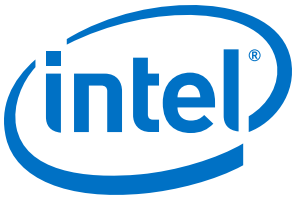


O-RAN Intel® FPGA IP User Guide

Updated for Intel® Quartus® Prime Design Suite: **20.2**

IP Version: **1.0.0**

Subscribe
Send Feedback

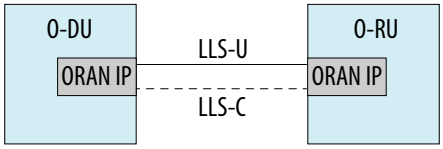
UG-20295 | 2020.09.04
Latest document on the web: **PDF** | **HTML**

Contents

1. About the O-RAN Intel® FPGA IP

The Extensible Radio Access Network (O-RAN WG4 Fronthaul Interface) defines a fronthaul interface between a lower-layer split distributed unit (DU) and remote unit (RU) in an Evolved Universal Terrestrial Access Network (E-UTRAN) and Next-Generation Radio Access Network (NG-RAN) system with a lower layer functional split-7-2x based architecture. The O-RAN IP implements control and user plane protocol specified in O-RAN-FH.CUS.0-v01.00. You can instantiate the O-RAN IP in both lower-layer split (LLS)-CU and RU modes. The IP does not support a synchronization plane. The IP splits protocol implementation between RTL targeting Intel® Arria® 10 and Intel Stratix® 10 devices and software targeting an ARM processor.

Figure 1. Architecture of eNB and gNB



For more information about O-RAN, refer to the *O-RAN Control and User Plane Specification* and the *O-RAN Management Plane Specification* on the O-RAN website.

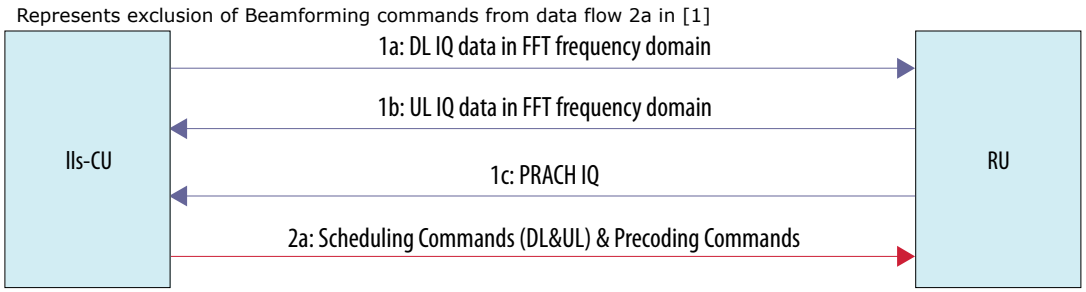
The O-RAN IP is compliant to *O-RAN Fronthaul Control, User and Synchronization Plane Version 1.0 - March, 2019 (O-RAN-WG4.CUS.0-v01.00)*

The O-RAN IP supports category A RUs and precoding for LTE TM2-TM4 in category B RUs.

The IP supports the following data flows:

- User-plane
  - Data Flow 1a: Flows of IQ Data in FFT frequency domain on downlink
  - Data Flow 1b: Flows of IQ Data in FFT frequency domain on uplink
  - Data Flow 1c: Flow of PRACH IQ data in FFT frequency domain
- C-plane
  - Data Flow 2a-: Scheduling commands (downlink and uplink) and precoding commands

Figure 2. Lower layer fronthaul data flows



The O-RAN IP provides delay management service to ensure that it receives correct data over the fronthaul interface despite packet delay variation (PDV). The IP refers to concept and latency models from the eCPRI specification. The IP manages transmission and receiver windows, which you can place relative to the air interface based on predefined or measured transport delay. The IP exchanges RU parameters and network characteristic over M-plane messages. The IP monitors and counts packets transmitted or received out of the window to warn the other node and discard them if necessary. The IP also transmits and receives non-delay managed U-plane traffic for which normal windows are not applicable.

The IP interface enables integration with either eCPRI or IEEE 1914.3 radio over Ethernet (RoE) transport layer with 10Gbps and 25Gbps Ethernet link rates.

The IP supports static-bit-width of 8 to 16 bits fixed-point IQ format. The IP supports μ-law and block floating-point companding to reduce fronthaul interface bandwidth.

The IP supports 64 bits of data width for O-RAN mapper and demapper logics and 128 bits of data widths in the compression and decompression blocks.

Related Information

O-RAN website

1.1. O-RAN Intel FPGA IP Features

- Support for CAT-A RU (up to 8 spatial streams)
- Support for CAT-B RU (precoding in RU)
- Bandwidth saving:
  - Programmable static bit-width fixed-point IQ
  - Real-time variable bit-width
  - Compressed IQ
  - Block floating-point compression
  - μ-law compression
  - Variable bit-width per channel (per data section)
  - Static configuration of U-plane IQ format and compression header
- Transmission blanking energy savings
- Preconfigured transport delay method CU–RU timing
- Section type 0 and type 1

1.2. O-RAN Intel FPGA IP Device Family Support

Intel offers the following device support levels for Intel FPGA IP:

- Advance support—the IP is available for simulation and compilation for this device family. FPGA programming file (`.pof`) support is not available for Quartus Prime Pro Stratix 10 Edition Beta software and as such IP timing closure cannot be guaranteed. Timing models include initial engineering estimates of delays based on early post-layout information. The timing models are subject to change as silicon testing improves the correlation between the actual silicon and the timing models. You can use this IP core for system architecture and resource utilization studies, simulation, pinout, system latency assessments, basic timing assessments (pipeline budgeting), and I/O transfer strategy (data-path width, burst depth, I/O standards tradeoffs).
- Preliminary support—Intel verifies the IP core 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. You can use it in production designs with caution.
- Final support—Intel verifies the IP with final timing models for this device family. The IP meets all functional and timing requirements for the device family. You can use it in production designs.

Table 1. O-RAN IP Device Family Support

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

1.3. Release Information for the O-RAN Intel FPGA IP

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

The IP version (X.Y.Z) number may change from one Intel Quartus Prime software version to another. A change in:

- X indicates a major revision of the IP. If you update your 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 2. O-RAN IP Release Information

Item	Description
Version	1.0.0
Release date	June 2020
Ordering code	IP-xRAN

1.4. O-RAN Intel FPGA IP Performance and Resource Utilization

The resource utilization of the O-RAN IP targeting an Intel Arria 10 device (10AS027E2F27E2SG) Intel Stratix 10 device (1SX280LU2F50E2VGS2).

The O-RAN IP operates at a 390.625 MHz synchronous clock frequency with the Ethernet MAC Intel FPGA IP.

Table 3. Resource Utilization

Intel generated the resource data with streaming mode.

Device	IP	ALMs	Logic Registers		Memory 20K
			Primary	Secondary	
Intel Arria 10	ORAN mapper and demapper	10,214	17,190	4,172	22
	Including compression and decompression	26,702	40,735	8,929	23
Intel Stratix 10	O-RAN mapper and demapper	12,836	22,881	3,729	23
	Including compression and decompression	35,909	67,788	11,802	24

Related Information

O-RAN IP Streaming Mode on page 28

2. Getting Started with the O-RAN Intel FPGA IP

Describes installing, parameterizing, simulating, and initializing the ORAN IP.

2.1. Obtaining, Installing, and Licensing the O-RAN IP

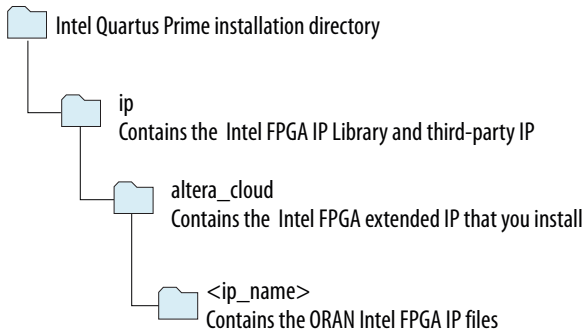
The O-RAN IP is an extended Intel FPGA IP that is not included with the Intel Quartus Prime release.

1. Create a My Intel account if you do not have one.
2. Log in to access the Self-Service Licensing Center (SSLC).
3. Purchase the O-RAN IP.
4. On the SSLC page, click **Run** for the IP.
   The SSLC provides an installation dialog box to guide your installation of the IP.
5. Install to the same location as Intel Quartus Prime folder.

Table 4. O-RAN 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*

Figure 3. O-RAN IP Installation Directory Structure



The O-RAN Intel FPGA IP now appears in the IP Catalog.

Related Information

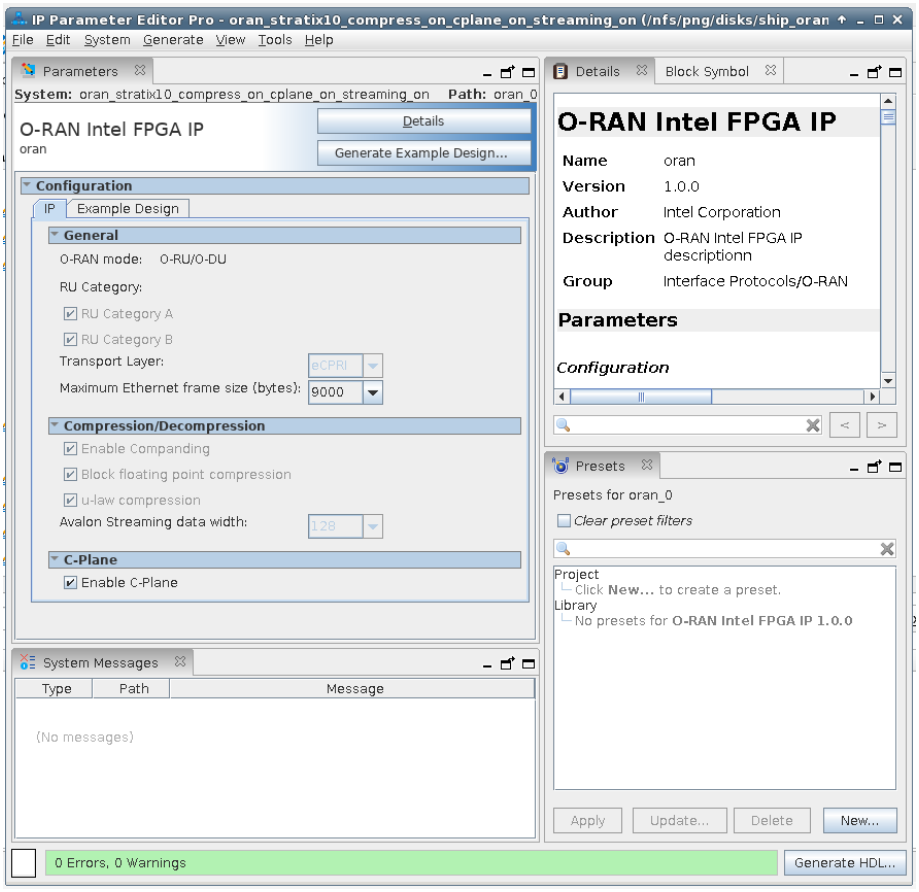
- Intel FPGA website
- Self-Service Licensing Center (SSLC)

2.2. Parameterizing the O-RAN IP

Quickly configure your custom IP variation in the IP Parameter Editor.

1. Create an Intel Quartus Prime Pro Edition project in which to integrate your IP core.
   a. In the Intel Quartus Prime Pro Edition, click **File ➤ New Project Wizard** to create a new Intel 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.
   c. Click **Finish**.
2. In the IP Catalog, select **O-RAN 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.

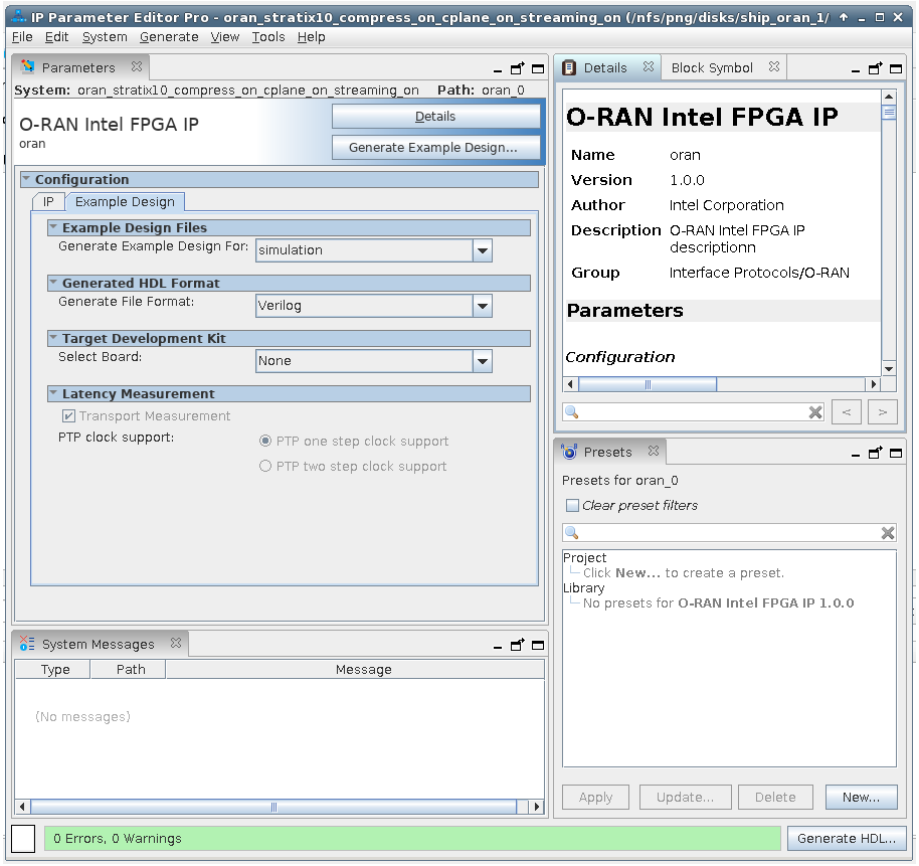
Figure 4. O-RAN IP Parameter Editor



5. Specify the parameters for your IP variation. Refer to *Parameters* for information about specific IP parameters.

6. Click the **Design Example** tab and specify the parameters for your design example.

Figure 5. Design Example Parameter Editor



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.

2.2.1. ORAN IP Parameters

Create custom variations of your IP.

Parameter Name	Values	Description
RU category	A or B	Select RU category, A or B.
Maximum Ethernet frame size	1500 or 9000	Specify the maximum Ethernet frame size. When value is greater than 1500, you must supply the packet size.
Enable companding	Off or on	Turn on for compression and decompression for U-plane IQ data.
ulaw compression	Off or on	Turn on for µ-law compression and decompression for U-plane IQ data. This parameter is available when you turn on **Enable companding**.
Block floating-point compression	Off or on	Turn on for block floating-point compression and decompression for U-plane IQ data. This parameter is available when you turn on **Enable compansion**.
Avalon Streaming data width	**128** when **Enable companding** is turned on. **64** when **Enable companding** is turned off.	Specify the Avalon streaming data width.
Enable C plane	Off or on	Turn on for the C-plane.

Related Information

O-RAN IP Streaming Mode on page 28

2.3. Generated IP File Structure

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

Table 5. Generated IP 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.
`<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.

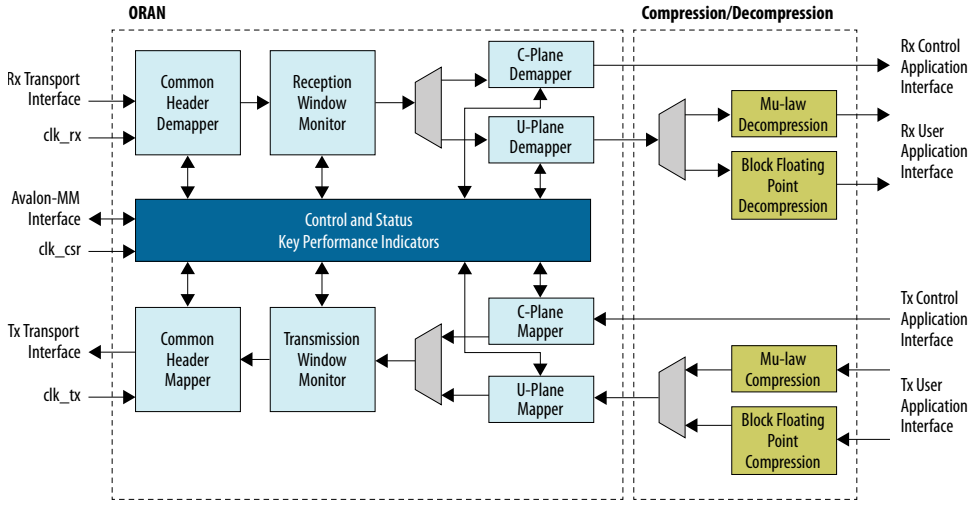
continued...

File Name	Description
	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 ModelSim* 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.
cadence/	Contains a shell script `ncsim_setup.sh` and other setup files to set up and run an NCSIM* simulation.
aldec/	Contains a shell script `rivierapro_setup.sh` to setup and run an Aldec* 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/` and `sim/` sub-directories.

3. O-RAN IP Functional Description

The ORAN IP comprises compression and decompression, mapper and demapper.

Figure 6. O-RAN IP Block Diagram

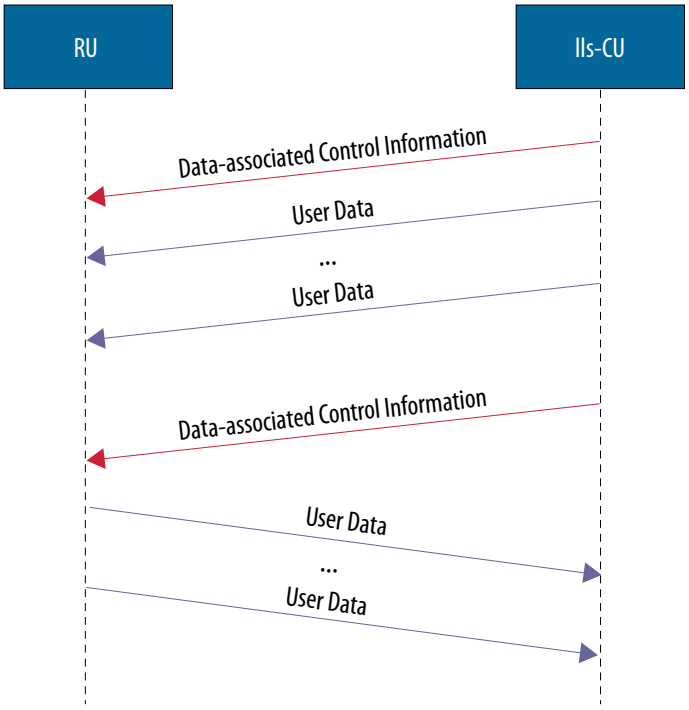


Mapper

The mapper includes section and common header mapping blocks. The common header consists of a time reference for each packet. The common header format is the same for C-plane and U-plane messages.

Figure 7. Scheduling control and user data transfer procedure

The IP transmits C-plane and U-plane messages at different times.



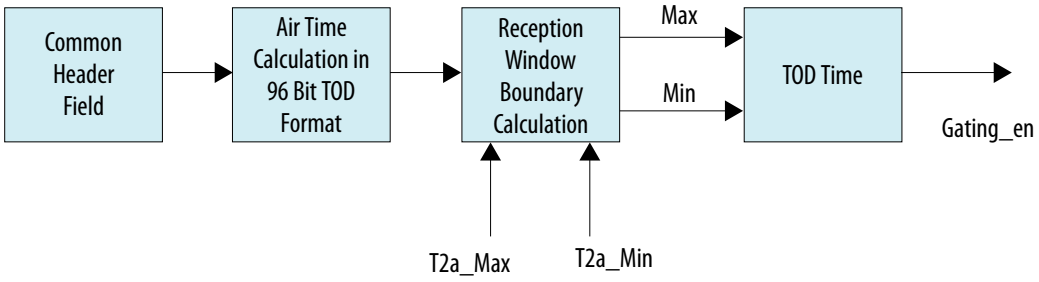
C-plane and U-plane messages have the same header format and different transmission time. The IP multiplexes one common header mapper instantiation for mapping both C-plane and U-plane messages. The IP stores information elements related to common header and `Rtcid` ID in a FIFO buffer to bypass section mapper.

Section Mapper

The section format is different for each section type in C-plane and U-plane messages. The IP instantiates a separate control and user mapper to interface with the client. A simple arbiter multiplexes the control and user mapper output for transmission window monitoring and common header mapping.

Figure 8. Transmission Window

Monitoring ensures the incoming packets fall under current time of day (TOD), if not the IP drops the current packet.



Common Header Mapper

The common header mapper appends the following fields to the start of every packet from the section mapper:

- `dataDirection`
- `payloadVersion`
- `filterIndex`
- `frameId`
- `subframeId`
- `slotId`
- `symbolId`

This block includes a dispatcher FSM to apply backpressure during insertion of header fields. The block also includes an output FIFO buffer to stream output data with zero or three cycle `readyLatency`.

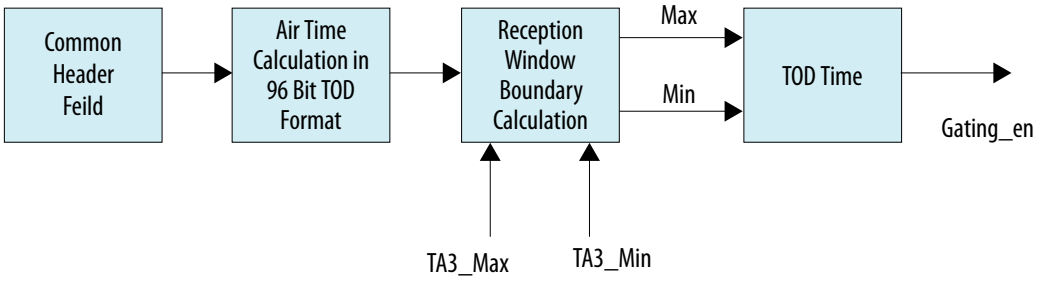
Common Header Demapper

The common header demapper demaps the radio application headers from the incoming eCPRI packet and forwards the O-RAN payloads to reception window monitor. This demapper is common for both U-plane and C-plane packets. For every SOP, the IP takes out the MSB nibble and decodes it as a common header. The IP appends the remaining LSB nibble with next clock cycle data and passes it to the next module.

Reception Window Monitoring

The window monitor monitors that the incoming packets fall under current time of day (TOD), if not it drops the current packet. The reception window monitoring shares the same module with the transmission window monitoring. They use different window thresholds, which you program through t2a registers.

Figure 9. Reception Window



Compression and Decompression

A preprocessing block-based bit shift block generates the optimum bit-shifts for a resource block of 12 resource elements (REs). The block reduces the quantization noise, especially for low-amplitude samples. Hence, it reduces the error vector magnitude (EVM) that compression introduces. The compression algorithm is almost

independent of the power value. Assuming the complex input samples is *x = x1 + jxQ*, the maximum absolute value of the real and imaginary components for the resource block is:

$$maxI_n = max\{\ |xI_{12(n-1)+1}|, |xI_{12(n-1)+2}|, \ldots, |xI_{12n}|\}$$

$$maxQ_n = max\{\ |xQ_{12(n-1)+1}|, |xQ_{12(n-1)+2}|, \ldots, |xQ_{12n}|\}$$

The maximum value of the resource block n is:

$$maxVal_n = max\{\ maxI_n\ ,\ maxQ_n\ \}$$

Having the maximum absolute value for the resource block, the following equation determines the left shift value assigned to that resource block:

$$lshift_n = \begin{cases} bitWidth - \lceil \log_2(maxVal_n) \rceil - 1 & if \quad maxVal_n < 2^{bitWidth-1} \\ 0 & else \end{cases}$$

Where `bitWidth` is the input bit width.

Mu-Law Compression and Decompression

The algorithm uses Mu-law companding technique, which speech compression widely uses. This technique passes the input uncompressed signal, x, through a compressor with function, f(x), before rounding and bit-truncation. The technique sends compressed data, y, over the interface. The received data passes through an expanding function (which is the inverse of the compressor, $F^{-1}(y)$. The technique reproduces the uncompressed data with minimal quantization error.

Equation 1. Compressor and decompressor functions

$$F(x) = sgn(x)\frac{ln(1+\mu|x|)}{ln(1+\mu)} \qquad -1 \le x \le 1$$

$$F^{-1}(y) = sgn(y)\frac{(1+\mu)^{|y|}-1}{\mu} \qquad -1 \le y \le 1$$

The IQ compression algorithm uses the Mu-val = 8 to balance orthogonal frequency-division multiplexing (OFDM) error vector magnitude (EVM) and implementation size. ITU-T Recommendation G.711 and G.191 specifies Mu-law compression.

3.1. O-RAN IP Signals

Connect and control the IP.

Table 6. Clock Signals

The IP operates at 390.625 MHz clock frequency asynchronously with the Ethernet MAC. The IP uses the same synchronous clock as the eCPRI IP, which runs at 390.625 MHz.

Signal Name	Direction	Description
clk_tx	Input	Clock for the transmitter logic. The frequency of this clock is 390.625 MHz for 25 Gbps (Intel Stratix 10 devices only) and 156.25 MHz for 10 Gbps. All transmitter interface signals are synchronous to clk_tx.
clk_rx	Input	Clock for the receiver logic. The frequency of this clock is 390.625 MHz for 10 or 25 Gbps (Intel Stratix 10 devices only) and 156.25 MHz for 10 Gbps. All receiver interface signals are synchronous to clk_rx.
clk_csr	Input	Clock for the control and status register interface. The frequency of this clock is 100 MHz.

Table 7. Reset Signals

Signal Name	Direction	Description
tx_rst_n	Input	Active low reset for transmitter interface synchronous to clk_tx.
rx_rst_n	Input	Active-low reset for receiver interface synchronous to clk_rx.
csr_rst_n	Input	Active-low reset for CSR interface synchronous to clk_csr.
tx_lanes_stable	Input	Indicates tx_clk clock signal is stable and transmitter path is ready to come out from reset.
rx_pcs_ready	Input	Indicates rx_clk clock signal is stable and receiver path is ready to come out from reset.

Table 8. Interrupt Signals

Signal	Bitwidth	Direction	Description
Irq	1	Output	Error interrupt signal. Indicates errors in the O-RAN IP. Software can poll Error Message register to determine error info. The signal is synchronous to clk_csr.

Table 9. Transmitter and Receiver TOD

Signal	Bitwidth	Direction	Description
tx_time_of_day_96b_data	96	Input	Current V2-format (96-bit) TOD in clk_txmac clock domain.
rx_time_of_day_96b_data	96	Input	Current V2-format (96-bit) TOD in clk_rxmac clock domain.

Table 10. CSR Signals

Signal	Bitwidth	Direction	Description
csr_address	16	Input	Config register address
csr_write	1	Input	Config register write enable.
csr_writedata	32	Input	Config register write data.
csr_readdata	32	Output	Config register read data.
csr_read	32	Output	Config register read enable.
csr_readdatavalid	1	Output	Config register read data valid.
csr_waitrequest	1	Output	Config register wait request.

Transport Interface

Table 11. Transmitter Signals

All transmitter interface signals are synchronous to `clk_tx`.

Signal	Bitwidth	Direction	Description
`avst_source_valid`	1	Output	When asserted, indicates valid data is available on `avst_source_data`.
`avst_source_data`	64	Output	Data to transport layer in network byte order.
`avst_source_startofpacket`	1	Output	Indicates first byte of a frame.
`avst_source_endofpacket`	1	Output	Indicates last byte of a frame.
`avst_source_ready`	1	Input	When asserted, indicates the transport layer is ready to accept data. `readyLayency` = 0 for this interface.
`avst_source_empty`	3	Output	Specifies the number of empty bytes on `avst_source_data` when `avst_source_endofpacket` is asserted.
`avst_source_error`	1	Output	When asserted in the same cycle as `avst_source_endofpacket`, indicates the current packet is an error packet.
`tx_transport_c_u`	1	Output	Indicates if packets transmitted to transport layer is C-plane or U-plane packet: 0 = User IQ data 1 = Control message.
`source_pc_id`	16	Output	`Pcid` for eCPRI transport and `RoEflowId` for RoE transport.
`source_rtc_id`	16	Output	`Rtcid` for eCPRI transport and `RoEflowId` for RoE transport.
`source_seq_id`	16	Output	Indicates the sequence ID of the packet. The eCPRI transport header uses this field.
`source_pkt_size`	16	Output	O-RAN packet size in bytes.

Figure 10. Transport Transmitter Interface Timing Diagram

Transport interface signals transmit O-RAN packet with `sectiontype=0` and `numberOfSections=10`

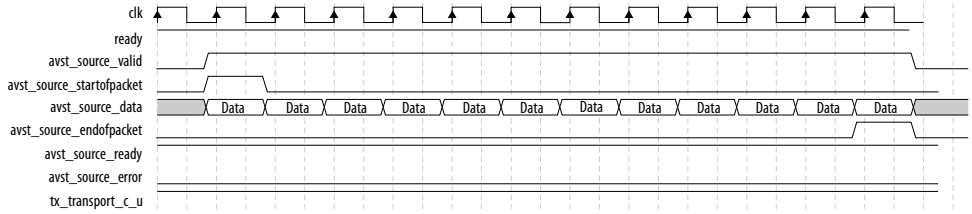


Table 12. Receiver Signals

All receiver interface signals are synchronous to `clk_rx`.

Signal	Bitwidth	Direction	Description
`avst_sink_valid`	1	Input	When asserted, indicates valid data is available on `avst_sink_data`.
`avst_sink_data`	64	Input	Data from transport layer in network byte order.
`avst_sink_startofpacket`	1	Input	Indicates first byte of a frame.
`avst_sink_endofpacket`	1	Input	Indicates last byte of a frame.

continued...

Signal	Bitwidth	Direction	Description
avst_sink_ready	1	Output	When asserted, indicates the O-RAN IP is ready to accept data from transport layer. readyLayency = 0 for this interface.
avst_sink_empty	3	Input	Specifies the number of empty bytes on avst_sink_data when avst_sink_endofpacket is asserted.
avst_sink_error	1	Input	When asserted in the same cycle as avst_sink_endofpacket, indicates the current packet is as an error packet.
rx_transport_c_u	1	Input	Indicates if packets received from transport layer is C-plane or U-plane packet 0 = User IQ data 1 = Control message.
sink_pc_id	16	Input	Pcid for eCPRI transport and RoEflowId for RoE transport.
sink_rtc_id	16	Input	Rtcid for eCPRI transport and RoEflowId for RoE transport.
sink_seq_id	16	Input	Indicates the sequence ID of the packet. The IP extracts this field from eCPRI transport header.

Application Interface Transmitter Signals

Table 13. Control Plane

All transmitter interface signals are synchronous to clk_tx.

Signal	Bitwidth	Direction	Description
avst_sink_c_valid	1	Input	When asserted, indicates valid section is available in this interface.
avst_sink_c_startofpacket	1	Input	Indicates the first section of a packet.
avst_sink_c_endofpacket	1	Input	Indicates the last section of a packet.
avst_sink_c_ready	1	Output	When asserted, indicates the O-RAN IP is ready to accept data from application interface. readyLatency = 0 for this interface.
tx_c_size	16	Input	C-plane packet size in bytes.
tx_c_rtc_id	16	Input	Rtcid for eCPRI transport and RoEflowId for RoE transport.
tx_c_seq_id	16	Input	SeqID of the packet appended to eCPRI transport header.
tx_c_dataDirection	1	Input	Drives common header IEs to the same value between avst_sink_c_startofpacket and avst_sink_c_endofpacket.
tx_c_filterIndex	4	Input	
tx_c_frameId	8	Input	
tx_c_subframeId	4	Input	
tx_c_slotID	6	Input	
tx_c_symbolid	6	Input	
tx_sectionType	8	Input	Drives section header IEs to the same value between avst_sink_c_startofpacket and avst_sink_c_endofpacket.
tx_timeOffset	16	Input	
tx_frameStructure	8	Input	
tx_cpLength	16	Input	

continued...

Signal	Bitwidth	Direction	Description
tx_c_udCompHdr	8	Input	
tx_c_sectionId	12	Input	Repeated section IEs synchronous with avst_sink_c_valid. For every cycle with avst_sink_c_valid = 1, the IP accepts new section IEs for mapping to the same packet between avst_sink_c_startofpacket and avst_sink_c_endofpacket.
tx_c_rb	1	Input	
tx_c_startPrb	10	Input	
tx_c_numPrb	8	Input	
tx_reMask	12	Input	
tx_ef	1	Input	
tx_beamid	15	Input	
tx_numSymbol	4	Input	
tx_frequencyOffset	24	Input	
tx_ext_sectionType	8	Input	Indicates which section type is associate for this section extension.
avst_sink_c_ext_valid	1	Input	When asserted, indicates valid section extension is available in this interface.
avst_sink_c_ext_startofpacket	1	Input	Indicates the first section extension of a packet.
avst_sink_c_ext_endofpacket	1	Input	Indicates the last section extension of a packet.
avst_sink_c_ext_data	64	Input	Section extension data from application layer in network byte order.
avst_sink_c_ext_empty	3	Input	Specifies the number of empty bytes on avst_sink_c_ext_data when avst_sink_c_ext_endofpacket is asserted.
avst_sink_c_ext_ready	1	Output	When asserted, indicates the O-RAN IP is ready to accept data from application interface. readyLatency = 0 for this interface.

Figure 11. Application Transmitter Interface Control Plane Timing Diagram

Application interface signals receives IEs with `sectionType=0` and `numberOfSections=10`. Avalon streaming sink, common header, and section header IEs are the same for entire packet. Only repeated section IEs vary for every `avst_sink_c_valid` cycle.

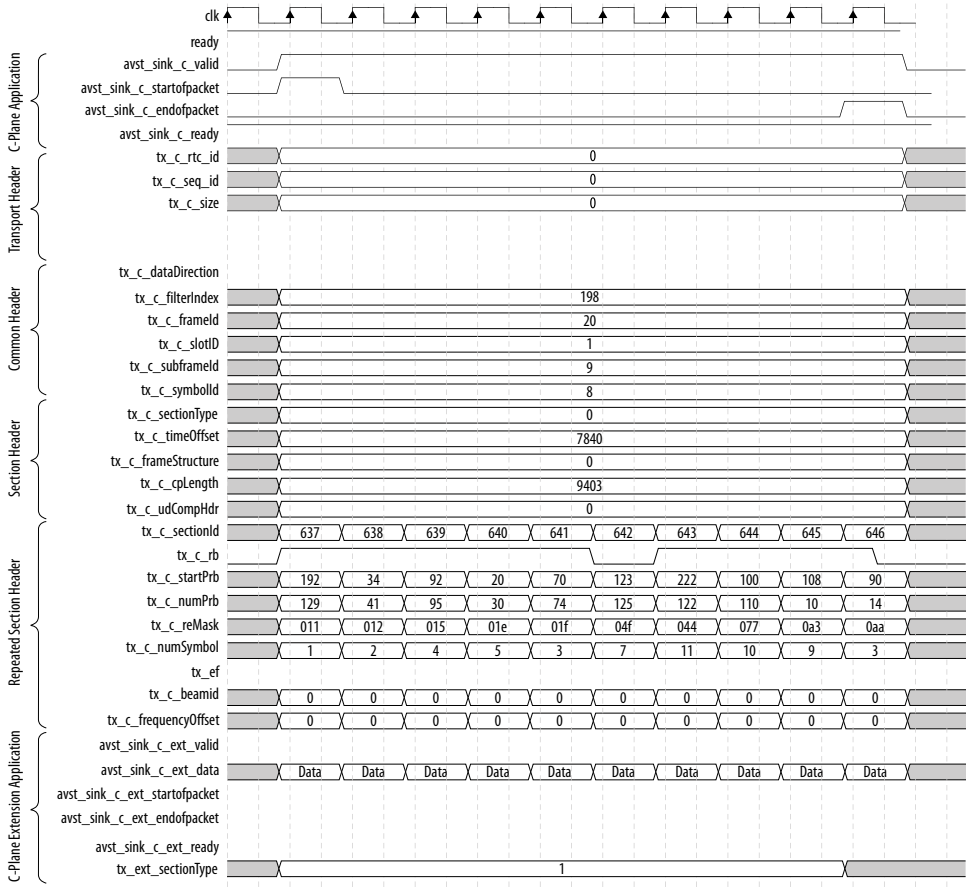


Table 14. User Plane Signals

All transmitter interface signals are synchronous to `clk_tx`.

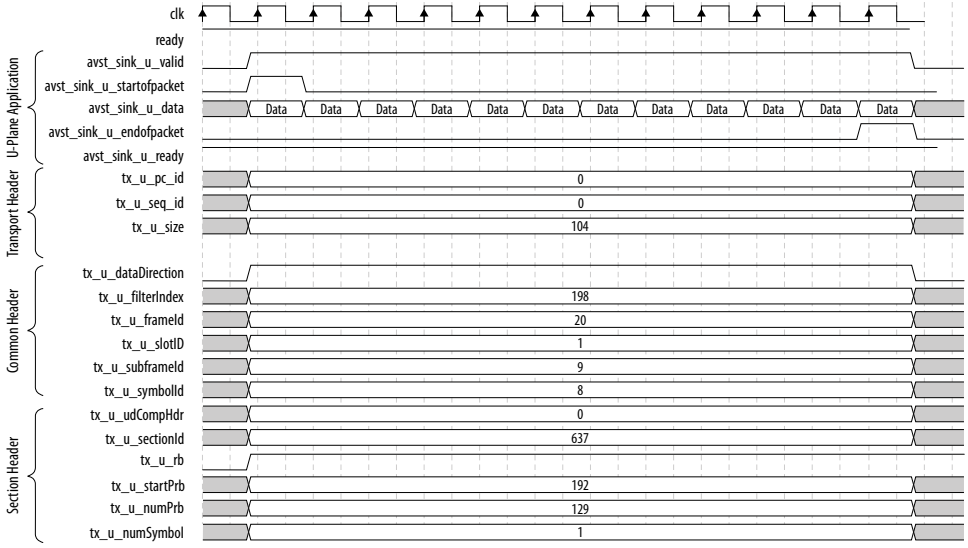
Signal	Bitwidth	Direction	Description
`avst_sink_u_valid`	1	Input	When asserted, indicates valid physical resource block (PRB) fields are available in this interface.
`avst_sink_u_data`	64/128	Input	PRB fields including `udCompParam`, `iSample` and `qSample`. Next section PRB fields are concatenated to the previous section PRB field. Data width is 128 when `EN_COMPANSION = 1`.
`avst_sink_u_startofpacket`	1	Input	Indicates the first PRB byte of a packet.
`avst_sink_u_endofpacket`	1	Input	Indicates the last PRB byte of a packet.
`avst_sink_u_empty`	3	Input	Indicates the number of empty bytes during end-of-packet. This signal only exists when `EN_COMPANSION = 0`.
`avst_sink_u_ready`	1	Output	When asserted, indicates the O-RAN IP is ready to accept data from the application interface. `readyLatency = 0` for this interface.

continued...

Signal	Bitwidth	Direction	Description
tx_u_size	16	Input	U-plane packet size in bytes.
tx_u_pc_id	16	Input	Pcid for eCPRI transport and RoEflowId for RoE transport.
tx_u_seq_id	16	Input	SeqID of the packet appended to the eCPRI transport header.
tx_u_dataDirection	1	Input	Drives common header IEs to the same value between avst_sink_u_startofpacket and avst_sink_u_endofpacket.
tx_u_filterIndex	4	Input	
tx_u_frameId	8	Input	
tx_u_subframeId	4	Input	
tx_u_slotID	6	Input	
tx_u_symbolid	6	Input	
tx_u_sectionId	12	Input	Repeated section IEs synchronous with avst_sink_u_valid. On presenting new section PRB fields in avst_sink_u_data, present new section IEs in these ports.
tx_u_rb	1	Input	
tx_u_startPrb	10	Input	
tx_u_numPrb	8	Input	
tx_u_udCompHdr	8	Input	

Figure 12. Application Interface Transmitter User Plane Timing Diagram

Application interface signals receive IEs with single PRB. The IP maintains the Avalon streaming sink, common header, and section header IEs the same for the entire packet



Application Interface Receiver Signals

Table 15. Control Plane Signals

All receiver interface signals are synchronous to `clk_rx`.

Signal	Bitwidth	Direction	Description
avst_source_c_valid	1	Output	When asserted, indicates valid section is available in this interface.
avst_source_c_startofpacket	1	Output	Indicates the first section of a packet.
avst_source_c_endofpacket	1	Output	Indicates the last section of a packet.
avst_source_c_ready	1	Input	When asserted, indicates the application interface is ready to accept data from O-RAN IP. `readyLatency` = 0 for this interface.
avst_source_c_error	1	Output	Indicates the packets contains error.
rx_c_rtc_id	16	Output	`Rtcid` for eCPRI transport and `RoEflowId` for RoE transport.
rx_c_seq_id	16	Output	`SeqID` of the packet, which the IP extracts from eCPRI transport header
rx_sec_hdr_valid	1	Output	Indicates the section data fields are valid. Drives a constant 0 when it is a C-plane packet.
rx_c_dataDirection	1	Output	Drives common header IEs to the same value between `avst_source_c_startofpacket` and `avst_source_c_endofpacket`.
rx_c_filterIndex	4	Output	
rx_c_frameId	8	Output	
rx_c_subframeId	4	Output	
rx_c_slotID	6	Output	
rx_c_symbolid	6	Output	
rx_sectionType	8	Output	Drives section header IEs to the same value between `avst_source_c_startofpacket` and `avst_source_c_endofpacket`.
rx_timeOffset	16	Output	
rx_frameStructure	8	Output	
rx_cpLength	16	Output	
rx_c_udCompHdr	8	Output	
rx_c_sectionId	12	Output	Repeated section IEs synchronous with `avst_source_c_valid`. For every cycle with `avst_source_c_valid` = 1, new section IEs stream out from same packet between `avst_source_c_startofpacket` and `avst_source_c_endofpacket`.
rx_c_rb	1	Output	
rx_c_startPrbc	10	Output	
rx_c_numPrbc	8	Output	
rx_reMask	12	Output	
rx_rf	1	Output	
rx_beamid	15	Output	
rx_numSymbol	4	Output	
rx_frequencyOffset	24	Output	

continued...

Signal	Bitwidth	Direction	Description
avst_source_c_ext_valid	1	Output	When asserted, indicates valid section extension is available in this interface.
avst_source_c_ext_startofpacket	1	Output	Indicates the first section extension of a packet.
avst_source_c_ext_endofpacket	1	Output	Indicates the last section extension of a packet.
avst_source_c_ext_error	1	Output	Indicates the packets contain errors.
avst_source_c_ext_data	64	Output	Section extension data to the application layer in network byte order.
avst_source_c_ext_empty	3	Output	Specifies the number of empty bytes on avst_source_c_ext_data when avst_source_c_ext_endofpacket is asserted.
avst_source_c_ext_ready	1	Input	When asserted, indicates the application interface is ready to accept data from O-RAN IP. readyLatency = 0 for this interface.
rx_ext_sectionType	8	Input	Indicates which section type is associate for this section extension.

Figure 13. Application Interface Receiver Control Plane Timing Diagram

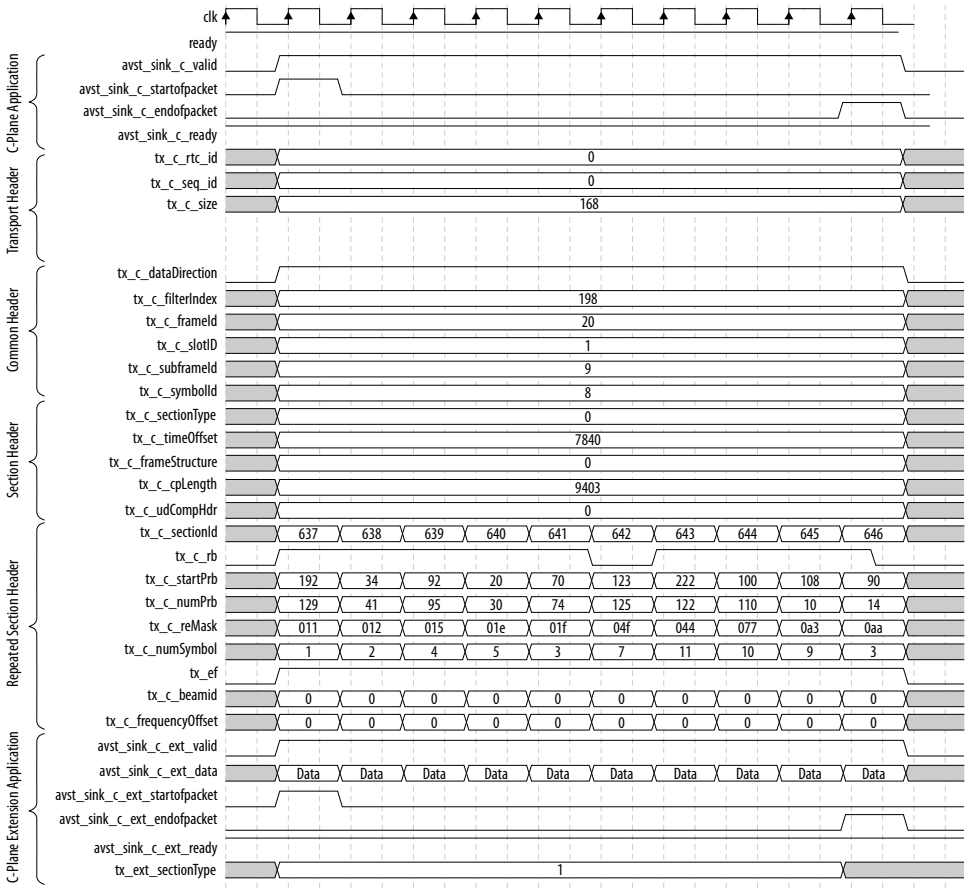


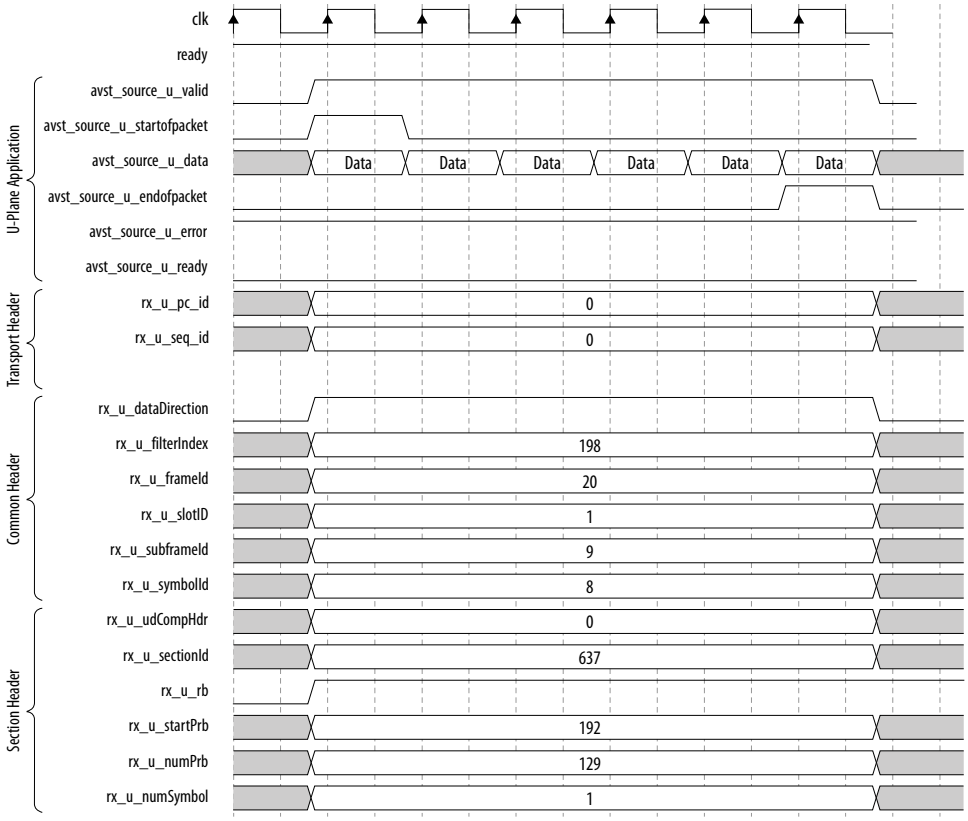
Table 16. User Plane Signals

All receiver interface signals are synchronous to `clk_rx`.

Signal	Bitwidth	Direction	Description
`avst_source_u_valid`	1	Output	When asserted, indicates valid PRB fields are available in this interface.
`avst_source_u_data`	64/128	Output	PRB fields including `udCompParam`, `iSample` and `qSample`. Next section PRB fields concatenate to previous section PRB field. Data width is 128 when `EN_COMPANSION = 1`.
`avst_source_u_startofpacket`	1	Output	Indicates the first PRB byte of a packet.
`avst_source_u_endofpacket`	1	Output	Indicates the last PRB byte of a packet.
`avst_source_u_empty`	3	Output	Indicates the number of empty bytes when `avst_source_c_ext_endofpacket` is asserted. This signal only exists when `EN_COMPANSION = 0`.
`avst_source_u_error`	1	Output	Indicates the packets contain errors.
`avst_source_u_ready`	1	Input	When asserted, indicates the O-RAN IP is ready to accept data from application interface. `readyLatency = 0` for this interface.
`rx_u_pc_id`	16	Output	`Pcid` for eCPRI transport and `RoEflowId` for RoE transport.
`rx_seq_id`	16	Output	`SeqID` of the packet, which the IP extracts from eCPRI transport header.
`rx_u_dataDirection`	1	Output	Common header IEs should produce the same value between `avst_sink_u_startofpacket` and `avst_sink_u_endofpacket`.
`rx_u_filterIndex`	4	Output	
`rx_u_frameId`	8	Output	
`rx_u_subframeId`	4	Output	
`rx_u_slotID`	6	Output	
`rx_u_symbolid`	6	Output	
`rx_u_sectionId`	12	Output	Repeated section IEs synchronous with `avst_sink_u_valid`. On presenting new section PRB fields in `avst_sink_u_data`, present new section IEs in these ports.
`rx_u_rb`	1	Output	
`rx_u_startPrb`	10	Output	
`rx_u_numPrb`	8	Output	
`rx_u_udCompHdr`	8	Output	

Figure 14. Application Interface Receiver User Plane Timing Diagram

The application interface signals send to the user logic with a single PRB. The IP maintains the Avalon streaming sink, common header and section header IEs the same for the entire packet.



3.2. O-RAN Intel FPGA IP Error Handling

Table 17. Error Handling

Events	Hardware Logging	Mitigations
Invalid transmission C-plane request section type	IP creates log in transmission error register.	IP drops request.
Transmission window check	IP creates log in transmission error register and interrupt signal asserted.	IP drops request.
Reception window check	IP creates log in receiver error register and asserts interrupt signal.	IP drops request.
Common or section FIFO overflow in section 0, 1, or, 3 mapper	IP creates log in transmission error register.	None.
Invalid common header (invalid section type, number of sections fields) for C-Plane data	IP creates log in receiver error register and asserts interrupt signal.	IP sends request to user application interface with Avalon streaming error indication.

continued...

Invalid section header (other than invalid `ef` field) for C- or U-plane data. Check `extType < MAX_EXTTYPE`	IP creates log in receiver error register and asserts interrupt signal.	IP sends request to user application interface with Avalon Streaming error indication.
Invalid PRB fields (invalid `uCompHdr, i/q` sample) for U-plane data. Static compression mode: ignore check Dynamic compression mode: • `udCompMeth` = 0000/0001/0011/ • `udIqWidth` = 1000 to 1111	IP creates log in receiver error register and asserts interrupt signal.	IP sends error request to user.
Invalid `numPrb` for U-plane data	IP asserts log in receiver error register and interrupt signal.	Invalid `numPrb`. IP sends request to user application interface with Avalon Streaming error indication
Incoming Avalon Streaming error packet	IP asserts log in receiver error register and interrupt signal.	Depends on the error events. IP sends request to user application interface with Avalon Streaming error indication

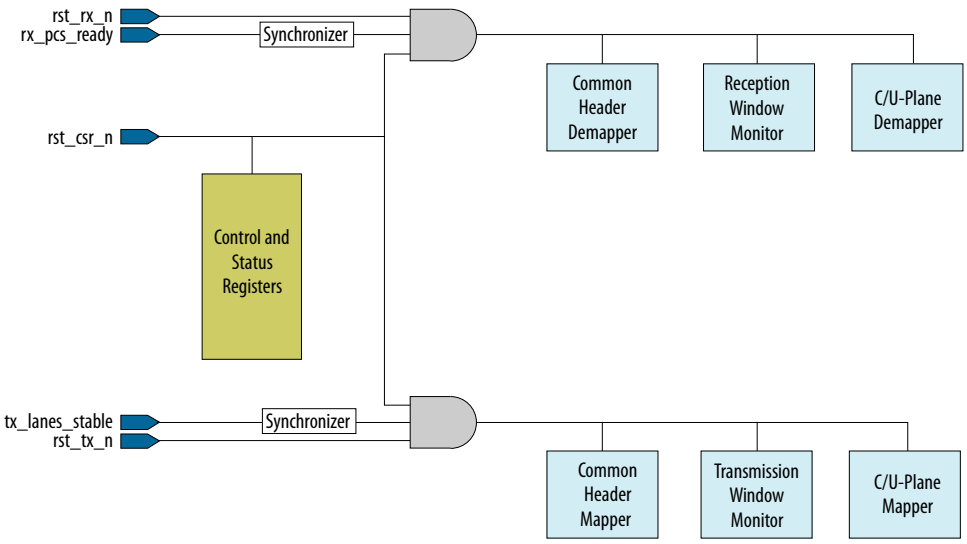
3.3. O-RAN Reset Transactions

The IP has five external reset ports.

The five external reset ports are:

- `rst_tx_n`. Resets the O-RAN IP in the transmission direction. Resets the common header mapper, transmission window monitor and C and U-plane mapper.
- `rst_rx_n`. Resets the O-RAN IP in the receiver direction. Resets the common header demapper, reception window monitor and C- and U-plane demapper.
- `rst_csr_n`. Resets the O-RAN IP control and status registers.
- `tx_lanes_stable`. Resets the O-RAN IP in the transmission direction. Deassertion indicates the transmitter clock is stable and O-RAN IP transmitter path is ready to come out from reset. Connect this reset to the Ethernet MAC output or tie to 1.
- `rx_pcs_ready`. Resets the O-RAN IP in the receiver direction. Deassertion indicates the receiver clock is stable and O-RAN IP receiver path is ready to come out from reset. Connect this reset to the Ethernet MAC output or tie to 1.

Figure 15. O-RAN IP Resets



Intel expects the three external reset ports to assert together to fully reset the O-RAN IP. You can deassert the three reset ports together or deassert `rst_csr_n`, then `reset_tx_n`, and `reset_rx_n` to reset CSR, transmitter path, receiver path, respectively.

The reset flow occurs before the O-RAN IP starts. Deassert the Avalon Streaming application interface ready signals to indicate IP is not ready to receive any transaction.

Alternatively, you can trigger a reset after reconfiguring O-RAN IP during run time.

Figure 16. Reset Deassertion

The timing diagram shows `tx_lanes_stable` and `rx_pcs_ready` tied to 1 and both the ORAN and Ethernet IP sharing the same CSR, transmit, and receive reset ports. On deasserting a reset,the ORAN IP deasserts the internal IP reset and sends a request when the eCPRI IP asserts the Avalon Streaming ready signal.

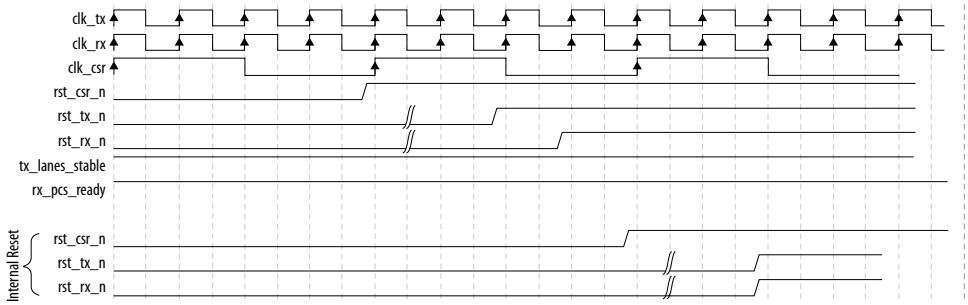
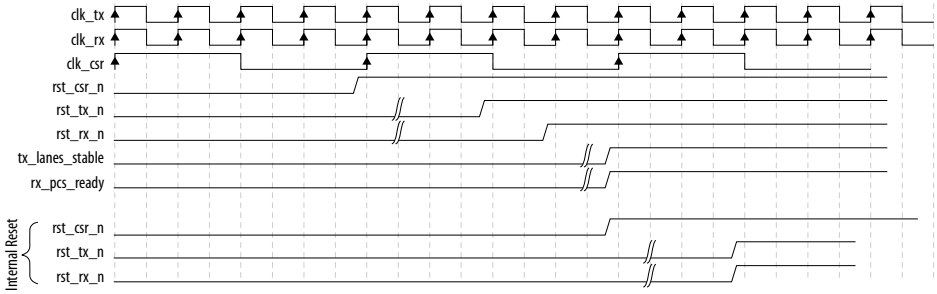


Figure 17. Reset Deassertion

The timing diagram shows the `tx_lanes_stable` and `rx_pcs_ready` connected to the Ethernet MAC output interface and both the ORAN and Ethernet IP sharing the same CST, transmit and receive reset ports. On deasserting a reset, the Ethernet MAC cycles through the reset deassertion flow and then asserts `tx_lanes_stable` and `rx_pcs_ready`. When the ORAN IP sees `tx_lanes_stable` and `rx_pcs_ready` assertions, the ORAN IP deasserts the internal IP reset.



3.4. O-RAN IP Streaming Mode

Enable by specifying **Maximum Ethernet frame size** to 9000.

In streaming mode, provide the packet size for C-plane packets that includes sections or section extension headers and their content. Similarly, provide the packet size for U-plane packets. If you turn on **Block Floating-point Compression**, the packet size is based on the compressed IQ output data.

The ORAN IP passes the packet size input to the eCPRI IP for appending into the eCPRI header of the final ORAN packet.

Figure 18. C-plane packet waveform

The C-plane packet contains 10 sections and 20 section extensions. Therefore, the `tx_c_size` is 168 (8B common header, 10 sections – 80B, 20 section extensions - 80B).

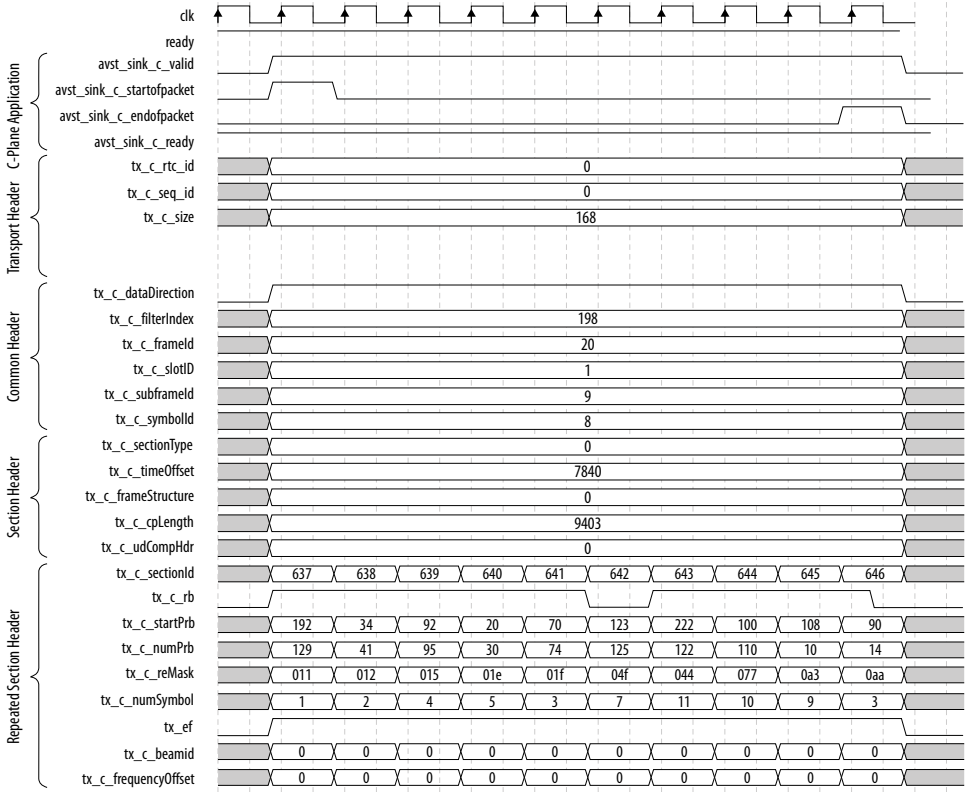
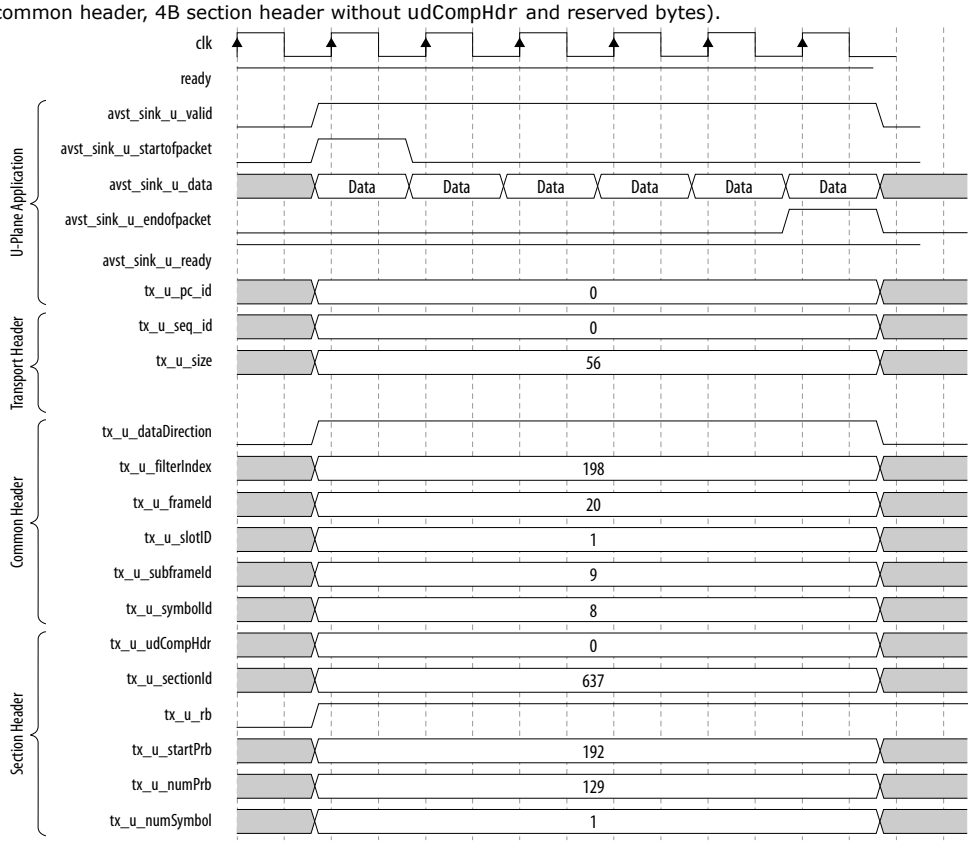


Figure 19. U-plane packet waveform

The U-plane packet contains two PRBs within single section and therefore the `tx_u_size` is 104 (2 PRB – 96B, 4B common header, 4B section header without `udCompHdr` and reserved bytes).



Related Information

ORAN IP Parameters on page 9

4. O-RAN IP Registers

Control and monitor O-RAN IP functionality through control and status interface.

Table 18. Register Map

CSR_ADDRESS (Word Offset)	Register Name
0x0	t2a_min_up
0x1	t2a_max_up
0x2	t2a_min_cp_ul
0x3	t2a_max_cp_ul
0x4	t2a_min_cp_dl
0x5	t2a_max_cp_dl
0x6	ta3_min_up
0x7	ta3_max_up
0x8	rx_window_enable
0x9	tx_window_enable
0xA	functional_mode
0xB	static_udCompHdr
0xC	Tx Error
0xD	Rx Error
0xE	Tx Error Mask
0xF	Rx Error Mask

Table 19. t2a_min_up Register

Bit Width	Description	Access	HW Reset Value
31:0	Minimum RU to antenna uplink delay	RW	0x0

Table 20. t2a_max_up Register

Bit Width	Description	Access	HW Reset Value
31:0	Maximum RU to antenna uplink delay	RW	0x0

Table 21. t2a_min_cp_ul Register

Bit Width	Description	Access	HW Reset Value
31:0	Minimum RU to antenna uplink control plane delay	RW	0x0

Table 22. t2a_max_cp_ul Register

Bit Width	Description	Access	HW Reset Value
31:0	Maximum RU to antenna uplink control plane delay	RW	0x0

Table 23. t2a_min_cp_dl Register

Bit Width	Description	Access	HW Reset Value
31:0	Minimum RU to antenna downlink control plane delay	RW	0x0

Table 24. t2a_max_cp_dl Register

Bit Width	Description	Access	HW Reset Value
31:0	Maximum RU to antenna uplink control plane delay	RW	0x0

Table 25. ta3_min_up Register

Bit Width	Description	Access	HW Reset Value
31:0	Minimum antenna to RU uplink delay	RW	0x0

Table 26. ta3_max_up Register

Bit Width	Description	Access	HW Reset Value
31:0	Maximum antenna to RU uplink delay	RW	0x0

Table 27. rx_window_enable Register

Bit Width	Description	Access	HW Reset Value
31:1	Reserved	RO	0x0
0:0	Receiver window enable	RW	0x0

Table 28. tx_window_enable Register

Bit Width	Description	Access	HW Reset Value
31:1	Reserved	RO	0x0
0:0	Transmission window enable	RW	0x0

Table 29. functional_mode Register

Bit Width	Description	Access	HW Reset Value
31:1	Reserved	RO	0x0
0:0	Functional mode: 0 – Static compression mode 1 – Dynamic compression mode	RW	0x0

Table 30. static_udCompHdr Register

Bit Width	Description	Access	HW Reset Value
31:8	Reserved	RO	0x0
7:0	Static user data compression header 7:4 – udIqWidth 4'b0000 – 16 bits 4'b1111 – 15 bits : 4'b0001 – 1 bit 3:0 - udCompMeth 4'b0000 – No compression 4'b0001 – Block Floating Point 4'b0011 - μ-law Others - reserved	RW	0x0

Table 31. tx_error Register

Bit Width	Description	Access	HW Reset Value
31:8	Reserved	RO	0x0
7:7	Invalid C-Plane request section type	RW1C	0x0
6:6	Section 3 mapper section FIFO overflow	RW1C	0x0
5:5	Section 3 mapper common FIFO overflow	RW1C	0x0
4:4	Section 1 mapper section FIFO overflow	RW1C	0x0
3:3	Section 1 mapper common FIFO overflow	RW1C	0x0
2:2	Section 0 mapper section fifo overflow	RW1C	0x0
1:1	Section 0 mapper common FIFO overflow	RW1C	0x0
0:0	Transmission window check error	RW1C	0x0

Table 32. rx_error Register

Bit Width	Description	Access	HW Reset Value
31:6	Reserved	RO	0x0
5:5	Incoming Avalon Streaming error packet	RW1C	0x0
4:4	Invalid receiver U-plane request—`udCompHdr` fields	RW1C	0x0
3:3	Invalid receiver U-plane request—PRB fields	RW1C	0x0
2:2	Invalid receiver C-plane request—section header	RW1C	0x0
1:1	Invalid receiver C-plane request—common header	RW1C	0x0
0:0	Reception window check error	RW1C	0x0

Table 33. tx_error_mask Register

Bit Width	Description	Access	HW Reset Value
31:1	Reserved	RO	0x0
0:0	Transmission window check error mask	RW	0x0

Table 34. rx_error_mask Register

Bit Width	Description	Access	HW Reset Value
31:6	Reserved	RO	0x0
5:5	Incoming Avalon Streaming error packet error mask	RW	0x0
4:4	Invalid receiver U-plane request—`udCompHdr` fields	RW	0x0
3:3	Invalid receiver U-plane request—PRB fields error mask	RW	0x0
2:2	Invalid receiver C-plane request—section header error mask	RW	0x0
1:1	Invalid receiver C-plane request—common header error mask	RW	0x0
0:0	Receiver window check error mask	RW	0x0

Table 35. error_log Register

Bit Width	Description	Access	HW Reset Value
31:8	Reserved	RO	0x0
7:0	Error C-plane request—section type	RO	0x0

5. Document Revision History for the O-RAN Intel FPGA IP User Guide

Date	IP Version	Intel Quartus Prime Software Version	Changes
2020.09.04	1.0.0	20.2	Initial release.