress_timestamp_96b_data	96	Input	Provides the V2-format timestamp when a 1588 PTP frame begins transmission on the Ethernet link. Value is valid when the ext_sink_sop signal is asserted.
ptp_tx_ingress_timestamp_96b_data	96	Output	Provides the V2-format timestamp when a 1588 PTP frame begins transmission on the Ethernet link. Synchronous with mac_source_valid signal.

5.11. eCPRI IP Source Interface

Table 39. Signals of the eCPRI IP Source Interface

This table lists the ports from eCPRI IP to client logic. All signals are synchronous to clk_rx.

Signal Name	Width (Bits)	I/O Direction	Description
avst_source_valid	1	Output	Avalon source valid from eCPRI to client logic.
avst_source_data	DATA_WIDTH[8]	Output	Avalon source write data from eCPRI to RRH PHY.
avst_source_sop	1	Output	Avalon source start of packet (SOP) from eCPRI to RRH PHY. Indicate the beginning of packet.
avst_source_eop	1	Output	Avalon source end of packet (EOP) from eCPRI to RRH PHY. Indicates the end of packet.

continued...

[8] This is set to 64. This parameter is hidden from user and you can't change it.

Signal Name	Width (Bits)	I/O Direction	Description
avst_source_empty	LOG2(DATA_WIDTH[8]/8)	Output	Avalon source empty from eCPRI to RRH PHY. Indicates the number of symbols that are empty, that is, do not represent valid data.
avst_source_error	1	Output	Avalon source error from eCPRI to RRH PHY. A bit mask to mark errors affecting the data being transferred in the current cycle.
Sideband			
source_pc_id	32	Output	Indicates physical channel ID of eCPRI message. For message type 3, the physical channel id is 32-bit wide. For other message types, it is 16-bit wide and the 16 bit MSB is ignored. Valid on SOP assertion and stable until EOP assertion.
source_seq_id	32	Output	Indicates sequence ID of eCPRI message. For message type 3, the sequence id is 32- bit wide. For other message types, it is 16-bit wide and the 16 bit MSB is ignored. Valid on SOP assertion and stable until EOP assertion.
source_rtc_id	16	Output	Real time control ID of eCPRI message. Valid on SOP assertion and stable until EOP assertion.
source_msg_type	8	Output	Indicates message type of the eCPRI message. Valid range is 0-7 and 64-255 for eCPRI v1.2 specification. Valid on SOP assertion and stable until EOP assertion.
source_mem_acc_id	8	Output	Indicates remote memory access id of the eCPRI message type 4. Valid on SOP assertion and stable until EOP assertion.
source_op_type	8	Output	Indicates the operation of the eCPRI message type 4: • 8’b0000 0000 – Read Request • 8’b0000 0001 – Read Response • 8’b0001 0000 – Write Request • 8’b0001 0001 – Write Response • 8’b0010 xxxx – Write No Response • 8’bxxxx 0010 – Failure No Response Valid on SOP assertion and stable until EOP assertion.
source_element_id	16	Output	Indicates element ID of the eCPRI message type 4. Valid on SOP assertion and stable until EOP assertion.
source_address	48	Output	Indicates memory address of the eCPRI message type 4. Valid on SOP assertion and stable until EOP assertion.
source_length	16	Output	Indicates memory access length of the eCPRI message type 4.

continued...

Signal Name	Width (Bits)	I/O Direction	Description
			Valid on SOP assertion and stable until EOP assertion.
`source_reset_id`	16	Output	Indicates reset ID of the eCPRI message type 6. Valid on SOP assertion and stable until EOP assertion.
`source_reset_op`	8	Output	Indicate reset operation type of the eCPRI message type 6: • 8’b0000 0000 – Remote Reset Request • 8’b0000 0001 – Remote Reset Response Others – Reserved Valid on SOP assertion and stable until EOP assertion.
`source_event_id`	8	Output	Indicates event ID of the eCPRI message type 7. Valid on SOP assertion and stable until EOP assertion.
`source_event_type`	8	Output	Indicates event type of the eCPRI message type 7. Valid on SOP assertion and stable until EOP assertion.
`source_notif`	8	Output	Indicates number of faults for within eCPRI message type 7. Valid on SOP assertion and stable until EOP assertion.

5.12. eCPRI IP Sink Interface

Table 40. Signals of the eCPRI IP Sink Interface

This table lists the ports from client logic to eCPRI IP. All signals are synchronous to `clk_tx`.

Signal Name	Width	Direction	Description
`avst_sink_valid`	1	Input	Avalon sink valid from client logic to eCPRI. When you set the Ethernet frame size to 9000 bytes, you must assert `avst_sink_valid` continuously between the assertions of `avst_sink_sop` and `avst_sink_eop`. The only exception is when `avst_sink_ready` signal deasserts, and you are required to deassert `avst_sink_valid` for three cycles of READY_LATENCY.
`avst_sink_data`	DATA_WIDTH[9]	Input	Avalon sink write data from RRH PHY to eCPRI
`avst_sink_sop`	1	Input	Avalon sink start of packet (SOP) from RRH PHY to eCPRI. Indicate the beginning of packet.
`avst_sink_eop`	1	Input	Avalon sink end of packet (EOP) from RRH PHY to eCPRI. Indicate the end of packet.
`avst_sink_empty`	LOG2(DATA_WIDTH[9]/8)	Input	Avalon sink empty from RRH PHY to eCPRI. Indicates the number of symbols that are empty, that is, do not represent valid data.

continued...

[9] This is set to 64. This parameter is hidden from user and you can't change it.

Signal Name	Width	Direction	Description
avst_sink_ready	1	Output	Avalon sink ready driven from eCPRI. Indicates eCPRI can accept data.
avst_sink_error	1	Input	Avalon sink error from RRH PHY to eCPRI. A bit mask to mark errors affecting the data being transferred in the current cycle.
Sideband			
sink_pkt_size	16	Input	Packet size in bytes for the data packet from client logic to eCPRI IP. The packet size must not include data on sideband signal interface. This port is available only when the Streaming mode is enabled in the eCPRI IP. Ensure that you send the data packet with the correct size. If the packet size (sink_pkt_size) does not match the packet size of the whole SOP and EOP, the Avalon streaming interface error to MAC is asserted. This sets eCPRI TX error register as well.
sink_pkt_checksum	16	Input	Checksum for data packet from client logic PHY to eCPRI IP. This signal is used for the UDP checksum. Intel recommends that you provide the correct signal because the eCPRI IP does not check for this signal.
sink_pc_id	32	Input	Physical channel ID of the eCPRI message. For message type 3, the physical channel is 32-bit wide. For other message types, it is 16-bit wide and the 16 bit MSB is ignored. Valid on SOP assertion and stable until EOP assertion.
sink_seq_id	32	Input	Sequence ID of eCPRI message. For message type 3, the physical channel is 32-bit wide. For other message types, it is 16-bit wide and the 16 bit MSB is ignored. Valid on SOP assertion and stable until EOP assertion.
sink_rtc_id	16	Input	Real time control ID of eCPRI message. Valid on SOP assertion and stable until EOP assertion.
sink_concatenation	1	Input	Concatenation indication on the eCPRI message: • 0 - Indicates that the eCPRI message is the last one inside the eCPRI PDU. • 1 - Indicates that another eCPRI message follows this one within the eCPRI PDU. Valid on SOP assertion and stable until EOP assertion.
sink_msg_type	8	Input	Indicate message type of the eCPRI message. Valid range is 0-7 and 64-255 for eCPRI v1.2 specification. Valid on SOP assertion and stable until EOP assertion.
sink_mem_acc_id	8	Input	Indicate remote memory access id of the eCPRI message type 4: Valid on SOP assertion and stable until EOP assertion.
sink_op_type	8	Input	Indicate operation of the eCPRI message type 4: • 8'b0000 0000 – Read Request • 8'b0000 0001 – Read Response • 8'b0001 0000 – Write Request • 8'b0001 0001 – Write Response • 8'b0010 xxxx – Write No Response • 8'bxxxx 0010 – Failure No Response Valid on SOP assertion and stable until EOP assertion.

continued...

Signal Name	Width	Direction	Description
sink_element_id	16	Input	Indicates element id of the eCPRI message type 4. Valid on SOP assertion and stable until EOP assertion.
sink_address	48	Input	Indicates memory address of the eCPRI message type 4. Valid on SOP assertion and stable until EOP assertion.
sink_length	16	Input	Indicates memory access length of the eCPRI message type 4. Valid on SOP assertion and stable until EOP assertion.
sink_reset_id	16	Input	Indicates reset ID of the eCPRI message type 6. Valid on SOP assertion and stable until EOP assertion.
sink_reset_op	8	Input	Indicates reset operation type of the eCPRI message type 6: • 8’b0000 0000 – Remote Reset Request • 8’b0000 0001 – Remote Reset Response • Others – Reserved Valid on SOP assertion and stable until EOP assertion.
sink_event_id	8	Input	Indicate event ID of the eCPRI message type 7. Valid on SOP assertion and stable until EOP assertion.
sink_event_type	8	Input	Indicates event type of the eCPRI message type 7. Valid on SOP assertion and stable until EOP assertion.

5.13. Miscellaneous Interface Signals

Table 41. Signals of Miscellaneous Interface

These signals are only available when you turn on the **Pair with ORAN** option in eCPRI IP Parameter editor.

Signal Name	Width (Bits)	I/O Direction	Description
tx_transport_c_u	1	Input	Indicates if the packets received to transport layer are C-plane or U-plane packets. • 0= User IQ data • 1= Control message
rx_transport_c_u	1	Output	Indicates if packets transmitted to transport layer are C-plane or U-plane packets. • 0= User IQ data • 1= Control message
tx_queue_<*N*>_fifo_full	1	Output	Indicates that the TX arbitration queue FIFO *N* is full where *N* can be 0 to 7.

5.14. IWF Type 0 eCPRI Interface

The IWF type 0 eCPRI interfaces are available only when you turn on **Interworking Function (IWF) support** parameter in eCPRI IP parameter editor.

5.14.1. IWF Source Interface

Table 42. Signals of the IWF Source Interface

All signals are synchronous to `clk_tx`.

Signal Name	Width (Bits)	I/O Direction	Description
`iwf_avst_source_valid`	1	Output	Avalon source valid from eCPRI IP to client logic.
`iwf_avst_source_data`	DATA_WIDTH[10]	Output	Avalon source write data from eCPRI IP to client logic.
`iwf_avst_source_sop`	1	Output	Avalon source start of packet from eCPRI IP to client logic. Indicates the beginning of packet.
`iwf_avst_source_eop`	1	Output	Avalon source end of packet from eCPRI IP to client logic. Indicates the end of packet.
`iwf_avst_source_empty`	LOG2(DATA_WIDTH[10]/8)	Output	Avalon source empty from eCPRI IP to client logic. Indicates the number of symbols that are empty, that is, do not represent valid data.
`iwf_avst_source_error`	1	Output	Avalon source error from eCPRI IP to client logic. A bit mask to mark errors affecting the data being transferred in the current cycle.
`iwf_gmii_rxdv[N]`	1	Input	Ethernet receive data valid. Indicates the presence of valid data or initial start-of-packet control character on `iwf_gmii_rxd`. This signal is from IWF to user logic.
`iwf_gmii_rxer[N]`	1	Input	Ethernet receive error. Indicates an error on `iwf_gmii_rxd`. When this signal is asserted, the value on `iwf_gmii_rxd` is 0x0E. This signal is from user logic to IWF.
`iwf_gmii_rxd[N]`	8	Input	Ethernet receive data. Data bus for data from the IWF to the external Ethernet block. This signal is from user logic to IWF.
`iwf_gmii_rxfifo_status[N]`	4	Input	Ethernet RX PCS FIFO fill level status. The individual bits have the following meanings: • Bit [3]: Empty • Bit [2]: Almost empty • Bit [1]: Full • Bit [0]: Almost full This signal is from user logic to IWF.
`iwf_gmii_txen[N]`	1	Output	Valid signal for GMII interface that indicate data is valid. This signal required to be asserted two clock cycles earlier for the character S to be inserted into the data stream as the start of packet before takes in the real GMII data. The deassertion of this signal trigger the assertion of /T/R as the representation of end of packet. This signal is from user logic to IWF.
`iwf_gmii_txer[N]`	1	Output	Ethernet transmit coding error. This signal is from user logic to IWF.
`iwf_gmii_txd[N]`	8	Output	Ethernet transmit data. This signal is from user logic to IWF.

continued...

[10] This is set to 64. This parameter is hidden from user and you can't change it.

<table>
<tr><th>Signal Name</th><th>Width (Bits)</th><th>I/O Direction</th><th>Description</th></tr>
<tr><td><code>iwf_gmii_txfifo_status[N]</code></td><td>4</td><td>Input</td><td>Ethernet TX PCS FIFO fill level status. The individual bits have the following meanings:
• Bit [3]: Empty
• Bit [2]: Almost empty
• Bit [1]: Full
• Bit [0]: Almost full
This signal is from user logic to IWF.</td></tr>
<tr><td colspan="4"><b>Sideband</b></td></tr>
<tr><td><code>iwf_source_pkt_size</code></td><td>16</td><td>Output</td><td>Indicates IWF to eCPRI IP packet size in bytes.
<table>
<tr><th rowspan="2">CPRI Line Bit Rate (Gbps)</th><th colspan="2">Packet Size (Byte)</th></tr>
<tr><th>Ctel_AxC and RTVS</th><th>IQ Data</th></tr>
<tr><td>0.6144</td><td>1</td><td>15</td></tr>
<tr><td>1.2288</td><td>2</td><td>30</td></tr>
<tr><td>2.4756</td><td>4</td><td>60</td></tr>
<tr><td>3.0720</td><td>5</td><td>75</td></tr>
<tr><td>4.9152</td><td>8</td><td>120</td></tr>
<tr><td>6.1440</td><td>10</td><td>120</td></tr>
<tr><td>8.11008</td><td>16</td><td>240</td></tr>
<tr><td>9.8304</td><td>16</td><td>240</td></tr>
<tr><td>10.1376</td><td>16
<i>Note:</i> For RTVS, the value is 4.</td><td>300</td></tr>
<tr><td>12.16512</td><td>16
<i>Note:</i> For RTVS, the value is 8.</td><td>360</td></tr>
<tr><td>24.33024</td><td>16
<i>Note:</i> For RTVS, the value is 32.</td><td>720</td></tr>
</table>
</td></tr>
<tr><td><code>iwf_source_pkt_checksum</code></td><td>16</td><td>Output</td><td>Checksum for data packet from IWF to eCPRI IP.
This signal is used for UPD checksum.</td></tr>
<tr><td><code>iwf_source_concatenation</code></td><td>1</td><td>Output</td><td>Concatenation indication on the eCPRI message:
• 0: Indicates that the eCPRI message is the last one inside the eCPRI PDU.
• 1: Indicates that another eCPRI message follows this one within the eCPRI PDU.
Valid on SOP assertion and stable until EOP assertion.</td></tr>
</table>

continued...

Signal Name	Width (Bits)	I/O Direction	Description
			Note: IWF do not support concatenation.
iwf_source_pc_id	16	Output	Physical channel ID of the eCPRI message. Lower byte maps to CPRI channel. Only channel 0 is supported in the current release. Valid on SOP assertion and stable until EOP assertion.
iwf_source_seq_id	16	Output	Sequence ID of the eCPRI message. Valid on SOP assertion and stable until EOP assertion.
iwf_source_rtc_id	16	Output	Real Time Control ID of the eCPRI message. Valid on SOP assertion and stable until EOP assertion. Valid values are: • ID [7:0]: CPRI channel • ID [11:8]: 'h0- Ctrl_AxC, 'h1- Vendor Specific, 'h2- Real-time Vendor Specific
iwf_source_msg_type	8	Output	Indicate message type of the eCPRI message. IWF type 0 supports only message type 0, 2, 6 and 7. Valid on SOP assertion and stable until EOP assertion.
iwf_source_reset_id	16	Output	Indicate reset ID of the eCPRI message type 6. Lower byte maps to CPRI channel. Only channel 0 is supported in the current release. Valid on SOP assertion and stable until EOP assertion.
iwf_source_reset_op	8	Output	Indicate reset operation type of the eCPRI message type 6: • 8'b00000000- Remote Reset Request • 8'b00000001- Remote Reset Response • Others- Reserved Valid on SOP assertion and stable until EOP assertion.
iwf_source_event_id	8	Output	Indicates event ID of the eCPRI message type 7. Lower byte maps to CPRI channel. Only channel 0 is supported in the current release. Valid on SOP assertion and stable until EOP assertion.
iwf_source_event_type	8	Output	Indicates event type of the eCPRI message type 7. Valid on SOP assertion and stable until EOP assertion.
iwf_source_notif	8	Output	Indicates number of faults withing eCPRI message type 7. Valid on SOP assertion and stable until EOP assertion.

5.14.2. IWF Sink Interface

Table 43. Signals of the IWF Sink Interface

All signals are synchronous to clk_rx.

Signal Name	Width (Bits)	I/O Direction	Description
iwf_avst_sink_valid	1	Input	Avalon sink valid from eCPRI IP to client logic.
iwf_avst_sink_data	DATA_WIDTH[11]	Input	Avalon sink write data from eCPRI IP to client logic.
iwf_avst_sink_sop	1	Input	Avalon sink start of packet from eCPRI IP to client logic. Indicate the beginning of packet.
iwf_avst_sink_eop	1	Input	Avalon sink end of packet from eCPRI IP to client logic. Indicates the end of packet.
iwf_avst_sink_empty	LOG2(DATA_WIDTH[11]/8)	Input	Avalon sink empty from eCPRI IP to client logic. . Indicates the number of symbols that are empty, that is, do not represent valid data.
iwf_avst_sink_ready	1	Output	Avalon sink ready driven from eCPRI. Indicates eCPRI can accept data.
iwf_avst_sink_error	1	Input	Avalon sink error from eCPRI IP to client logic. A bit mask to mark errors affecting the data being transferred in the current cycle.
Sideband			
iwf_sink_pc_id	16	Input	Physical channel ID of the eCPRI message. Lower byte maps to CPRI channel. Valid on SOP assertion and stable until EOP assertion.
iwf_sink_seq_id	16	Input	Sequence ID of the eCPRI message. Valid on SOP assertion and stable until EOP assertion.
iwf_sink_rtc_id	16	Input	Real Time Control ID of the eCPRI message. Valid values are: • ID [7:0]: CPRI channel • ID [11:8]: 'h0- Ctrl_AxC, 'h1- Vendor Specific, 'h2- Real-time Vendor Specific Valid on SOP assertion and stable until EOP assertion.
iwf_sink_msg_type	8	Input	Indicate message type of the eCPRI message. IWF type 0 supports only message type 0 to 7 and 64-255 for eCPRI v1.2 specifications. Valid on SOP assertion and stable until EOP assertion.

continued...

[11] This is set to 64. This parameter is hidden from user and you can't change it.

Signal Name	Width (Bits)	I/O Direction	Description
iwf_sink_reset_id	16	Input	Indicate reset ID of the eCPRI message type 6. Lower byte maps to CPRI channel. Valid on SOP assertion and stable until EOP assertion.
iwf_sink_reset_op	8	Input	Indicate reset operation type of the eCPRI message type 6: • 8’b00000000- Remote Reset Request • 8’b00000001- Remote Reset Response • Others- Reserved Valid on SOP assertion and stable until EOP assertion.
iwf_sink_event_type	8	Input	Indicates event type of the eCPRI message type 7. Valid on SOP assertion and stable until EOP assertion.
iwf_sink_event_id	8	Input	Indicates event ID of the eCPRI message type 7. Lower byte maps to CPRI channel. Valid on SOP assertion and stable until EOP assertion.

5.15. IWF Type 0 CPRI MAC Interface

The IWF type 0 CPRI MAC interfaces listed in the following sections are available only when you turn on **Interworking Function (IWF) support** parameter in eCPRI IP parameter editor.

All signals are synchronous to cpri_clkout[N].

5.15.1. CPRI 32-bit IQ Data TX Interface

Table 44. Signals of CPRI 32-bit IQ Data Interface

Signal Name	Width (Bits)	I/O Direction	Description
TX Interface			
iq32_tx_ready[N]	4	Input	Each asserted bit indicates the IP core is ready to write IQ data into iq_tx_datain the next clock cycle. Each bit represents readiness of each byte.
iq32_tx_valid[N]	4	Output	Write valid for iq_tx_data.
iq32_tx_data[N]	64	Output	Respective IQ data word or bytes to be written based on iq_tx_ready signal.
RX Interface			
iq32_rx_valid[N]	4	Input	Assertion of the bit indicates the corresponding byte on the current iq_rx_data bus is valid IQ data.
iq32_rx_data[N]	64	Input	IQ data received from the CPRI frame. The iq_rx_valid signal indicates valid I/Q data bytes.

5.15.2. CPRI 64-bit IQ Data TX Interface

Table 45. Signals of CPRI 64-bit IQ Data Interface

Signal Name	Width (Bits)	I/O Direction	Description
TX Interface			
iq64_tx_ready[N]	8	Input	Each asserted bit indicates the IP core is ready to write IQ data into iq_tx_datain the next clock cycle. Each bit represents readiness of each byte.
iq64_tx_valid[N]	8	Output	Write valid for iq_tx_data.
iq64_tx_data[N]	64	Output	Respective IQ data word or bytes to be written based on iq64_tx_ready signal.
RX Interface			
iq64_rx_valid[N]	8	Input	Assertion of the bit indicates the corresponding byte on the current iq_rx_data bus is valid IQ data.
iq64_rx_data[N]	64	Input	IQ data received from the CPRI frame. The iq_rx_valid signal indicates valid I/Q data bytes.

5.15.3. CPRI 32-bit Ctrl_AxC TX Interface

Table 46. Signals of CPRI 32-bit Ctrl_AxC Interface

Signal Name	Width (Bits)	I/O Direction	Description
TX Interface			
ctrl32_axc_tx_ready[N]	4	Input	Assertion of the bits indicate the CPRI mapper is ready to read Ctrl_AxC data from the corresponding byte of ctrl_axc_tx_data on the next clock cycle.
ctrl32_axc_tx_valid[N]	4	Output	Write valid for ctrl_axc_tx_data. Assert bit [n] to indicate that the corresponding byte on the current ctrl_axc_tx_data bus is valid Ctrl_AxC data.
ctrl32_axc_tx_data[N]	64	Output	Ctrl_AxC data to be written to the CPRI frame. The CPRI mapper writes the individual bytes of the current value on the ctrl_axc_tx_data bus to the CPRI frame based on the ctrl_axc_tx_ready signal from the previous cycle, and the ctrl_axc_tx_valid signal in the current cycle.
RX Interface			
ctrl32_axc_rx_valid[N]	4	Input	Each asserted bit indicates the corresponding byte on the current ctrl_axc_rx_data bus is valid Ctrl_AxC data.
ctrl32_axc_rx_data[N]	64	Input	Ctrl_AxC data received from the CPRI frame. The ctrl_axc_rx_valid signal indicates valid Ctrl_AxC data bytes.

5.15.4. CPRI 64-bit Ctrl_AxC TX Interface

Table 47. Signals of CPRI 64-bit Ctrl_AxC Interface

Signal Name	Width (Bits)	I/O Direction	Description
TX Interface			
ctr64_axc_tx_ready[N]	8	Input	Each asserted bit indicates the CPRI mapper is ready to read Ctrl_Axc data from the corresponding byte of ctrl_axc_tx_data on the next clock cycle.
ctrl64_axc_tx_valid[N]	8	Output	Write valid for ctrl_axc_tx_data. Assert bit [n] to indicate that the corresponding byte on the current ctrl_axc_tx_data bus is valid Ctrl_AxC data.
ctrl64_axc_tx_data[N]	64	Output	Ctrl_AxC data to be written to the CPRI frame. The CPRI mapper writes the individual bytes of the current value on the ctrl_axc_tx_data bus to the CPRI frame based on the ctrl_axc_tx_ready signal from the previous cycle, and the ctrl_axc_tx_valid signal in the current cycle.
RX Interface			
ctrl64_axc_rx_valid[N]	4	Input	Assertion of the bit indicates the corresponding byte on the current ctrl64_axc_rx_data bus is valid Ctrl_AxC data.
ctrl64_axc_rx_data[N]	64	Input	IQ data received from the CPRI frame. The ctrl64_axc_rx_valid signal indicates valid Ctrl AxC data bytes.

5.15.5. CPRI 32-bit Vendor Specific TX Interface

Table 48. Signals of CPRI 32-bit Ctrl_AxC Interface

Signal Name	Width (Bits)	I/O Direction	Description
TX Interface			
vs32_tx_ready[N]	4	Input	Indicates that CPRI mapper is ready to read a real-time vendor specific byte from vs_tx_data on the next clock cycle.
vs32_tx_valid[N]	4	Output	Write valid for vs_tx_data. Assert this signal to indicate vs_tx_data holds a valid value in the current clock cycle
vs32_tx_data[N]	64	Output	Real-time vendor specific word to be written to the CPRI frame. The CPRI mapper writes the current value of the vs_tx_data bus to the CPRI frame based on the vs_tx_ready signal from the previous cycle, and the vs_tx_valid signal in the current cycle.
RX Interface			
vs_rx_valid[N]	4	Input	Each asserted bit indicates the corresponding byte on the current vs_rx_data bus is a valid vendor specific byte.
vs_rx_data[N]	64	Input	Indicates vendor specific word received from the CPRI frame. The vs_rx_valid signal indicates which bytes are valid vendor specific bytes.

5.15.6. CPRI 64-bit Vendor Specific TX Interface

Table 49. Signals of CPRI 32-bit Ctrl_AxC Interface

Signal Name	Width (Bits)	I/O Direction	Description
TX Interface			
`vs64_tx_ready[N]`	8	Input	Indicates that CPRI mapper is ready to read a real-time vendor-specific byte from `vs_tx_data` on the next clock cycle.
`vs64_tx_valid[N]`	8	Output	Write valid for `vs_tx_data`. Assert this signal to indicate `vs_tx_data` holds a valid value in the current clock cycle
`vs64_tx_data[N]`	64	Output	Real-time vendor-specific word to be written to the CPRI frame. The CPRI mapper writes the current value of the `vs_tx_data` bus to the CPRI frame based on the `vs_tx_ready` signal from the previous cycle, and the `vs_tx_valid` signal in the current cycle.
RX Interface			
`vs64_rx_valid[N]`	8	Input	Each asserted bit indicates the corresponding byte on the current `vs_rx_data` bus is a valid vendor-specific byte.
`vs64_rx_data[N]`	64	Input	Indicates Vendor-specific word received from the CPRI frame. The `vs_rx_valid` signal indicates which bytes are valid vendor specific bytes.

5.15.7. CPRI 32-bit Real-time Vendor Specific TX Interface

Table 50. Signals of CPRI 32-bit Ctrl_AxC Interface

Signal Name	Width (Bits)	I/O Direction	Description
TX Interface			
`rtvs32_tx_ready[N]`	1	Input	Indicates that CPRI mapper is ready to read a real-time vendor-specific byte from `rtvs_tx_data` on the next clock cycle.
`rtvs32_tx_valid[N]`	1	Output	Write valid for `rtvs_tx_data`. Assert this signal to indicate `rtvs_tx_data` holds a valid value in the current clock cycle
`rtvs32_tx_data[N]`	64	Output	Real-time vendor-specific word to be written to the CPRI frame. The CPRI mapper writes the current value of the `rtvs_tx_data` bus to the CPRI frame based on the `rtvs_tx_ready` signal from the previous cycle, and the `rtvs_tx_valid` signal in the current cycle.
RX Interface			
`rtvs32_rx_valid[N]`	1	Input	Each asserted bit indicates the corresponding byte on the current `rtvs_rx_data` bus is a valid real-time vendor-specific byte.
`rtvs32_rx_data[N]`	64	Input	Indicates real-time vendor-specific word received from the CPRI frame. The `rtvs_rx_valid` signal indicates which bytes are valid vendor specific bytes.

5.15.8. CPRI 64-bit Real-time Vendor Specific TX Interface

Table 51. Signals of CPRI 32-bit Ctrl_AxC Interface

Signal Name	Width (Bits)	I/O Direction	Description
TX Interface			
rtvs64_tx_ready[N]	1	Input	Indicates that CPRI mapper is ready to read a real-time vendor-specific byte from rtvs_tx_data on the next clock cycle.
rtvs64_tx_valid[N]	1	Output	Write valid for rtvs_tx_data. Assert this signal to indicate rtvs_tx_data holds a valid value in the current clock cycle
rtvs64_tx_data[N]	64	Output	Real-time vendor-specific word to be written to the CPRI frame. The CPRI mapper writes the current value of the rtvs_tx_data bus to the CPRI frame based on the rtvs_tx_ready signal from the previous cycle, and the rtvs_tx_valid signal in the current cycle.
RX Interface			
rtvs64_rx_valid[N]	1	Input	Each asserted bit indicates the corresponding byte on the current rtvs_rx_data bus is a valid real-time vendor-specific byte.
rtvs64_rx_data[N]	64	Input	Indicates real-time vendor-specific word received from the CPRI frame. The rtvs_rx_valid signal indicates which bytes are valid vendor specific bytes.

5.15.9. CPRI Gigabit Media Independent Interface (GMII)

Table 52. Signals of CPRI GMII Interface

Signal Name	Width (Bits)	I/O Direction	Description
TX Interface			
gmii_txen[N]	1	Output	Valid signal for GMII interface that indicate data is valid. This signal required to be asserted two clock cycles earlier for the character S to be inserted into the data stream as the start of packet before takes in the real GMII data. The deassertion of this signal trigger the assertion of /T/R as the representation of end of packet. This signal is going to CPRI MAC interface.
gmii_txer[N]	1	Output	Ethernet transmit coding error. When this signal is asserted, char /V/ will be inserted and pass into the CPRI link. This signal is going to CPRI MAC interface.
gmii_txd[N]	8	Output	Ethernet transmit data. The data transmitted from the external Ethernet block to the CPRI IP core, for transmission on the CPRI link. This input bus is synchronous to the rising edge of gmii_txclk clock. This signal is going to CPRI MAC interface.
gmii_txfifo_status[N]	4	Input	Ethernet TX PCS FIFO fill level status. The individual bits have the following meanings:

continued...

Signal Name	Width (Bits)	I/O Direction	Description
			- Bit [3[: FIFO empty - Bit [2]: FIFO almost empty - Bit [1]: FIFO full - Bit [0]: FIFO almost full This signal is going to CPRI MAC interface.
RX Interface			
`gmii_rxdv[N]`	1	Input	Ethernet receive data valid. Indicates the presence of valid data or initial start-of-packet control character on `gmii_rxd`. This signal is going to CPRI MAC interface.
`gmii_rxer[N]`	1	Input	Ethernet receive error. Indicates an error on `gmii_rxd`. When this signal is asserted, the value on `gmii_rxd` is 0x0E. This signal is going to CPRI MAC interface.
`gmii_rxd[N]`	8	Input	Ethernet receive data. Data bus for data from the CPRI IP to the external Ethernet block. All bits are deasserted during reset, and all bits are asserted after reset until the CPRI IP achieves frame synchronization. This signal is going to CPRI MAC interface.
`gmii_rxfifo_status[N]`	4	Input	Ethernet RX PCS FIFO fill level status. The individual bits have the following meanings: - Bit [3[: FIFO empty - Bit [2]: FIFO almost empty - Bit [1]: FIFO full - Bit [0]: FIFO almost full This signal is going to CPRI MAC interface.

5.15.10. CPRI IP L1 Control and Status Interface

Table 53. Signals of CPRI IP L1 Control and Status Interface

Signal Name	Width (Bits)	I/O Direction	Description
TX Interface			
`cpri_nego_bitrate_in[N]`	6	Input	CPRI line bit rate to be used in next attempt to achieve frame synchronization, encoded according to the following valid values: - 6'b000001: 0.6144 Gbps - 6'b000010: 1.2288 Gbps - 6'b000100: 2.4576 Gbps - 6'b000101: 3.0720 Gbps - 6'b001000: 4.9150 Gbps - 6'b001010: 6.1440 Gbps - 6'b010110: 8.11008 Gbps - 6'b010000: 9.8304 Gbps - 6'b010100: 10.1376 Gbps - 6b'011000: 12.16512 Gbps - 6'b110000 : 24.33024 Gbps *Note:* IWF uses this information to determine the active interface (either 32-bit or 64-bit).
`cpri_state_startup_seq[N]`	6	Input	Indicates the state of the CPRI start-up sequence state machine. This signal has the following valid values:

continued...

Signal Name	Width (Bits)	I/O Direction	Description
			• 3'b000: State A: Standby • 3'b001: State B: L1 Synchronization • 3'b011: State C: Protocol Setup • 3'b010: State D: Control and Management Setup • 3'b110: State E: Interface and VSS Negotiation • 3'b111: State F: Operation • 3'b101: State G: Passive Link *Note:* Drive clk_csr with the same clock source as CPRI's reconfig_clk.
cpri_state_l1_synch[N]	3	Input	State B condition indicator. Indicates the state of the CPRI receiver L1 synchronization state machine. This signal has the following valid values: • 3'b000: XACQ1 • 3'b001: XACQ2 • 3'b011: XSYNC1 • 3'b010: XSYNC2 • 3'b110: HFNSYNC
cpri_local_lof[N]	1	Input	The CPRI IP notifies the loss of frame detection to IWF block. In this case, the state_l1_synch signal indicates the L1 synchronization state machine is in state XACQ1 or XACQ2.
cpri_local_los[N]	1	Input	The CPRI IP notifies the loss of frame detection to IWF block. The CPRI IP asserts this flag if it detects excessive 8B/10B or 64B/66B errors.
cpri_sdi_assert[N]	1	Output	Indicates that the master service access point (SAP) is not available. Possible causes for this situation are equipment error or that the connected slave IP core is forwarding an SDI request it detected to the current RE CPRI master IP core through a direct connection.
cpri_local_rai[N]	1	Input	Indicates that either the cpri_local_lof or the cpri_local_los signal is high; clears when both of those two signals are low. Logical OR of two output signals cpri_local_lof and cpri_local_los.
cpri_reset_assert[N]	1	Output	Reset request from the application or from an RE slave to the current RE CPRI master IP core through a direct connection.
cpri_remote_lof[N]	1	Input	Indicates LOF received in Z.130.0 control byte from remote CPRI link partner. In this case the IP core also asserts the remote_lof bit in the FLSAR register at offset 0x2C.
cpri_remote_los[N]	1	Input	Indicates LOS received in Z.130.0 control byte from remote CPRI link partner. In this case the IP core also asserts the remote_los bit in the FLSAR register at offset 0x2C.
cpri_sdi_req[N]	1	Input	Indicates remote SAP defect indication received in Z.130.0 control byte from remote CPRI link master. If the current CPRI IP core is an RE slave in a multi-hop configuration, you should connect this output signal directly to the cpri_sdi_assert input signal of the downstream RE master.
cpri_remote_rai[N]	1	Input	Asserts when either cpri_remote_lof or cpri_remote_los is asserted, and clears when both cpri_remote_lof and z130_remote_los have the value of 0.

continued...

Signal Name	Width (Bits)	I/O Direction	Description
			In this case the IP core also asserts the `rai_detected` bit in the FLSAR register at offset 0x2C.
`cpri_reset_req[N]`	1	Input	If the current IP core is a CPRI link slave, indicates the IP core received a reset request in the Z.130.0 control byte from the remote CPRI link master. If the current IP core is a CPRI link master, indicates the IP core received a reset acknowledgement in the Z.130.0 control byte from the remote CPRI link slave.

6. IP Registers

The eCPRI IP core registers are 32-bits wide and are accessible using the Avalon memory-mapped interface. This table lists the registers available in the IP core. All unlisted locations are reserved.

- eCPRI Intel FPGA IP Register Map

Table 54. Register Access Codes

Code	Description
RW	Read and write
RO	Read only
RW1C	Read, write, and clear. The user application writes 1 to the register bit(s) to invoke a defined instruction. The IP core clears the bit(s) upon executing instructions.

7. eCPRI Intel FPGA IP User Guide Archives

For the latest and previous versions of this user guide, refer to the eCPRI Intel FPGA IP User Guide HTML version. Select the version and click **Download**. If an IP or software version is not listed, the user guide for the previous IP or software version applies.

8. Document Revision History for eCPRI Intel FPGA IP User Guide

Document Version	Intel Quartus Prime Software Version	IP Version	Changes
2023.02.24	22.4	2.0.2	Bug fixes
2022.11.15	22.3	2.0.1	• Updated the *Device Speed Grade Support*. • Updated the *Resource Utilization*.
2022.08.26	22.2	2.0.0	• Added information about the packets arbitration scheme in section: *Transmit TX Path*. • Added information about the Data Flow Identification in section: *Receive RX Path*. • Added new IP parameters: — **Default VLAN ID** — **Data Flow Identification** — **Packets Arbitration Scheme** — **TX Packets Default Priority** — **TX Arbitration Queue 0 Depth** — **TX Arbitration Queue 1 Depth** — **TX Arbitration Queue 2 Depth** — **TX Arbitration Queue 3 Depth** — **TX Arbitration Queue 4 Depth** — **TX Arbitration Queue 5 Depth** — **TX Arbitration Queue 6 Depth** — **TX Arbitration Queue 7 Depth** • Added new signals: — `tx_queue_<N>_fifo_full` — `ext_source_pkt_type` — `ext_tx_ingress_timestamp_96b_data` — `ptp_tx_ingress_timestamp_96b_data` • Added a new register eCPRI `ORAN C/U Plane VLAN ID Match` in section: *IP Registers*.
2022.07.01	22.1	1.4.1	• Corrected the I/O direction for the following signals: — `avst_source_sop` — `avst_source_eop` — `avst_source_empty` • Corrected the signal name to which signals of the eCPRI IP Source and Sink Interface are synchronized. • Corrected the figure: *eCPRI IP Core Reset Logic*. • Added support for QuestaSim simulator. • Removed support for ModelSim* SE simulator.

continued...

Document Version	Intel Quartus Prime Software Version	IP Version	Changes
2021.12.14	21.3	1.4.1	• Corrected the signal descriptions for *Configuration Avalon Memory-Mapped Interface*. • Corrected the register descriptions for *Ethertype Register* and *UDP Port Register*. • Corrected IP version for Intel Quartus Prime software version 21.2.
2021.11.11	21.2	1.4.0	Clarified information about the streaming mode in section: *eCPRI IP Sink Interface* and *IP Parameters*.
2021.10.01	21.2	1.4.0	• Added support for Intel Agilex F-tile devices. • Added support for multi-channel designs. For more information, refer to the *eCPRI Intel FPGA IP Design Example User Guide*. • Removed support for NCSim.
2021.02.26	20.4	1.3.0	• Added support for Intel Agilex E-tile devices. • Updated the following signal descriptions: — `tx_egress_timestamp_96b_fingerprint` — `ptp_timestamp_request_fingerprint` • Added the following signals in section *External ST Sink Interface*: — `ext_ptp_timestamp_request_fingerprint` — `ext_tx_egress_timestamp_96b_fingerprint`
2021.01.08	20.3	1.2.0	• The IP now supports interworking function (IWF) type 0. • Supports pairing of eCPRI Intel FPGA IP with O-RAN Intel FPGA IP. • Updated resource utilization numbers for IWF in *Resource Utilization* section. • Updated *Table: eCPRI Intel FPGA IP Core Release Information* for 20.3 release. • Updated *Figure: eCPRI IP Parameter Editor* with new parameters. • Updated *Parameter Settings* section. • • Added following new interfaces in section *Interfaces*: — IWF Type 0 eCPRI Source Interface — IWF Type 0 eCPRI Sink Interface — IWF Type 0 CPRI MAC Interface • Updated *Figure: eCPRI Intel FPGA IP High-Level System Overview*. • Added description for two new blocks in section *Operation of the eCPRI IP Blocks*: — eCPRI IWF Type 0 • Added IWF related new clock signals in section *eCPRI IP Input Clocks*. • Added following reset signals in *Table: eCPRI IP Reset, Power, and Firewalls Signals*. • Created following new sections to document IWF Type 0 related signals: — *IWF Type 0 eCPRI Interface* — *IWF Type 0 eCPRI MAC Interface* • Corrected one field in *Table: eCPRI Common Header Format*.

continued...

Document Version	Intel Quartus Prime Software Version	IP Version	Changes
2020.05.19	20.1	1.1.0	• Added support for Intel Arria 10 devices. • The IP now supports 10G data rate with Intel Stratix 10 and Intel Arria 10 devices. • IP supports streaming of Ethernet frame size up to 9,000 bytes. • Added new *Table: eCPRI Intel FPGA IP Feature Matrix* in section *Supported Features*. • Updated resource utilization numbers in *Table: Resource Utilization*. • Added following new parameters in *Table: Parameters: Configuration Tab*: — **Streaming** — **Pair with ORAN** — **One-way Delay Measurement Timer Bit-width** — **Remote Memory Access Timer Bit-width** — **Remote Reset Timer Bit-width** • Modified *Figure: eCPRI Intel FPGA IP High-Level System Overview* to include client logic. • Updated *Section: Supported Ethernet Variants*. • Updated *Section: Error Handling*. • Added new signals in the following: — *Table: eCPRI IP Input Clocks* — *Table: Signals of the TX Time of Day Interface* — *Table: Signals of the External ST Sink Interface* — *Table: Signals of the eCPRI IP Sink Interface* • Added new *Table: Miscellaneous Interface Signals*. • Updated the following register tables: — *Table: eCPRI Version Register at Offset 0x000* — *Table: eCPRI TX Error Message Register at Offset 0x0004* — *Table: eCPRI RX Error Message Register at Offset 0x0005* — *Table: eCPRI Error Mask Message Register at Offset 0x0006* — *Table: RX Error Register at Offset 0x003E*
2020.04.15	19.4	1.0.0	Corrected information in *Table: eCPRI Version Register at Offset 0x000*.
2020.04.13	19.4	1.0.0	Initial release.