AXI Streaming Intel® FPGA IP for PCI Express* User Guide

Updated for Intel® Quartus® Prime Design Suite: **23.4**

IP Version: **1.0.0**

Online Version
Send Feedback

790711
2024.02.12

Contents

1. Introduction

The AXI Streaming Intel® FPGA IP for PCI Express* allows you to implement PCI Express (PCIe) in your design using Intel's technology leading PCIe* hardened protocol stack with an AXI4 user interface. The IP includes the hardened transaction, data link and physical layers, as well as optional blocks and interface adapters to interface with Direct Memory Access (DMA) and Scalable Switch Intel FPGA IPs for applications requiring high-bandwidth data transfer between the host or virtual machine and the I/O devices. This document introduces the various Intel FPGA PCI Express IP offerings and details the AXI Streaming Intel FPGA IP for PCI Express, including features and functional descriptions of the various blocks within the IP. This document also describes the design flow requirements and guidelines, IP parameters, interfaces, and signals available to you when you use this IP.

1.1. Goal of the AXI Streaming Intel FPGA IP for PCI Express User Guide

The goal of the AXI Streaming Intel FPGA IP for PCI Express User Guide is to help you:

- Understand the features supported by this IP.
- Parameterize the IP for a specific application.
- Understand the IP interfaces and how to connect them to the user logic.
- Learn how to drive clock and reset inputs to the IP.
- Simulate and compile the IP (standalone).
- Simulate and compile the IP as part of the design examples.

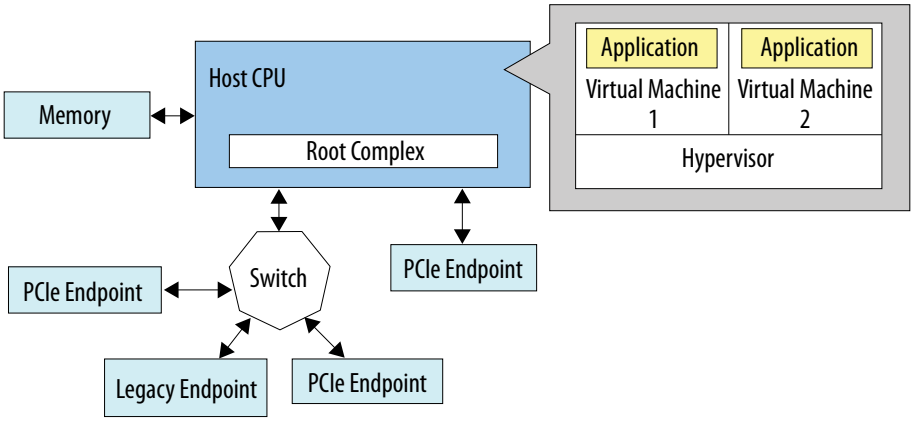
1.2. Intended Audience for the AXI Streaming Intel FPGA IP for PCI Express User Guide

This guide is to be used by FPGA designers who will be implementing PCIe using the AXI Streaming Intel FPGA IP for PCI Express.

1.3. What is PCI Express?

PCI Express is a point-to-point, serial interconnect bus with protocol stack that includes Transaction, Data Link, and Physical Layer.The protocol is scalable – from 1 lane to 16 lanes per link, with data on the link serialized and sent from one device to another. It uses differential signaling with complementary pair of signals for transmit and receive sides and uses packet-based transactions. You can use the Intel FPGA PCI Express IPs available in the Intel Quartus® Prime Pro Edition Edition Catalogue to implement PCI Express in your designs.

Figure 1. PCI Express Topology



1.4. What are the Intel FPGA IPs for PCI Express?

The Intel FPGA devices offer a wider variety of IPs for users to implement PCI Express in their designs. Along with the AXI Streaming Intel FPGA IP for PCI Express, the table below shows the various Intel IPs that integrate PCIe as part of the IP. Features that are enabled are indicated by an "X" in the table below. If you select an IP below that does not support a required feature, you can implement it in your application logic. For example, TLP Packet Formation in the AVST IP will need to be handled by the application logic.

- Avalon® Streaming Intel FPGA IP for PCI Express:
  - R-Tile Avalon Streaming Intel FPGA IP for PCI Express
  - F-Tile Avalon Streaming Intel FPGA IP for PCI Express
  - P-Tile Avalon Streaming Intel FPGA IP for PCI Express
- Multi Channel DMA Intel FPGA IP for PCI Express
- Scalable Switch Intel FPGA IP for PCI Express
- AXI Streaming Intel FPGA IP for PCI Express

Table 1. Intel FPGA PCI Express IPs

Features	Avalon Streaming			Multi Channel DMA			Scalable Switch			AXI Streaming		
Tile	P	F	R	P	F	R	P	F	R	P	F	R
Device / IP												
Intel Agilex® 7 device support	X	X	X	X	X	X	X	X	X	X	X	X
Intel Stratix® 10 device support	X	N/A	N/A	X	N/A	N/A	X	N/A	N/A	N/A	N/A	N/A
Simulation support	X	X	X	X	X	X	X	X	X	X	X	X
Hardware support	X	X	X	X	X	X	X			X	X	X
Static port bifurcation	X	X	X	X	X	X	N/A	N/A	N/A	X	X	X

continued...

Features	Avalon Streaming			Multi Channel DMA			Scalable Switch			AXI Streaming		
Tile	**P**	**F**	**R**	**P**	**F**	**R**	**P**	**F**	**R**	**P**	**F**	**R**
Independent Reference clock support for 2x8 bifurcated port	X	X	X	X	X	X	N/A	N/A	N/A	X	X	X
Independent PERST support (GPIO)	X	X	X	X	X	X	N/A	N/A	N/A	X	X	X
Independent PERST support (pin) (* For select Intel Agilex® I-Series devices only)			X*				N/A	N/A	N/A			X*
Autonomous HIP	X	X	X	X	X	X	N/A	N/A	N/A	X	X	X
Configuration via Protocol (CvP) (Init, Update)	X	X	X	X	X	X	N/A	N/A	N/A	X	X	X
TLP packet formation				X	X	X						
Link partner credit handling				X	X	X						
Transaction ordering	X	X	X	X	X	X						
Completion reordering				X	X	X						
Device-dependent programmable application clock frequency	X	X	X	X	X	X	X	X	X	X	X	X
Avalon streaming interface support	X	X	X	X	X	X	X	X	X			
Avalon Memory-Mapped interface support				X	X	X						
AXI-4 streaming interface support (datapath)										X	X	X
Error interface for application to report errors	X	X	X				X	X	X	X	X	X
Completion timeout interface	X	X	X							X	X	X
Configuration intercept interface	X	X	X	X	X	X	X	X	X	X	X	X
Debug toolkit	X	X	X	X	X	X						
Design Example Generation	X	X	X	X	X	X	X	X		X* (Simulation only)		X* (Simulation only)

continued...

Features	Avalon Streaming			Multi Channel DMA			Scalable Switch			AXI Streaming		
Tile	P	F	R	P	F	R	P	F	R	P	F	R
Design Example Driver support	X	X	X	X	X	X	X	X				
PCI Express Features												
Native Gen3 speed	X	X	X	X	X	X	X	X	X	X	X	X
Native Gen4 speed	X	X	X	X	X	X	X	X	X	X	X	X
Native Gen5 speed			X			X						X
Multi-lane link (x16, x8, x4) (* Only x16 and x8 supported currently)	X	X	X	X	X	X	X	X	X (x4, x8 only)	X*	X*	X*
Native Endpoint	X	X	X	X	X	X	N/A	N/A	N/A	X	X	X
Root Port	X	X	X	X	X	X	N/A	N/A	N/A			
Transaction layer bypass (TL Bypass)	X	X	X				N/A	N/A	N/A			
Separate reference clock with Independent Spread Spectrum Clocking (SRIS)	X	X	X	X	X	X	X	X	X	X	X	X
Separate Reference clock with no Spread Spectrum Clocking (SRNS)	X	X	X	X	X	X	X	X	X	X	X	X
Common reference clock architecture	X	X	X	X	X	X	X	X	X	X	X	X
Advanced error reporting (AER)	X	X	X	X	X	X	X	X	X	X	X	X
Up to 512-byte maximum payload size (MPS)	X	X	X	X	X	X	X	X	X	X	X	X
Up to 4096-byte (4K) maximum read request size (MRRS)	X	X	X				X	X	X	X	X	X
32/64-bit BAR support (prefetchable/ non-prefetchable)	X	X	X	X	X	X	X	X	X	X	X	X
Expansion ROM BAR support	X	X	X	X	X	X				X	X	X
Single virtual channel (VC)	X	X	X	X	X	X	X	X	X	X	X	X

continued...

Features	Avalon Streaming			Multi Channel DMA			Scalable Switch			AXI Streaming		
Tile	**P**	**F**	**R**	**P**	**F**	**R**	**P**	**F**	**R**	**P**	**F**	**R**
MSI (Capability registers only)	X	X	X	X	X	X	X	X	X	X	X	X
MSI-X (Capability registers only)	X	X	X	X	X	X	X	X	X	X	X	X
PM (Capability registers only)	X	X	X				X	X	X	X	X	X
PRS (Capability registers only)	X	X	X	X	X	X				X	X	X
LTR (Capability registers only)	X	X	X							X	X	X
ACS (Capability registers only)	X	X	X							X	X	X
Vendor specific (Capability registers only)	X	X	X	X	X	X				X	X	X
10-bit tag support	X	X	X	X	X	X	X	X	X	X	X	X
MSI-X Table				X	X	X						
Address Remapping Between Remote Host and Local Fabric Address Map (Device-ATT) (* Root Port only)				X*	X*	X*						
Multi-function and virtualization												
Single root IO virtualization (SR-IOV)	X	X	X	X	X	X	X	X	X	X	X	X
Functional level reset (FLR)	X	X	X	X	X	X	X	X	X	X	X	X
TLP processing hint (TPH)	X	X	X	X	X	X	X	X	X	X	X	X
Alternative Routing-ID Interpretation (ARI)	X	X	X				X	X	X	X	X	X
Address Translation Services (ATS)	X	X	X	X	X	X	X	X	X	X	X	X
Process Address Space ID (PasID)	X	X	X				X	X	X	X	X	X
VirtIO (Capability registers only)	X	X	X				X	X	X	X	X	X

Note: For more details on the support for the features in the table above, refer to the respective IP User Guides.

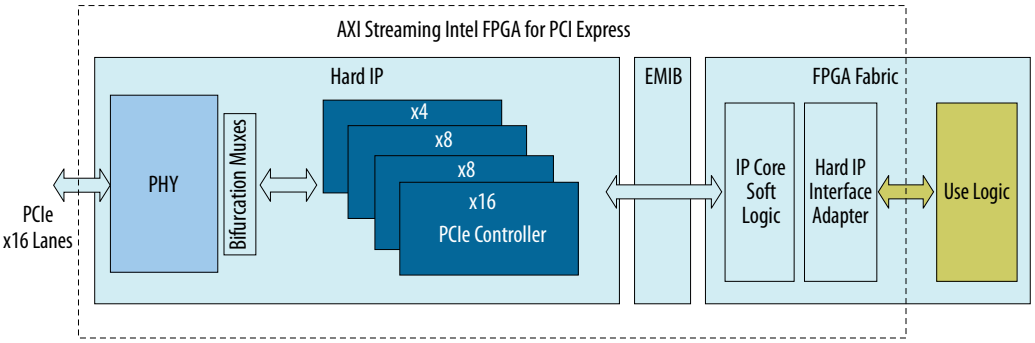
Refer to the Intel FPGA PCI Express IP Support Center for details on each IP.

1.5. What is the AXI Streaming Intel FPGA IP for PCI Express?

The AXI Streaming Intel FPGA IP for PCI Express supports PCI Express Gen3, Gen4, and Gen5 in Endpoint mode. It includes the PCIe Hard IP (HIP) and HIP Interface Adaptor that converts the native Hard IP interface to an AXI-ST interface. It allows you to choose various blocks and integrate with other Intel FPGA PCI Express IPs like MC-DMA based on the application requirement. The IP provides you the flexibility and control over the transaction layer packets by providing parametrization capabilities, functional modes, optional interfaces, error reporting, and debug capabilities. It implements basic telemetry functionality as well.

The figure below shows the block diagram of the AXI Streaming Intel FPGA IP for PCI Express. This document covers functional mode description, parameterization, and interface definitions for the IP and its various modes.

Figure 2. AXI Streaming Intel FPGA IP for PCI Express Block Diagram



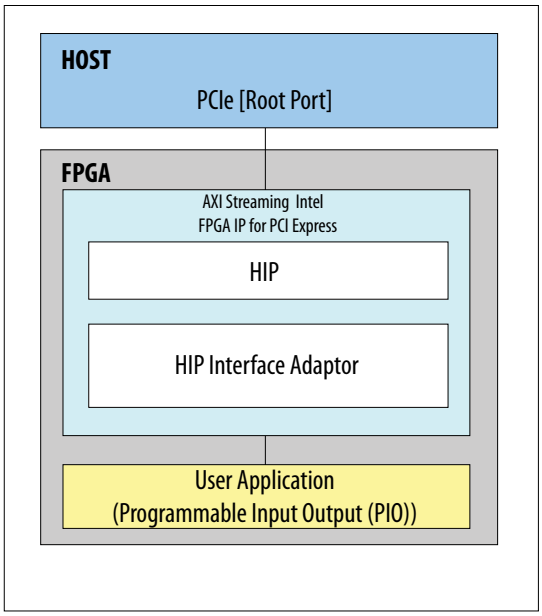
Refer to the IP Architecture and Functional Description chapter for details on each of the blocks.

1.6. Example Use Models

The AXI Streaming Intel FPGA IP for PCI Express can be used in various applications such as an endpoint, virtualization, inline processing, lookaside memory processing, etc. to move data between the source and destination.

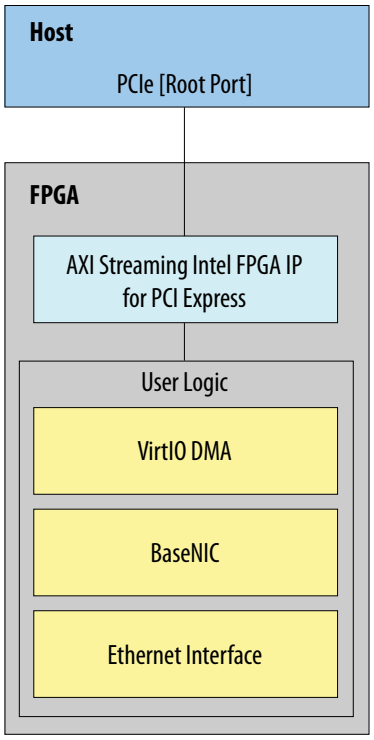
The following figure shows a simple block diagram of the IP in Endpoint mode connected to a root port on a host. You can run an application like the Programmable Input Output (PIO) to perform writes/reads to the host memory.

Figure 3. Example of the Programmable Input Output (PIO) Example Using the AXI Streaming Intel FPGA IP for PCI Express



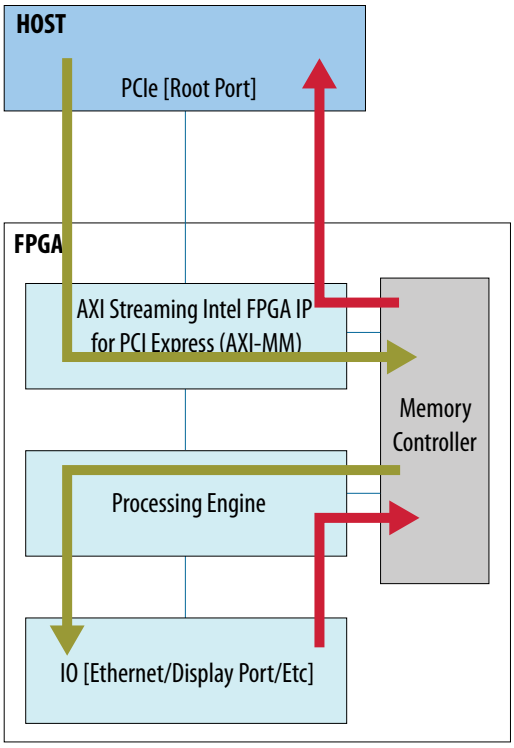
The following figure shows two examples of the IP connected to a processing engine directly using AXI streaming interface. The processing engine can be custom user logic implemented in the FPGA fabric, and can perform functions like DMA, e.g., VirtIO DMA connected to a BaseNIC. It receives data from HOST over AXI streaming interface and sends data towards HOST over AXI streaming interface.

Figure 4. Examples of the Inline Processing Model Using the AXI Streaming Intel FPGA IP for PCI Express



The AXI Streaming Intel FPGA IP for PCI Express can also be used in applications like lookaside memory processing. The following figure shows an example of the IP interfacing with a Memory controller. The processing engine can be custom user logic implemented in the FPGA fabric and can perform functions like DMA. The IP stores data coming from HOST into external memory. The processing engine then reads this data from external memory and performs required operations. Similarly, data coming from other IO devices are stored by the processing engine into external memory. If the processing engine wishes to transfer this data to HOST, it requests the IP to read data from external memory and send it to HOST using the command interface.

Figure 5. Example of Lookaside Memory Processing Model Using the AXI Streaming Intel FPGA IP for PCI Express



1.7. Design Flow Requirements

1.7.1. Design Software

Intel Quartus Prime Pro Edition Software Requirements for the AXI Streaming Intel FPGA IP for PCI Express are the following:

- Intel Quartus Prime Pro Edition 23.4
- Synopsys* VCS Simulator version - U/2023.03-1.

1.7.2. Hardware

Hardware Requirements for the AXI Streaming Intel FPGA IP for PCI Express are the following:

- Intel Agilex 7 FPGA with P-Tile and E-Tile (e.g., AGFB014R24B2E2V)
- Intel Agilex 7 FPGA with F-Tile and R-Tile (e.g., AGIB027R29A1E2VR2)
- Intel Agilex 7 FPGA with R-Tile (e.g., AGIB027R29A1E2VR3)
- Standards and Specifications Compliance

Table 2. AXI Streaming Intel FPGA IP for PCI Express Standards and Specifications Revision/Version

Standard	Revision/Version
PCI Express Base Specification	4, 5
Single Root I/O Virtualization and Sharing Specification	1.1
Address Translation Services	1.1
Virtual I/O Device (VIRTIO)	1
AMBA Stream Protocol Specification	AXI-4

2. Features

2.1. Supported Features

The AXI Streaming Intel FPGA IP for PCI Express provides you control over the PCIe HIP, by providing you with finer control over the PCIe Transaction Layer Packet (TLP), credit handling and various modes directly to the application layer. The IP sends TLPs received from the Link to the user logic side with some additional information like BAR number, function number, etc. The IP supports the following features:

PCIe Features:

- Complete protocol stack including the Transaction, Data Link, and Physical Layers implemented as Hard IP.
- Configurations supported:

	Gen3/Gen4/Gen5 1x16	Gen4 1x8	Gen3/Gen4/Gen5 2x8
Endpoint (EP)	Yes	Yes	Yes

Note: 1. Currently supported with Intel Agilex 7 devices with P-Tile (e.g., AGFB014R24B2E2V), Intel Agilex 7 devices with F-Tile (e.g., AGIB027R29A1E2VR2), and Intel Agilex 7 devices with R-Tile (e.g., AGIB027R29A1E2VR3).

2. Gen1/Gen2 configurations are supported via link down-training.

- Static port bifurcation: two x8s endpoints
- Clocking architecture:
  - Separate reference clock with independent spread spectrum clocking (SRIS)
  - Separate reference clock with no spread spectrum clocking (SRNS)
  - Common reference clock architecture
- Single Virtual Channel (VC)

- Capability Registers:
  - Message Signaled Interrupt (MSI)
  - Message Signal Interrupt Extended (MSI-X)
  - Advanced Error Reporting (AER) (PF only)
  - Power Management (PM – D0 and D3 PCIe power states) (PF only)
  - Alternative Routing ID (ARI)
  - Address Translation Services (ATS)
  - Page Request Service (PRS)
  - Transaction Processing Hints (TPH) ("No Steering Tag (ST)" mode only)
  - Access Control Services (ACS) (For ACS, only ports 0 and 1 are supported)
  - Latency Tolerance Reporting (LTR)
  - Process Address Space ID (PASID)
  - Vendor Specific Capability
- PCI Express Advanced Error Reporting (AER) (PF only)
- Supports up to 512-byte maximum payload size (MPS)
- Supports up to 4096-byte (4 KB) maximum read request size (MRRS)
- 32/64-bit BAR support (Prefetchable/Non-Prefetchable)
- Expansion ROM BAR support
- Number of tags – 32, 64, 128, 256, 512, 768 (Gen5 x16 only)
- Application error handling (UR/CA/Completion Timeout/Poison)
- MSI support - Supports multiple MSI, per-vector masking

Multifunction and Virtualization Features (Optional):

- SR-IOV support (Maximum 8PFs, 2048 VFs across all Endpoints in a design)
- Supports single TLP prefix per TLP
- Supports VIRTIO PCI* Configuration Registers
- Function Level Reset (FLR) – communicated to application through separate interface

User Interface Features:

- AXI4 (Streaming, Lite) user interface for data and control signals.
- AXI Streaming Interface: There are separate interfaces for header and data in both the Transmit and Receive directions.
  - The AXI Streaming Source Interface comprises the master signals, and provides the start of the transaction.
  - The AXI Streaming Source Interface is a single stream interface.
  - The AXI Streaming Sink Interface comprises the slave signals, and provides the response to the transaction from the source.
  - The AXI Streaming Sink Interface provides support for basic bare metal mode (e.g., Single physical function, AER, etc.) and virtualization mode (e.g., Multiple physical functions, function level reset, etc).
  - Selects the application's AXI streaming data bus width. The interface width is defined in terms of number of Bytes. This interface supports a scalable data bus width (32, 64, 128-byte widths) and frequency.
  - Supports the following data packing schemes:
    - HIP Native: Interface width, segment size of the AXI-ST interface are the same as those of the Native Hard IP.
    - Non-HIP Native/Compact: Interface width, segment size of the AXI-ST interface are different than those of the Native Hard IP.
  - Supports operating frequency selection options of 250, 350, 400, 470, 500 MHz. Refer to Device Family Support for the valid combinations of data bus width and frequencies.
- AXI Lite Responder Interface
  - This is the Control and Status Register Interface to access registers implemented in the IP modules, including PCI/PCIe Configuration Registers of all Functions.
  - 32-bit or 64-bit at 100-250 MHz.
- Configuration Extension Bus (CEB) Interface provided to extend the configuration capabilities beyond the PCI/PCIe capabilities and implement Customer Specific Capabilities.
- Configuration Intercept Interface (CII) allows the application logic to detect the occurrence of a Configuration (CFG) request on the link and to modify its behavior.

  Note: This interface is mutually exclusive with the Configuration Extension Bus (CEB) interface.

  Note: This interface is provided so that the IP is backward compatible with any legacy application logic that relies on CII for their functionality. Newly defined application logic should avoid using the CII interface and move to the CEB interface.
- Error reporting by the application logic: The IP implements Error Reporting registers. These registers allow user to indicate various errors. The IP then forwards this error information to the HIP block (UR/CA/Completion Timeout/ Poison).

- Supports Link Partner Credits (exposed via credit interface) - The IP exposes link partner credit to user in the Transmit and Receive directions depending on the tile used. The credits are advertised as a limit value specified in the PCIe spec. You must check the availability of credits for transmitting and receiving the TLP.
- Transaction ordering, deadlock avoidance
  - You must implement transaction ordering in the user application logic.
- Control Shadow Interface provided to shadow the control information from control / command registers (Optional).
- Completion timeout interface (Optional) - The PCIe IP can optionally track outgoing non-posted packets to report completion timeout information to the application.
- Supports Autonomous Hard IP mode - This mode allows the PCIe Hard IP to communicate with the Host before the FPGA configuration and entry into User mode are complete.

  Note: Unless Readiness Notifications mechanisms are used, the Root Complex or system software must allow at least 1.0 s after a Conventional Reset of a device before it may determine that a device that fails to return a Successful Completion status for a valid Configuration Request is a broken device. This period is independent of how quickly Link training completes.
- FPGA core configuration via Protocol (CvP Init and CvP Update) (Optional)

  Note: For Gen3, Gen4 and Gen5 x16 variants, Port 0 (corresponding to lanes 0 - 15) supports the CvP features. For Gen3, Gen4 and Gen5 x8 variants, only Port 0 (corresponding to lanes 0 - 7) supports the CvP features. Port 1 (corresponding to lanes 8 - 15) does not support CvP.
- Debug Toolkit for register accesses and debug (Optional).

  Note: This feature is not supported in the current Intel Quartus Prime release.
- Design example generation: Currently available when using P-Tile (Gen4 x16) and R-Tile (Gen5 x16).
- Software Driver support.
  - Available along with the Design Examples and Intel Open FPGA Stack reference design.

    Note: This feature is not supported in the current Intel Quartus Prime release.

Related Information

- P-Tile Avalon Streaming Intel FPGA IP for PCI Express User Guide
- F-Tile Avalon Streaming Intel FPGA IP for PCI Express User Guide
- R-Tile Avalon Streaming Intel FPGA IP for PCI Express User Guide
- AMBA AXI4-Stream Protocol Specification

2.2. Device Family Support

The AXI Streaming Intel FPGA IP for PCI Express currently supports only Intel Agilex 7 devices with P-Tile, F-Tile, and R-Tile.

The following table presents the resource utilization of the IP. These results come from the compilation of the design examples created through the IP Parameter Editor Pro for the devices AGFB014R24B2E2V (P-Tile), AGIB027R29A1E2VR2 (F-Tile), and AGIB027R29A1E2VR3 (R-Tile).

Table 3. AXI Streaming Intel FPGA IP for PCI Express Resources Utilization – (P-Tile)

IP Core Variation	Logic Utilization - (in ALMs)	Dedicated Logic Registers	M20K RAM Blocks
Gen4 1x16	12,506 / 487,200 (3%)	31,520	121/7110 (2%)
Gen4 2x8	18,411/487200 (4%)	49,138	198/7110 (3%)
Gen4 1x8	20,466/487,200 (4%)	63,766	95/7110 (1%)
Gen3 1x16	11,751/487200 (2%)	26,865	121/7110 (2%)
Gen3 2x8	17,555/487200 (4%)	38,963	198/7110 (3%)
Gen3 1x8	19,466/487,200 (4%)	58,654	95/7110 (1%)

Table 4. AXI Streaming Intel FPGA IP for PCI Express Resources Utilization – (F-Tile)

Variant	Logic Utilization (in ALMs)	Dedicated Logic Registers	M20K RAM Blocks
Gen4 1x16	16,755/912,800 (2%)	37,784	130/13,272 (<1%)
Gen4 2x8	22,720/912,800 (2%)	51,407	206/13,272 (2%)
Gen4 1x8	28,984/912,800 (3%)	47,304	105/13,272 (<1%)
Gen3 1x16	15,943/912,800 (2%)	33,223	130/13,272 (<1%)
Gen3 2x8	21,891/912,800 (2%)	45,375	206/13,272 (2%)
Gen3 1x8	26,954/912,800 (3%)	44,804	105/13,272 (<1%)

Table 5. AXI Streaming Intel FPGA IP for PCI Express Resources Utilization – (R-Tile)

Variant	Logic Utilization (in ALMs)	Dedicated Logic Registers	M20K RAM Blocks
Gen5 1x16	39,752/782,400 (5%)	65,227	222/10,464 (2%)
Gen5 2x8	61,977/782,400 (8%)	78,105	349/10,464 (3%)
Gen4 1x16	30,941/782,400 (4%)	40,172	175/10,464 (2%)
Gen4 2x8	52,697/782,400 (7%)	54,797	299/10,464 (3%)
Gen3 1x16	30,750/782,400 (4%)	38,136	175/10,464 (2%)
Gen3 2x8	52,588/782,400 (7%)	53,767	299/10,464 (3%)

Note: The above numbers are obtained with a design containing five physical functions and four virtual functions.

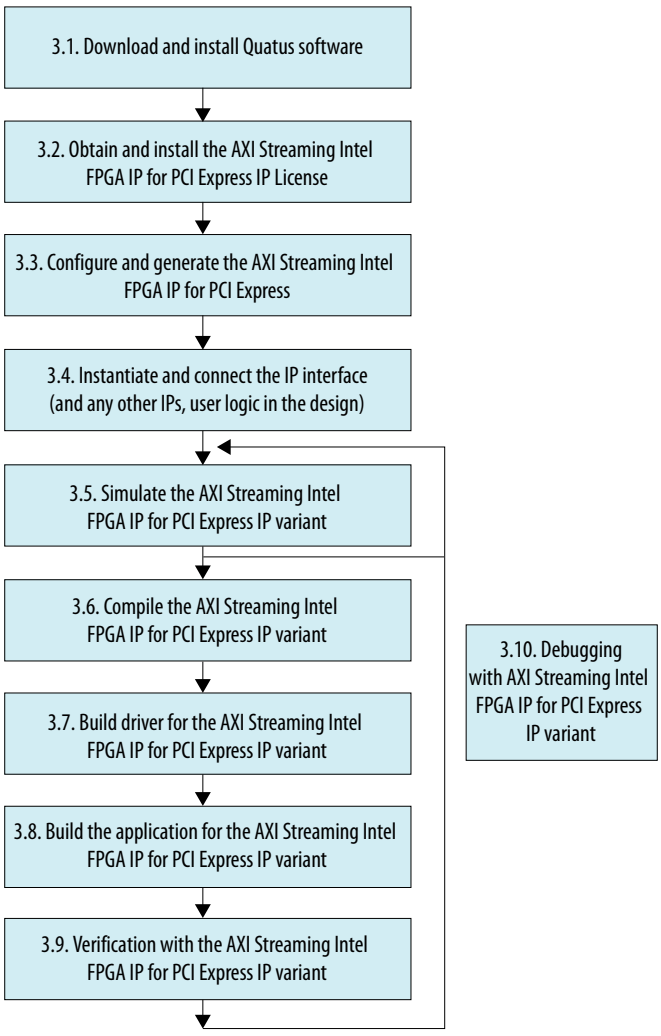
2.3. Performance

This information may be available in a future release.

3. Getting Started with the AXI Streaming Intel FPGA IP for PCI Express

This chapter provides the steps to get started with the AXI Streaming Intel FPGA IP for PCI Express, from installing the required software, to instantiating the IP, to simulating and compiling the IP(s) and verifying the functionality.

Figure 6. Flow Diagram for Getting Started with the AXI Streaming Intel FPGA IP for PCI Express



3.1. Download and Install Quartus Software

Refer to the section *Design Flow Requirements* in *Intel Quartus Prime Pro Edition User Guide* for details on downloading and installing the Quartus software and the necessary patches.

User Guides for both Intel Quartus Prime Pro Edition and Intel Quartus Prime Standard Edition can be found at Intel Quartus Prime User Guides.

3.2. Obtain and Install Intel FPGA IPs and Licenses

Obtain and install Intel FPGA IPs and licenses.

3.3. Configure and Generate the AXI Streaming Intel FPGA IP for PCI Express

You can generate the AXI Streaming Intel FPGA IP for PCI Express as per the following process:

Table 6. Generating the AXI Streaming Intel FPGA IP for PCI Express

AXI Streaming Intel FPGA IP for PCI Express in	Description
Standalone mode	Refer to the steps listed below to generate the IP in standalone mode.
Design example	Refer to the steps listed below to generate the IP as part of the design example.
Intel Open FPGA Stack (OFS) reference design	You can use the Intel pre-designed and verified system level shell designs (e.g., Open FPGA Stack (OFS)) with the IP and other Intel FPGA IPs like DMA, etc., and software stack to run example workloads. *Note:* For examples on connecting the various interfaces of the AXI Streaming Intel FPGA IP for PCI Express, you can access the following design repository: Open FPGA Stack Overview.

Note: OFS will not support the AXI Streaming Intel FPGA IP for PCI Express until the Intel Quartus Prime Pro Edition 24.1 version of this IP.

Following is the procedure to generate the AXI Streaming Intel FPGA IP for PCI Express and bring up a PCI Express link using Intel Quartus Prime Pro Edition software in standalone mode.

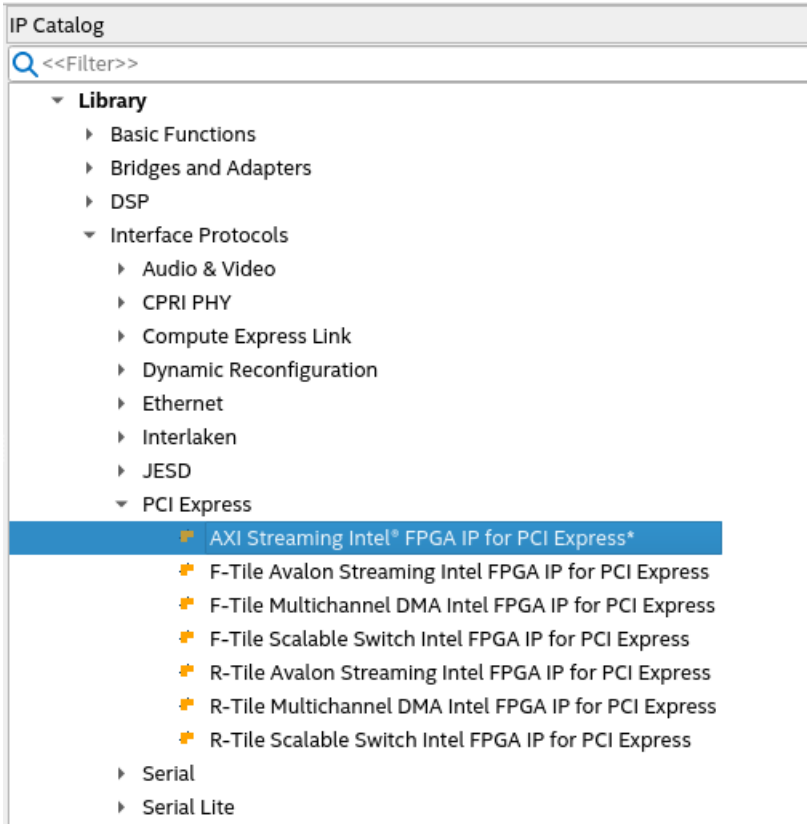
1. Use the Intel Quartus Prime Pro Edition software to create a Quartus Project and select the device. Currently, the AXI Streaming Intel FPGA IP for PCI Express is only supported on Intel Agilex 7 devices with P-Tile (e.g., AGFB014R24B2E2V), F-Tile (e.g. AGIB027R29A1E2VR2), and R-Tile (AGIB027R29A1E2VR3).
2. Use the following steps to specify IP core options and parameters:
   a. In the Intel Quartus Prime Pro Edition software, create a new project (**File ➤ New Project Wizard**).
   b. Specify the **Directory, Name**, and **Top-Level Entity.**
   c. For **Project Type**, accept the default value, **Empty project**. Click **Next**.
   d. For **Add Files** click **Next**.

e. For **Family, Device & Board Settings** under **Family**, select **Intel Agilex 7**.

f. Select the **Target Device** for your design.

g. Click **Finish**.

h. In the IP Catalog, locate and add the AXI Streaming Intel FPGA IP for PCI Express. This IP is available under the PCI Express IPs.



i. In the **New IP Variant** dialogue box, specify a name for your IP. The Parameter Editor saves the IP variation settings in a file named <your_ip>.ip

j. Click **Create**. The Parameter Editor appears.

k. Specify the parameters for your IP core variation. Refer to Chapter 5.1 Parameter Editor Parameters, for information about specific IP core parameters.

3. Generate the IP:

a. **Generation** dialogue box appears from the previous step. This allows you to generate the IP in the stand-alone mode.

b. Specify output file generation options, and then click **Generate**. The IP variation files are generated according to your specifications.

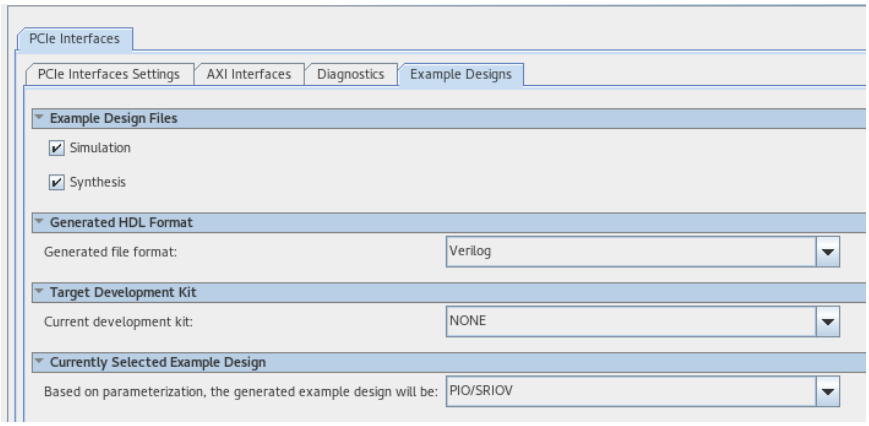
c. Click **Close**. The Parameter Editor adds the top-level.ip file to the current project automatically. If you are prompted to manually add the .ip file to the project, click **Project > Add/Remove Files** in Project to add the file.

4. To generate the design example, go to the **Example Designs** tab. Make the following selections:

a. For Example Design Files, turn on the **Simulation** and **Synthesis** options. If you do not need these simulation or synthesis files, leaving the corresponding option(s) turned off significantly reduces the design example generation time.

b. For General HDL Format, only Verilog is available in the current release.

c. For Target Development Kit, select the FPGA development kit corresponding to the device being used in the project.



5. Select **Generate Example Design** to create a design example that you can simulate and download to hardware. If you select one of the development boards, the device on that board overwrites the device previously selected in the Intel Quartus Prime project if the devices are different. When the prompt asks you to specify the directory for your design example, you can accept the default directory, `./<ip__example_design>`, or choose another directory.
6. Close the window when the design example generation is done. You may save your .ip file when prompted, but it is not required to be able to use the design example.
7. Open the design example project by navigating to `<project_dir>/<ip_example_design>/` and opening the file `pcie_ed.qpf`.
8. Compile the design example project to generate the .sof file for the complete design example. This file is what you download to a board to perform hardware verification. For details on how to compile the design, refer to Compiling the Design Example.
9. Close your design example project. Note that you cannot change the PCIe pin allocations in the Intel Quartus Prime project. However, to ease PCB routing, you can take advantage of the lane reversal and polarity inversion features supported by this IP.

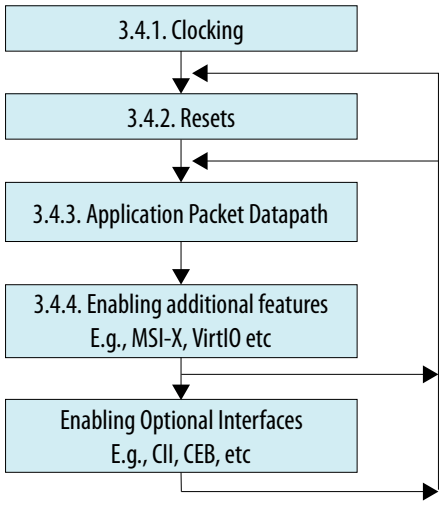
Note: For Design Examples in the current release of Intel Quartus Prime Pro Edition:

- Only simulation support is available.
- Only Gen5 x16 (R-Tile) and Gen4 x16 (P-Tile) are available.

3.4. Instantiate and Connect the AXI Streaming Intel FPGA IP for PCI Express Interfaces

You can use the Intel Quartus Prime Platform Designer and IPs in the IP catalog, and/or use RTL to design to add any additional IPs, user logic required in the design and connect the IPs and user logic. You can also make appropriate pin assignments to connect ports and set any appropriate per-instance RTL parameters. For Quartus generated design examples, the required IPs, clocking, reset and application logic to run basic traffic are already provided by the generated design.

Figure 7. Flow Diagram for Instantiating and Connecting the Interfaces in the AXI Streaming Intel FPGA IP for PCI Express



3.4.1. Clocking and Resets

For details on clocking architecture, guidelines, as well as clock and reset interfaces and signals, refer to the Clocks and Resets section.

For details on parameters available in the IP parameter editor, refer to the Parameter Editor Parameters section.

3.4.2. Application Packet Datapath

For details on the application Datapath architecture and guidelines, refer to the AXI-Streaming Interface section.

For details on parameters available in the IP parameter editor, refer to the Parameter Editor Parameters section.

For details on the interfaces and signals available in the IP, refer to the AXI-Streaming Interface section.

3.4.3. Enabling Additional Features

For details on the architecture and guidelines for additional features like SR-IOV, FLR etc., refer to the Interfaces and Signals chapter.

For details on parameters available in the IP parameter editor, refer to the Parameter Editor Parameters section.

For details on the interfaces and signals available in the IP, refer to the Interfaces and Signals chapter.

3.4.4. Enabling Optional Interfaces

For details on the architecture and guidelines for optional interfaces like CII, Debug Toolkit and Reconfiguration interface, etc, refer to the Interfaces and Signals chapter.

For details on parameters available in the IP parameter editor, refer to the Parameter Editor Parameters section.

For details on the interfaces and signals available in the IP, refer to the Interfaces and Signals chapter.

3.5. Simulate the AXI Streaming Intel FPGA IP for PCI Express IP Variant

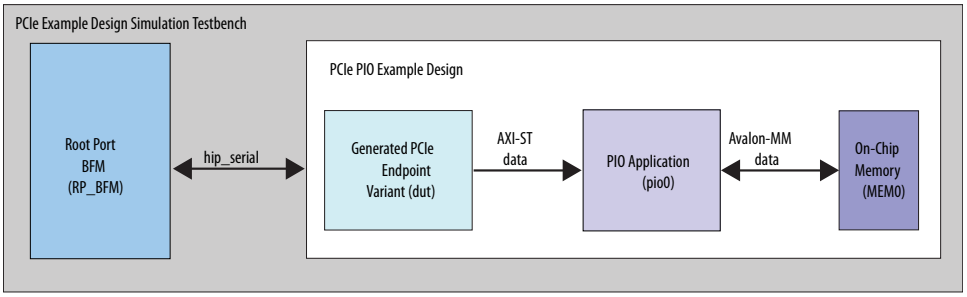
Table 7. Simulating the AXI Streaming Intel FPGA IP for PCI Express

AXI Streaming Intel FPGA IP for PCI Express in	Description
Standalone mode	You must create your own simulation environment to simulate and verify the functionality of the IP and design, based on the application requirement. You can also use third-party Bus Functional Models (BFMs), and Verification IPs (VIPs) to verify the IP in simulation. The IP simulation files can be found in the /ip/sim/<simulator> folder in the Intel Quartus Prime project directory.
Design example	The simulation testbench (Intel BFM) for the IP is generated as part of the design example generation. Refer to the steps below for simulating the design example using the Intel BFM.
Intel Open FPGA Stack (OFS) reference design	Simulation environment for the IP and example workloads are provided as part of the reference design. *Note:* For examples on simulating the IP with the Intel OFS reference design, contact your Intel Sales Representative for access to the Intel OFS design repository.

3.5.1. Simulating the Design Example

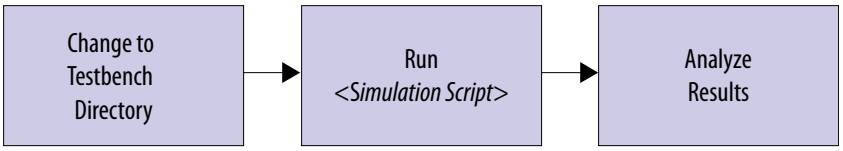
The simulation setup involves the use of a Root Port Bus Functional Model (BFM) to exercise the AXI Streaming Intel FPGA IP for PCI Express (DUT) as shown in the following figure.

Figure 8. PIO Design Example Simulation Testbench



The following flow diagram shows the steps to simulate the design example:

Figure 9. Procedure



1. Change to the testbench simulation directory, `<project_directory>/pcie_ss_ed_tb/pcie_ss_ed_tb/sim/<EDA_vendor>/simulator`.
2. Run the simulation script for the simulator of your choice. Refer to the table below.
3. Analyze the results.

Note: **The AXI Streaming Intel FPGA IP for PCI Express does not support parallel PIPE simulations.**

The following figure shows the link status information for a Gen5 x16 Endpoint simulation:

Figure 10. Link Status Information for a Gen5 x16 Endpoint Simulation

```
32974  INFO:          405261 ns   EP PCI Express Link Status Register (1105):
32975  INFO:          405261 ns     Negotiated Link Width: x16
32976  INFO:          405261 ns          Slot Clock Config: System Reference Clock Used
32977  INFO:          405261 ns             New Link Speed: 32.0GT/s
32978  INFO:          405261 ns
32979  INFO:          405845 ns   EP PCI Express Link Control Register (0000):
32980  INFO:          405845 ns       Common Clock Config: Local Clock Used
32981  INFO:          405845 ns
32982  INFO:          407365 ns
32983  INFO:          407365 ns   EP PCI Express Capabilities Register (0002):
32984  INFO:          407365 ns         Capability Version: 2
32985  INFO:          407365 ns                  Port Type: Native Endpoint
32986  INFO:          407365 ns
32987  INFO:          407365 ns   EP PCI Express Device Capabilities Register (00008022):
32988  INFO:          407365 ns     Max Payload Supported: 512 Bytes
32989  INFO:          407365 ns              Extended Tag: Supported
32990  INFO:          407365 ns    Acceptable L0s Latency: Less Than 64 ns
32991  INFO:          407365 ns    Acceptable L1  Latency: Less Than 1 us
32992  INFO:          407365 ns          Attention Button: Not Present
32993  INFO:          407365 ns       Attention Indicator: Not Present
32994  INFO:          407365 ns            Power Indicator: Not Present
32995  INFO:          407365 ns
32996  INFO:          407365 ns   EP PCI Express Link Capabilities Register (01400105):
32997  INFO:          407365 ns         Maximum Link Width: x16
32998  INFO:          407365 ns      Supported Link Speed: 32.0GT/s or 16.0GT/s or 8.0GT/s or 5.0GT/s or 2.5GT/s
32999  INFO:          407365 ns                  L0s Entry: Not Supported
33000  INFO:          407365 ns                  L1  Entry: Not Supported
33001  INFO:          407365 ns          L0s Exit Latency: Less Than 64 ns
33002  INFO:          407365 ns          L1  Exit Latency: Less Than 1 us
33003  INFO:          407365 ns                Port Number: 01
33004  INFO:          407365 ns   Surprise Dwn Err Report: Not Supported
33005  INFO:          407365 ns    DLL Link Active Report: Not Supported
33006  INFO:          407365 ns
33007  INFO:          407365 ns   EP PCI Express Device Capabilities 2 Register (30731B9F):
33008  INFO:          407365 ns   Completion Timeout Rnge: ABCD (50us to 64s)
```

After a successful simulation, the `simulation.log` file contains a "successful completion" message.

This testbench simulates up to a Gen5 x16 variant.

3.5.1.1. Steps to Run Simulations

The following sections describe the steps to run simulations on the AXI Streaming Intel FPGA IP for PCI Express using different simulators.

3.5.1.1.1. VCS* Simulator

To run simulations using the VCS* simulator, follow these steps:

1. Go to the working directory <*example_design*>`/pcie_ss_ed_tb/pcie_ss_ed_tb/sim/synopsys/vcs`.
2. Type `sh vcs_setup.sh USER_DEFINED_COMPILE_OPTIONS="" USER_DEFINED_ELAB_OPTIONS="-xlrm\ uniq_prior_final\ -debug_access+all" USER_DEFINED_SIM_OPTIONS="" TOP_LEVEL_NAME="pcie_ss_ed_tb" | tee simulation.log`

   Note: The command above is a single-line command.
3. A successful simulation ends with the following message, "Simulation stopped due to successful completion!".

Note: To run a simulation in interactive mode, use the following steps: (if you already generated a `simv` executable in non-interactive mode, delete the `simv` and `simv.diadir`)

a. Open the `vcs_setup.sh` file and add a debug option to the VCS command: `vcs -debug_access+all`.

b. Compile the design example: `sh vcs_setup.sh USER_DEFINED_ELAB_OPTIONS="-xlrm\ uniq_prior_final" SKIP_SIM=1 TOP_LEVEL_NAME="pcie_ss_ed_tb"`.

c. Start the simulation in interactive mode: `simv -gui &`.

```
33038  INFO:          412557 ns   PCI MSI-X Capability Register:
33039  INFO:          412557 ns               Table Size: 0003
33040  INFO:          413141 ns                Table BIR: 0
33041  INFO:          413141 ns             Table Offset: 00000000
33042  INFO:          413725 ns                 PBA BIR: 0
33043  INFO:          413725 ns               PBA Offset: 00000000
33044  INFO:          413725 ns
33045  INFO:          414949 ns   EP PCI Express AER Capability Register:
33046  INFO:          416477 ns        ECRC Check Capable: Supported
33047  INFO:          416477 ns   ECRC Generation Capable: Supported
33048  INFO:          416477 ns
33049  INFO:          422397 ns
33050  INFO:          422397 ns BAR Address Assignments:
33051  INFO:          422397 ns BAR    Size       Assigned Address  Type
33052  INFO:          422397 ns ---    ----       ----------------
33053  INFO:          422397 ns BAR1:0  64 KBytes 00000001 00000000 Prefetchable
33054  INFO:          422397 ns BAR2   Disabled
33055  INFO:          422397 ns BAR3   Disabled
33056  INFO:          422397 ns BAR4   Disabled
33057  INFO:          422397 ns BAR5   Disabled
33058  INFO:          422397 ns ExpROM Disabled
33059  INFO:          425245 ns
33060  INFO:          425245 ns Completed configuration of Endpoint BARs.
33061  INFO:          425885 ns Issue Mwr/MRd to PIO ED.
33062  INFO:          425885 ns ---------
33063  INFO:          426485 ns Passed: 0004 same bytes in BFM mem addr 0x00000040 and 0x00000840
33064  INFO:          427085 ns Passed: 0004 same bytes in BFM mem addr 0x00000040 and 0x00000840
33065  INFO:          427685 ns Passed: 0004 same bytes in BFM mem addr 0x00000040 and 0x00000840
33066  INFO:          428285 ns Passed: 0004 same bytes in BFM mem addr 0x00000040 and 0x00000840
33067  INFO:          428885 ns Passed: 0004 same bytes in BFM mem addr 0x00000040 and 0x00000840
33068  INFO:          429485 ns Passed: 0004 same bytes in BFM mem addr 0x00000040 and 0x00000840
33069  INFO:          430085 ns Passed: 0004 same bytes in BFM mem addr 0x00000040 and 0x00000840
33070  INFO:          430685 ns Passed: 0004 same bytes in BFM mem addr 0x00000040 and 0x00000840
33071  INFO:          431285 ns Passed: 0004 same bytes in BFM mem addr 0x00000040 and 0x00000840
33072  INFO:          431885 ns Passed: 0004 same bytes in BFM mem addr 0x00000040 and 0x00000840
33073  INFO:          431885 ns PIO ED MWr/Mrd Completed.
33074  SUCCESS: Simulation stopped due to successful completion!
33075  Simulation passed
```

Note: For details on the testbench, refer to Simulating the Design Example.

3.6. Compile the AXI Streaming Intel FPGA IP for PCI Express IP Variant

You must complete the following steps to compile the AXI Streaming Intel FPGA IP for PCI Express IP Variant.

Table 8. Compiling the AXI Streaming Intel FPGA IP for PCI Express

AXI Streaming Intel FPGA IP for PCI Express in	Description
Standalone mode	Use the **Quartus Prime Pro software -> Processing** menu to select **Start Compilation**. Timing can be verified using the **TimeQuest Timing Analyzer** of the Quartus Prime Pro. Use the assembler to generate the configuration bit stream as a .sof (or .pof) file. This file is what you download to a board to perform hardware verification.
Design example	1. Navigate to `<project_dir>/<AXI Streaming IP path>/` and open the file `pcie_ed.qpf`. 2. If you select either of the two following development kits, the VID-related settings are included in the .qsf file of the generated design example, and you are not required to add them manually. Note that these settings are board-specific. a. Intel Agilex 7 F-Series FPGA development kit b. Intel Agilex 7 I-Series FPGA development kit 3. On the **Processing** menu, select **Start Compilation**.
Intel Open FPGA Stack (OFS) reference design	Pre-compiled OFS design with IP and example workloads are provided as part of the reference design. *Note:* Contact your Intel Sales Representative for access to the Intel OFS design repository.

Note: OFS will not support the AXI Streaming Intel FPGA IP for PCI Express until the Intel Quartus Prime Pro Edition 24.1 version of this IP.

Note: Hardware support is not available for the standalone IP and design examples in the current release of Intel Quartus Prime Pro Edition.

Download the bit stream resulting from the compilation onto the device and bring up the PCIe link(s) in the design. Ensure that your device is linked up and enumerated in the PCI Express topology. You can use utilities like lspci, setpci to obtain general information of the device like link speed, link width, etc.

For example, to read the negotiated link speed for the given device in a system, you can use the following commands:

```
sudo lspci –s $bdf -vvv
```

`-s` refers to "slot" and is used with the bus/device/function number (bdf) information. Use this command if you know the bdf of the device in the system topology.

```
sudo lspci –d :$did -vvv
```

`-d` refers to device and is used with the device ID as configured in the parameter settings of the PCIe IP (vid:did). Use this command to search using the device ID.

Use the steps below to compile the IP when using the Quartus generated design.

3.7. Software Drivers for AXI Streaming Intel FPGA IP for PCI Express IP Variant

The software drivers for the AXI Streaming Intel FPGA IP for PCI Express vary when the IP is configured in different modes (e.g., Root Port vs Endpoint mode). You must ensure the correct software drivers are used in each of these modes for proper functionality of the IP. You can download, customize, build, and install the software drivers for the IP as mentioned in the following table.

Table 9. Software Drivers for AXI Streaming Intel FPGA IP for PCI Express

AXI Streaming Intel FPGA IP for PCI Express in	Description
Standalone mode	You must create your own software drivers based on the application requirement.
Design example	The software drivers for the IP are generated as part of the design example generation. Refer to the following section for instructions on how to install and run the software drivers for the Quartus generated design example.
Intel FPGA IP OFS reference design	Software drivers for the IP and example workloads are provided as part of the reference design. *Note:* For examples on software drivers of the AXI Streaming Intel FPGA IP for PCI Express, please contact your Intel Sales Representative for access to the Intel OFS design repository.

Note: OFS will not support the AXI Streaming Intel FPGA IP for PCI Express until the Intel Quartus Prime Pro Edition 24.1 version of this IP.

Note: Hardware support is not available for the standalone IP and design examples in the current release of Intel Quartus Prime Pro Edition.

3.7.1. Installing the Linux Kernel Driver

Before you can test the design example in hardware, you must configure the FPGA and then restart your computer. After that, install the Linux kernel driver. You can use this driver to perform the following tests:

- A PCIe link test that performs 10 writes and reads
- Memory space DWORD(1) reads and writes
- Configuration Space DWORD reads and writes

In addition, you can use the driver to change the value of the following parameters:

- The BAR being used
- The selected device (by specifying the bus, device and function (BDF) numbers for the device)

Complete the following steps to install the kernel driver:

1. Navigate to `./software/kernel/linux` under the example design generation directory.
2. Change the permissions on the `install`, `load`, and `unload` files:

   ```
   $ chmod 777 install load unload
   ```

3. Install the driver:

   ```
   $ sudo ./install
   ```

4. Verify the driver installation:

   ```
   $ lsmod | grep intel_fpga_pcie_drv
   ```

   Expected result:

(1) Throughout this user guide, the terms word, DWORD and QWORD have the same meaning that they have in the PCI Express Base Specification. A word is 16 bits, a DWORD is 32 bits, and a QWORD is 64 bits.

```
intel_fpga_pcie_drv 17792 0
```

5. Verify that Linux recognizes the PCIe design example:

   ```
   $ lspci -d 1172:000 -v | grep intel_fpga_pcie_drv
   ```

   Note: If you have changed the Vendor ID, substitute the new Vendor ID for Intel's Vendor ID in this command.

   Expected result:

   ```
   Kernel driver in use: intel_fpga_pcie_drv
   ```

3.8. Build the Application for the AXI Streaming Intel FPGA IP for PCI Express IP Variant

After configuring the software drivers for the AXI Streaming Intel FPGA IP for PCI Express IP variant, you must configure the PCIe link to run applications for traffic tests, Read/Write transactions, measure performance etc.

Table 10. Application for the AXI Streaming Intel FPGA IP for PCI Express

AXI Streaming Intel FPGA IP for PCI Express in	Description
Standalone mode	You must create your own software application based on the application requirement.
Design example	Example application for the IP is generated as part of the design example generation. Refer to the following section for instructions on how to run the application for the Quartus generated design examples.
Intel OFS reference design	Example application for the IP and example workloads are provided as part of the reference design. *Note:* For examples on application and workloads of the AXI Streaming Intel FPGA IP for PCI Express, please contact your Intel Sales Representative for access to the Intel OFS design repository.

Note: OFS will not support the AXI Streaming Intel FPGA IP for PCI Express until the Intel Quartus Prime Pro Edition 24.1 version of this IP.

Note: Hardware support is not available for the standalone IP and design examples in the current release of Intel Quartus Prime Pro Edition.

3.8.1. Running the Design Example

Here are the test operations you can perform on the AXI Streaming Intel FPGA IP for PCI Express design examples:

Table 11. Test Operations Supported by the AXI Streaming Intel FPGA IP for PCI Express

Operations	Required BAR	Supported by the AXI Streaming Intel FPGA IP for PCI Express
		PIO
0: Link test - 10 writes and reads	0	Yes
1: Write memory space	0	Yes
2: Read memory space	0	Yes

3.8.1.1. Running the PIO Design Example

1. Navigate to `./software/user/example` under the design example directory.
2. Compile the design example application:

   ```
   $ make
   ```

3. Run the test:

   ```
   $ sudo ./intel_fpga_pcie_link_test
   ```

   You can run the Intel FPGA IP PCIe link test in manual or automatic mode. Choose from:

   - In automatic mode, the application automatically selects the device. The test selects the Intel PCIe device with the lowest BDF by matching the Vendor ID. The test also selects the lowest available BAR.
   - In manual mode, the test queries you for the bus, device, and function number and BAR.

   For the Intel Agilex 7 Development Kit, you can determine the BDF by typing the following command:

   ```
   $ lspci -d 1172:
   ```

4. Here are sample transcripts for automatic and manual modes:

Automatic mode:

```
*******************************************************************
Intel FPGA PCIe Link Test
Version 2.0
0: Automatically select a device
1: Manually select a device
*******************************************************************
> 0
Opened a handle to BAR 0 of a device with BDF 0x8200

*******************************************************************
 0: Link test - 100 writes and reads
 1: Write memory space
 2: Read memory space
 3: Write configuration space
 4: Read configuration space
 5: Change BAR for PIO
 6: Change device
 7: Enable SRIOV
 8: Do a link test for every enabled virtual function
    belonging to the current device
 9: Perform DMA for Throughput
10: Quit program
*******************************************************************
> 0
Doing 100 writes and 100 reads..
Number of write errors:       0
Number of read errors:        0
Number of dword mismatches:   0
```

Manual mode:

```
*********************************************************
Intel FPGA PCIe Link Test
Version 2.0
0: Automatically select a device
1: Manually select a device
*********************************************************
> 1
Enter bus number, in hex:
> 4b
Enter device number, in hex:
> 0
Enter function number, in hex:
> 0
BDF is 0x4b00
B:D.F, in hex, is 4b:0.0
Enter BAR number (-1 for none):
> 0
Opened a handle to BAR 0 of a device with BDF 0x4b00
```

3.9. Verification with the AXI Streaming Intel FPGA IP for PCI Express IP Variant

After simulation, compilation, configuring software drivers, and building application for the AXI Streaming Intel FPGA IP for PCI Express IP Variant, you must verify the IP.

Table 12. Verification with the AXI Streaming Intel FPGA IP for PCI Express

AXI Streaming Intel FPGA IP for PCI Express in	Description
Standalone mode	You must create your own verification test suite based on the application requirement.
Design example	Example testbench for the IP are generated as part of the design example generation. Refer to Simulate the AXI Streaming Intel FPGA IP for PCI Express for details on the verification testbench provided as part of the Quartus generated design examples.
Intel OFS reference design	Example verification suite for the IP and example workloads are provided as part of the reference design. *Note:* For examples on verification suite of the AXI Streaming Intel FPGA IP for PCI Express, contact your Intel Sales Representative for access to the Intel OFS design repository.

Note: OFS will not support the AXI Streaming Intel FPGA IP for PCI Express until the Intel Quartus Prime Pro Edition 24.1 version of this IP.

3.10. Debugging with the AXI Streaming Intel FPGA IP for PCI Express IP Variant

Typically, PCI Express link-up involves the following steps:

1. Link training
2. BIOS enumeration and data transfer

This section describes the flow to debug link issues during the hardware bring-up. Intel recommends a systematic approach to diagnosing issues as illustrated in the following figure.

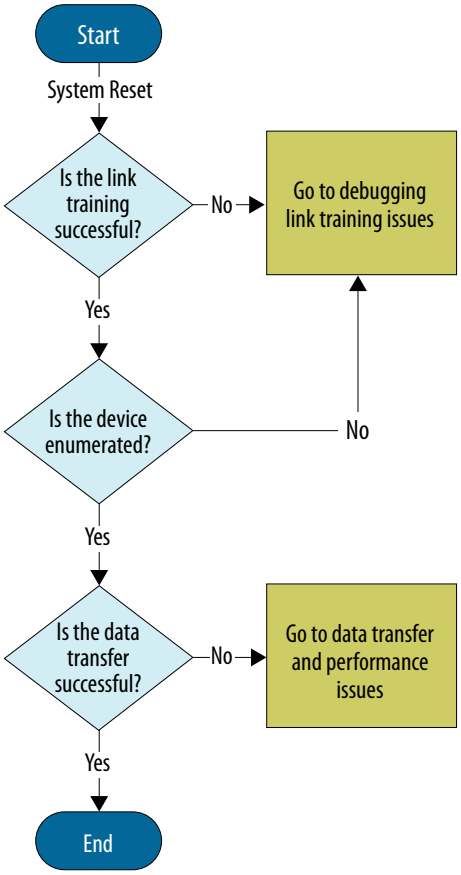
You can use the following Quartus tools to identify the issues:

- Intel Quartus Signal Tap II Analyzer
- In-System Sources and Probes (ISSP) tools

Additionally, you can use the IP Debug Toolkit for debugging the PCIe links when using the AXI Streaming Intel FPGA IP for PCI Express. The Debug Toolkit includes the following features:

- Protocol and link status information
- Basic and advanced debugging capabilities including register read access and Eye viewing capability
- System Console based interface to access status registers of the AXI Streaming Intel FPGA IP for PCI Express IP using scripts

Figure 11. PCI Express Debug Flow Chart



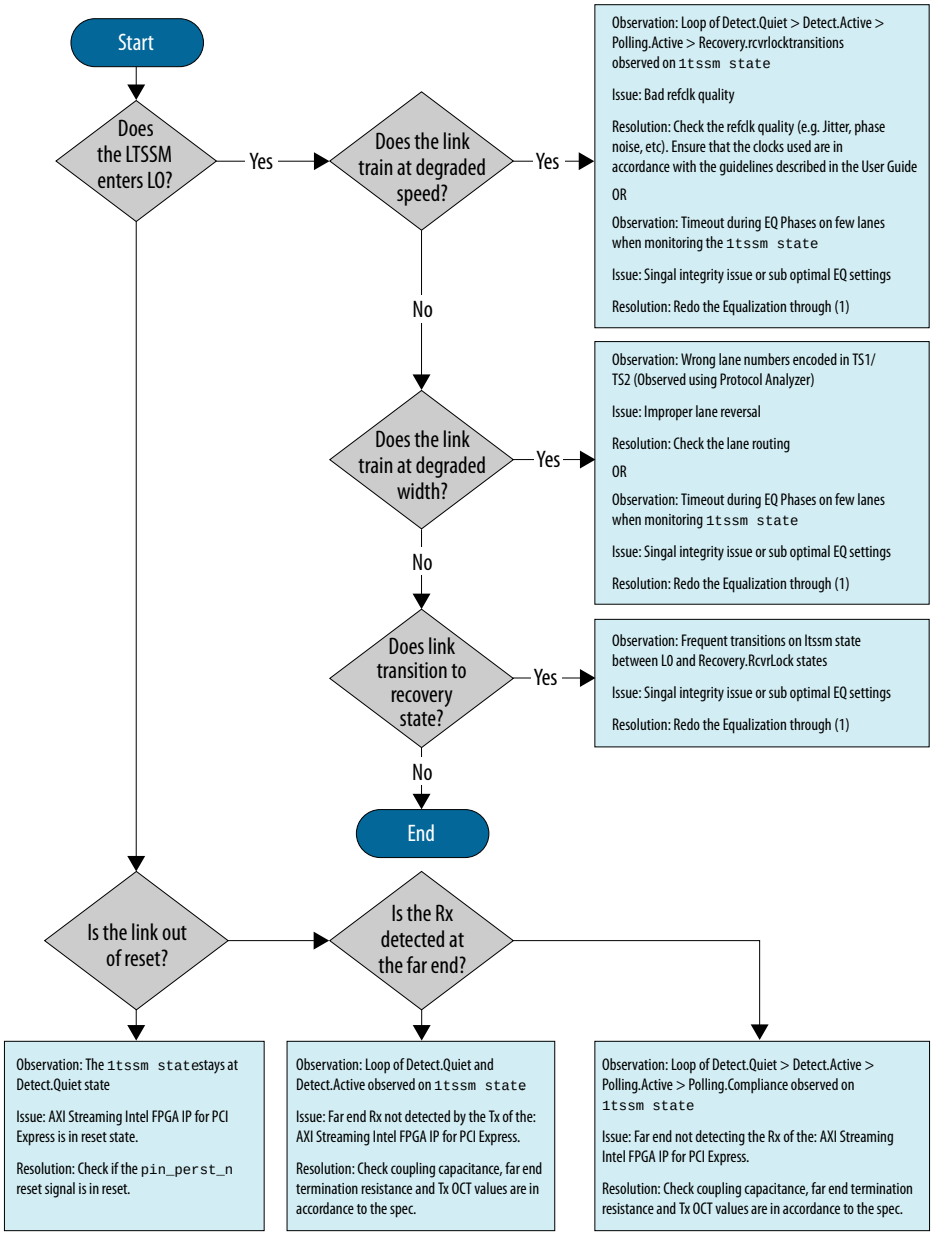
3.10.1. Debugging Link Training Issues

The Physical Layer automatically performs link training and initialization without software intervention. This is a well-defined process to configure and initialize the device's physical layer and link so that PCIe packets can be transmitted. Some examples of link training issues include:

- Link fails to negotiate to expected link speed
- Link fails to negotiate to the expected link width
- LTSSM fails to reach/stay stable at L0

The following flow chart identifies the potential cause of the issue seen during link training when using the AXI Streaming Intel FPGA IP for PCI Express.

Figure 12. Link Training Debugging Flow



Note: Redo the equalization using the Link Equalization Request 8.0 GT/s bit of the Link Status 2 register for 8.0 GT/s, or the Link Equalization Request 16.0 GT/s bit of the 16.0 GT/s Status Register for 16.0 GT/s, or the Link Equalization Request 32.0 GT/s bit of the 32.0 GT/s Status Register for 32 GT/s.

You may use the following debug tools for debugging link training issues observed on the PCI Express link when using the AXI Streaming Intel FPGA IP for PCI Express.

3.10.1.1. Operating System Tools and Utilities

You can use Linux* utilities like lspci, setpci to obtain general information of the device like link speed, link width, etc.

The following commands reads the negotiated link speed for the AXI Streaming Intel FPGA IP for PCI Express in a system.

```
sudo lspci –s $bdf -vvv
```

`-s` refers to "slot" and is used with the bus/device/function number (bdf) information. Use this command if you know the bdf of the device in the system topology

```
sudo lspci –d :$did -vvv
```

`-d` refers to device and is used with the device ID as configured in the parameter settings of the PCIe IP (vid:did). Use this command to search using the device ID.

Figure 13. lspci Output

```
lspci -vvv -s 0a:00.0
0a:00.0 Non-VGA unclassified device: Altera Corporation Device 0000 (rev 01)
        Control: I/O- Mem+ BusMaster+ SpecCycle- MemWINV- VGASnoop- ParErr- Stepping- SERR- FastB2B- DisINTx-
        Status: Cap+ 66MHz- UDF- FastB2B- ParErr- DEVSEL=fast >TAbort- <TAbort- <MAbort- >SERR- <PERR- INTx-
        Latency: 0, Cache Line Size: 64 bytes
        Interrupt: pin A routed to IRQ 72
        Region 0: Memory at f0000000 (64-bit, prefetchable) [size=64K]
        Capabilities: [40] Power Management version 3
                Flags: PMEClk- DSI- D1- D2- AuxCurrent=0mA PME(D0-,D1-,D2-,D3hot-,D3cold-)
                Status: D0 NoSoftRst- PME-Enable- DSel=0 DScale=0 PME-
        Capabilities: [70] Express (v2) Endpoint, MSI 00
                DevCap: MaxPayload 512 bytes, PhantFunc 0, Latency L0s <64ns, L1 <1us
                        ExtTag+ AttnBtn- AttnInd- PwrInd- RBE+ FLReset- SlotPowerLimit 0.000W
                DevCtl: Report errors: Correctable+ Non-Fatal+ Fatal+ Unsupported-
                        RlxdOrd+ ExtTag+ PhantFunc- AuxPwr- NoSnoop+
                        MaxPayload 512 bytes, MaxReadReq 512 bytes
                DevSta: CorrErr+ UncorrErr- FatalErr- UnsuppReq+ AuxPwr- TransPend-
                LnkCap: Port #1, Speed 16GT/s, Width x16, ASPM not supported, Exit Latency L0s <64ns, L1 <1us
                        ClockPM- Surprise- LLActRep- BwNot- ASPMOptComp+
                LnkCtl: ASPM Disabled; RCB 64 bytes Disabled- CommClk+
                        ExtSynch- ClockPM- AutWidDis- BWInt- AutBWInt-
                LnkSta: Speed 16GT/s, Width x16, TrErr- Train- SlotClk+ DLActive- BWMgmt- ABWMgmt-
                DevCap2: Completion Timeout: Range ABCD, TimeoutDis+, LTR+, OBFF Not Supported
                DevCtl2: Completion Timeout: 50us to 50ms, TimeoutDis-, LTR-, OBFF Disabled
                LnkCtl2: Target Link Speed: 16GT/s, EnterCompliance- SpeedDis-
                         Transmit Margin: Normal Operating Range, EnterModifiedCompliance- ComplianceSOS-
                         Compliance De-emphasis: -6dB
                LnkSta2: Current De-emphasis Level: -3.5dB, EqualizationComplete+, EqualizationPhase1+
                         EqualizationPhase2+, EqualizationPhase3+, LinkEqualizationRequest-
        Capabilities: [100 v2] Advanced Error Reporting
                UESta:  DLP- SDES- TLP- FCP- CmpltTO- CmpltAbrt- UnxCmplt- RxOF- MalfTLP- ECRC- UnsupReq- ACSViol-
                UEMsk:  DLP- SDES- TLP- FCP- CmpltTO- CmpltAbrt- UnxCmplt- RxOF- MalfTLP- ECRC- UnsupReq- ACSViol-
                UESvrt: DLP+ SDES+ TLP- FCP+ CmpltTO- CmpltAbrt- UnxCmplt- RxOF+ MalfTLP+ ECRC- UnsupReq- ACSViol-
                CESta:  RxErr- BadTLP- BadDLLP- Rollover- Timeout- NonFatalErr+
                CEMsk:  RxErr- BadTLP- BadDLLP- Rollover- Timeout- NonFatalErr+
                AERCap: First Error Pointer: 00, GenCap+ CGenEn- ChkCap+ ChkEn-
        Capabilities: [148 v1] Virtual Channel
                Caps:   LPEVC=0 RefClk=100ns PATEntryBits=1
                Arb:    Fixed- WRR32- WRR64- WRR128-
                Ctrl:   ArbSelect=Fixed
                Status: InProgress-
                VC0:    Caps:   PATOffset=00 MaxTimeSlots=1 RejSnoopTrans-
                        Arb:    Fixed- WRR32- WRR64- WRR128- TWRR128- WRR256-
```

The **LnkCap** under Capabilities indicates the advertised link speed and width capabilities of the device. The **LnkSta** under Capabilities indicates the negotiated link speed and width of the device.

You can rescan the PCIe bus using the following commands. You must have root privileges to perform the below commands:

```
#To detach the device from the tree:
% echo 1 > /sys/bus/pci/devices/0a:00.0/remove
```

```
#To rescan the bus:
% echo 1 > /sys/bus/pci/rescan
```

3.10.1.2. Signal Tap Logic Analyzer

Using the Signal Tap Logic Analyzer, you can monitor the following top-level signals from the AXI Streaming Intel FPGA IP for PCI Express to confirm the failure symptom for any port type (Root Port, Endpoint) and configuration (Gen5/Gen4/Gen3).

Table 13. Top-Level Signals to be Monitored for Debugging

Signal	Description	Expected value for successful link up
p<n>_pin_perst_n where n = 0, 1	Active-low asynchronous output signal from the PCIe IP. It is derived from the pin_perst_n input signal.	1'b1
ninit_done	Active-low asynchronous output signal from the Reset Release Intel FPGA IP. High indicates that the FPGA device is not yet fully configured, and low indicates the device has been configured and is in normal operating mode. For more details on the Reset Release Intel FPGA IP.	1'b0
p<n>_reset_status_n where n = 0, 1	Active-low output signal from the IP, synchronous to coreclkout_hip_toapp of the IP. The reset_status_n output of HIP drives this signal. Held low until pin_perst_n is deasserted and the PCIe Hard IP comes out of reset. When port bifurcation is used, there is one such signal for each port. The application logic can use this signal to drive its reset network.	1'b1
p<n>_ss_app_linkup where n = 0,1	Indicates the Physical link layer is up. Synchronous to coreclkout_hip_toapp clock of the IP.	1'b1
p<n>_ss_app_dlup where n = 0, 1	Indicates the data link layer is up. Synchronous to coreclkout_hip_toapp clock of the IP.	1'b1
p<n>_ss_app_serr where n = 0, 1	Indicates system error is detected. Synchronous to coreclkout_hip_toapp clock of the IP. EP mode: Asserted when the P-Tile PCIe Hard IP sends a message of correctable/ non-fatal/fatal error.	1'b0
link_up_o	Active-high output signal from the PCIe Hard IP, synchronous to coreclkout_hip clock of the HardIP. Indicates that the Physical Layer link is up. (This signal is currently available at the Hard IP interface).	1'b1
dl_up_o	Active-high output signal from the PCIe Hard IP, synchronous to coreclkout_hip of the Hard IP. Indicates that the Data Link Layer is active.	1'b1
ltssm_state_o[5:0] (P-Tile) p<n>_ss_app_ltssmstate[5:0] where n=0,1 (F/R-Tile)	Indicates the LTSSM state, synchronous to coreclkout_hip of the Hard IP. (This signal is currently available at the Hard IP interface) • 6'h00: S_DETECT_QUIET • 6'h01: S_DETECT_ACT • 6'h02: S_POLL_ACTIVE	6'b11 (L0)

continued...

Signal	Description	Expected value for successful link up
	• 6'h03: S_POLL_COMPLIANCE • 6'h04: S_POLL_CONFIG • 6'h05: S_PRE_DETECT_QUIET • 6'h06: S_DETECT_WAIT • 6'h07: S_CFG_LINKWD_START • 6'h08: S_CFG_LINKWD_ACCEPT • 6'h09: S_CFG_LANENUM_WAIT • 6'h0A: S_CFG_LANENUM_ACCEPT • 6'h0B: S_CFG_COMPLETE • 6'h0C: S_CFG_IDLE • 6'h0D: S_RCVRY_LOCK • 6'h0E: S_RCVRY_SPEED • 6'h0F: S_RCVRY_RCVRCFG • 6'h10: S_RCVRY_IDLE • 6'h11: S_L0 • 6'h12: S_L0S • 6'h13: S_L123_SEND_EIDLE • 6'h14: S_L1_IDLE • 6'h15: S_L2_IDLE • 6'h16: S_L2_WAKE • 6'h17: S_DISABLED_ENTRY • 6'h18: S_DISABLED_IDLE • 6'h19: S_DISABLED • 6'h1A: S_LPBK_ENTRY • 6'h1B: S_LPBK_ACTIVE • 6'h1C: S_LPBK_EXIT • 6'h1D: S_LPBK_EXIT_TIMEOUT • 6'h1E: S_HOT_RESET_ENTRY • 6'h1F: S_HOT_RESET • 6'h20: S_RCVRY_EQ0 • 6'h21: S_RCVRY_EQ1 • 6'h22: S_RCVRY_EQ2 • 6'h23: S_RCVRY_EQ3	
p<n>_ss_app_surprise_down_err where n=0,1	Active high asynchronous output signal. Indicates that a surprise down event is occurring in the HardIP controller. This error event is triggered when the PHY layer reports to the Data Link Layer that the link is down.	1'b0
p<n>_ss_app_rx_par_err where n=0,1	Indicates a parity error detected at the input of the HIP'S RX buffer. Asserts for a single cycle. Synchronous to the axi_st_clk clock. *Note:* Application must reset the HardIP if this occurs because parity errors can leave the Hard IP in an unknown state.	1'b0
p<n>_ss_app_tx_par_err where n=0,1	Indicates a parity error during TX TLP transmission at the HIP. Asserts for a single cycle. Synchronous to the axi_st_clk clock.	1'b0

3.10.1.3. Additional Debug Tools

To access additional registers like the IP debug registers (for example, receiver detection, etc.), you can use the Debug Toolkit, Control and Status Responder Interfaces (lite_csr) of the AXI Streaming Intel FPGA IP for PCI Express.

3.10.2. Debugging Data Transfer and Performance Issues

There are many possible reasons causing the PCIe link to stop transmitting data. The PCI Express base specification defines three types of errors, outlined in the following table:

Table 14. Error Types Defined by the PCI Express Base Specification

Type	Responsible Agent	Description
Correctable	Hardware	While correctable errors may affect system performance, data integrity is maintained.
Uncorrectable, non-fatal	Device software	Uncorrectable, non-fatal errors are defined as errors in which data is lost, but system integrity is maintained. For example, the fabric may lose a particular TLP, but it still works without problems.
Uncorrectable, fatal	System software	Errors generated by a loss of data and system failure are considered uncorrectable and fatal. Software must determine how to handle such errors: whether to reset the link or implement other means to minimize the problem.

3.10.2.1. Advanced Error Reporting (AER)

Each PCI Express compliant device must implement a basic level of error management and can optionally implement advanced error management. The PCI Express Advanced Error Reporting Capability is an optional Extended Capability that may be implemented by PCI Express device functions supporting advanced error control and reporting.

The AXI Streaming Intel FPGA IP for PCI Express implements both basic and advanced error reporting. Error handling for a Root Port is more complex than that of an Endpoint. In this IP, the Physical Functions (PFs) are always capable of AER (enabled by default). There is no AER implementation for Virtual Functions (VFs).

Use the AER capability of the IP to identify the type of error and the protocol stack layer in which the error may have occurred. Refer to the PCI Express Capability Structures section of the Configuration Space Registers appendix for the AER Extended Capability Structure and the associated registers.

Table 15. Correctable Error Status Register (AER)

Observation	Issue	Resolution
Receiver error bit set	Physical layer error which may be due to a PCS error when a lane is in L0, or a Control symbol being received in the	Use the Control and Status Register Responder Interface (lite_csr) to access the PCIe configuration space registers, IP Debug Registers

continued...

Observation	Issue	Resolution
	wrong lane, or signal Integrity issues where the link may transition from L0 to the Recovery state.	to obtain more information about the error. Also refer to the flow chart in Debugging Link Training Issues to obtain more information about the error.
Bad DLLP bit set	Data link layer error which may occur when a CRC verification fails.	Use the Control and Status Register Responder Interface (lite_csr) to access the PCIe configuration space registers, IP Debug Registers to obtain more information about the error.
Bad TLP bit set	Data link layer error which may occur when an LCRC verification fails or when a sequence number error occurs.	Use the Control and Status Register Responder Interface (lite_csr) to access the PCIe configuration space registers, IP Debug Registers to obtain more information about the error.
Replay_num_rollover bit set	Data link layer error which may be due to TLPs sent without success (no ACK) four times in a row.	Use the Control and Status Register Responder Interface (lite_csr) to access the PCIe configuration space registers, IP Debug Registers to obtain more information about the error.
replay timer timeout status bit set	Data link layer error which may occur when no ACK or NAK was received within the timeout period for the TLPs transmitted.	Use the Control and Status Register Responder Interface (lite_csr) to access the PCIe configuration space registers, IP Debug Registers to obtain more information about the error.
Advisory non-fatal	Transaction layer error which may be due to higher priority uncorrectable error detected.	Use the Control and Status Register Responder Interface (lite_csr) to access the PCIe configuration space registers, IP Debug Registers to obtain more information about the error.
Corrected internal error bits set	Transaction layer error which may be due to an ECC error in the internal Hard IP RAM.	Use the Control and Status Register Responder Interface (lite_csr) to access the PCIe configuration space registers, IP Debug Registers to obtain more information about the error.

Table 16. Uncorrectable Error Status Register (AER)

Observation	Issue	Resolution
Data link protocol error	Data link layer error which may be due to transmitter receiving an ACK/NAK whose Seq ID does not correspond to an unacknowledged TLP or ACK sequence number.	Use the Control and Status Register Responder Interface (lite_csr) to access the PCIe configuration space registers, IP Debug Registers to obtain more information about the error.
Surprise down error	Data link layer error which may be due to link going down during L0, indicating the physical layer link is going down unexpectedly.	Use the Control and Status Register Responder Interface (lite_csr) to access the PCIe configuration space registers, IP Debug Registers to obtain more information about the error.
Flow control protocol error	Transaction layer error which can be due to the receiver reporting more than the allowed credit limit. This error occurs when a component does not receive updated flow control credits with the 200 µs limit.	Use the Control and Status Register Responder Interface (lite_csr) to access the PCIeconfiguration space registers, IP Debug Registers, TX/RX flow control credit interfaces to obtain more information about the error.
Poisoned TLP received	Transaction layer error which can be due to a received TLP with the EP bit set.	Use the Control and Status Register Responder Interface (lite_csr) to access the PCIe configuration space registers, IP Debug Registers to obtain more information about the error.
Completion timeout	Transaction layer error which can be due to a completion not received within the required amount of time after a non-posted request was sent.	Use the Control and Status Register Responder Interface (lite_csr), completion timeout interface to access the PCIe configuration space registers, IP Debug Registers to obtain more information about the error.

continued...

Observation	Issue	Resolution
Completer abort	Transaction layer error which can be due to a completer being unable to fulfill a request due to a problem with the requester or a failure of the completer.	Use the Control and Status Register Responder Interface (lite_csr) to access the PCIe configuration space registers, IP Debug Registers to obtain more information about the error.
Unexpected completion	Transaction layer error which can be due to a requester receiving a completion that does not match any request awaiting a completion. The TLP is deleted by the Hard IP and not presented to the Application Layer.	Use the Control and Status Register Responder Interface (lite_csr) to access the PCIe configuration space registers, IP Debug Registers to obtain more information about the error.
Receiver overflow	Transaction layer error which can be due to a receiver receiving more TLPs than the available receive buffer space. The TLP is deleted by the Hard IP and not presented to the Application Layer.	Use the Control and Status Register Responder Interface (lite_csr) to access the PCIe configuration space registers, IP Debug Registers, TX/RX flow control credit interfaces to obtain more information about the error.
Malformed TLP	Transaction layer error which can be due to errors in the received TLP header. The TLP is deleted by the Hard IP and not presented to the Application Layer	Use the Control and Status Register Responder Interface (lite_csr) to access the PCIe configuration space registers, IP Debug Registers to obtain more information about the error.
ECRC error	Transaction layer error which can be due to an ECRC check failure at the receiver despite the fact that the TLP is not malformed and the LCRC check is valid. The Hard IP block handles this TLP automatically. If the TLP is a nonposted request, the Hard IP block generates a completion with a completer abort status. The TLP is deleted by the Hard IP and not presented to the Application Layer.	Use the Control and Status Register Responder Interface (lite_csr) to access the PCIe configuration space registers, IP Debug Registers to obtain more information about the error.
Unsupported request	Transaction layer error which can be due to the completer being unable to fulfill the request. The TLP is deleted in the Hard IP block and not presented to the Application Layer. If the TLP is a non-posted request, the Hard IP block generates a completion with Unsupported Request status.	Use the Control and Status Register Responder Interface (lite_csr) to access the PCIe configuration space registers, IP Debug Registers to obtain more information about the error.
ACS violation	Transaction layer error which can be due to access control error in the received posted or non-posted request.	Use the Control and Status Register Responder Interface (lite_csr) to access the PCIe configuration space registers, IP Debug Registers to obtain more information about the error
Uncorrectable internal error	Transaction layer error which can be due to an internal error that cannot be corrected by the hardware.	Use the Control and Status Register Responder Interface (lite_csr) to access the PCIe configuration space registers, IP Debug Registers to obtain more information about the error.
Atomic egress blocked		Use the Control and Status Register Responder Interface (lite_csr) to access the PCIe configuration space registers, IP Debug Registers to obtain more information about the error.
TLP prefix blocked	EP or RP only	Use the Control and Status Register Responder Interface (lite_csr) to access the PCIe configuration space registers, IP Debug Registers to obtain more information about the error.
Poisoned TLP egress blocked	EP or RP only	Use the Control and Status Register Responder Interface (lite_csr) to access the PCIe configuration space registers, IP Debug Registers to obtain more information about the error.

3.10.2.2. Second-Level Debug Tools

Use the following debug tools for second-level debug of any issue observed on the PCI Express link when using the AXI Streaming Intel FPGA IP for PCI Express:

- **Using the Completion Timeout Interface**: Refer to Transaction Ordering, Completion Timeout Interface (st_cplto) for details on this interface.
- **Using the Control Shadow Interface**: Refer to Control Shadow, Control Shadow Interface (st_ctrlshadow) on page 124 for details on this interface.
- **Using the Configuration Intercept Interface**: Refer to Configuration Intercept Interface for details on this interface.
- **Using the Configuration Space Extension Interface**: Refer to Configuration Space Extension, Configuration Extension Bus Interface on page 117 for details on this interface.
- **Using the Flow Control Credit Handling Interfaces**: Refer to Completion Timeout, Application Transmit Flow Control Credit Interface (st_txcrdt) for details on these interfaces.
- **Using the Control and Status Register Responder Interface**: Refer to Control and Status Register Responder Interface (lite_csr) for details on this interface.
- **Using the Application Error Interface**: Refer to Application Error Reporting for details on this interface.
- **Using the Debug toolkit**: Refer to Debug Toolkit and Hard IP (HIP) Reconfiguration Interface for details on this interface.

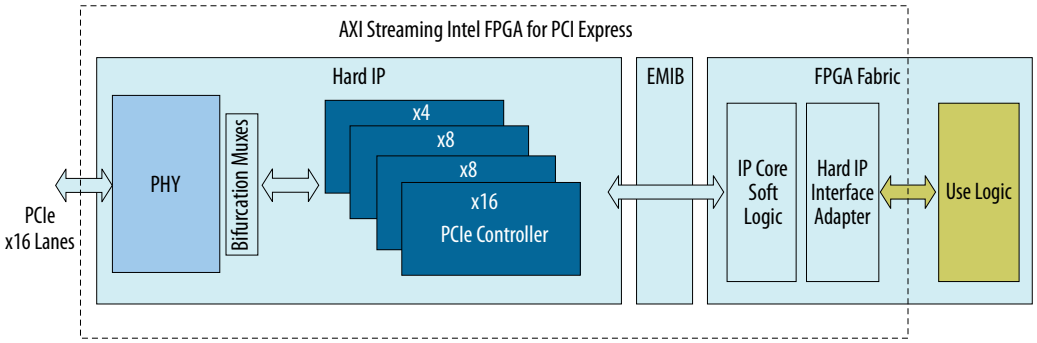
Related Information

- P-tile Avalon Streaming Intel FPGA IP for PCI Express User Guide
- F-tile Avalon Streaming Intel FPGA IP for PCI Express User Guide
- R-tile Avalon Streaming Intel FPGA IP for PCI Express User Guide
- AMBA AXI4-Stream Protocol Specification
- Intel Quartus Prime Pro Edition User Guide

4. IP Architecture and Functional Description

This chapter describes the architecture details of the IP and details the various blocks and modes available in the IP that you can use. The following figure displays the IP block diagram, showing important blocks and their interfaces.

Figure 14. AXI Streaming Intel FPGA IP for PCI Express Block Diagram



The IP gives you complete control over the PCIe HIP. You can implement any functionality of interest with finer control over the PCIe Transaction Layer Packet (TLP), credit handling and various modes provided by HIP.

On the transmit side, the IP forwards TLPs received from application to the Link. Your application user logic is responsible for constructing TLPs as per PCIe rules, and for implementing credit management logic using the credit interfaces provided. The tag allocation and management are also done by the application user logic.

On the receive side, the IP sends TLPs received from Link to user side with some additional information like BAR number and function number. Apart from forwarding received TLPs, additional sideband interfaces are provided for error reporting, reading, and writing registers in Hard IP and reset handshake.

This mode also provides basic telemetry and debug functionality blocks.

The following table shows the various profiles available when using the IP. The profiles when selected will populate the below default settings in the Parameter Editor and can be used as a starting point by the user.

If you want settings that do not match any of the profiles, you can choose the Basic profile and configure the settings you want via the "expand" tabs. For example, you can choose 4 PFs in an Endpoint after choosing the Basic profile.

Table 17. Functional Mode Profiles

Profile	Default Parameter Editor Selections
Basic	• One PF in an endpoint • One 64-bit BAR of 64 kB • AER enabled • Max payload size of 512 bytes • Maximum read request size (MRRS)=max supported by tile • Tags=max supported by tile
Basic +	All features from Basic Profile plus: • VirtIO PCIe Capability present • FLR Enabled • MSI-X with 256 vectors
Virtual	• Two Physical Functions (PF) in an Endpoint • PF0 has no Virtual Functions (Supervisory Role) • 128 Virtual Functions per PF • AER Enabled • FLR Enabled • All functions have one 64-bit BAR of 64Kbyte • MSI-X enabled with 4 vectors per function • ATS, TPH, PASID capabilities Enabled • Maximum Payload size is 512 Bytes • MRRS=Max Supported by Tile • Tags = Max Supported by Tile
Virtual+	All features from Virtual profile plus: • VirtIO PCI Capability Present • Four PFs in an endpoint

4.1. Clocks and Resets

The AXI Streaming Intel FPGA IP for PCI Express has the following clock domains to drive the various interfaces.

Table 18. Clock domains in the AXI Streaming Intel FPGA IP for PCI Express

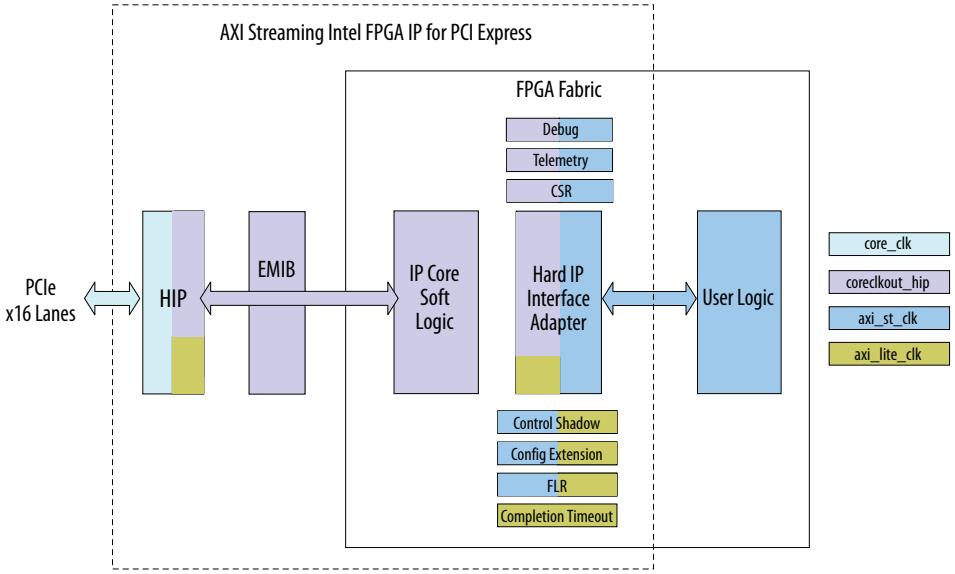
Clock Domain	Description
core_clk	This clock is synchronous to the SerDes parallel clock Gen4/Gen5: 1000 MHz Gen3: 500 MHz Gen2: 250 MHz Gen1: 125 MHz
coreclkout_hip_toapp	The coreclkout_hip output of the HIP drives this clock. Application can use this clock to generate PCIe IP clocks. Gen4/Gen5: 500 MHz Gen3: 250 MHz Gen2/Gen1: Gen1/Gen2 is supported only via link down-training and not natively. Hence, the coreclkout_hip_toapp clock frequency depends on the configuration you choose in the IP Parameter Editor. For example, if you choose a Gen3 configuration, the application clock frequency is 250 MHz.
axi_st_clk	This global clock signal is an input to the IP. This clock is used to clock the AXI-ST Datapath interfaces (TX, RX) to the application logic. All signals of the AXI-ST Datapath interface are sampled on the rising edge of axi_st_clk.

continued...

Clock Domain	Description
	The frequency of this clock depends on the mode in which the IP is configured: • In Native HIP mode, this clock uses the Hard IP's coreclkout_hip. • In non-Native HIP mode, the clock frequency can be selectable based on the speed of the link as shown below: — Gen5/Gen4: 500/470/400/350/250 MHz (32-, 64-, 128-byte widths) — Gen3: 300/275/250 MHz (32-, 64-byte widths)
axi_lite_clk	This global clock signal is an input to the IP. This clock is used to clock the sideband interfaces, e.g., control and status register interface, completion timeout interface, etc. All signals are sampled on the rising edge of axi_lite_clk. Frequency: 100-250 MHz (Default 250MHz)

The figure below shows clock domains in the IP. All the clocks must be always on for the correct functioning of a design.

Figure 15. Clock Domains in the AXI Streaming Intel FPGA IP for PCI Express



The AXI Streaming Intel FPGA IP for PCI Express has two types of resets:

- Bus Resets - The bus resets are AXI specification defined reset signals, which are used to reset the logic in the IP interfacing with AXI fabric.
- IP Resets - The IP reset signals perform cold/warm reset sequences.

You must implement a user reset sequencer in the application user logic and follow the assertion and deassertion sequence for graceful entry and exit for each of the resets (cold, warm etc). Refer to Clocks and Resets for the sequence to be followed for these reset signals.

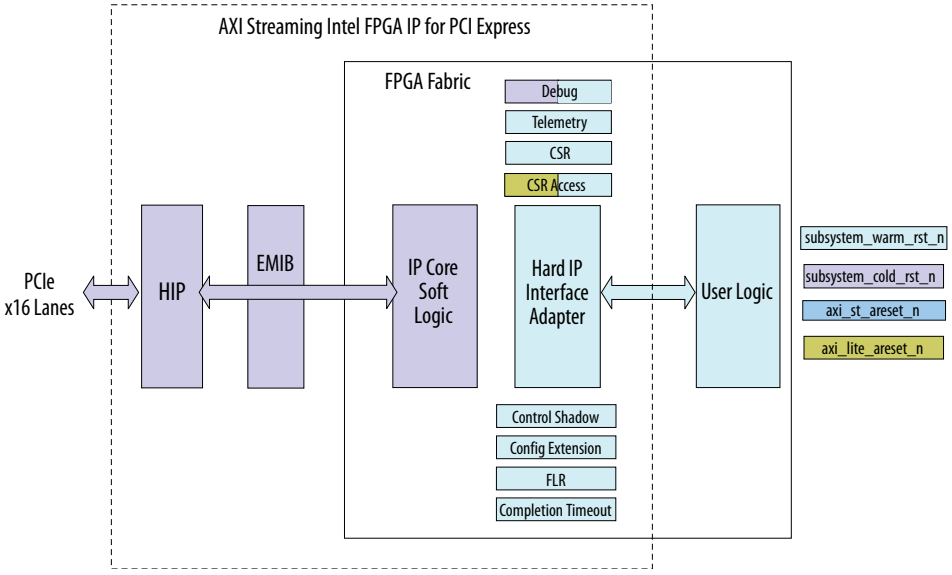
The IP has the following reset domains to drive the various interfaces.

Table 19. Reset Domains in the AXI Streaming Intel FPGA IP for PCI Express

Reset Domain	Type	Description
Cold reset	IP reset	A Reset following the application of power. This will reset the following: • Bus resets (AXI-ST/AXI-Lite) • Hard IP • Sticky registers of configuration space • When cold reset is triggered, warm reset and bus resets must be asserted Refer to *PCI Express Base Specification Revision 5.0 for more details on warm reset*.
Warm reset	IP reset	A Fundamental Reset without cycling main power. This will reset the following: • Bus resets (AXI-ST/AXI-Lite) • Hard IP • The warm reset can be triggered multiple times by user without going through cold reset sequence. • When warm reset is triggered, Bus resets must be asserted Refer to *PCI Express Base Specification Revision 5.0 for more details on warm reset*.
AXI-ST reset	Bus reset	This will reset the AXI-ST main data path interface (e.g., AXI-ST TX/RX)
AXI-Lite reset	Bus reset	This will reset the AXI-Lite sideband interfaces (e.g., Completion timeout, control and status register)

The following figure indicates the reset domains in the IP.

Figure 16. Reset Domains



The list below specifies reset sequencing handshake requirement. The user must implement the reset sequencer in application user logic and follow the assertion and deassertion sequence for graceful entry and exit from reset.

Reset assertion:

- Assertion happens concurrently for all.

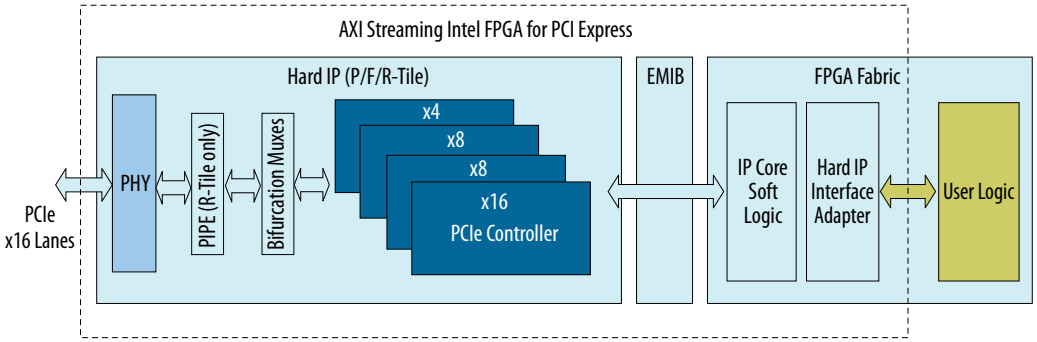
Reset deassertion:

- Cold reset
  - Req/Rdy handshake is used for graceful reset entry and exit.
- Warm reset
  - Req/Rdy handshake is used for graceful reset entry and exit.
- AXI4-Lite reset

4.2. PCIe Hard IP (HIP)

The PCIe Hard IP implements the functionality of the PCIe protocol. The HIP implements Physical, Data Link and Transaction Layers of the protocol. The HIP handles link training, DLLP exchanges, credit handling, BAR decode, and error handling in normal mode. It also implements SRIOV functionality for handling virtualization. The HIP's main data path interface is the Avalon Streaming interface.

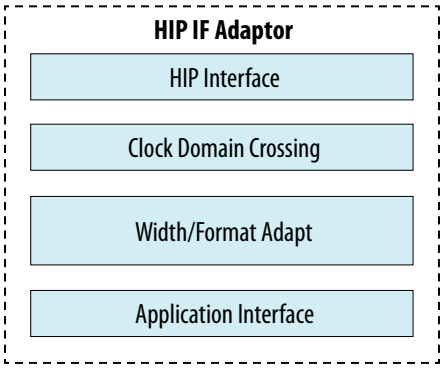
Figure 17. PCIe Hard IP



4.3. HIP Interface (IF) Adaptor

The Hard IP (HIP) interfaces with the HIP IF Adaptor in the AXI Streaming Intel FPGA IP for PCI Express. The HIP IF adaptor acts as an interface between the HIP and the downstream logic. The HIP IF Adaptor provides a standardized interface to the downstream logic by performing the required width and format adaptation depending on the AVST and sideband interfaces of the tile. The clock domain crossing module allows downstream logic to run at different frequencies.

Figure 18. Hard IP IF Adaptor



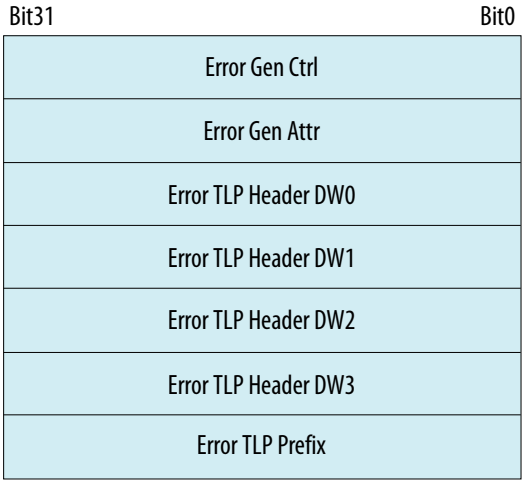
4.4. Application Error Reporting

The IP implements Application Error Reporting registers to allow you to indicate various errors. The IP logic then forwards this error information to the HIP block. The HIP block then responds to the assertion of these error bits by performing the following:

- Logging the status in the error reporting registers of the Function.
- Sending error messages as per Basic Error Reporting policies or as per AER policies.

The following figure shows registers implemented in the IP register space for error reporting. Refer to Register Descriptions for more details on the AER registers.

Figure 19. Error Reporting



Note: VF-related errors detected in application layer cannot be logged in HIP status registers as HIP's Application Error Interface doesn't provide error handling down to VF granularity. Additionally, the Hard IP Reconfiguration Interface doesn't provide access to set Error Status registers. In summary, VF related errors reported through the Application Error Reporting registers will not have any effect.

4.5. Debug Toolkit and Hard IP (HIP) Reconfiguration Interface

The IP instantiates the Debug Toolkit module to provide debug functionality. The Debug Toolkit can access the link related debug information from the Hard IP (using the Hard IP reconfiguration interface) available as tabs on the Debug Toolkit's graphical user interface. The Debug Toolkit can also access the IP's soft register space for control and status information per core (e.g., core_x16, core_x8, core_x4, core_x4) through the system console

The Debug Toolkit provides the following features:

- Real time monitoring of physical layer.
- View of Protocol and Link status information.
- View of PLL and per-channel status of link.

- Indicates presence of a re-timer connected between link partners.
- Basic and advance debugging capabilities including PMA register access and Eye margining capability.
- The IP's soft register space for control and status information using system console.

Refer to the P-Tile Avalon Streaming Intel FPGA IP for PCI Express User Guide for more details on the Hard IP registers that can be accessed using the Debug Toolkit in devices with P-Tile.

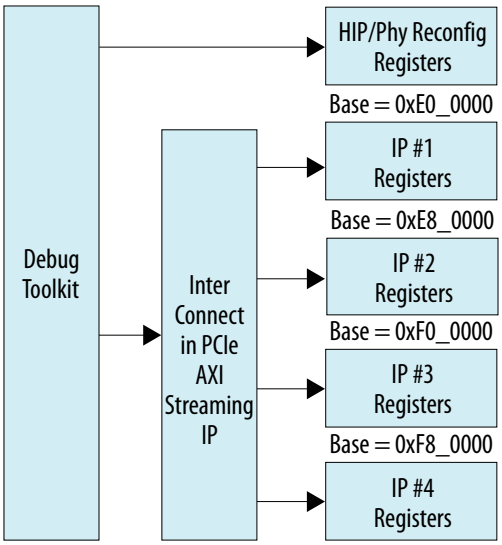
Refer to the F-Tile Avalon Streaming Intel FPGA IP for PCI Express User Guide for more details on the Hard IP registers that can be accessed using the Debug Toolkit in devices with F-Tile.

Refer to the R-Tile Avalon Streaming Intel FPGA IP for PCI Express User Guide for more details on the Hard IP registers that can be accessed using the Debug Toolkit in devices with R-Tile.

To access the IP's soft control and status registers, you must assign a value of '111' on address [23:21]. For the lower 20-bit address offsets, use the address map defined in Register Descriptions for the respective PCIe IP's base address above. The IP's soft register space for each core will start from the base addresses below:

- PCIe IP Instance #1 (core_x16) - 0xE0_0000
- PCIe IP Instance #2 (core_x8) - 0xE8_0000
- PCIe IP Instance #3 (core_x4) - 0xF0_0000
- PCIe IP Instance #4 (core_x4) - 0xF8_0000

Figure 20. Debug Toolkit and Reconfiguration Interfaces



4.6. Configuration Space Extension

The Hard IP implements mandatory PCI and PCIe capabilities. The AXI Streaming Intel FPGA IP for PCI Express provides the Configuration Extension Bus (CEB) interface to extend the configuration capabilities available in an IP's protocol stack HIP block.

- The configuration TLPs with a destination address not matching with internally implemented registers are routed to the configuration extension interface.
- The user application is responsible for returning data on read.
- The user application returns zero if a transaction targets unimplemented address space.
- The write access to unimplemented address is dropped by application.
- Maximum one outstanding read request is allowed.
- The next pointer field of the last capability structure within HIP is set by the external capability pointer parameter.
  - Separate parameters are provided for PCI Compatible Region of Physical Function (PF) and Virtual Function (VF).
  - Separate parameters are provided for PCIe extended capability region of Physical Function (PF) and Virtual Function (VF).
- The IP implements timeout mechanism for request issued on the CEB interface.
  - The timeout value is configurable, and you can set this value during compilation.
  - The IP sends the completion back to the host with "SC" status and data as all zeros in case the application failed to return data before the timeout counter expires.

Note: The CEB interface and the CII interface are mutually exclusive. Hence, both cannot be enabled at the same time.

Note: Refer to Interfaces and Signals for details on the interface signals.

4.7. Control Shadow

The control shadow interface is used to bring out the settings of the various configuration register fields of the function. These fields are often required in designing the control path of the application layer logic. The application logic decodes information provided on this interface to create a shadow copy. The interface provides updates for primary control signals only. The application logic must read extra information required through the lite_csr interface by reading the configuration register of interest.

Note: Refer to Interfaces and Signals for details on the interface signals.

4.8. Configuration Intercept Interface

The Configuration Intercept Interface (CII) allows the application logic to detect the occurrence of a Configuration (CFG) request on the link and to modify its behavior. The application logic should detect the CFG request at the assertion of ss_app_st_ciireq_tvalid on the ss_app_st_ciireq* interface.

The application logic can use the CII to:

- Delay the processing of a CFG request by the controller. This allows the application to perform any housekeeping task first. This can be achieved by withholding the assertion of app_ss_st_ciireq_tready.
- Overwrite the data payload of a CfgWr request. The application logic can also overwrite the data payload of a CfgRd completion TLP. This can be achieved using the app_ss_st_ciiresp* interface.

Note:

1. This interface is provided so that the IP is backward compatible with legacy application logic that relies on CII for their functionality. Newly defined application logic should avoid using the CII interface and move to the CEB interface.
2. The CEB interface and the CII interface are mutually exclusive. Hence, both cannot be enabled at the same time.

Note: Refer to Interfaces and Signals for details on the interface signals.

4.9. Power Management

The power management registers contain the power management messages. The generation of power management messages based on user input is not supported. The IP exposes these registers only if the tile supports this capability.

4.10. Legacy Interrupt

The Legacy Interrupt register controls generation of assert and deassert messages. This register allows user to generate legacy interrupt through a side band interface instead of sending it over the main band AXI-ST interface.

4.11. Credit Handling

The IP exposes link partner credit to you in the Transmit direction and application logic credits in the Receive direction. The credits are advertised as a limit value specified in the PCIe specification. Apart from the AXI Streaming ready-valid handshake, you must check the availability of credits for transmitting and receiving the TLP. The IP handles credits using the Transmit and Receive flow control credit interfaces and depends on the configuration and modes used.

Note: Refer to Interfaces and Signals for details on the interface signals.

4.12. Completion Timeout

The IP communicates completion timeout events to the application logic through the dedicated completion timeout interface.

Note: Refer to Interfaces and Signals for details on the interface signals.

4.13. Transaction Ordering

The IP does not have separate receiving queues to handle PCIe transaction ordering or prevent deadlocks. The application logic needs to ensure the transactions adhere to PCIe ordering rules that prevent deadlocks, namely:

- Allow posted writes to pass blocked read transactions.
- Allow posted writes to pass blocked configuration write transactions.
- Allow completion to pass blocked read transactions.
- Allow completion to pass blocked configuration write transactions.

4.14. Page Request Service (PRS) Events

The Page Request Service (PRS) Control register allows the IP to generate events on the HIP's PRS interface. This register allows user logic to generate these PRS events through a sideband interface.

4.15. TX Non-Posted Metering Requirement on Application

The PCIe HIP implements a finite number of RX Completion Buffers for its header and data. However, in endpoint mode, it advertises infinite credit to the host. Hence, the application logic needs to implement metering logic on its TX Non-Posted requests such that it does not issue more requests than allowed to avoid overflowing the completion buffer. Refer to the Appendix sections for detailed completion buffer sizes for each different tile.

4.16. MSI Pending

The MSI Pending registers (MSI PENDING CTRL and MSI PENDING) allow application layer to indicate their MSI Pending bits to the HIP.

4.17. D-State Status

The D-State Status register (D-State STS) allows application to read the D-State value of each function from the HIP.

4.18. Configuration Retry Status Enable

The application layer can use the Configuration Retry Control register (CFG RETRY CTRL) to update the per PF Configuration Retry Status Enable controls (CRS En Controls) driven to the HIP. All VFs share the same control as their parent PF. When the corresponding PF's CRS En Control is asserted, HIP responds to Configuration TLPs with a CRS (Configuration Retry Status) if it has not already responded to a Configuration TLP with non-CRS status since the last reset. You can use this to hold off on enumeration.

4.19. AXI-Streaming Interface

The IP uses an AXI4 Streaming interface for transporting header and data information to and from the application logic. The header and data are presented as separate interfaces. The PCIe header, PF Number, VF Number, BAR number and Prefix information are grouped as a 32-byte header on the AXI Streaming interface.

The data is presented as 128-, 64- or 32-bytes wide bus, segmented into a number of segments depending on the configuration (Gen5/4/3 x8/x16) and mode used. The following modes are supported by the IP:

- HIP Native mode
- Non-HIP Native mode
  - Simple packing mode
  - Compact packing mode

The HIP Native mode packing scheme is only available when you choose R-Tile. In this packing scheme, the AXI-ST Transmit and Receive interfaces follow all the rules that the AVST interface of the Native Hard IP follows for packing TLPs.

A packet must start from segment0 for a new cycle even for the compact packing mode. This constrains the design to send one packet per cycle. This is applicable to P/F/R-Tiles. Note that the HIP Native mode is not applicable to P/F-Tiles.

Note: The simple packing mode is not supported in the current Intel Quartus Prime release.

The compact packing scheme allows the header to be available in fixed locations. This is applicable to P/F/R-Tiles.

Refer to Application Packet Interface for details on how the application user logic must handle this interface in each of the configurations and modes.

4.20. Precision Time Measurement (PTM) [F/R-Tiles Only]

Note: This feature is planned for a future release.

Precision Time Measurement (PTM) enables precise coordination of events across multiple components with independent local time clocks. Ordinarily, such precise coordination would be difficult given that individual time clocks have differing notions of the value and rate of change of time. To work around this limitation, PTM enables components to calculate the relationship between their local times and a shared PTM Master Time, which is an independent time domain associated with a PTM Root. Each PTM Root supplies the PTM Master Time for a PTM Hierarchy.

Note: Only applicable when operating in Endpoint Mode (PTM Requester).

The endpoint generates a PTM request message that goes to the Root Complex. The PTM requester updates the time stamp t1 when generating a request. It updates the time stamp t4 when it receives the response. Refers to the PCIe Base Specification for t1/t4 definitions. Requester automatically updates the PTM context (starting dialogues) when enabled using all or any of the following:

- Automatic trigger every 10ms - always enabled
- Manual trigger through user input (ptm_manual_update)

The received PTM messages are also forwarded to the application layer. You may drop the messages if not useful. The PTM context valid indicates if the context is valid. The PTM context is automatically invalidated when:

- Clock stops or runs at the wrong frequency (for example, when the link speed is changing), or
- PTM is disabled, or
- PTM response times out (the requester restarts the PTM dialogue when the auto-update or manual update start conditions are met), or
- A duplicate PTM TLP is received or a replay TLP is sent (if waiting for a response, the requester waits for 100µs since the last non-duplicate request was sent before allowing a new PTM dialogue to be started).

The committed PTM accuracy targets for F/R-Tile are:

- +/-50ns for the common clock scheme.
- +/100ns for the separate clock scheme.

5. AXI Streaming Intel FPGA IP for PCI Express Parameters

The AXI Streaming Intel FPGA IP for PCI Express Parameter Editor provides the parameters you can set to configure your IP variant and simulation and hardware design examples.

5.1. Parameter Editor Parameters

The AXI Streaming Intel FPGA IP for PCI Express has one tab for parameterization, the **PCIe Interfaces** tab.

Table 20. AXI Streaming Intel FPGA IP for PCI Express Parameters: - PCIe Interfaces Tab

Parameter	Default Setting	Parameter Description
PCIe Interface 0 Settings		
PCIe Tile	P-Tile	Selects the supported Tile of the PCIe interface based on the device. • P-Tile • F-Tile • R-Tile
PCIe Profile	Basic	Selects functional features based on profile like virtualization, additional interfaces, number of endpoints, etc. • Basic • Basic+ • Virtual • Virtual+
PCIe Mode	Gen4 2x8	Selects the width of the data interface between the transaction layer and the application layer implemented in the PLD fabric, the lane data rate and the lane rate. Gen5 1x16 Gen5 2x8 Gen4 1x16 Gen4 1x8 Gen4 2x8 Gen3 1x16 Gen3 2x8 Gen3 1x8
Enable TLP-Bypass Mode	Disabled	Enables the TLP Bypass mode.
Port Mode	Native Endpoint	Selects Port Mode. For Endpoint mode with multiple ports, all ports are set to endpoint mode.

continued...

Parameter	Default Setting	Parameter Description
Enable PHY Reconfiguration	Disabled	When on, creates an Avalon-MM slave interface that software can drive to update Transceiver reconfiguration registers.
PLD Clock Frequency	400MHz	Selects the PLD Clock Frequency. 500MHz (Gen4/Gen5) 450MHz (Gen4) 400MHz (Gen4) 350MHz (Gen4) 300MHz (Gen3)
Enable SRIS Mode	Disabled	Enables separate reference clock with independent Spread Spectrum Clocking (SSC).
Enable System PLL Clock	Enabled	Enables the System PLL Clock when using F-Tile.
Enable PCS and controller user reset	Disabled	Enables PCS and controller user reset in Endpoint and Bypass modes.
Enable Debug Toolkit	Disabled	Enables the Debug Toolkit.
Enable CVP (Intel VSEC)	Disabled	Enablement of CVP for single tile only.
Optional Side Interfaces		
Enable PCIe0 Control Shadow Interface	Disabled	Enables Control Shadow Interface. Host write to specific PCIe configuration space register's bit is indicated through this interface.
Enable PCIe0 Completion Timeout Interface	Disabled	Enables Completion Timeout Interface. Completion Timeout event is indicated through this interface.
Enable PCIe0 Configuration Extension Bus Interface	Disabled	Enables Configuration Extension Bus Interface. User can add additional PCIe capabilities using this interface.
PCIe0 Standard next address pointer for PF	0x00000000	This parameter is available when "Enable Configuration Extension Bus Interface" parameter is enabled. Enables CEB pointer address for PF (DW address in Hex). Valid range from 0x000 to 0x03F.
PCIe0 Extended next address pointer for PF	0x00000000	This parameter is available when "Enable Configuration Extension Bus Interface" parameter is enabled. Enables CEB pointer address for PF (DW address in Hex). Valid range from 0x040 to 0x3FF.
PCIe0 Standard next address pointer for VF	0x00000000	This parameter is available when "Enable Configuration Extension Bus Interface" parameter is enabled. Enables CEB pointer address for PF (DW address in Hex). Valid range from 0x000 to 0x03F.
PCIe0 Extended next address pointer for VF	0x00000000	This parameter is available when "Enable Configuration Extension Bus Interface" parameter is enabled. Enables CEB pointer address for PF (DW address in Hex). Valid range from 0x040 to 0x3FF.

continued...

Parameter	Default Setting	Parameter Description
PCIe0 CEB REQ to ACK Latency Timeout value	100	This parameter is available when "Enable Configuration Extension Bus Interface" parameter is enabled. Enables CEB REQ to ACK Latency Timeout value (in Clock Cycles). Valid range from 1 to 256.
Enable PCIe0 Configuration Intercept Interface	Disabled	Enables Configuration Intercept Interface (CII). User can intercept PCIe Configuration cycles using this interface (Power User Mode only).
PCIe0 CII REQ to ACK Latency Timeout value	100	This parameter is available when "Enable Configuration Intercept Interface" parameter is enabled. Enables CII REQ to ACK Latency Timeout value (in Clock Cycles). Valid range from 1 to 256.
Enable PCIe0 Configuration Monitoring	Disabled	CII monitoring feature is used to monitor configuration write cycle that happens with no intention to override the values being written. When CII monitoring feature is enabled, only the st_ciireq interface are exposed to the user and not st_ciiresp. The IP decodes the CII request received and outputs it on the st_ciireq interface if it is a write request. A "cii_halt" deassertion happens towards the HIP after the shadow registers operation is done, without depending on app_ss_st_ciiresp_tvalid.
Enable PCIe0 VirtIO PCI CFG Interface	Disabled	HOST read and write accesses to VIRTIO PCI Configuration Access Data Register will use this interface for its alternate access functionality.
PCIe Interfaces Ports Settings		
Extend PCIe Ports Settings	Disabled	Extends PCIe IP Settings. When enabled, the PCIe IP setting can be further configured.
Top Level Settings		
Enable ASPM Support Control	No ASPM support	Enables Active State Power Management (ASPM) Control in the Link Control Register. Valid settings are: • No ASPM support • L0s supported • L0s and L1 supported • L1 supported
Enable CVP (Intel VSEC)	Disabled	Enables Configuration Via Protocol (CVP) for a single tile only.
Port 0		
***PCIe0 Avalon Settings when* Extend PCIe Ports Settings=On**		
Enable byte parity Port on Avalon-ST interface	Disabled	Enables or disables parity Port on Avalon-ST interface.
Enable Power Management Interface	Enabled	When selected, Power Management interface will be exported.

continued...

Parameter	Default Setting	Parameter Description
Power management interface: Enable p0_apps_ready_entr_l23_i port	Disabled	When selected, input port p0_apps_ready_entr_l23_i is exposed in the Power Management interface. The application logic asserts this signal to indicate that it is ready to enter L2/L3 Ready state. It is provided for applications that must control L2/L3 Ready entry (in case certain task must be performed before going into L2/L3 Ready).
Power management interface: Enable p0_app_xfer_pending_i port	Disabled	When selected, input port p0_app_xfer_pending_i is exposed in the Power Management interface. This port is used to prevent the entry to L1 or initiates the exit from L1.
Enable Legacy Interrupt	Enabled	When selected, the Interrupt interface is exported.
Enable Completion Timeout Interface	Enabled	Select to enable completion timeout interface.
Enable Configuration Intercept Interface	Enabled	Select to enable configuration intercept interface.
Enable PRS Event	Enabled	Select to enable PRS Event Interface.
Enable Error Interface	Enabled	Select to enable Error Interface.
Export `pld_warm_rst_rdy` and `link_req_rst_n` interface to top level	Enabled	Exports the `pld_warm_rst_rdy` and `link_req_rst_n` interface to the top level.
Export `user_mode_to_pld` and `pld_in_use` interface	Enabled	Exports `user_mode_to_pld` and `pld_in_use` interface to the top level.
Enable 10-bit tag support Interface	Disabled	Enables 10-bit tag support enable interface.
PCIe Header Format	Enabled	When On, the P-Tile header format for PCIe is used (the first byte of the header dword is located in the most significant byte of the dword). When Off, the first byte of the header dword is located in the least significant byte of the dword.
PCIe0 Configuration Intercept Interface (CII)		
Enable Range 0 CII	Enabled	Indicates Physical Functions (PFs) that are subject to having Configuration cycles be intercepted or halted. Each bit refers to one PF. Together with the Range 0 Start Address and Range 0 Address Size parameters, this parameter defines the range for CII to take effect.
Range 0 Start Address	0x00000000	Indicates the starting address where Configuration cycles can be intercepted or halted. Together with the Range 0 Address Size and Enable Range 0 CII parameters, this parameter defines the range for CII to take effect.
Range 0 Address Size	0x00000fff	Indicates the address size for Configuration cycles to be intercepted or halted. Together with the Range 0 Start

continued...

Parameter	Default Setting	Parameter Description
		Address and Enable Range 0 CII parameters, this parameter defines the range for CII to take effect.
Enable Range 1 CII	Enabled	Indicates Physical Functions (PFs) that are subject to having Configuration cycles be intercepted or halted. Each bit refers to one PF. Together with the Range 1 Start Address and Range 1 Address Size parameters, this parameter defines the range for CII to take effect.
Range 1 Start Address	0x00000000	Indicates the starting address where Configuration cycles can be intercepted or halted. Together with the Range 1 Address Size and Enable Range 1 CII parameters, this parameter defines the range for CII to take effect.
Range 1 Address Size	0x00000000	Indicates the address size for Configuration cycles to be intercepted or halted. Together with the Range 1 Start Address and Enable Range 1 CII parameters, this parameter defines the range for CII to take effect.
Enable Range 2 CII	Enabled	Indicates Physical Functions (PFs) that are subject to having Configuration cycles be intercepted or halted. Each bit refers to one PF. Together with the Range 2 Start Address and Range 2 Address Size parameters, this parameter defines the range for CII to take effect.
Range 2 Start Address	0x00000000	Indicates the starting address where Configuration cycles can be intercepted or halted. Together with the Range 2 Address Size and Enable Range 2 CII parameters, this parameter defines the range for CII to take effect.
Range 2 Address Size	0x00000000	Indicates the address size for Configuration cycles to be intercepted or halted. Together with the Range 2 Start Address and Enable Range 2 CII parameters, this parameter defines the range for CII to take effect.
PCIe0 Configuration, Debug and Extension Options		
Enable HIP dynamic reconfiguration of PCIe registers	Enabled	When on, creates an Avalon-MM slave interface that software can drive to update global configuration registers which are read-only at run time.
Gen 3 Requested equalization far-end TX preset vector	0x00000200	Specifies the Gen 3 requested phase 2/3 far-end TX preset vector. Choosing a value different from the default is not recommended for most designs.
Gen 4 Requested equalization far-end TX preset vector	0x00000008	Specifies the Gen 4 requested phase 2/3 far-end TX preset vector. Choosing a value different from the default is not recommended for most designs.
Gen 5 Requested equalization far-end TX preset vector	0x00000200	Specifies the Gen 5 requested phase 2/3 far-end TX preset vector. Choosing a value different from the default is not recommended for most designs.

continued...

Parameter	Default Setting	Parameter Description
Predetermined Number of Lanes	8	Indicates the number of lanes to check for exit from Electrical Idle in Polling.Active and L2.Idle. This value is referred to as the Predetermined Number of Lanes in Section 4.2.6.2.1 of the PCIe Base Specification. • 16 • 8 • 4 • 2 • 1
Enable Rx Buffer Limit Ports	Disabled	When selected, RX buffer limit ports is exported for you to control RX Posted, Non-Posted and CplD Packets. Otherwise, the Max Buffer Size will be used.
Bypass Posted Rx Buffer Limit	Disabled	When selected, the RX buffer limit selected for Posted packets is bypassed.
Bypass Non-Posted Rx Buffer Limit	Disabled	When selected, the RX buffer limit selected for Non-Posted packets is bypassed.
Bypass CplD Rx Buffer Limit	Disabled (Power User Mode)	When selected, the RX buffer limit selected for CplD packets is bypassed.
PCIe0 Base Address Registers		
PCIe0 PF0 BAR Configuration		
PCIe0 PF0 BAR		
BAR0 Type	64-bit prefetchable memory	Sets the BAR type (64-bit prefetchable or 32-bit non-prefetchable).
BAR0 Size	64 Kbytes – 16 bits	Sets from 7-64 bits per base address register (BAR).
BAR1 Type	Disabled	Sets the BAR type.
BAR2 Type	Disabled	Sets the BAR type.
BAR3 Type	Disabled	Sets the BAR type.
BAR4 Type	Disabled	Sets the BAR type.
BAR5 Type	Disabled	Sets the BAR type.
Expansion ROM Size	Disabled	Specifies an expansion ROM from 4 KBytes - 16 MBytes when enabled.
PCIe0 PF0 VF BAR		
BAR0 Type	64-bit prefetchable memory	Sets the BAR type.
BAR0 Size	64 Kbytes – 16 bits	Sets from 7-64 bits per base address register (BAR).
BAR1 Type	Disabled	Sets the BAR type.
BAR2 Type	Disabled	Sets the BAR type.
BAR3 Type	Disabled	Sets the BAR type.
BAR4 Type	Disabled	Sets the BAR type.

continued...

Parameter	Default Setting	Parameter Description
BAR5 Type	Disabled	Sets the BAR type.
Expansion ROM Size	Disabled	Specifies an expansion ROM from 4 KBytes - 16 MBytes when enabled.
PCIe0 PF1 BAR Configuration		
PCIe0 PF1 BAR		
BAR0 Type	64-bit prefetchable memory	Sets the BAR type.
BAR0 Size	64 Kbytes – 16 bits	Sets from 7-64 bits per base address register (BAR).
BAR1 Type	Disabled	Sets the BAR type.
BAR2 Type	Disabled	Sets the BAR type.
BAR3 Type	Disabled	Sets the BAR type.
BAR4 Type	Disabled	Sets the BAR type.
BAR5 Type	Disabled	Sets the BAR type.
Expansion ROM Size	Disabled	Specifies an expansion ROM from 4 KBytes - 16 MBytes when enabled.
PCIe0 PF1 VF BAR		
BAR0 Type	64-bit prefetchable memory	Sets the BAR type.
BAR0 Size	64 Kbytes – 16 bits	Sets from 7-64 bits per base address register (BAR).
BAR1 Type	Disabled	Sets the BAR type.
BAR2 Type	Disabled	Sets the BAR type.
BAR3 Type	Disabled	Sets the BAR type.
BAR4 Type	Disabled	Sets the BAR type.
BAR5 Type	Disabled	Sets the BAR type.
Expansion ROM Size	Disabled	Specifies an expansion ROM from 4 KBytes - 16 MBytes when enabled.
PCIe0 PF2 BAR Configuration		
PCIe0 PF2 BAR		
BAR0 Type	64-bit prefetchable memory	Sets the BAR type.
BAR0 Size	64 Kbytes – 16 bits	Sets from 7-64 bits per base address register (BAR).
BAR1 Type	Disabled	Sets the BAR type.
BAR2 Type	Disabled	Sets the BAR type.
BAR3 Type	Disabled	Sets the BAR type.
BAR4 Type	Disabled	Sets the BAR type.
BAR5 Type	Disabled	Sets the BAR type.
Expansion ROM Size	Disabled	Specifies an expansion ROM from 4 KBytes - 16 MBytes when enabled.
PCIe0 PF2 VF BAR		

continued...

Parameter	Default Setting	Parameter Description
BAR0 Type	64-bit prefetchable memory	Sets the BAR type.
BAR0 Size	64 Kbytes – 16 bits	Sets from 7-64 bits per base address register (BAR).
BAR1 Type	Disabled	Sets the BAR type.
BAR2 Type	Disabled	Sets the BAR type.
BAR3 Type	Disabled	Sets the BAR type.
BAR4 Type	Disabled	Sets the BAR type.
BAR5 Type	Disabled	Sets the BAR type.
Expansion ROM Size	Disabled	Specifies an expansion ROM from 4 KBytes - 16 MBytes when enabled.
PCIe3 PF1 BAR Configuration		
PCIe3 PF1 BAR		
BAR0 Type	64-bit prefetchable memory	Sets the BAR type.
BAR0 Size	64 Kbytes – 16 bits	Sets from 7-64 bits per base address register (BAR).
BAR1 Type	Disabled	Sets the BAR type.
BAR2 Type	Disabled	Sets the BAR type.
BAR3 Type	Disabled	Sets the BAR type.
BAR4 Type	Disabled	Sets the BAR type.
BAR5 Type	Disabled	Sets the BAR type.
Expansion ROM Size	Disabled	Specifies an expansion ROM from 4 KBytes - 16 MBytes when enabled.
PCIe3 PF1 VF BAR		
BAR0 Type	64-bit prefetchable memory	Sets the BAR type.
BAR0 Size	64 Kbytes – 16 bits	Sets from 7-64 bits per base address register (BAR).
BAR1 Type	Disabled	Sets the BAR type.
BAR2 Type	Disabled	Sets the BAR type.
BAR3 Type	Disabled	Sets the BAR type.
BAR4 Type	Disabled	Sets the BAR type.
BAR5 Type	Disabled	Sets the BAR type.
Expansion ROM Size	Disabled	Specifies an expansion ROM from 4 KBytes - 16 MBytes when enabled.
PCIe0 Device Identification Registers		
PCIe0 PF0 IDs		
Vendor ID	0x00001172	Sets the read-only value for the Vendor IP registers.

continued...

Parameter	Default Setting	Parameter Description
		Note: 1172 is the Intel/Altera Vendor ID and should NOT be used by customers in production products.
Device ID	0x00000000	Sets the read-only value of the Device ID register.
Revision ID	0x00000001	Sets the read-only value of the Revision ID register.
Class code	0x00FF0000	Sets the read-only value of the Class code register.
Subsystem Vendor ID	0x00000000	Sets the read-only value of the AXI Streaming IP Vendor ID register.
Subsystem Device ID	0x0000000	Sets the read-only value of the AXI Streaming IP Device ID register.
PCIe0 PF0 VF IDs		
Device ID	0x00000000	
Subsystem ID	0x00000000	Sets the read-only value of the AXI Streaming IP ID register for the virtual functions
PCIe0 PF1 IDs		
Vendor ID	0x00000000	Sets the read-only value for the Vendor IP registers. *Note:* 1172 is the Intel/Altera Vendor ID and should NOT be used by customers in production products.
Device ID	0x00000000	Sets the read-only value of the Device ID register.
Revision ID	0x00000001	Sets the read-only value of the Revision ID register.
Class code	0x00FF0000	Sets the read-only value of the Class code register.
Subsystem Vendor ID	0x00000000	Sets the read-only value of the AXI Streaming IP Vendor ID register.
Subsystem Device ID	0x0000000	Sets the read-only value of the AXI Streaming IP Device ID register.
PCIe0 PF1 VF IDs		
Device ID	0x00000000	Sets the read-only value of the Device ID register for the virtual functions
Subsystem ID	0x00000000	Sets the read-only value of the AXI Streaming IP ID register for the virtual functions.
PCIe0 PF2 IDs		
Vendor ID	0x00000000	Sets the read-only value for the Vendor IP registers. *Note:* 1172 is the Intel/Altera Vendor ID and should NOT be used by customers in production products.

continued...

Parameter	Default Setting	Parameter Description
Device ID	0x00000000	Sets the read-only value of the Device ID register.
Revision ID	0x00000001	Sets the read-only value of the Revision ID register.
Class code	0x00FF0000	Sets the read-only value of the Class code register.
Subsystem Vendor ID	0x00000000	Sets the read-only value of the AXI Streaming IP Vendor ID register.
Subsystem Device ID	0x0000000	Sets the read-only value of the AXI Streaming IP Device ID register.
PCIe0 PF2 VF IDs		
Device ID	0x00000000	Sets the read-only value of the Device ID register for the virtual functions.
Subsystem ID	0x00000000	Sets the read-only value of the AXI Streaming IP ID register for the virtual functions.
PCIe0 PF3 IDs		
Vendor ID	0x00000000	Sets the read-only value for the Vendor IP registers. *Note:* 1172 is the Intel/Altera Vendor ID and should NOT be used by customers in production products.
Device ID	0x00000000	Sets the read-only value of the Device ID register.
Revision ID	0x00000001	Sets the read-only value of the Revision ID register.
Class code	0x00FF0000	Sets the read-only value of the Class code register.
Subsystem Vendor ID	0x00000000	Sets the read-only value of the AXI Streaming IP Vendor ID register.
Subsystem Device ID	0x0000000	Sets the read-only value of the AXI Streaming IP Device ID register.
PCIe0 PF3 VF IDs		
Device ID	0x00000000	Sets the read-only value of the Device ID register for the virtual functions.
Subsystem ID	0x00000000	Sets the read-only value of the AXI Streaming IP ID register for the virtual functions.
PCIe0 PCI Express / PCI Capabilities		
PCIe0 Device		
Maximum payload size supported	512 Bytes	Sets the read-only value of the max payload size of the Device Capabilities register and optimizes for this payload size.
Support Extended Tag Field	Enabled	Sets the Extended Tag Field Supported bit in Configuration Space Device Capabilities Register.

continued...

Parameter	Default Setting	Parameter Description
PCIe0 Multifunction and SR-IOV System Settings		
Enable multiple physical functions	Enabled	Enables multiple physical functions.
Total physical functions (PFs)	4	Sets the number of physical functions.
Enable SR-IOV support	Enabled	Enables SR-IOV.
Total virtual functions of physical function 0 (PF0 VFs)	0	Sets the number of VFs to be assigned to Physical Function 0.
Total virtual functions of physical function 1 (PF1 VFs)	128	Sets the number of VFs to be assigned to Physical Function 1.
Total virtual functions of physical function 2 (PF2 VFs)	128	Sets the number of VFs to be assigned to Physical Function 2.
Total virtual functions of physical function 3 (PF3 VFs)	128	Sets the number of VFs to be assigned to Physical Function 3.
PCIe0 Link		
Link port number (Root Port only)	1	Sets the read-only value of the port number field in the Link Capabilities register.
Slot clock configuration	Enabled	Sets the read-only value of the slot clock configuration bit in the link status register.
PCIe0 Legacy Interrupt Pin Register		
PCIe0 PF0 Int Pin		
Set Interrupt Pin for PF0	NO INT	Sets interrupt Pin for PF0.
PCIe0 PF1 Int Pin		
Set Interrupt Pin for PF1	NO INT	Sets interrupt Pin for PF1.
PCIe0 PF2 Int Pin		
Set Interrupt Pin for PF2	NO INT	Sets interrupt Pin for PF2.
PCIe0 PF3 Int Pin		
Set Interrupt Pin for PF3	NO INT	Sets interrupt Pin for PF3.
PCIe0 LTR		
PCIe0 Enable LTR	Disabled	LTR (Latency Tolerance Reporting) New Mechanism that enables Endpoints to send information about their latency requirements for memory reads/writes and interrupts.
PCIe0 MSI		
PCIe0 PF0 MSI		
PF0 Enable MSI	Disabled	Enables or disables MSI capability for PF0.
PF0 MSI 64-bit addressing	Disabled	Enables or disables MSI 64-bit addressing for PF0.
PF0 MSI extended data capable	Disabled	Enables or disables MSI extended data capability for PF0.

continued...

Parameter	Default Setting	Parameter Description
PF0 Number of MSI messages requested	1	Sets the number of messages that the application can request in the multiple messages capable field of the Message Control register. 1 2 4 8 16 32
PCIe0 PF1 MSI		
PF1 Enable MSI	Disabled	Enables or disables MSI capability for PF1.
PF1 MSI 64-bit addressing	Disabled	Enables or disables MSI 64-bit addressing for PF1.
PF1 MSI extended data capable	Disabled	Enables or disables MSI extended data capability for PF1
PF1 Number of MSI messages requested	1	Sets the number of messages that the application can request in the multiple messages capable field of the Message Control register 1 2 4 8 16 32
PCIe0 PF2 MSI		
PF2 Enable MSI	Disabled	Enables or disables MSI capability for PF2.
PF2 MSI 64-bit addressing	Disabled	Enables or disables MSI 64-bit addressing for PF2.
PF2 MSI extended data capable	Disabled	Enables or disables MSI extended data capability for PF2.
PF2 Number of MSI messages requested	1	Sets the number of messages that the application can request in the multiple messages capable field of the Message Control register. 1 2 4 8 16 32
PCIe0 PF3 MSI		
PF3 Enable MSI	Disabled	Enables or disables MSI capability for PF3.
PF3 MSI 64-bit addressing	Disabled	Enables or disables MSI 64-bit addressing for PF3.
PF3 MSI extended data capable	Disabled	Enables or disables MSI extended data capability for PF3.

continued...

Parameter	Default Setting	Parameter Description
PF3 Number of MSI messages requested	1	Sets the number of messages that the application can request in the multiple messages capable field of the Message Control register. 1 2 4 8 16 32
PCIe0 MSI-X		
PCIe0 PF MSI-X		
PCIe0 PF0 MSI-X		
Enable MSI-X	Enabled	Enables or disables the MSI-X capability
Table size	4	Sets the number of entries in the MSI-X table.
Table offset	0x0000000000000000	Sets the read-only base address of the MSI-X table. The low-order 3 bits are automatically set to 0.
Table BAR indicator	0	Specifies which one of a function's base address registers, located beginning at 0x10 in the Configuration Space, maps the MSI-X table into memory space. This field is read-only.
Pending bit array (PBA) offset	0x0000000000000000	Specifies the offset from the address stored in one of the function's base address registers that points to the base of the MSI-X PBA. This field is read-only.
PBA BAR indicator	0	Indicates which of a function's base address registers, located beginning at 0x10 in the Configuration Space, maps the function's MSI-X PBA into memory space. This field is read-only.
PCIe0 PF1 MSI-X		
Enable MSI-X	Enabled	Enables or disables the MSI-X capability
Table size	4	Sets the number of entries in the MSI-X table.
Table offset	0x0000000000000000	Sets the read-only base address of the MSI-X table. The low-order 3 bits are automatically set to 0.
Table BAR indicator	0	Specifies which one of a function's base address registers, located beginning at 0x10 in the Configuration Space, maps the MSI-X table into memory space. This field is read-only.
Pending bit array (PBA) offset	0x0000000000000000	Specifies the offset from the address stored in one of the function's base address registers that points to the base of the MSI-X PBA. This field is read-only.

continued...

Parameter	Default Setting	Parameter Description
PBA BAR indicator	0	Indicates which of a function's base address registers, located beginning at 0x10 in the Configuration Space, maps the function's MSI-X PBA into memory space. This field is read-only.
PCIe0 PF2 MSI-X		
Enable MSI-X	Enabled	Enables or disables the MSI-X capability
Table size	4	Sets the number of entries in the MSI-X table.
Table offset	0x0000000000000000	Sets the read-only base address of the MSI-X table. The low-order 3 bits are automatically set to 0.
Table BAR indicator	0	Specifies which one of a function's base address registers, located beginning at 0x10 in the Configuration Space, maps the MSI-X table into memory space. This field is read-only.
Pending bit array (PBA) offset	0x0000000000000000	Specifies the offset from the address stored in one of the function's base address registers that points to the base of the MSI-X PBA. This field is read-only.
PBA BAR indicator	0	Indicates which of a function's base address registers, located beginning at 0x10 in the Configuration Space, maps the function's MSI-X PBA into memory space. This field is read-only.
PCIe0 PF3 MSI-X		
Enable MSI-X	Enabled	Enables or disables the MSI-X capability
Table size	4	Sets the number of entries in the MSI-X table.
Table offset	0x0000000000000000	Sets the read-only base address of the MSI-X table. The low-order 3 bits are automatically set to 0.
Table BAR indicator	0	Specifies which one of a function's base address registers, located beginning at 0x10 in the Configuration Space, maps the MSI-X table into memory space. This field is read-only.
Pending bit array (PBA) offset	0x0000000000000000	Specifies the offset from the address stored in one of the function's base address registers that points to the base of the MSI-X PBA. This field is read-only.
PBA BAR indicator	0	Indicates which of a function's base address registers, located beginning at 0x10 in the Configuration Space, maps the function's MSI-X PBA into memory space. This field is read-only.
PCIe0 VF MSI-X		
PCIe0 PF0 VF MSI-X		
Enable VF MSI-X	Enabled	Enables or disables the MSI-X capability.

continued...

Parameter	Default Setting	Parameter Description
Table size	4	Sets the number of entries in the MSI-X table.
Table offset	0x0000000000000000	Sets the read-only base address of the MSI-X table. The low-order 3 bits are automatically set to 0.
Table BAR indicator	0	Specifies which one of a function's base address registers, located beginning at 0x10 in the Configuration Space, maps the MSI-X table into memory space. This field is read-only.
Pending bit array (PBA) offset	0x0000000000000000	Specifies the offset from the address stored in one of the function's base address registers that points to the base of the MSI-X PBA. This field is read-only.
PBA BAR indicator	0	Indicates which of a function's base address registers, located beginning at 0x10 in the Configuration Space, maps the function's MSI-X PBA into memory space. This field is read-only.
PCIe0 PF1 VF MSI-X		
Enable VF MSI-X	Enabled	Enables or disables the MSI-X capability.
Table size	4	Sets the number of entries in the MSI-X table.
Table offset	0x0000000000000000	Sets the read-only base address of the MSI-X table. The low-order 3 bits are automatically set to 0.
Table BAR indicator	0	Specifies which one of a function's base address registers, located beginning at 0x10 in the Configuration Space, maps the MSI-X table into memory space. This field is read-only.
Pending bit array (PBA) offset	0x0000000000000000	Specifies the offset from the address stored in one of the function's base address registers the points to the base of the MSI-X PBA. This field is read-only.
PBA BAR indicator	0	Indicates which of a function's base address registers, located beginning at 0x10 in the Configuration Space, maps the function's MSI-X PBA into memory space. This field is read-only.
PCIe0 PF2 VF MSI-X		
Enable VF MSI-X	Enabled	Enables or disables the MSI-X capability.
Table size	4	Sets the number of entries in the MSI-X table.
Table offset	0x0000000000000000	Sets the read-only base address of the MSI-X table. The low-order 3 bits are automatically set to 0.
Table BAR indicator	0	Specifies which one of a function's base address registers, located beginning at 0x10 in the Configuration Space, maps the MSI-X table into memory space. This field is read-only.

continued...

Parameter	Default Setting	Parameter Description
Pending bit array (PBA) offset	0x0000000000000000	Specifies the offset from the address stored in one of the function's base address registers that points to the base of the MSI-X PBA. This field is read-only.
PBA BAR indicator	0	Indicates which of a function's base address registers, located beginning at 0x10 in the Configuration Space, maps the function's MSI-X PBA into memory space. This field is read-only.
PCIe0 PF3 VF MSI-X		
Enable VF MSI-X	Enabled	Enables or disables the MSI-X capability.
Table size	4	Sets the number of entries in the MSI-X table.
Table offset	0x0000000000000000	Sets the read-only base address of the MSI-X table. The low-order 3 bits are automatically set to 0.
Table BAR indicator	0	Specifies which one of a function's base address registers, located beginning at 0x10 in the Configuration Space, maps the MSI-X table into memory space. This field is read-only.
Pending bit array (PBA) offset	0x0000000000000000	Specifies the offset from the address stored in one of the function's base address registers that points to the base of the MSI-X PBA. This field is read-only.
PBA BAR indicator	0	Indicates which of a function's base address registers, located beginning at 0x10 in the Configuration Space, maps the function's MSI-X PBA into memory space. This field is read-only.
PCIe0 PASID		
PCIe0 PF0 PASID		
PCIE0 PF0 Enable PASID	Enabled	PASID (Process Address Space ID) Optional feature which allows a single endpoint to be shared by multiple processes by providing each a virtual 64-bit address space.
PCIe0 PF0 Enable Execute Permission Support	Disabled	Enables or disables PASID Execute Permission Support for PCIe0 PF0.
PCIe0 PF0 Enable Privileged Mode Support	Disabled	Enables or disables PASID Privileged Mode Support for PCIe0 PF0.
PCIe0 PF0 Max PASID Width	0	Sets the Max PASID Width for PCIe0 PF0.
PCIe0 PF1 PASID		
PCIE0 PF1 Enable PASID	Enabled	PASID (Process Address Space ID) Optional feature which allows a single endpoint to be shared by multiple processes by providing each a virtual 64-bit address space.
PCIe0 PF1 Enable Execute Permission Support	Disabled	Enables or disables PASID Execute Permission Support for PCIe0 PF1.

continued...

Parameter	Default Setting	Parameter Description
PCIe0 PF1 Enable Privileged Mode Support	Disabled	Enables or disables PASID Privileged Mode Support for PCIe0 PF1.
PCIe0 PF1 Max PASID Width	0	Sets the Max PASID Width for PCIe0 PF1.
PCIe0 PF2 PASID		
PCIE0 PF2 Enable PASID	Enabled	PASID (Process Address Space ID) Optional feature which allows a single endpoint to be shared by multiple processes by providing each a virtual 64-bit address space.
PCIe0 PF2 Enable Execute Permission Support	Disabled	Enables or disables PASID Execute Permission Support for PCIe0 PF2.
PCIe0 PF2 Enable Privileged Mode Support	Disabled	Enables or disables PASID Privileged Mode Support for PCIe0 PF2.
PCIe0 PF2 Max PASID Width	0	Sets the Max PASID Width for PCIe0 PF2.
PCIe0 PF3 PASID		
PCIE0 PF3 Enable PASID	Enabled	PASID (Process Address Space ID) Optional feature which allows a single endpoint to be shared by multiple processes by providing each a virtual 64-bit address space.
PCIe0 PF3 Enable Execute Permission Support	Disabled	Enables or disables PASID Execute Permission Support for PCIe0 PF3.
PCIe0 PF3 Enable Privileged Mode Support	Disabled	Enables or disables PASID Privileged Mode Support for PCIe0 PF3.
PCIe0 PF3 Max PASID Width	0	Sets the Max PASID Width for PCIe0 PF3.
PCIe0 DEV SER		
Enable Device Serial Number Capability	Disabled	Enables Device Serial Number Capability (DEVSER) optional extended capability is a 64-bit value that is unique for any given PCIe device.
Device Serial Number (DW1)	Disabled	Sets the lower 32 bits of IEEE 64-bit Device Serial Number (DW1).
Device Serial Number (DW2)	Disabled	Sets the upper 32 bits of IEEE 64-bit Device Serial Number (DW2).
PCIe0 PRS		
PCIe0 PF0 PRS		
PF0 Enable PRS	Disabled	Enables PF0 Page Request Service (PRS).
PCIe0 PF1 PRS		
PF1 Enable PRS	Disabled	Enables PF1 Page Request Service (PRS).
PCIe0 PF2 PRS		
PF2 Enable PRS	Disabled	Enables PF2 Page Request Service (PRS).

continued...

Parameter	Default Setting	Parameter Description
PCIe0 PF3 PRS		
PF3 Enable PRS	Disabled	Enables PF3 Page Request Service (PRS).
PCIe0 Power Management		
Endpoint L0s acceptable latency	Maximum of 64ns	Sets the read-only value of the endpoint L0s acceptable latency field of the Device Capabilities register. This value should be based on the latency that the application layer can tolerate. This setting is disabled for root ports.
Endpoint L1s acceptable latency	Maximum of 1 us	Sets the acceptable latency that an endpoint can withstand in the transition from the L1 to L0 state. It is an indirect measure of the endpoint internal buffering. This setting is disabled for root ports.
PCIe0 VSEC		
Vendor Specific Extended Capability	Disabled	Enables Vendor Specific Extended Capability (VSEC). *Note:* Please enable Configuration Intercept Interface (CII) when using Vendor Specific Extended Capability
User ID register from the Vendor Specific Extended Capability	0x00000000	Sets the read-only value of the 16-bit User ID register from the Vendor Specific Extended Capability
Drops Vendor Type0 Messages	Disabled	When selected, received Vendor MSG Type0 will be dropped as an Unsupported Request(UR). Otherwise, received Vendor MSG Type0 will not be dropped, but visible on RX AVST interface. This option is not applicable for TLP Bypass mode. In TLP Bypass mode, received Vendor MSG Type0 will always be visible on RX AVST interface.
Drops Vendor Type1 Messages	Disabled	When selected, received Vendor MSG Type1 will be dropped silently. Otherwise, received Vendor MSG Type1 will not be dropped, but visible on RX AVST interface. This option is not applicable for TLP Bypass mode. In TLP Bypass mode, received Vendor MSG Type1 will always be visible on RX AVST interface.
Note: Please enable PCIe0 Configuration Intercept Interface (CII) when using PCIe0 Vendor Specific Extended Capability.		
PCIe0 ATS		
PCIe0 ATS for Physical Functions		
PCIe0 ATS for PF0		
Enable Address Translation Services (ATS)	Enabled	When Address Translation Services (ATS) is enabled, senders can request and cache translated addresses using the RP memory space for later use.
PCIe0 ATS for PF1		

continued...

Parameter	Default Setting	Parameter Description
Enable Address Translation Services (ATS)	Enabled	When Address Translation Services (ATS) is enabled, senders can request and cache translated addresses using the RP memory space for later use.
PCIe0 ATS for PF2		
Enable Address Translation Services (ATS)	Enabled	When Address Translation Services (ATS) is enabled, senders can request and cache translated addresses using the RP memory space for later use.
PCIe0 ATS for PF3		
Enable Address Translation Services (ATS)	Enabled	When Address Translation Services (ATS) is enabled, senders can request and cache translated addresses using the RP memory space for later use.
PCIe0 ATS for Virtual Functions		
PCIe0 ATS for PF0 VF		
Enable Address Translation Services (ATS)	Enabled	When Address Translation Services (ATS) is enabled, senders can request and cache translated addresses using the RP memory space for later use.
PCIe0 ATS for PF1 VF		
Enable Address Translation Services (ATS)	Enabled	When Address Translation Services (ATS) is enabled, senders can request and cache translated addresses using the RP memory space for later use.
PCIe0 ATS for PF2 VF		
Enable Address Translation Services (ATS)	Enabled	When Address Translation Services (ATS) is enabled, senders can request and cache translated addresses using the RP memory space for later use.
PCIe0 ATS for PF3 VF		
Enable Address Translation Services (ATS)	Enabled	When Address Translation Services (ATS) is enabled, senders can request and cache translated addresses using the RP memory space for later use.
PCIe0 TPH		
PCIe0 TPH for Physical Functions		
PCIe0 TPH for PF0		
Enable TLP Processing Hints (TPH)	Enabled	Using TLP Processing Hints (TPH) may improve latency and traffic congestion.
PCIe0 TPH for PF1		
Enable TLP Processing Hints (TPH)	Enabled	Using TLP Processing Hints (TPH) may improve latency and traffic congestion.
PCIe0 TPH for PF2		
Enable TLP Processing Hints (TPH)	Enabled	Using TLP Processing Hints (TPH) may improve latency and traffic congestion.
PCIe0 TPH for PF3		

continued...

Parameter	Default Setting	Parameter Description
Enable TLP Processing Hints (TPH)	Enabled	Using TLP Processing Hints (TPH) may improve latency and traffic congestion.
PCIe0 TPH for Virtual Functions		
PCIe0 TPH for PF0 VF		
Enable TLP Processing Hints (TPH)	Enabled	Using TLP Processing Hints (TPH) may improve latency and traffic congestion.
PCIe0 TPH for PF1 VF		
Enable TLP Processing Hints (TPH)	Enabled	Using TLP Processing Hints (TPH) may improve latency and traffic congestion.
PCIe0 TPH for PF2 VF		
Enable TLP Processing Hints (TPH)	Enabled	Using TLP Processing Hints (TPH) may improve latency and traffic congestion.
PCIe0 TPH for PF3 VF		
Enable TLP Processing Hints (TPH)	Enabled	Using TLP Processing Hints (TPH) may improve latency and traffic congestion.
PCIe0 ACS Capabilities		
PCIe0 ACS for Physical Functions		
PCIe0 ACS for PF0		
Enable Access Control Service (ACS)	Disabled	ACS defines a set of control points within a PCI Express topology to determine whether a TLP is to be routed normally, blocked, or redirected.
Enable ACS P2P Traffic Support	Disabled	Indicates if the component supports Peer to Peer Traffic.
Enable ACS P2P Egress Control	Disabled	Indicates if the component implements ACS P2P Egress Control.
PCIe0 ACS for PF1		
Enable Access Control Service (ACS)	Disabled	ACS defines a set of control points within a PCI Express topology to determine whether a TLP is to be routed normally, blocked, or redirected.
Enable ACS P2P Traffic Support	Disabled	Indicates if the component supports Peer to Peer Traffic.
Enable ACS P2P Egress Control	Disabled	Indicates if the component implements ACS P2P Egress Control.
PCIe0 ACS for PF2		
Enable Access Control Service (ACS)	Disabled	ACS defines a set of control points within a PCI Express topology to determine whether a TLP is to be routed normally, blocked, or redirected.
Enable ACS P2P Traffic Support	Disabled	Indicates if the component supports Peer to Peer Traffic.
Enable ACS P2P Egress Control	Disabled	Indicates if the component implements ACS P2P Egress Control.
PCIe0 ACS for PF3		

continued...

Parameter	Default Setting	Parameter Description
Enable Access Control Service (ACS)	Disabled	ACS defines a set of control points within a PCI Express topology to determine whether a TLP is to be routed normally, blocked, or redirected.
Enable ACS P2P Traffic Support	Disabled	Indicates if the component supports Peer to Peer Traffic.
Enable ACS P2P Egress Control	Disabled	Indicates if the component implements ACS P2P Egress Control.
PCIe0 ACS for Virtual Functions		
PCIe0 ACS for PF0 VF		
Enable Access Control Service (ACS)	Disabled	ACS defines a set of control points within a PCI Express topology to determine whether a TLP is to be routed normally, blocked, or redirected.
PCIe0 ACS for PF1 VF		
Enable Access Control Service (ACS)	Disabled	ACS defines a set of control points within a PCI Express topology to determine whether a TLP is to be routed normally, blocked, or redirected.
PCIe0 ACS for PF2 VF		
Enable Access Control Service (ACS)	Disabled	ACS defines a set of control points within a PCI Express topology to determine whether a TLP is to be routed normally, blocked, or redirected.
PCIe0 ACS for PF3 VF		
Enable Access Control Service (ACS)	Disabled	ACS defines a set of control points within a PCI Express topology to determine whether a TLP is to be routed normally, blocked, or redirected.
PCIe0 VIRTIO		
Enable VIRTIO support	Enabled	If set, enables VIRTIO Capabilities for PFs and VFs.
PCIe0 PF0 VIRTIO STRUCTURES		
Enable VIRTIO Capabilities for PF0	Disabled	Exposes VIRTIO Capabilities for VIRTIO Capable Devices.
Enable Device Specific Capability for PF0	Enabled	Enables Device Specific Capability for VIRTIO Device on PF0.
PCIe0 PF0 COMMON CONFIGURATION STRUCTURE		
BAR Indicator	Disabled	Indicates BAR holding the Common Configuration Structure.
Offset within BAR	Disabled	Indicates starting position of Common Config Structure in given BAR.
Structure Length in Bytes	Disabled	Indicates length of Common Config Structure.
PCIe0 PF0 NOTIFICATION STRUCTURE		
BAR Indicator	Disabled	Indicates BAR holding the Notification Structure.

continued...

Parameter	Default Setting	Parameter Description
Offset within BAR	Disabled	Indicates starting position of Notification Structure in given BAR.
Structure Length in Bytes	Disabled	Indicates length of Notification Structure.
Notify Off Multiplier	Disabled	Indicates Multiplier for queue_notify_off.
PCIe0 PF0 ISR STATUS STRUCTURE		
BAR Indicator	Disabled	Indicates BAR holding the ISR STATUS Structure.
Offset within BAR	Disabled	Indicates starting position of ISR STATUS Structure in given BAR.
Structure Length in Bytes	Disabled	Indicates length of ISR STATUS Structure.
PCIe0 PF0 PCI CONFIGURATION ACCESS STRUCTURE		
BAR Indicator	Disabled	Indicates BAR holding the PCI Configuration Access Structure.
Offset within BAR	Disabled	Indicates starting position of PCI Configuration Access Structure in given BAR.
Structure Length in Bytes	Disabled	Indicates length of PCI Configuration Access Structure.
Note: BAR Indicator, Offset within BAR and Structure Length are 0 by default.		
PCIe0 PF1 VIRTIO STRUCTURES		
Enable VIRTIO Capabilities for PF1	Enabled	Exposes VIRTIO Capabilities for VIRTIO Capable Devices.
Enable Device Specific Capability for PF1	Enabled	Enables Device Specific Capability for VIRTIO Device on PF1.
PCIe0 PF1 COMMON CONFIGURATION STRUCTURE		
BAR Indicator	Disabled	Indicates BAR holding the Common Configuration Structure.
Offset within BAR	Disabled	Indicates starting position of Common Configuration Structure in given BAR.
Structure Length in Bytes	Disabled	Indicates length of Common Configuration Structure.
PCIe0 PF1 NOTIFICATION STRUCTURE		
BAR Indicator	Disabled	Indicates BAR holding the Notification Structure.
Offset within BAR	Disabled	Indicates starting position of Notification Structure in given BAR.
Structure Length in Bytes	Disabled	Indicates length of Notification Structure.
Notify Off Multiplier	Disabled	Indicates Multiplier for queue_notify_off.
PCIe0 PF1 ISR STATUS STRUCTURE		
BAR Indicator	Disabled	Indicates BAR holding the ISR STATUS Structure.

continued...

Parameter	Default Setting	Parameter Description
Offset within BAR	Disabled	Indicates starting position of ISR STATUS Structure in given BAR.
Structure Length in Bytes	Disabled	Indicates length of ISR STATUS Structure.
PCIe0 PF1 PCI CONFIGURATION ACCESS STRUCTURE		
BAR Indicator	Disabled	Indicates BAR holding the PCI Configuration Access Structure.
Offset within BAR	Disabled	Indicates starting position of PCI Configuration Access Structure in given BAR.
Structure Length in Bytes	Disabled	Indicates length of PCI Configuration Access Structure.
Note: BAR Indicator, Offset within BAR and Structure Length are 0 by default.		
PCIe0 PF2 VIRTIO STRUCTURES		
Enable VIRTIO Capabilities for PF2	Enabled	Exposes VIRTIO Capabilities for VIRTIO Capable Devices.
Enable Device Specific Capability for PF2	Enabled	Enables Device Specific Capability for VIRTIO Device on PF2.
PCIe0 PF2 COMMON CONFIGURATION STRUCTURE		
BAR Indicator	0	Indicates BAR holding the Common Configuration Structure.
Offset within BAR	0x00000000	Indicates starting position of Common Configuration Structure in given BAR.
Structure Length in Bytes	0x00000000	Indicates length of Common Configuration Structure.
PCIe0 PF2 NOTIFICATION STRUCTURE		
BAR Indicator	0	Indicates BAR holding the Notification Structure.
Offset within BAR	0x00000000	Indicates starting position of Notification Structure in given BAR.
Structure Length in Bytes	0x00000000	Indicates length of Notification Structure.
Notify Off Multiplier	0x00000000	Indicates Multiplier for queue_notify_off.
PCIe0 PF2 ISR STATUS STRUCTURE		
BAR Indicator	0	Indicates BAR holding the ISR STATUS Structure.
Offset within BAR	0x00000000	Indicates starting position of ISR STATUS Structure in given BAR.
Structure Length in Bytes	0x00000000	Indicates length of ISR STATUS Structure.
PCIe0 PF2 DEVICE SPECIFIC STRUCTURE		
BAR Indicator	0	Indicates BAR holding the Device Specific Structure.
Offset within BAR	0x00000000	Indicates starting position of Device Specific Structure in given BAR.

continued...

Parameter	Default Setting	Parameter Description
Structure Length in Bytes	0x00000000	Indicates length of Device Specific Structure.
PCIe0 PF2 PCI CONFIGURATION ACCESS STRUCTURE		
BAR Indicator	0	Indicates BAR holding the PCI Configuration Access Structure.
Offset within BAR	0x00000000	Indicates starting position of PCI Configuration Access Structure in given BAR.
Structure Length in Bytes	0x00000000	Indicates length of PCI Configuration Access Structure.
Note: BAR Indicator, Offset within BAR and Structure Length are 0 by default.		
PCIe0 PF3 VIRTIO STRUCTURES		
Enable VIRTIO Capabilities for PF3	Enabled	Exposes VIRTIO Capabilities for VIRTIO Capable Devices.
Enable Device Specific Capability for PF3	Enabled	Enables Device Specific Capability for VIRTIO Device on PF3.
PCIe0 PF3 COMMON CONFIGURATION STRUCTURE		
BAR Indicator	0	Indicates BAR holding the Common Configuration Structure.
Offset within BAR	0x00000000	Indicates starting position of Common Configuration Structure in given BAR.
Structure Length in Bytes	0x00000000	Indicates length of Common Configuration Structure.
PCIe0 PF3 NOTIFICATION STRUCTURE		
BAR Indicator	0	Indicates BAR holding the Notification Structure.
Offset within BAR	0x00000000	Indicates starting position of Notification Structure in given BAR.
Structure Length in Bytes	0x00000000	Indicates length of Notification Structure.
Notify Off Multiplier	0x00000000	Indicates Multiplier for queue_notify_off.
PCIe0 PF3 ISR STATUS STRUCTURE		
BAR Indicator	0	Indicates BAR holding the ISR STATUS Structure.
Offset within BAR	0x00000000	Indicates starting position of ISR STATUS Structure in given BAR.
Structure Length in Bytes	0x00000000	Indicates length of ISR STATUS Structure.
PCIe0 PF3 DEVICE SPECIFIC STRUCTURE		
BAR Indicator	0	Indicates BAR holding the Device Specific Structure.
Offset within BAR	0x00000000	Indicates starting position of Device Specific Structure in given BAR.
Structure Length in Bytes	0x00000000	Indicates length of Device Specific Structure.

continued...

Parameter	Default Setting	Parameter Description
PCIe0 PF3 PCI CONFIGURATION ACCESS STRUCTURE		
BAR Indicator	0	Indicates BAR holding the PCI Configuration Access Structure.
Offset within BAR	0x00000000	Indicates starting position of PCI Configuration Access Structure in given BAR.
Structure Length in Bytes	0x00000000	Indicates length of PCI Configuration Access Structure.
Note: BAR Indicator, Offset within BAR and Structure Length are 0 by default.		
PCIe0 PF0 VFs VIRTIO STRUCTURES		
Enable VIRTIO Capabilities for PF0 VFs	Enabled	Exposes VIRTIO Capabilities for VIRTIO Capable Devices.
Enable Device Specific Capability for PF0 VFs	Enabled	Enables Device Specific Capability for VIRTIO Device on PF1 VFs.
PCIe0 PF0 VFs COMMON CONFIGURATION STRUCTURE		
BAR Indicator	0	Indicates BAR holding the Common Configuration Structure.
Offset within BAR	0x00000000	Indicates starting position of Common Configuration Structure in given BAR.
Structure Length in Bytes	0x00000000	Indicates length of Common Configuration Structure.
PCIe0 PF0 VFs NOTIFICATION STRUCTURE		
BAR Indicator	0	Indicates BAR holding the Notification Structure.
Offset within BAR	0x00000000	Indicates starting position of Notification Structure in given BAR.
Structure Length in Bytes	0x00000000	Indicates length of Notification Structure.
Notify Off Multiplier	0x00000000	Indicates Multiplier for queue_notify_off.
PCIe0 PF0 VFs ISR STATUS STRUCTURE		
BAR Indicator	0	Indicates BAR holding the ISR STATUS Structure.
Offset within BAR	0x00000000	Indicates starting position of ISR STATUS Structure in given BAR.
Structure Length in Bytes	0x00000000	Indicates length of ISR STATUS Structure.
PCIe0 PF0 VFs DEVICE SPECIFIC STRUCTURE		
BAR Indicator	0	Indicates BAR holding the Device Specific Structure.
Offset within BAR	0x00000000	Indicates starting position of Device Specific Structure in given BAR.
Structure Length in Bytes	0x00000000	Indicates length of Device Specific Structure.
PCIe0 PF0 VFs PCI CONFIGURATION ACCESS STRUCTURE		

continued...

Parameter	Default Setting	Parameter Description
BAR Indicator	0	Indicates BAR holding the PCI Configuration Access Structure.
Offset within BAR	0x00000000	Indicates starting position of PCI Configuration Access Structure in given BAR.
Structure Length in Bytes	0x00000000	Indicates length of PCI Configuration Access Structure.
Note: BAR Indicator, Offset within BAR and Structure Length are 0 by default.		
PCIe0 PF1 VFs VIRTIO STRUCTURES		
Enable VIRTIO Capabilities for PF1	Enabled	Exposes VIRTIO Capabilities for VIRTIO Capable Devices.
Enable Device Specific Capability for PF1	Enabled	Enables Device Specific Capability for VIRTIO Device on PF2 VFs.
PCIe0 PF1 VFs COMMON CONFIGURATION STRUCTURE		
BAR Indicator	0	Indicates BAR holding the Common Configuration Structure.
Offset within BAR	0x00000000	Indicates starting position of Common Configuration Structure in given BAR.
Structure Length in Bytes	0x00000000	Indicates length of Common Configuration Structure.
PCIe0 PF1 VFs NOTIFICATION STRUCTURE		
BAR Indicator	0	Indicates BAR holding the Notification Structure.
Offset within BAR	0x00000000	Indicates starting position of Notification Structure in given BAR.
Structure Length in Bytes	0x00000000	Indicates length of Notification Structure.
Notify Off Multiplier	0x00000000	Indicates Multiplier for queue_notify_off.
PCIe0 PF1 VFs ISR STATUS STRUCTURE		
BAR Indicator	0	Indicates BAR holding the ISR STATUS Structure.
Offset within BAR	0x00000000	Indicates starting position of ISR STATUS Structure in given BAR.
Structure Length in Bytes	0x00000000	Indicates length of ISR STATUS Structure.
PCIe0 PF1 VFs DEVICE SPECIFIC STRUCTURE		
BAR Indicator	0	Indicates BAR holding the Device Specific Structure.
Offset within BAR	0x00000000	Indicates starting position of Device Specific Structure in given BAR.
Structure Length in Bytes	0x00000000	Indicates length of Device Specific Structure.
PCIe0 PF1 VFs PCI CONFIGURATION ACCESS STRUCTURE		
BAR Indicator	0	Indicates BAR holding the PCI Configuration Access Structure.

continued...

Parameter	Default Setting	Parameter Description
Offset within BAR	0x00000000	Indicates starting position of PCI Configuration Access Structure in given BAR.
Structure Length in Bytes	0x00000000	Indicates length of PCI Configuration Access Structure.
Note: BAR Indicator, Offset within BAR and Structure Length are 0 by default.		
PCIe0 PF2 VFs VIRTIO STRUCTURES		
Enable VIRTIO Capabilities for PF2	Enabled	Exposes VIRTIO Capabilities for VIRTIO Capable Devices.
Enable Device Specific Capability for PF2	Enabled	Enables Device Specific Capability for VIRTIO Device on PF3 VFs.
PCIe0 PF2 VFs COMMON CONFIGURATION STRUCTURE		
BAR Indicator	0	Indicates BAR holding the Common Configuration Structure.
Offset within BAR	0x00000000	Indicates starting position of Common Configuration Structure in given BAR.
Structure Length in Bytes	0x00000000	Indicates length of Common Configuration Structure.
PCIe0 PF2 VFs NOTIFICATION STRUCTURE		
BAR Indicator	0	Indicates BAR holding the Notification Structure.
Offset within BAR	0x00000000	Indicates starting position of Notification Structure in given BAR.
Structure Length in Bytes	0x00000000	Indicates length of Notification Structure.
Notify Off Multiplier	0x00000000	Indicates Multiplier for queue_notify_off.
PCIe0 PF2 VFs ISR STATUS STRUCTURE		
BAR Indicator	0	Indicates BAR holding the ISR STATUS Structure.
Offset within BAR	0x00000000	Indicates starting position of ISR STATUS Structure in given BAR.
Structure Length in Bytes	0x00000000	Indicates length of ISR STATUS Structure.
PCIe0 PF2 VFs DEVICE SPECIFIC STRUCTURE		
BAR Indicator	0	Indicates BAR holding the Device Specific Structure.
Offset within BAR	0x00000000	Indicates starting position of Device Specific Structure in given BAR.
Structure Length in Bytes	0x00000000	Indicates length of Device Specific Structure.
PCIe0 PF2 VFs PCI CONFIGURATION ACCESS STRUCTURE		
BAR Indicator	0	Indicates BAR holding the PCI Configuration Access Structure.

continued...

Parameter	Default Setting	Parameter Description
Offset within BAR	0x00000000	Indicates starting position of PCI Configuration Access Structure in given BAR.
Structure Length in Bytes	0x00000000	Indicates length of PCI Configuration Access Structure.
Note: BAR Indicator, Offset within BAR and Structure Length are 0 by default.		
PCIe0 PF3 VFs VIRTIO STRUCTURES		
Enable VIRTIO Capabilities for PF3	Enabled	Exposes VIRTIO Capabilities for VIRTIO Capable Devices.
Enable Device Specific Capability for PF3	Enabled	Enables Device Specific Capability for VIRTIO Device on PF3 VFs.
PCIe0 PF3 VFs COMMON CONFIGURATION STRUCTURE		
BAR Indicator	0	Indicates BAR holding the Common Configuration Structure.
Offset within BAR	0x00000000	Indicates starting position of Common Configuration Structure in given BAR.
Structure Length in Bytes	0x00000000	Indicates length of Common Configuration Structure.
PCIe0 PF3 VFs NOTIFICATION STRUCTURE		
BAR Indicator	0	Indicates BAR holding the Notification Structure.
Offset within BAR	0x00000000	Indicates starting position of Notification Structure in given BAR.
Structure Length in Bytes	0x00000000	Indicates length of Notification Structure.
Notify Off Multiplier	0x00000000	Indicates Multiplier for queue_notify_off.
PCIe0 PF3 VFs ISR STATUS STRUCTURE		
BAR Indicator	0	Indicates BAR holding the ISR STATUS Structure.
Offset within BAR	0x00000000	Indicates starting position of ISR STATUS Structure in given BAR.
Structure Length in Bytes	0x00000000	Indicates length of ISR STATUS Structure.
PCIe0 PF3 VFs DEVICE SPECIFIC STRUCTURE		
BAR Indicator	0	Indicates BAR holding the Device Specific Structure.
Offset within BAR	0x00000000	Indicates starting position of Device Specific Structure in given BAR.
Structure Length in Bytes	0x00000000	Indicates length of Device Specific Structure.
PCIe0 PF3 VFs PCI CONFIGURATION ACCESS STRUCTURE		
BAR Indicator	0	Indicates BAR holding the PCI Configuration Access Structure.

continued...

Parameter	Default Setting	Parameter Description
Offset within BAR	0x00000000	Indicates starting position of PCI Configuration Access Structure in given BAR.
Structure Length in Bytes	0x00000000	Indicates length of PCI Configuration Access Structure.
Note: BAR Indicator, Offset within BAR and Structure Length are 0 by default.		

Figure 21. PCIe Interfaces Settings with Extended PCIe Port Settings Disabled

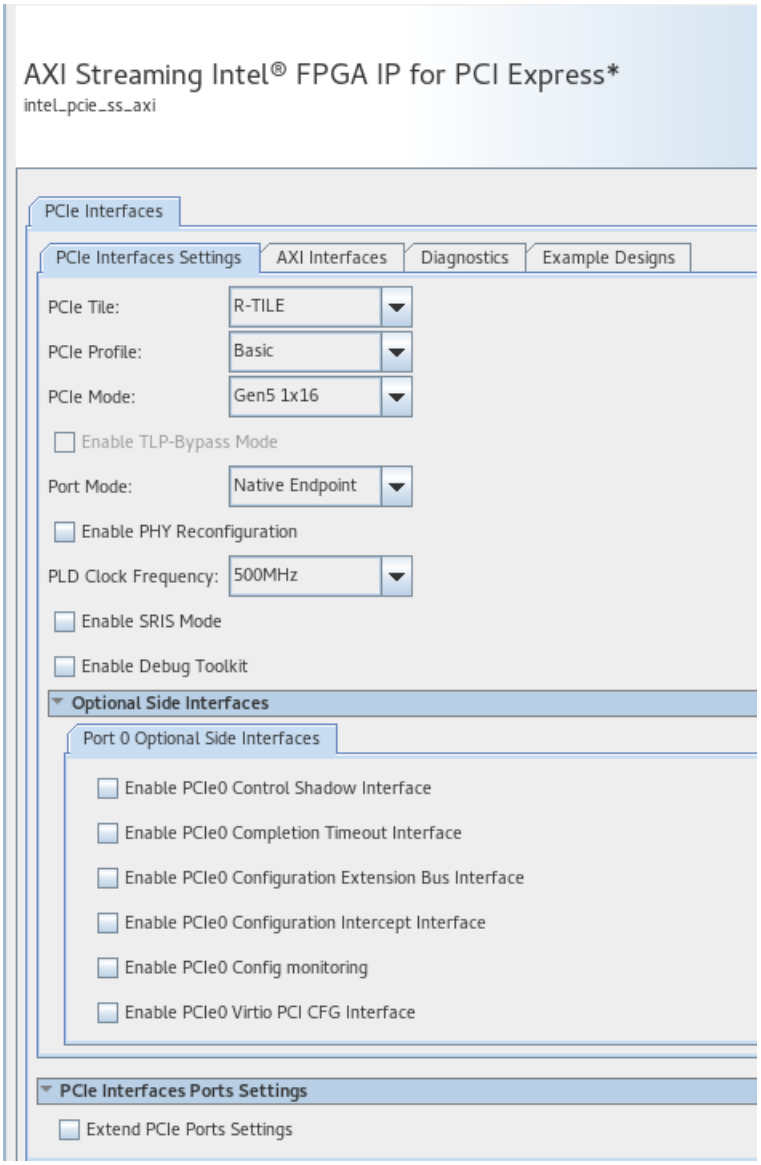


Figure 22. PCIe Interfaces Settings with Extended PCIe Port Settings Enabled

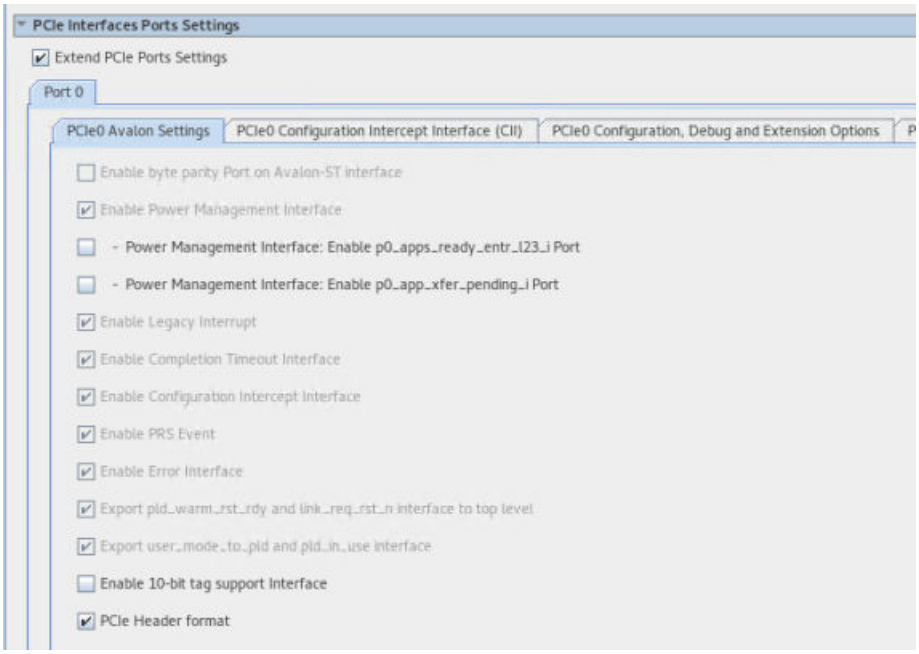


Table 21. AXI Streaming Intel FPGA IP for PCIe Parameters: AXI Interfaces Tab

Parameter	Default Setting	Parameter Description
AXI Streaming Configuration		
AXI Interfaces		
AXI-Lite Clock Frequency (in MHz)	250	Selects the PCIe IP AXI-Lite Operating Clock Frequency.
AXI-ST Clock Frequency (in MHz)	470	Selects the PCIe IP AXI-ST Operating Clock Frequency. 500 (Gen5/4) 470 (Gen5/4) 400 (Gen5/4) 350 (Gen5/4) 250 (Gen3)
AXI-ST Interfaces		
Port 0 AXI-ST Interface 0 Settings		
PCIe0 AXI-ST Interface Data Bus Width (in Byte)	64	Selects application's AXI streaming data bus width. The interface width is defined in terms of number of Bytes. • 32 • 64 • 128
PCIe0 AXI-ST HIP Native mode	Enabled	Available only for R-Tile. When enabled, the IP enables the configuration supported natively by the HIP's AVST interface for interface width, segment size, AXI-ST clock frequency and AXI-ST header interface type combinations. When disabled, you can choose the AXI-ST interface width, segment size, header interface type and clock frequency. *Note:* Only the HIP Native mode is supported in the current release.

continued...

Parameter	Default Setting	Parameter Description
PCIe0 AXI-ST Header scheme	Sideband	Selects the AXI-ST header scheme. The following schemes are available: • In-band: selects an in-band header for the AXI-ST TX, RX interfaces. • Sideband: selects a sideband header for the TX, RX interfaces. *Note:* Only the sideband header scheme is supported in the current release.
PCIe0 AXI-ST number of segments	4,2,1	Selects the number of segments available on the AXI-ST interface. Indicates simple packing when the number of segments equals 1 and compact packing when the number of segments equals 2 or 4.
PCIe0 AXI-ST Segment Size (in Byte)	32	Indicates smallest fragment of Data bus when it is divided into multiple segments. This allows start of packet at different positions on data bus • 32
AXI-Lite Interfaces		
Port 0 AXI-Lite Interface Settings		
PCIe0 AXI-Lite Responder Interface Data Bus Width (in Byte)	4	Selects application's AXI-Lite Responder interface's data bus width. The interface width is defined in terms of number of Bytes • 4 • 8

Figure 23. AXI Interface and AXI-ST Interface Settings Tab

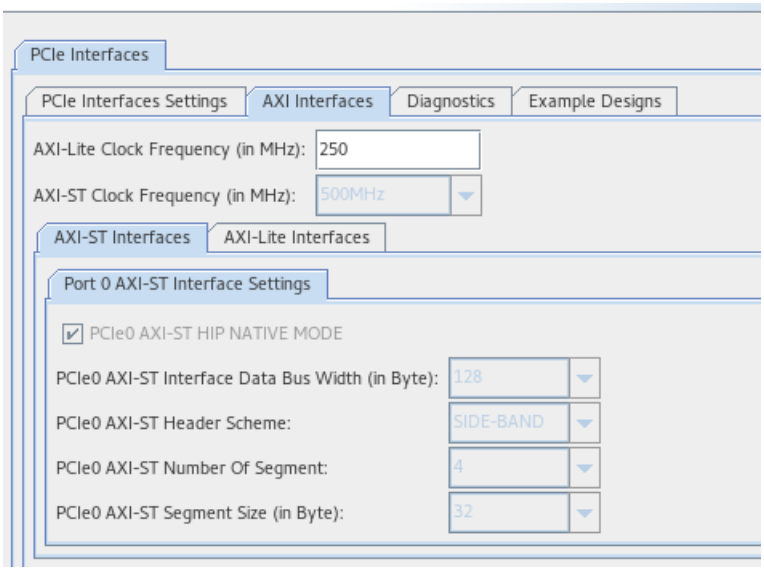


Figure 24. AXI Interface and AXI-Lite Interface Settings Tab

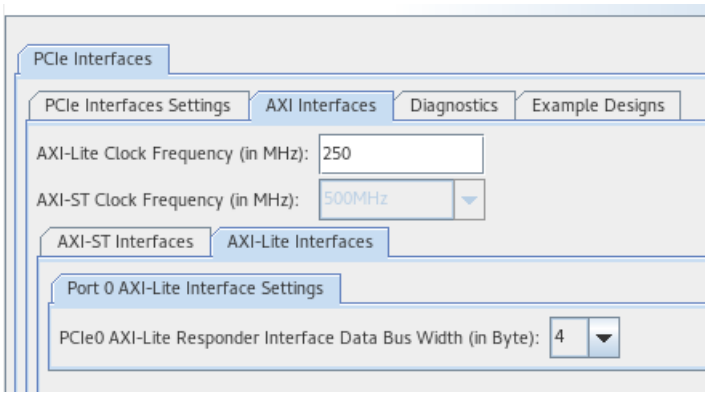


Table 22. AXI Streaming Intel FPGA IP for PCIe Parameters: Diagnostics Tab

Parameter	Default Setting	Parameter Description
Enable HIP Interface Adaptor Debug Monitor	False	Enables the HIP Interface Adaptor Debug Monitor. • True • False
Enable HIP Interface Adaptor Performance Monitor	False	Enables the HIP Interface Adaptor Performance Monitor. • True • False

Figure 25. Diagnostics Tab

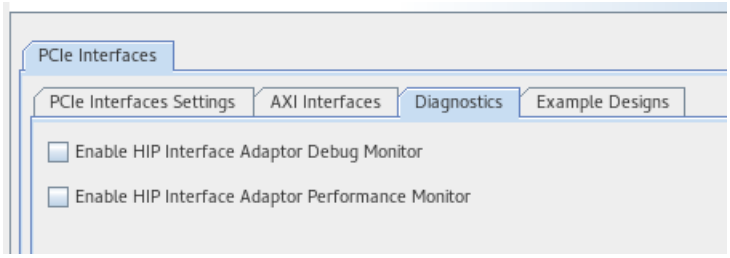
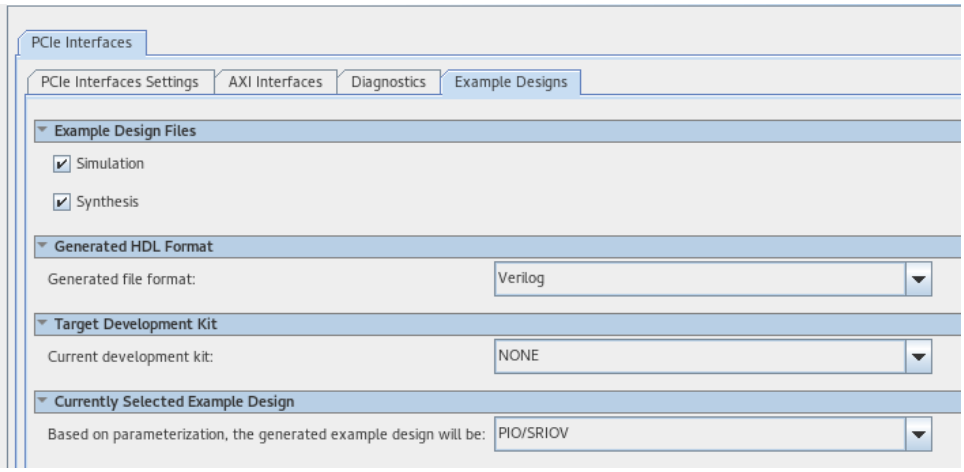


Table 23. AXI Streaming Intel FPGA IP for PCIe Parameters: Example Designs Tab

Parameter	Default Setting	Parameter Description
Simulation	**True**	When checked, all necessary filesets for simulation will be generated. Otherwise, a Platform Designer example design system will be generated.
Synthesis	**True**	When checked, all necessary filesets for synthesis will be generated. Otherwise, a Platform Designer example design system will be generated.
Generated HDL format	**Verilog**	HDL format. Currently, only Verilog is available.
Target development kit	**None** **Intel Agilex 7 I-Series FPGA DevKit DK-DEV-AGI027R1BES**	Provides support for various Development Kits listed. If an Intel FPGA Development Kit is selected, the Target Device used for generation will be the one that matches the device on the Development Kit.
Currently Selected Example Design	**PIO/SRIOV**	Only the PIO design example is available in the current release of Intel Quartus Prime.

Figure 26. Example Designs Tab



6. Interfaces and Signals

This section focuses mainly on the signal interfaces that the AXI Streaming Intel FPGA IP for PCI Express uses to communicate with the Application Layer in the FPGA fabric core. It also briefly covers the Serial Data Interface, which allows the IP to communicate with the link partner across the PCIe link.

6.1. Overview

You can determine each of the interface sections from the prefixes in the signal names.

- p0: x16 core
- p1: x8 core
- p2: x4 core
- p3: x4 core

The AXI Streaming Intel FPGA IP for PCI Express Top-Level Signals figure below shows the top-level signals of this IP. Note that the signal names in the figure will get the appropriate prefix p<n> (where n = 0, 1, 2, 3) depending on which of the supported configurations (1x16, 2x8 or 4x4) the IP is in.

As an example, the ss_app_st_rx_tdata bus can take on the following names:

- In the 1x16 configuration, only the x16 core is active. For Gen5x16, this bus appears as p0_ss_app_st_rx_tdata [1023:0].
- In the 2x8 configuration, both the x16 core and x8 core are active. In this case, this bus is split into p0_ss_app_st_rx_tdata[511:0] and p1_ss_app_st_rx_tdata [511:0] in Gen5.

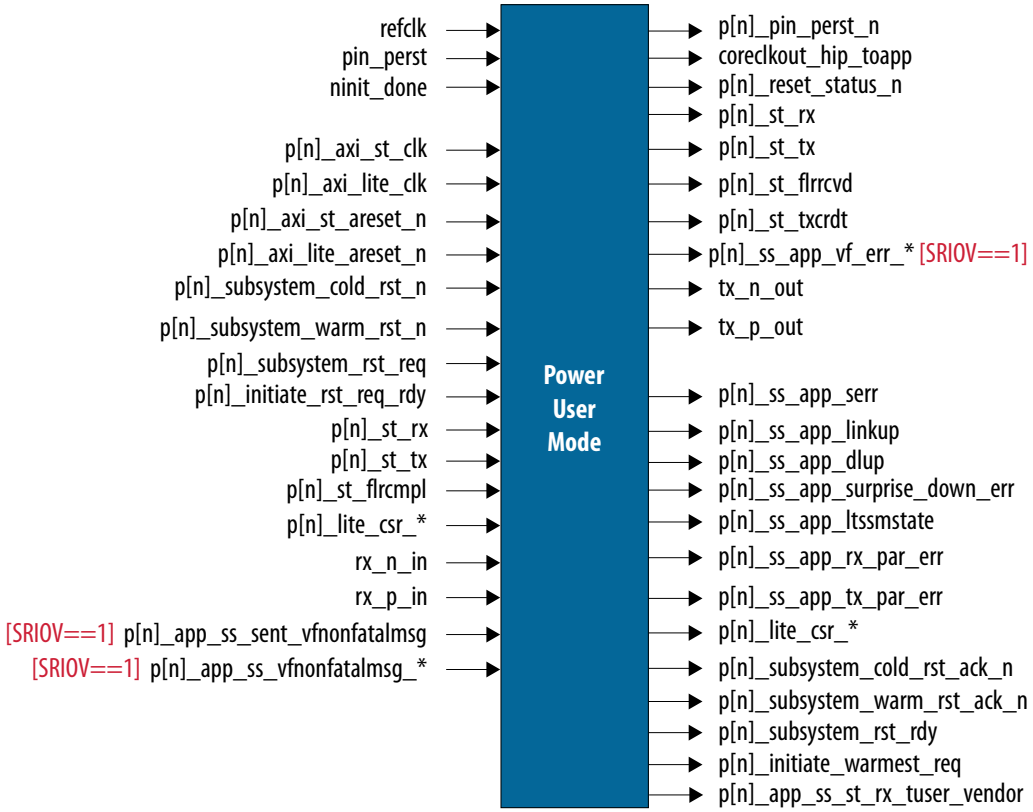
The only cases where the interface signal names do not get the pn prefixes are the interfaces that are common for all the cores, like the PHY reconfiguration interface, clocks and resets. For example, there is only one xcvr_reconfig_clk that is shared by all the cores.

You can enable the PHY reconfiguration interface from the PCIe Interface Settings in the Parameter editor.

Each of the cores has its own AXI-ST interface to the user logic. The number of IP-to-User Logic interfaces exposed to the FPGA fabric are different based on the configuration modes.

For a summary of the IP to FPGA fabric interfaces, refer to Application Packet Interface.

Figure 27. AXI Streaming Intel FPGA IP for PCI Express Top-Level Signals



The following table shows the values of the variable n that is used to define the bus indices for top-level signal busses shown in the top-level block diagram above. The value of this variable changes depending on which configuration is active (1x16, 1x8, 2x8).

Table 24. Variable Used in the Bus Indices

Variable	1 x16 configuration	1 x8 configuration	2 x8 configuration
n	1	1	2
b	16	8	16

6.2. Clocks and Resets

6.2.1. Interface Clock Signals

Table 25. Interface Clock Signals

Signal Name	Direction	EP/RP/BP	Description
refclk[1:0]	I	EP	These are the input reference clocks for the IP core. These clocks must be free-running. For more details on how to connect these clocks, refer to the section Clock Sharing in Bifurcation Modes. EP 100 MHz ± 300 ppm
p<n>_axi_st_clk	I	EP	Global clock signal for AXI -ST interface. All AXI-ST signals are sampled on the rising edge of this clk. This clock drives main data path. The frequency of this clock depends on the mode in which the IP is configured: • In Native HIP mode, this clock uses the Hard IP's coreclkout_hip. • In non-Native HIP mode, the clock frequency can be selectable based on the speed of the link as below: — Gen5: 500/470/400/350/250 MHz (32-, 64-, 128-byte width) — Gen4: 500/470/400/350/250 MHz (32-, 64-, 128-byte width) — Gen3: 300/275/250 MHz (32-, 64-, 128-byte width)
axi_lite_clk	I	EP	The global clock signal for the AXI-Lite interface. All AXI-Lite signals are sampled on the rising edge of p<n>_axi_lite_clk. This clock drives control and status register interfaces in the design. Frequency: 100-250 MHz (250 MHz default)
coreclkout_hip_toapp	O	EP	The coreclkout_hip output of Hard IP drives this clock. Application can use this clock to generate PCIe SS clocks. Gen5: 500 MHz Gen4: 500 MHz Gen3: 250 MHz Gen2/Gen1: Gen1/Gen2 is supported only via link down-training and not natively. Hence, the coreclkout_hip clock frequency depends on the configuration you choose in the IP Parameter Editor. For example, if you choose a Gen3 configuration, the application clock frequency is 250 MHz.
xcvr_reconfig_clk	I	EP	Clock for the PHY reconfiguration interface. This is an Avalon-MM interface. This optional interface is enabled when you turn on the **Enable PHY Reconfiguration** option in the **PCIe Interfaces Settings** tab. This interface is shared among all the cores. 50 MHz - 125 MHz (range) 100 MHz (recommended)

6.2.2. Interface Reset Signals

Table 26. Interface Reset Signals

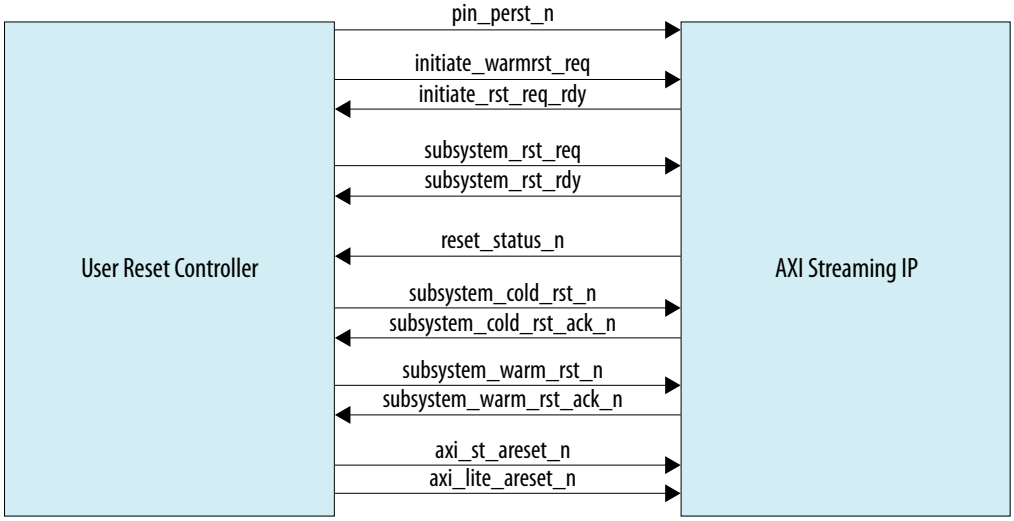
Signal Name	Direction	Type	Description
pin_perst	I	Asynchronous	This is an active-low input to the PCIe Hard IP. It implements the PERST# function defined by the PCIe specification.
p<n>_pin_perst_n where n = 0, 1, 2, 3	O	Asynchronous	This is the PERST output signal from the Hard IP. It is derived from the pin_perst_n input signal.
p<n>_Subsystem_cold_rst_n	I	Could be implemented as synchronous or asynchronous reset.	IP global reset. Resets sticky register bits. Active low.
p<n>_Subsystem_warm_rst_n	I	Could be implemented as synchronous or asynchronous reset.	IP warm reset. Does not reset sticky register bits. Active low.
p<n>_Subsystem_cold_rst_ack_n	O	Asynchronous	Handshake signal. Indicates cold reset action is completed by the IP.
p<n>_Subsystem_warm_rst_ack_n	O	Asynchronous	Handshake signal. Indicates warm reset action is completed by the IP.
p<n>_axi_st_areset_n	I	The reset signal can be asserted asynchronously, but deassertion must be synchronous after the rising edge of axi_st_clk.	AXI-Streaming main datapath reset. Active-LOW reset signal. Used to reset the AXI-ST datapath interface.
p<n>_axi_lite_areset_n	I	The reset signal can be asserted asynchronously, but deassertion must be synchronous after the rising edge of axi_lite_clk.	AXI-Lite reset. Active-LOW reset signal. Used to reset the AXI Lite interface.
p<n>_Subsystem_rst_req	I	Asynchronous	Reset entry indication from Central Reset Sequencer block implemented by the user logic.
p<n>_Subsystem_rst_rdy	O	Asynchronous	Ready signal for reset entry indication from the IP to the Central Reset Sequencer block.
p<n>_initiate_warmrst_req	O	Asynchronous	Warm Reset entry required indication from the IP block to the Central Reset Sequencer in user logic. Initiator block cannot issue new reset entry request until previous reset sequence (entire reset operation) is completed.
p<n>_initiate_rst_req_rdy	I	Asynchronous	Indicates the Central Reset Sequencer block in the user logic has accepted the initiation request and will start issuing resets.
p<n>_reset_status_n	O	Synchronous to coreclkout_hip of Hard IP	Active low signal. When asserted, indicates HARD IP is in reset state. When asserted,

continued...

Signal Name	Direction	Type	Description
			will continue to stay asserted until pin perstn is deasserted and HARD IP is out of reset state. The application logic can use this signal to drive its reset network. The reset_status_n output of HIP drives this signal.

Note: You must implement the user reset sequencer in your application user logic and follow the assertion and deassertion sequence for graceful entry and exit for each of the resets (cold, warm etc). The following figure shows the reset connections between a user-implemented reset controller and the AXI Streaming Intel FPGA IP for PCI Express.

Figure 28. User Reset Controller Connections



The following table indicates the signals/blocks used for each type of reset:

Table 27. Signals and Blocks Used for Reset Type

Reset Type	Signals/Blocks Under Reset
Cold Reset	• Subsystem_cold_rst_n and Subsystem_warm_rst_n will be asserted • Bus resets (AXI-ST/AXI-MM/AXI-Lite) will be asserted • HIP will undergo reset
Warm Reset (e.g., LTSSM Hot reset)	• Subsystem_cold_rst_n will not be asserted • Bus resets (AXI-ST/AXI-MM/AXI-Lite) will be asserted also • HIP will undergo reset also

Expected reset isolation requirement for reset domain crossings are shown in the following table.

Table 28. Reset Isolation Requirement for Reset Domain Crossings

- No: No reset isolation required for Column->Row Reset Domain Crossing
- Yes: Reset isolation required for Column->Row Reset Domain Crossing
- N/A: Not applicable since same Reset Domain Crossing

Column: Source Row: Destination	Cold Reset	HIP Reset	Warm Reset	AXI-ST/MM Reset	AXI-Lite Reset
Cold Reset	N/A	No	Yes	No	No
HIP Reset	No	N/A	No	No	No
Warm Reset	No	No	N/A	No	No
AXI-ST Reset	No	No	No	N/A	No
AXI-Lite Reset	No	No	No	No	N/A

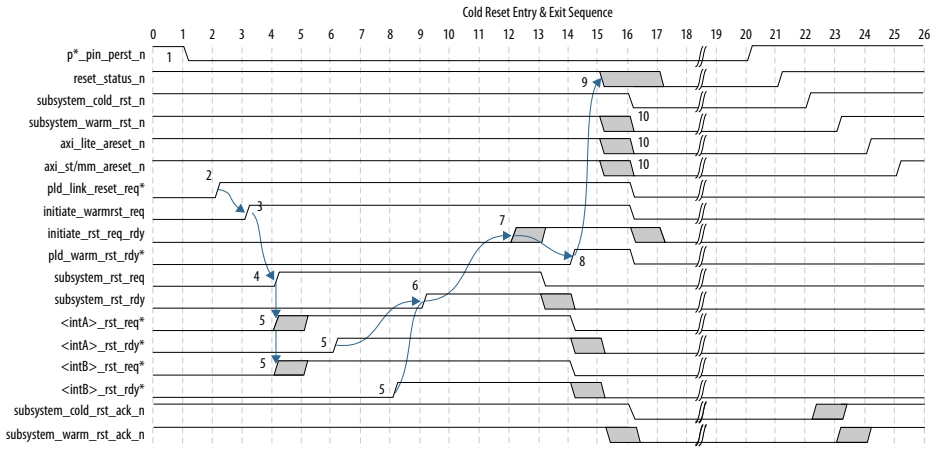
Note: In Endpoint mode, IP warm reset could be asserted without IP cold reset in scenarios such LTSSM Hot Reset.

Cold Reset Entry and Exit Sequence

The following is the sequence for Cold Reset Entry.

1. Cold reset is initiated by the assertion of Hard IP's p*_pin_perst_n.
2. Hard IP asserts pld_link_reset_req to the AXI IP soft logic.
3. The AXI IP notifies the user reset sequencer by asserting initiate_warmrst_req.
4. User reset sequencer then asserts _rst_req.
5. The AXI IP sequences its internal blocks for reset entry (intA_rst_req, intB_rst_req, intA_rst_rdy, intB_rst_rdy, …).
6. The AXI IP asserts Subsystem_rst_rdy to user reset sequencer, indicating the IP's internal blocks are ready for reset.
7. User reset sequencer acknowledges to the AXI IP that it is ready for reset by asserting initiate_rst_req_rdy.
8. The AXI IP soft logic then asserts pld_warm_rst_rdy to Hard IP.
9. Hard IP asserts reset_status_n indicating the application logic needs to be in reset.
10. User reset sequencer asserts Subsystem_cold_rst_n, Subsystem_warm_rst_n and AXI bus resets.

Figure 29. Cold Reset Entry and Exit Sequence Timing Diagram



Note: * indicates the signals between the AXI Streaming Intel FPGA IP for PCI Express soft logic and the Hard IP. These signals are not available to the application logic.

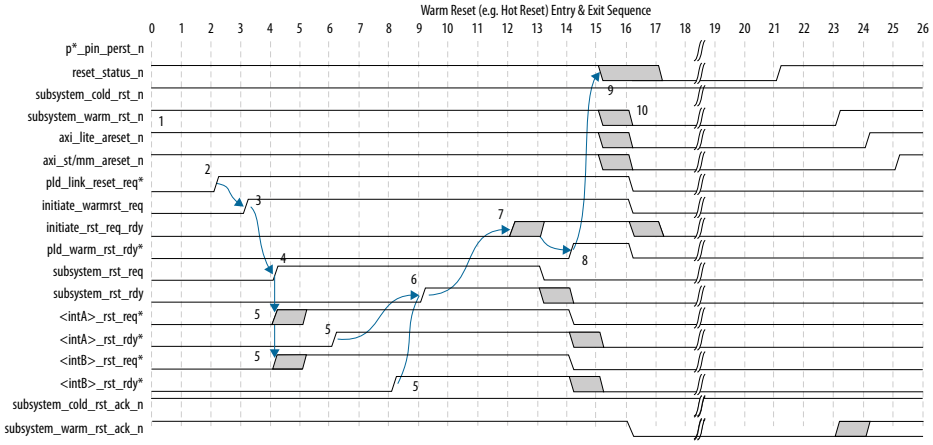
Warm Reset Entry & Exit Sequence

The following is the sequence for Warm Reset Entry.

1. Warm reset is initiated by the assertion of Hard IP event, e.g., Hot reset.
2. Hard IP asserts pld_link_reset_req to the AXI IP soft logic.
3. The AXI IP notifies the user reset sequencer by asserting initiate_warmrst_req.
4. The user reset sequencer asserts Subsystem_rst_req.
5. The AXI IP sequences its internal blocks for reset entry (intA_rst_req, intB_rst_req, intA_rst_rdy, intB_rst_rdy, …).
6. The AXI IP asserts Subsystem_rst_rdy to the user reset sequencer, indicating the IP's internal blocks are ready for reset.
7. The user reset sequencer acknowledges to the IP that it is ready for reset by asserting initiate_rst_req_rdy.
8. The AXI IP soft logic then asserts pld_warm_rst_rdy to Hard IP.
9. Hard IP asserts reset_status_n indicating the application logic needs to be in reset.
10. The user reset sequencer asserts Subsystem_warm_rst_n and AXI bus resets.

Note: Warm Reset flow is similar to Cold Reset flow, with the exception that p*_pin_perst_n and Subsystem_cold_rst_n are not asserted for warm reset

Figure 30. Warm Reset Entry and Exit Sequence Timing Diagram

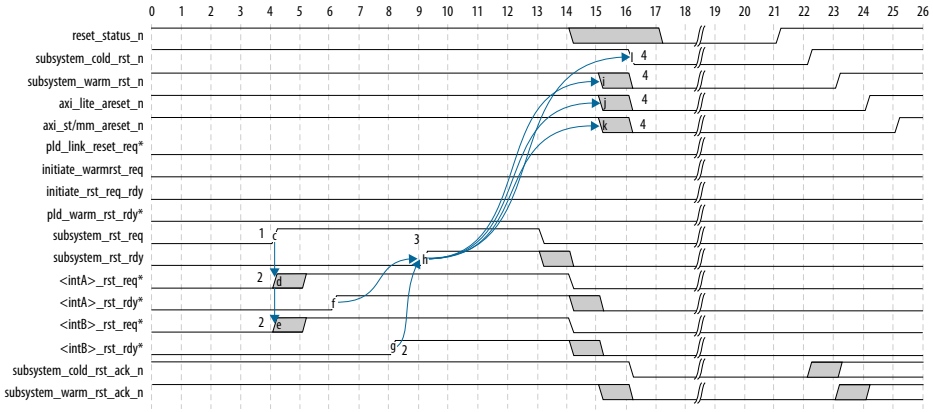


Note: * indicates the signals between the AXI Streaming Intel FPGA IP for PCI Express soft logic and the Hard IP. These signals are not available to the application logic.

User reset sequencer initiated Cold Reset Entry and Exit Sequence

1. Cold reset is initiated by the user reset sequencer by the assertion of the Subsystem_rst_req.
2. The AXI IP sequences its internal blocks for reset entry (intA_rst_req, intB_rst_req, intA_rst_rdy, intB_rst_rdy, …).
3. The AXI IP asserts Subsystem_rst_rdy to user reset sequencer, indicating the IP's internal blocks are ready for reset.
4. The user reset sequencer asserts Subsystem_cold_rst_n, Subsystem_warm_rst_n and AXI bus resets.

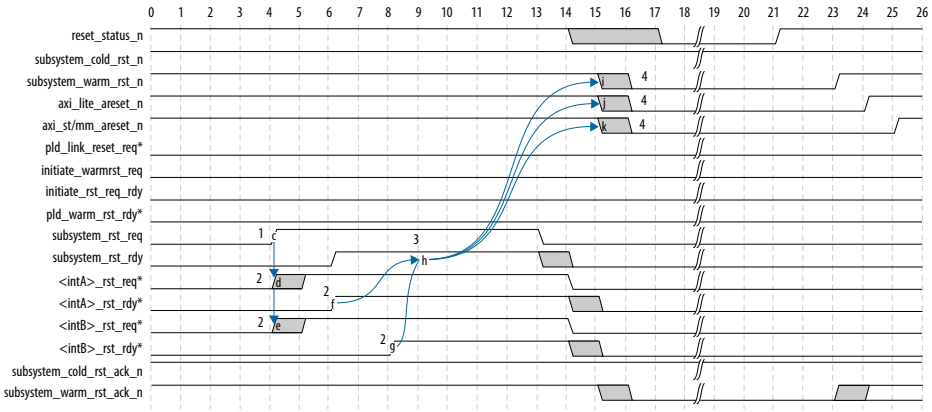
Figure 31. User Reset Sequencer Initiated Cold Reset Entry and Exit Sequence Timing Diagram



Note: * indicates the signals between the AXI Streaming Intel FPGA IP for PCI Express soft logic and the Hard IP. These signals are not available to the application logic.

User reset sequencer triggered Warm Reset flow is the same as the user reset sequencer triggered Cold Reset flow, with the exception that the *Subsystem_cold_rst_n* will not be asserted for this flow.

Figure 32. User Reset Sequencer Initiated Warm Reset Entry and Exit Sequence Timing Diagram



Note: * indicates the signals between the AXI Streaming Intel FPGA IP for PCI Express soft logic and the Hard IP. These signals are not available to the application logic.

6.3. Application Packet Interface

The IP uses an AXI4 Streaming interface for transporting header and data information. The header and data are presented as separate interfaces. The PCIe header, PF Number, VF Number, BAR number and Prefix information are grouped as a 32-byte header on the AXI Streaming interface. The data is presented as a 128-, 64- or 32-bytes (1024-, 512- or 256-bits) wide data bus, segmented into a number of segments depending on the configuration (Gen5/4/3 x16/x8x8) and mode (HIP Native, Simple or Compact packing) used.

Table 29. IP to FPGA Fabric Interfaces Summary (Supported in Intel Quartus Prime 23.4)

Link Width	Link Speed	Tile	Mode	Data Width (Each Interface)	Header Width (Each Interface)	# of Segments	Application Clock Frequency (MHz)
x16	Gen5	R	HIP Native	1024	1024	4	Same as PLD Clock frequency (500/475/450/425/400)
x8x8	Gen5	R	HIP Native	512	512	2	Same as PLD Clock frequency (500/475/450/425/400)
x16	Gen4	R	HIP Native	512	512	2	Same as PLD Clock frequency (500/475/450/425/400)
x8x8	Gen4	R	HIP Native	256	256	1	Same as PLD Clock frequency (500/475/450/425/400)
x16	Gen3	R	HIP Native	512	512	2	Same as PLD Clock frequency (300/275/250)

continued...

Link Width	Link Speed	Tile	Mode	Data Width (Each Interface)	Header Width (Each Interface)	# of Segments	Application Clock Frequency (MHz)
x8x8	Gen3	R	HIP Native	256	256	1	Same as PLD Clock frequency (300/275/250)
x16	Gen4	P/F	Compact	512	512	2	500/470/450/400/350/250/225/200/175
x8x8	Gen4	P/F	Compact	256	256	1	500/470/450/400/350/250/225/200/175
x16	Gen3	P/F	Compact	512	512	2	250
x8x8	Gen3	P/F	Compact	256	256	1	250
x8	Gen4	P/F	Compact	256	256	1	500/470/450/400/350/250/225/200/175
x8	Gen3	P/F	Compact	256	256	1	250

6.3.1. Header Format

The following table lists header fields, their byte positions and bit positions on the user header bus.

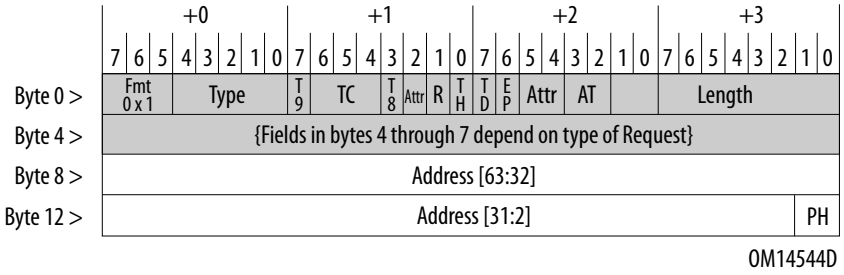
Table 30. Header Format

Header Byte Index	Header Fields	Bits	Header Bit Position Start	Header Bit Position End
Byte 15 - Byte 0	PCIe Header	128	0	127
Byte 19 - Byte 16	Prefix	24	128	151
	Prefix Type	5	152	156
	Prefix Present	1	157	157
	Reserved	2	158	159
Byte 23 - Byte 20	PF Number	3	160	162
	VF Number	11	163	173
	VF Active	1	174	174
	BAR number	4	175	178
	Slot number	5	179	183
	Reserved*	8	184	191
Byte 31 - Byte 24	Reserved	64	192	255

Note: Bits [186:185] are reserved for future use.

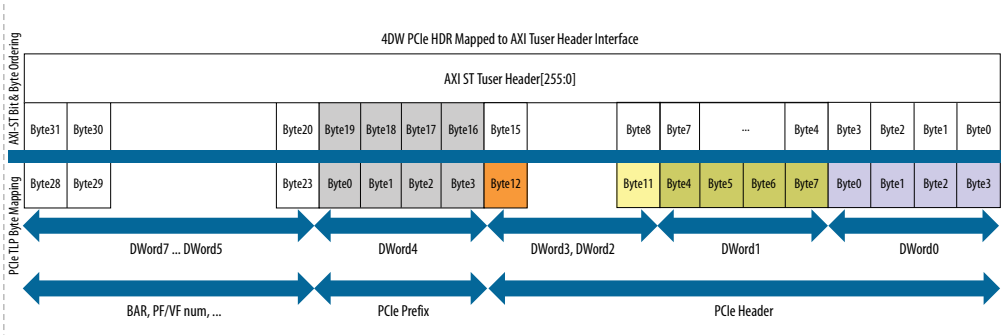
The following figure shows a standard PCIe header format.

Figure 33. PCIe Header for Memory TLP with 64Bit Addressing (4DW Header)



The PCI specification standard header format is mapped to a Tuser Header interface as shown in the following figure:

Figure 34. 4DW PCIe Header Mapping on the User Header Bus



The 3DW PCIe Header will be mapped to AXI-ST Tdata bit[95:0], the bit[127:96] are considered do not care in this case.

Figure 35. PCIe Header for Memory TLP with 64Bit Addressing (3DW Header)

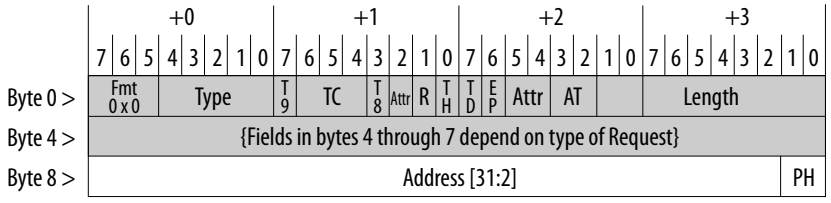
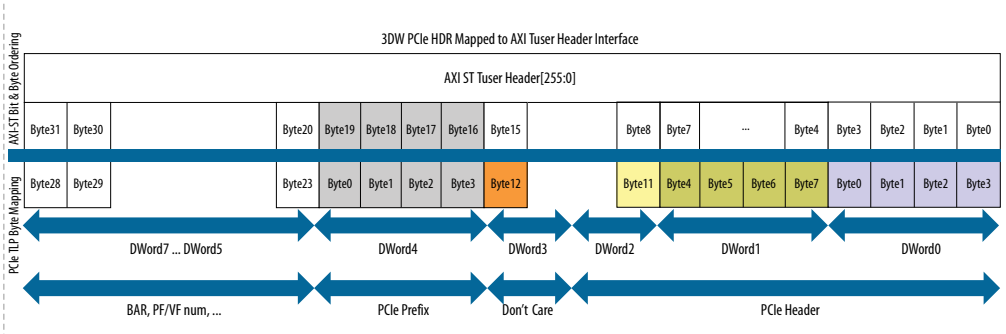


Figure 36. 3DW PCIe Mapping on the User Header Bus



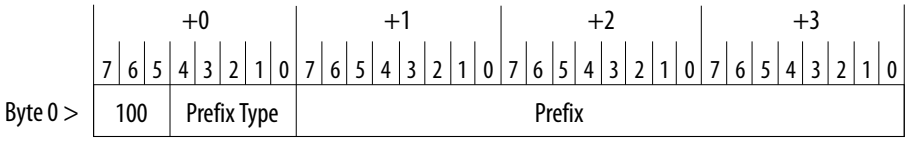
6.3.2. Prefix Format

The Prefix and Prefix Type fields carry the prefix information of the current TLP. The Prefix Present field indicates presence of prefix information. The Prefix Present field '0' indicates current TLP has no prefix associated with it. It is recommended to drive zero on Prefix and Prefix Type fields when Prefix Present is equal to '0'.

Note: The design supports one DW prefix per TLP in the current release.

The following figure shows PCIe Prefix TLP Format.

Figure 37. PCIe TLP Prefix



The table below shows the mapping of PCIe TLP Prefix shown in the figure above on to the Tdata bus.

Table 31. PCIe TLP Prefix Mapping to Tdata Header Field

PCIe TLP Prefix Bytes	AXI-ST Tdata Header Byte Index
Prefix - Byte 3	Byte 16
Prefix - Byte 2	Byte 17
Prefix - Byte 1	Byte 18
Prefix Type - Byte 0 [Bits 4:0]	Byte 19 [Bits 4:0]

6.3.3. Function Number Format

The PF Number, VF Number and VF Active fields specify the function number of a TLP. The VF Active high indicates the request/completion is targeting a virtual function. The VF Active low indicates the request/completion is targeting a physical function in a device.

The Slot number indicates which endpoint in a design the transaction is targeting. The slot number field is always zero when the Switch functionality is not present in the design.

The definitions of these function fields apply not only to the header but also for all occurrences in this document, for example on interface pins, register fields, and register names.

Table 32. Function Number Field Description

Function Number Fields	Description
Slot Number	Indicates target slot number of received TLP
PF Number	Indicates target physical function number of received TLP
VF Number	Indicates target virtual function number of received TLP
VF Active	When asserted high, indicates access is for Virtual Function. When low, indicates access is for Physical Function

6.3.4. BAR Number Format

The BAR Number indicates matching BAR for MMIO transaction coming from HOST side.

Table 33. BAR Number Field Description

BAR Number	Description
0000	BAR 0 (when configured as 32-bit BAR), or BAR 0-1 (when configured as 64-bit BAR)
0001	BAR 1 (when configured as 32-bit BAR); reserved when BAR 1 is combined with BAR 0 to form a 64-bit BAR
0010	BAR 2 (when configured as 32-bit BAR), or BAR 2-3 (when configured as 64-bit BAR)
0011	BAR 3 (when configured as 32-bit BAR); reserved when BAR 2 is combined with BAR 3 to form a 64-bit BAR
0100	BAR 4 (when configured as 32-bit BAR), or BAR 4-5 (when configured as 64-bit BAR)
0101	BAR 5 (when configured as 32-bit BAR); reserved when BAR 4 is combined with BAR 5 to form a 64-bit BAR
0111	Expansion ROM BAR
All others	Reserved

6.3.5. Data and Header Packing Schemes

The AXI Streaming Intel FPGA IP for PCI Express provides separate interfaces for data and header. The IP supports the following packing schemes:

- HIP Native mode packing
- Simple packing
- Compact packing

6.3.5.1. HIP Native Mode Packing

This packing scheme is only available when you choose R-Tile. In this packing scheme, the AXI-ST Transmit and Receive interfaces follow all the rules that the Native Hard IP's AVST interface follows for packing TLPs.

When using R-Tile and in 1x16 mode, the IP provides a data bus that is 1024-bits wide with four segments (each segment having 256 bits of data) qualified by a data valid, and 1024 bits of header with four segments (each segment having 256 bits of header) qualified by a header valid for each of the corresponding 4 data segments.

When using R-Tile and in 2x8 mode, for each of the x8 ports, the IP provides a data bus that is 512-bits wide with two segments (each segment having 256 bits of data) qualified by a data valid, and 512 bits of header with two segments (each segment having 256 bits of header) qualified by a header valid for each of the corresponding 2 data segments.

The application logic must follow the AXI4-ST Interface Specification for ready - valid handshake while driving data on the data bus interface. **This interface also does not follow a fixed latency between the data ready and valid signals as specified by the AXI4-ST Interface Specifications.**

6.3.5.1.1. Application Logic Guidelines for the AXI Streaming TX Interface in HIP Native Mode (R-Tile)

The following guidelines must be considered by the Application logic:

- Application logic must adhere to the requirements for the pX_app_ss_st_tx_ready and pX_app_ss_st_tx_valid signal behavior as outlined in the AXI4 Streaming Specifications.
- Transmission of a TLP must be uninterrupted when the pX_app_ss_st_tx_ready is asserted, unless there is backpressure from the IP as indicated by the deassertion of pX_app_ss_st_tx_ready.

  Note: Failing to meet this guideline may cause the transmission of a TLP with an invalid LCRC.
- For the 1x16 mode, the start of a command (header) can only happen in segment 0 (S0) or segment 2 (S2) (i.e. a given command cannot start on segment 1 (S1) or segment 3 (S3)).
- For the 1x16 mode, the header for segment 2 is allowed depending on the utilization for segment 0 and segment 1. Refer to the table below for the allowed conditions. Note that the table does not include all the signals for the AXI-ST TX interface. It only shows the Header (H) and Data (D) combinations to highlight the valid cases where a command can be started on segment 2.

Table 34. TX AXI-ST HIP Native Mode Data Packing for x16 1024 Bits with 4 Segments

S0	S1	S2	S3
H, D	-	H, D	-
H, D	D	H, D	D
D	-	H, D	D
D	D	H, D	-

- For Configuration Mode 1 (2x8) and Configuration Mode 2 (4x4), the header for segment 1 is allowed depending on the utilization for segment 0. Refer to the table below for the allowed conditions. Note that the table does not include all the signals for the AXI-ST TX interface. It only shows the Header (H) and Data (D) combinations to highlight the valid cases where a TLP can be started on segment 1.

Table 35. TX AXI-ST HIP Native Mode Data Packing for x8

S0	S1
H, D	H, D
H, D	D
D	H, D

- For a single command spanning across multiple segments, the application logic needs to send the TLP in the order of the segment indices (segment S0 → S1 → S2 → S3 → S0).

- If the length of the TLP being transmitted is greater than the segment size, the segment used to assert the pX_app_ss_st_tx_tuser_last_segment signal is dictated by the TLP length.
- The app_ss_st_tx_tkeep signal must be used to qualify the byte-wise data on each segment to indicate if the content of the associated byte is valid. The invalid bytes are allowed only during the app_axi_st_tx_tlast cycle.
- The maximum number of clock cycles allowed between the deassertion of pX_app_ss_st_tx_ready and pX_app_ss_st_tx_valid is 4 axi_st_clk cycles.

Example of the HIP Native mode packing scheme in Gen5x16 with a 1024-bit, 4-segment bus on the TX ASI-ST Interface:

1st Command with Data - Payload 16 Bytes

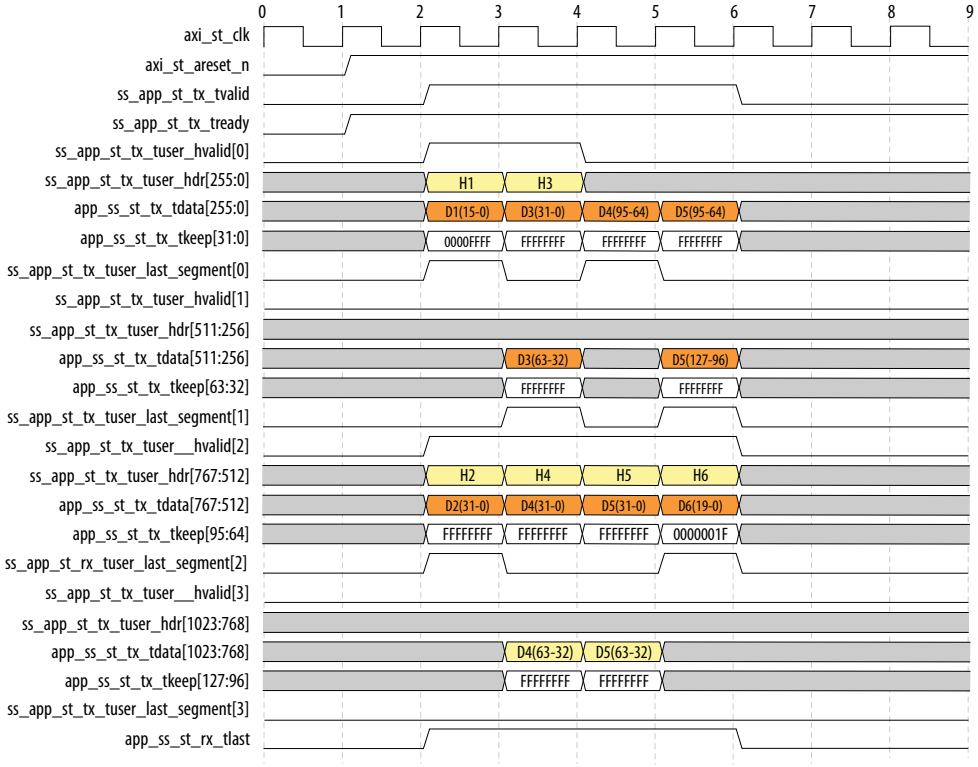
2nd Command with Data - Payload 32 Bytes

3rd Command with Data - Payload 64 Bytes

4th Command with Data - Payload 96 Bytes

5th Command with Data - Payload 128 Bytes

6th Command with Data - Payload 20 Bytes



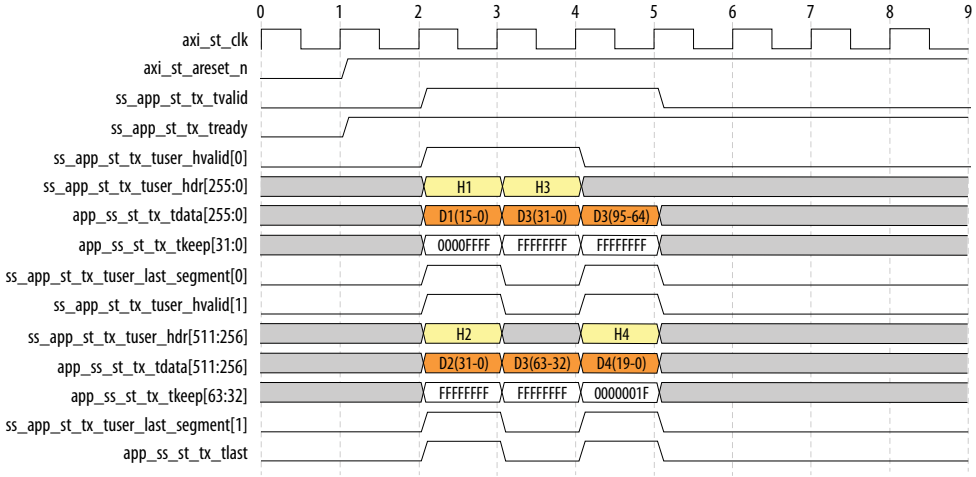
Example of the HIP Native mode packing scheme in Gen5x8 with a 512-bit, 2-segment bus on the TX ASI-ST Interface:

1st Command with Data - Payload 16 Bytes

2nd Command with Data - Payload 32 Bytes

3rd Command with Data - Payload 96 Bytes

4th Command with Data - Payload 20 Bytes



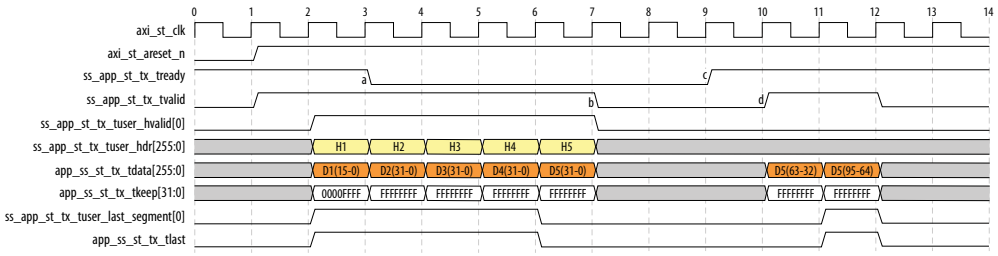
AXI Streaming TX Interface ss_app_st_tx_tready Behavior in HIP Native Mode

The following timing diagram illustrates the behavior of ss_app_st_tx_tready, which is deasserted to pause the data transmission to the IP, and then reasserted. The application logic deasserts app_ss_st_tx_tvalid four clock cycles after ss_app_st_tx_tready is deasserted (from the point marked with the letter a to the point marked with the letter b). This is the maximum number of clock cycles allowed between the deassertion of ss_app_st_tx_tready and app_ss_st_tx_tvalid.

When the IP reasserts the ss_app_st_tx_tready signal (from the point marked with letter c to the point marked with letter d), the following two cases must be considered:

- Case 1: If there is a TLP suspended due to the deassertion of ss_app_st_tx_tready, then the maximum number of clock cycles for app_ss_st_tx_tvalid to go high after the assertion of ss_app_st_tx_tready is one (as illustrated in the following figure).
- Case 2: If there is no TLP suspended due to the deassertion of ss_app_st_tx_tready, then there is no requirement, and the application logic can reassert app_ss_st_tx_valid as soon as it has a TLP available to transmit. The application must not deassert app_ss_st_tx_valid during the packet transmission unless there is backpressure from the IP indicated by the deassertion of ss_app_st_tx_tready.

  Note: Failing to meet this guideline may cause the transmission of a TLP with an invalid LCRC.



6.3.5.1.2. Application Logic Guidelines for the AXI Streaming RX Interface in HIP Native Mode

The Application Layer receives data from the AXI Streaming Intel FPGA IP for PCI Express over the AXI-ST RX interface. For R-Tile in HIP Native mode, the pX_ss_app_st_rx_ready has to be always high. The buffer control in the application logic needs to be handled by the RX Flow Control Credit interface. Refer to RX Flow Control Interface on page 75 for more details.

The following guidelines must be considered by the Application logic:

- The pX_ss_app_st_rx_ready signal has to be always high. The buffer control and backpressure need to be handled with the RX Flow Control Credit interface. Refer to RX Flow Control Credit Interface for more details.
- The header may occur in any of the segments (S0/1/2/3).
- For a single TLP spanning across multiple segments, the application logic needs to process the TLP in the order of the segment indices (segment S0 → S1 → S2 → S3 → S0).

- For multiple TLPs arriving on the same clock cycle, the application logic needs to process the TLPs in the order of the segment indices (segment S0 → S1 → S2 → S3 → S0).
- The IP does not use segment 2 and segment 3 if segment 0 AND segment 1 are unused.

  Note that this behavior only applies in Production devices or Engineering Samples with the following OPNs:

  — AGIx027R29AxxxxR2
  — AGIx027R29AxxxxR3
  — AGIx027R29BxxxxR3
  — AGIx023R18AxxxxR0
  — AGIx041R29DxxxxR0
  — AGIx041R29DxxxxR1
  — AGMx039R47AxxR0

  For more details on OPN decoding, refer to the Available Options section of the Intel Agilex 7 FPGAs and SoCs Device Overview.
- - There is a maximum of three headers in a single clock cycle. The following table describes the possible combinations across segments:

Table 36. RX AXI-ST with 1024-Bit Tdata Bus (x16 HIP Native Mode - 1024 Bits, 4 Segments x 256 Bits Wide)

S0	S1	S2	S3
H, D	H, D	H, D	-
H, D	H, D	H, D	D
-	H, D	-	H, D
-	H, D	D	H, D
	H, D	D	H
H, D	D	H, D	H, D
H, D	D	H, D	H

Example of HIP-Native mode packing scheme in Gen5x16 with a 1024-bit 4-segment bus on the RX AXI-ST Interface:

1st Command with Data - Payload 16 Bytes

2nd Command with Data - Payload 32 Bytes

3rd Command with Data - Payload 64 Bytes

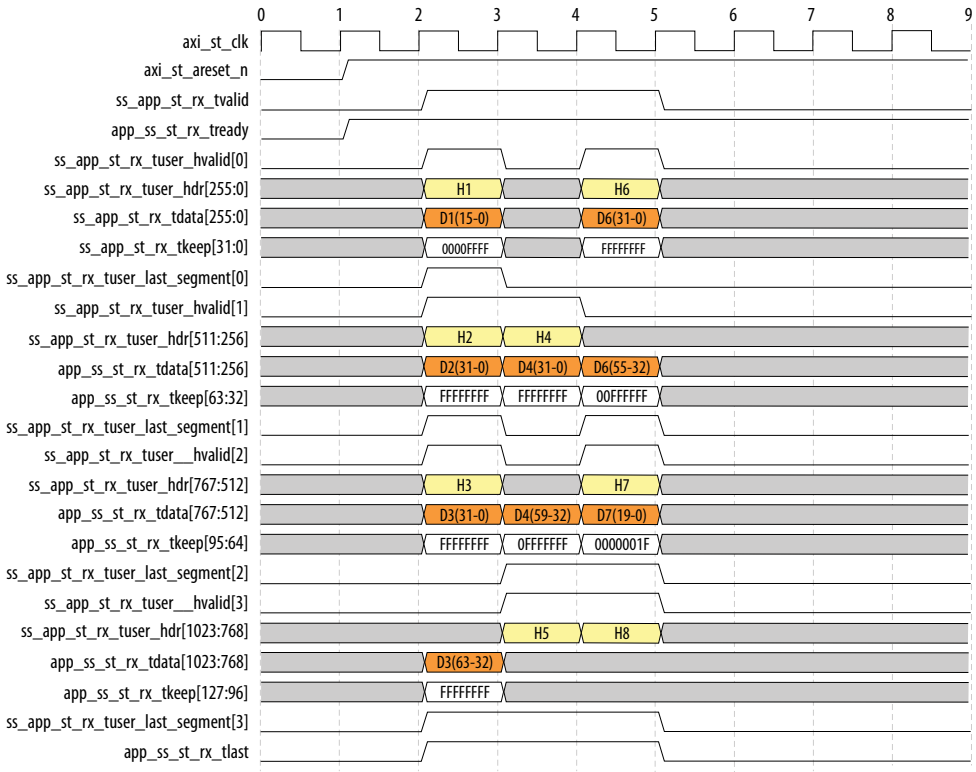
4th Command with Data - Payload 60 Bytes

5th Command without Data

6th Command with Data – Payload 56 Bytes

7th Command with Data – Payload 20 Bytes

8th Command without Data



6.3.5.2. Compact Packing

This packing scheme allows the header to be available in fixed locations. This is applicable to P/F/R-Tiles. When using R-Tile, the compact packing scheme is only available when using the non-HIP Native mode.

6.3.5.2.1. Application Logic Guidelines for the AXI Streaming TX Interface in Compact Mode (P/R-Tiles)

Following are the guidelines for the application logic when using the compact packing scheme:

- For a given clock cycle, the header always starts from segment 0, or on the following segment after the end of the previous packet. The header for segment 1 is allowed depending on the utilization for segment 0. Refer to the table below for the allowed conditions. Note that the table does not include all the signals for the AXI-ST TX interface. It only shows the Header (H) and Data (D) combinations to highlight the valid cases where a TLP can be started on segment 1.
- When sending multiple packets on multiple segments in a single clock cycle, the headers must be placed on contiguous segments starting with segment 0. For example, when using a 1024-bit data bus with 4 segments, the header can be on segments 0, 1, 2 but not on segments 0, 1, 3 nor on segments 1, 2, 3.

  The following table describes the possible combinations across segments:

The following table describes the possible combinations across segments:

Table 37. TX Compact Packing Scheme with 512-Bit Tdata Bus (Gen4x16 P-Tile - 512 Bits, 2 Segments x 256 Bits Wide)

S0	S1
H, D	H, D
H, D	D
D	H, D

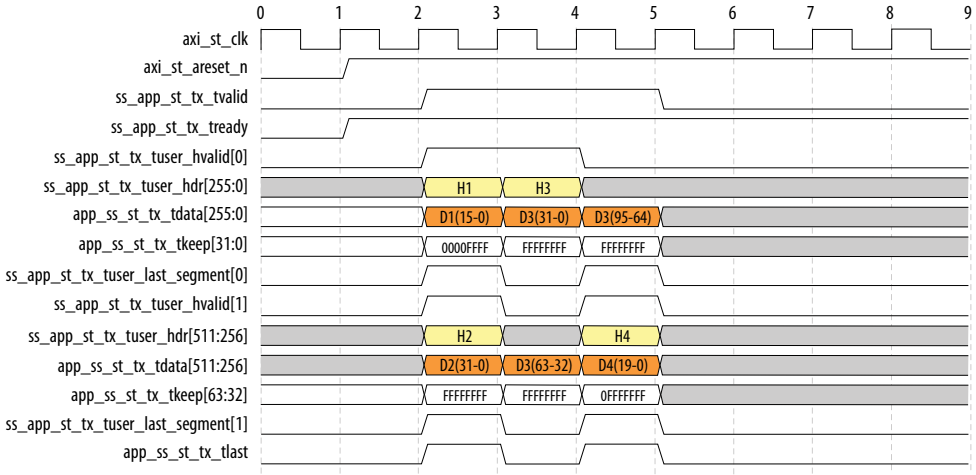
Example of the compact packing scheme in Gen4x16 with a 512-bit, 2-segment bus on the TX AXI-ST Interface:

1st Command with Data - Payload 16 Bytes

2nd Command with Data - Payload 32 Bytes

3rd Command with Data - Payload 96 Bytes

4th Command with Data – Payload 20 Bytes



6.3.5.2.2. Application Logic Guidelines for the AXI Streaming RX Interface in Compact Mode (P/R-Tiles)

Following are the guidelines for the application logic when using the compact packing scheme:

- For a given clock cycle, the header always starts from segment 0, or on the following segment after the end of the previous packet. The header for segment 1 is allowed depending on the utilization for segment 0. Refer to the table below for the allowed conditions. Note that the table does not include all the signals for the AXI-ST RX interface. It only shows the Header (H) and Data (D) combinations to highlight the valid cases where a TLP can be started on segment 1.
- When sending multiple packets on multiple segments in a single clock cycle, the headers are placed on contiguous segments starting with segment 0. For example, when using a 1024-bit data bus with 4 segments, the header can be on segments 0, 1, 2 but not on segments 0, 1, 3 nor on segments 1, 2, 3.

The following table describes the possible combinations across segments:

Table 38. RX AXI-ST Compact Packing Scheme with a 512-Bit Tdata Bus (Gen4x16 P-Tile - 512 Bits, 2 Segments x 256 Bits Wide)

S0	S1
H, D	H, D
H, D	D
D	H, D

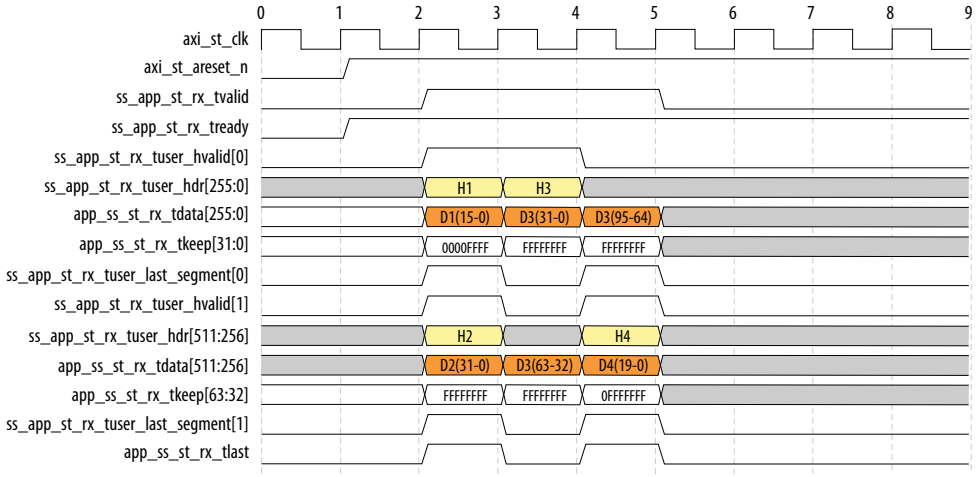
Example of the compact packing scheme in Gen4x16 with a 512-bit, 2-segment bus on the RX AXI-ST Interface:

1st Command with Data - Payload 16 Bytes

2nd Command with Data - Payload 32 Bytes

3rd Command with Data - Payload 96 Bytes

4th Command with Data – Payload 20 Bytes



6.3.5.3. Simple Packing

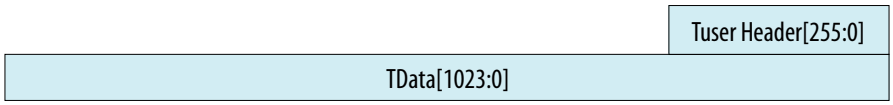
This packing scheme allows the header to be packed on segment0 for any given clock cycle. This constrains the design to send one packet per cycle. This is applicable to P/F/R-Tiles. When using R-Tile, the simple packing scheme is only available when using the non-HIP Native mode.

Note: This mode is not supported in the current Intel Quartus Prime release.

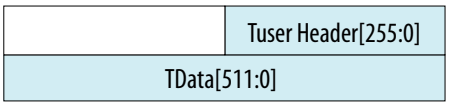
6.3.5.3.1. Application Logic Guidelines for the AXI Streaming TX Interface in Simple Mode (P/R-Tiles)

Below is a figure that shows the possible header positions in the simple packing scheme for 1024-, 512- and 256-bit wide data buses.

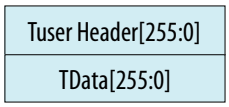
Simple Packing Scheme with 1024-Bit Tdata Bus



Simple Packing Scheme with 512-Bit Tdata Bus



Simple Packing Scheme with 256-Bit Tdata Bus



Example of the simple packing scheme in Gen5x16 with a 1024-bit, 4-segment bus on the TX AXI-ST Interface:

1st Command with Data - Payload 16 Bytes

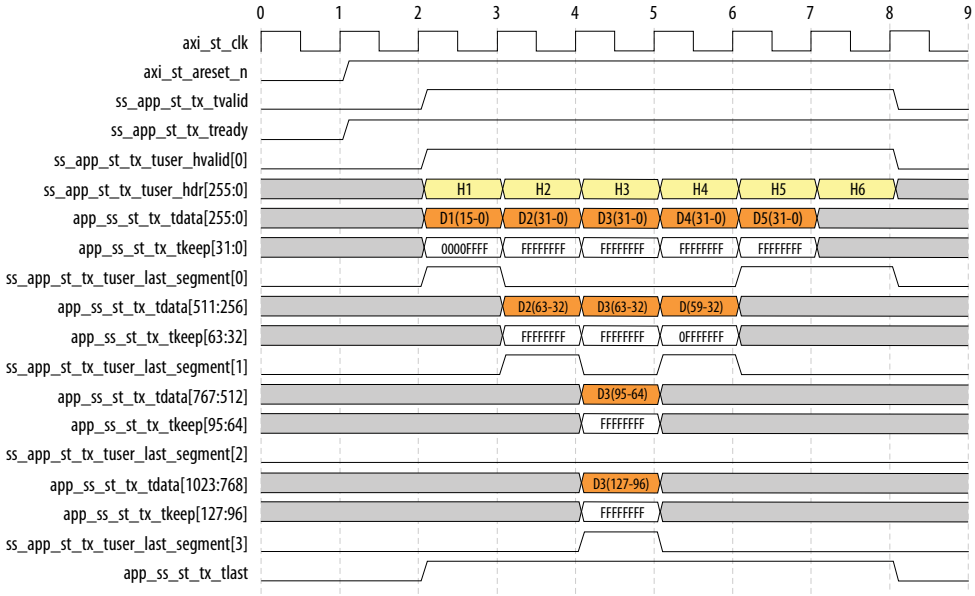
2nd Command with Data - Payload 64 Bytes

3rd Command with Data - Payload 128 Bytes

4th Command with Data - Payload 60 Bytes

5th Command with Data - Payload 32 Bytes

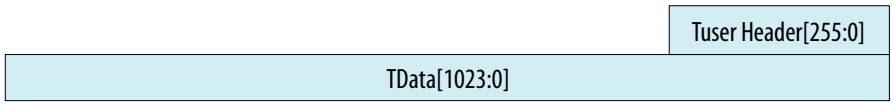
6th Command without Data



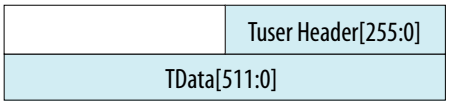
6.3.5.3.2. Application Logic Guidelines for the AXI Streaming RX Interface in Simple Mode (P/R-Tiles)

Below is a figure that shows the possible header positions in the simple packing scheme for 1024-, 512- and 256-bit wide data buses.

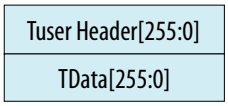
Simple Packing Scheme with 1024-Bit Tdata Bus



Simple Packing Scheme with 512-Bit Tdata Bus



Simple Packing Scheme with 256-Bit Tdata Bus



Example of the simple packing scheme in Gen5x16 with a 1024-bit, 4-segment bus on the RX AXI-ST Interface:

1st Command with Data - Payload 16 Bytes

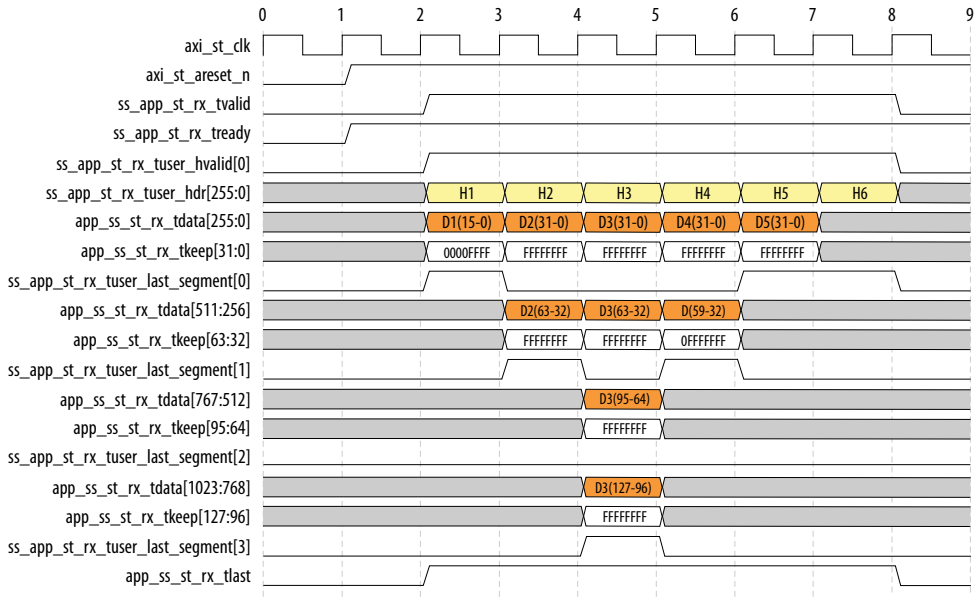
2nd Command with Data - Payload 64 Bytes

3rd Command with Data - Payload 128 Bytes

4th Command with Data - Payload 60 Bytes

5th Command with Data - Payload 32 Bytes

6th Command without Data



6.3.6. Application Packet Transmit Interface (st_tx)

The outbound packet towards the link from application side is transmitted through this interface with separate header and data interfaces. The PCIe header, PF Number, VF Number, BAR number and Prefix information are grouped as a 32-byte header on the AXI Streaming interface. The data is presented as a 1024-, 512- or 256-bit wide data bus, segmented into a number of segments depending on the configuration (Gen3/4/5 x16/x8x8) and mode (HIP Native, Simple or Compact packing) used.

Table 39. APP AXI ST TX Interface

Signal Name	Direction	Clock Domain	Description
app_ss_st_tx_tvalid	Input	axi_st_clk	app_ss_st_tx_valid indicates that the source is driving a valid transfer.
ss_app_st_tx_tready	Output	axi_st_clk	app_axi_st_tx_ready indicates that the sink can accept a transfer in the current cycle. readyLatency parameter defined in Avalon spec shall be supported. By default, the value is '0'.
app_ss_st_tx_tdata[(DWIDTH*NUM_OF_SEG)-1:0]	Input	axi_st_clk	Data Interface with configurable width specified by the DWIDTH_SEG*NUM_OF_SEG parameter. tdata carries only the payload. Default DWIDTH = DWIDTH_SEG*NUM_OF_SEG = 1024.
app_ss_st_tx_tkeep[((DWIDTH_SEG/8)*NUM_OF_SEG)-1:0]	Input	axi_st_clk	A byte qualifier used to indicate whether the content of the associated byte is valid.

continued...

Signal Name	Direction	Clock Domain	Description
			The invalid bytes are allowed only during app_axi_st_tx_tlast cycle.
app_ss_st_tx_tlast	Input	axi_st_clk	Indicates End of Data/ Command Transmission.
ss_app_st_tx_tuser_last_segment[NUM_OF_SEG-1:0]	Input	axi_st_clk	Indicates the last segment of the packet. Not applicable for the Simple packing scheme. For a header without payload, this signal is asserted during the header cycle.
ss_app_st_tx_tuser_vendor[NUM_OF_SEG-1:0]	Input	axi_st_clk	Vendor Specific Tuser bits. Indicates Header Format. Reserved (don't-care).
ss_app_st_tx_tuser_hvalid[NUM_OF_SEG-1:0]	Input	axi_st_clk	Indicates the tuser_hdr and tuser_vendor bits are valid in the respective segment.
app_ss_st_tx_tuser_hdr[256*NUM_OF_SEG-1:0]	Input	axi_st_clk	Carries header format for the respective segment. Refer to Header Format for the bit positions and mapping.

Refer to the section Data and Header Packing Schemes for timing diagrams in each of the modes when using the Application Packet Transmit Interface.

6.3.7. Application Packet Receive Interface (st_rx)

The packet received from the link is presented to the application logic on this interface with separate header and data interfaces. The PCIe header, PF Number, VF Number, BAR number and Prefix information are grouped as a 32-byte header on the AXI Streaming interface. The data is presented as a 1024-, 512- or 256-bit wide data bus, segmented into a number of segments depending on the configuration (Gen3/4/5 x16/ x8x8) and mode (HIP Native, Simple or Compact packing) used.

Signal Name	Direction	Clock Domain	Description
ss_app_st_rx_tvalid	Output	axi_st_clk	ss_app_st_rx_tvalid indicates that the source is driving a valid transfer.
app_ss_st_rx_tready	Input	axi_st_clk	app_ss_st_rx_tready indicates that the sink can accept a transfer in the current cycle readyLatency parameter defined in Avalon spec shall be supported. By default, the value is '0'.
ss_app_st_rx_tdata[(DWIDTH_SEG*NUM_OF_SEG)-1:0]	Output	axi_st_clk	Data interface with configurable width specified by the DWIDTH_SEG*NUM_OF_SEG parameter. Default DWIDTH = DWIDTH_SEG*NUM_OF_SEG = 1024 tdata carries only the payload.

continued...

Signal Name	Direction	Clock Domain	Description
ss_app_st_rx_tkeep[((DWIDTH_SEG/8)*NUM_OF_SEG)-1:0]	Output	axi_st_clk	A byte qualifier used to indicate whether the content of the associated byte is valid. The invalid bytes are allowed only during the ss_app_st_tx_tlast cycle.
ss_app_st_rx_tlast	Output	axi_st_clk	Indicates End of Data/ Command Transmission. tlast may seem redundant with the tuser.last_segment. However, tlast can be used by the front-end layer of the AXI BFM/routing fabric that does not deal with the decoding of the multipacket data.
ss_app_st_rx_tuser_last_segment[NUM_OF_SEG-1:0]	Output	axi_st_clk	Indicates Packet End position on tdata bus. Only applicable with Variable Header Position and Compact Packing scheme. *Note:* Variable and compact packing not currently supported.
ss_app_st_rx_tuser_vendor[NUM_OF_SEG-1:0]	Output	axi_st_clk	Vendor Specific Tuser bits. Indicates Header Format. Is a don't-care in Power User mode. [0] - Indicates Header format of First packet in a cycle [1] - Indicates Header format of Second packet in a cycle
ss_app_st_rx_tuser_hvalid[NUM_OF_SEG-1:0]	Output	axi_st_clk	Indicates the ss_app_st_rx_tuser_hdr and ss_app_st_rx_tuser_vendor bits are valid in the respective segment.
ss_app_st_rx_tuser_hdr[NUM_OF_SEG*256-1:0]	Output	axi_st_clk	If tuser_vendor = 0, tuser_hdr carries the Power User Mode header format for the respective segment. If tuser_vendor = 1, tuser_dr carries the header format for the respective segment.

Refer to the section Data and Header Packing Schemes for timing diagrams in each of the modes when using the Application Packet Receive Interface.

6.3.8. Flow Control Credit Handling

The IP implements credit handling by providing TX and RX flow control credit interfaces to/from the application logic. Each header type (P,NP,CPL) and data type (P,NP,CPL) has independent credit handling. The credits are advertised as limit values as per the PCIe specification. One data credit consists of 16 bytes. One header credit includes the TLP Header, 1DW prefix (if present) and the digest (if present). The credit interfaces on the IP for TX and RX are not time division multiplexed. The initial buffer space for P, NP, CPL header and data is communicated on the first set of clock cycles

with a valid assertion. Every time there is an update to credits for a particular type, the credit interface updates the value for that particular type with a valid assertion. A value of 0 credits initially indicates infinite credits.

6.3.8.1. Application Transmit Flow Control Credit Interface (st_txcrdt)

On the Transmit side, the link partner's receive buffer space information is provided to the application through the Transmit Flow Control Credit Interface for P/F/R-Tiles. Apart from the AXI streaming ready valid handshake, the application logic must check for the availability of credits on the TX flow control credit interface before transmitting the TLP.

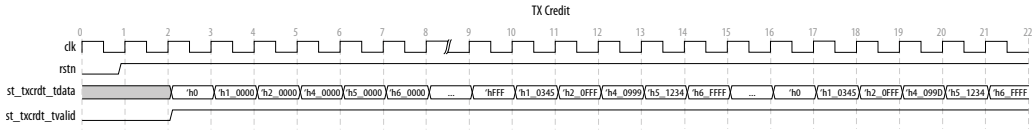
Table 40. Application Transmit Flow Control Credit Interface

Signal Name	Direction	Clock Domain	Description
ss_app_st_txcrdt_tvalid	Output	axi_st_clk	tvalid indicates that the credit information on tdata is valid.
ss_app_st_txcrdt_tdata[18:0]	Output	axi_st_clk	Carries the credit limit information and type of credit. [15:0] - Credit Limit Value [18:16] - Credit Type • 3'b000 - Posted Header Credit • 3'b001 - Non-Posted Header Credit • 3'b010 - Completion Header Credit • 3'b011 - Reserved • 3'b100 - Posted Data Credit • 3'b101 - Non-Posted Data Credit • 3'b110 - Completion Data Credit • 3'b111 - Reserved

Note: Although infinite credits may be advertised, the IP may still backpressure on the st_tx interface.

The figure below shows the credit limit update on the Transmit Flow Control Credit Interface.

In the example below, updated credit limit is output from cycle 9 to cycle 14. When the HOST returns the credit after receiving the packet, the credit limit is incremented by the number of credits returned. At cycle 16, one Posted Header credit is returned. At cycle 19, four Posted Data credits are returned.



6.3.8.2. Application Receive Flow Control Credit Interface (st_rxcrdt) (R-Tile Only)

The RX flow control credit interface is not available for P/F-Tiles. The Receive side of the IP operates on an AXI Streaming ready-valid handshake for P- and F-Tiles.

For R-Tile:

- In HIP Native mode, the IP provides the RX flow control credit interface for the application logic to advertise the available credits in the application to the IP.
- In non-HIP Native (Simple/Compact) mode, the receive side of the IP operates on an AXI streaming ready-valid handshake. There is no RX flow control credit interface available for the application logic in this mode.

In the R-Tile HIP Native mode, the RX flow control interface provides information on the application's available RX buffer space to the IP for header and data for Posted (P), Non-Posted (NP) and Completion (CPL) transactions. It reports the space available in a number of credits as defined by the PCIe Specification.

The Application logic must ensure that enough credits are provided to the IP initially. This is to prevent a performance impact caused by the lack of credits available between the IP and the Application logic. Depending on the number of P, NP and CPL transactions happening at the PCIe link level, a lack of credit scenario may happen if the number of credits advertised by the Application logic is less than the credits advertised by the IP to the link partner.

Flow control credits are available for the following TLP categories:

- Posted (P) transactions: TLPs that do not require a response.
- Non-posted (NP) transactions: TLPs that require a completion.
- Completions (CPL): TLPs that respond to non-posted transactions.

Table 41. Credits Advertised by the IP to the Link Partner in Endpoint Mode

The credit unit in this table follows the PCIe Specification where one credit = 4 DWs = 16 Bytes.

Port	Posted Headers	Posted Data	Non-Posted Headers	Non-Posted Data	Completion Headers	Completion Data
Port 0	784	1456	784	392	0 (infinite)	0 (infinite)
Port 1	392	760	392	196	0 (infinite)	0 (infinite)

Note: In some systems, there are broadcast Message TLPs being sent by the link partner during the PCIe enumeration process. Failing to properly initialize and return the corresponding credits from the application logic to the IP may cause enumeration issues due to the priority order between Message TLPs and the Completions required for Configuration TLPs. Application logic must ensure proper credits are returned to the IP for any TLP that it receives.

Table 42. Categorization of Transaction Types

TLP Type	Category
Memory Write	Posted
Memory Read	Non-Posted
Memory Read Lock	
I/O Read	
I/O Write	
Configuration Read	
Configuration Write	

continued...

TLP Type	Category
Fetch and Add AtomicOp	
Message	Posted
Completion	Completion
Completion with Data	
Completion Lock	
Completion Lock with Data	

Table 43. Application Receive Flow Control Credit Interface

Signal Name	Direction	Clock Domain	Description
ss_app_st_rxcrdt_tvalid	Input	axi_st_clk	tvalid indicates that the credit information on tdata is valid.
ss_app_st_rxcrdt_tdata[18:0]	Input	axi_st_clk	Carries the credit limit information and type of credit. [15:0] - Credit Limit Value [18:16] - Credit Type • 3'b000 - Posted Header Credit • 3'b001 - Non-Posted Header Credit • 3'b010 - Completion Header Credit • 3'b011 - Reserved • 3'b100 - Posted Data Credit • 3'b101 - Non-Posted Data Credit • 3'b110 - Completion Data Credit • 3'b111 - Reserved

The credit limit is first initialized to 0 for all the credit types.

6.4. Configuration Extension Bus Interface

6.4.1. Configuration Extension Bus Request Interface (st_cebreq)

The IP sends configuration read and configuration write requests using this interface. The interface follows AXI Streaming interface protocol with ready valid handshake. The interface will support a maximum of one outstanding read request.

This interface is mutually exclusive with the Configuration Intercept Request Interface.

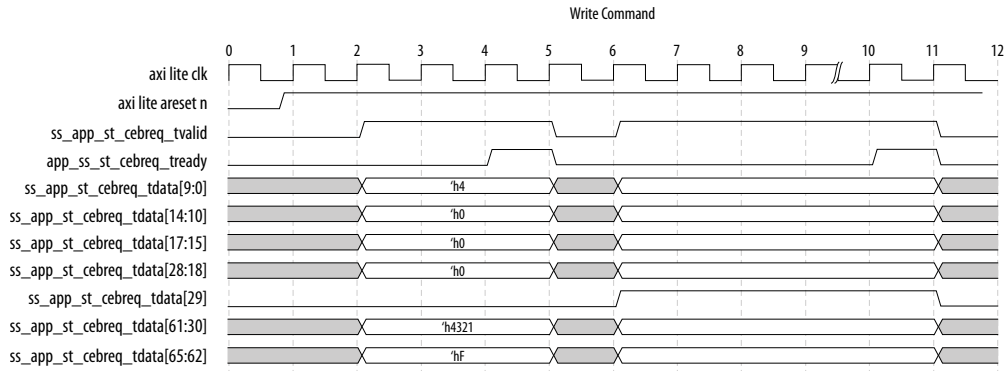
Table 44. Configuration Extension Bus Request Interface

Signal Name	Direction	Clock Domain	Description
ss_app_st_cebreq_tvalid	Output	axi_lite_clk	When asserted, indicates a valid Configuration Extension access cycle. Deasserted when app_ss_st_cebreq_tready is asserted.
app_ss_st_cebreq_tready	Input	axi_lite_clk	Application asserts this signal for one clock to acknowledge ss_app_st_cebreq_tvalid is seen by responder.
ss_app_st_cebreq_tdata[9:0]	Output	axi_lite_clk	DWORD Address of register being accessed.
ss_app_st_cebreq_tdata[14:10]	Output	axi_lite_clk	The slot Number of register access.
ss_app_st_cebreq_tdata[17:15]	Output	axi_lite_clk	The PF Number of register access [2:0].
ss_app_st_cebreq_tdata[28:18]	Output	axi_lite_clk	Indicates child VF Number of parent PF indicated by ss_app_st_cebreq_tdata[17:15].
ss_app_st_cebreq_tdata[29]	Output	axi_lite_clk	Indicates access is for Virtual Function implemented in slot's physical function.
ss_app_st_cebreq_tdata[61:30]	Output	axi_lite_clk	Write data for write access.
ss_app_st_cebreq_tdata[67:62] [67:66] = pf_num[4:3]	Output	axi_lite_clk	Indicates the configuration register access type, read or write. For writes, indicates the byte enables: The following encodings are defined: 4'b0000: Read 4'b0001: Write byte 0 4'b0010: Write byte 1 4'b0100: Write byte 2 4'b1000: Write byte 3 4'b1111: Write all bytes. Combinations of byte enables, for example,4'b 0101b are also valid.

The following figure shows timing diagram for write command; the first command sends write for all four bytes of register located at address=4. The ss_app_st_cebreq_tdata[29] low indicates the access is for a physical function.

The second command sends write for byte3 and byte2 of register located at address =8. The ss_app_st_cebreq_tdata[29] high indicates access is for a virtual function.

Figure 38. Timing Diagram for Configuration Extension Bus Request Interface



6.4.2. Configuration Extension Bus Response Interface (st_cebresp)

The application returns read data for requests received from "st_cebreq" interface using "st_cebresp" interface. The IP is always ready to accept responses from the application. The application will provide response data with valid qualifier.

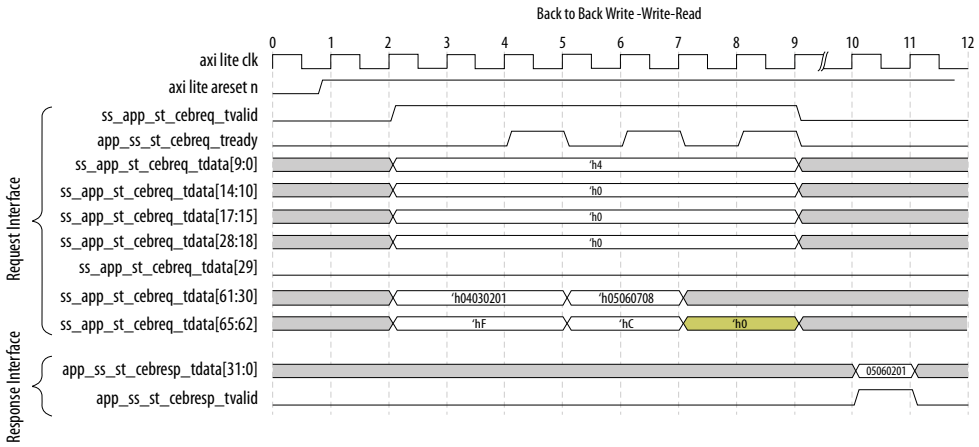
This interface is mutually exclusive with the Configuration Intercept Response Interface.

Table 45. Configuration Extension Bus Response Interface

Signal Name	Direction	Clock Domain	Description
app_ss_st_cebresp_tvalid	Input	axi_lite_clk	Application assert this signal for one clock to indicate that valid data is driven on app_ss_st_cebresp_tdata bus.
app_ss_st_cebresp_tdata[31:0]	Input	axi_lite_clk	Response data from application for read request issued using "st_cebreq interface".

The following figure shows timing diagram for back-to-back write and read command; the first command sends write for all four bytes of register located at address=4. The second command sends write for byte3 and byte2 of same register. The third command sends read for same register. Upon receiving the read command on st_cebreq interface, the application returns data on st_cebresp interface. You must note that the data returned is 5621. The upper two bytes were modified by the second write.

Figure 39. Timing Diagram for Configuration Extension Bus Response Interface



6.5. Configuration Intercept Interface

The IP allows application logic to intercept configuration read and configuration write requests using this interface. The interface follows AXI Streaming interface protocol with ready valid handshake. The interface will support a maximum of one outstanding request at a time. The IP provides st_ciireq and st_ciiresp interfaces for intercepting packets.

Note:

1. This interface is provided so that the PCIe IP is backward compatible to legacy application logic that relies on CII for their functionality. Newly defined application logic should avoid using the CII interface and move to the CEB interface.
2. This interface is mutually exclusive with the Configuration Extension Bus Request Interface.

This interface is applicable only when operating as an Endpoint in Power User mode.

6.5.1. Configuration Intercept Request Interface (st_ciireq)

Table 46. Configuration Intercept Request Interface

Signal Name	Direction	Clock Domain	Description
ss_app_st_ciireq_tvalid	Output	axi_lite_clk	When asserted, indicates a valid CFG request cycle is waiting to be intercepted. Deasserted when app_ss_st_ciireq_tready is asserted.
app_ss_st_cciireq_tready	Input	axi_lite_clk	Application asserts this signal for one clock to acknowledge ss_app_st_ciireq_tvalid is seen by responder.
ss_app_st_ciireq_tdata[0]	Output	axi_lite_clk	hdr_poisoned: The poisoned bit in the received TLP header on the CII.
ss_app_st_ciireq_tdata[4:1]	Output	axi_lite_clk	hdr_first_be: The first dword byte enable field in the received TLP header on the CII.

continued...

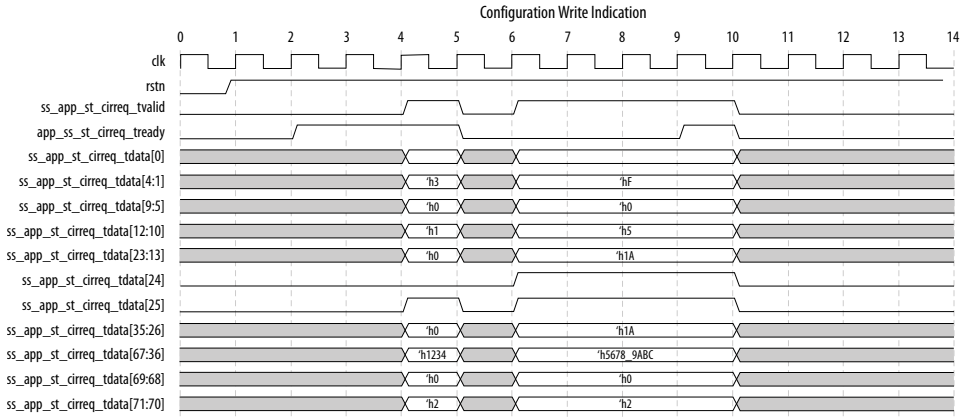
Signal Name	Direction	Clock Domain	Description
ss_app_st_ciireq_tdata[9:5]	Output	axi_lite_clk	slot_num: The slot number in the received TLP header on the CII.
ss_app_st_ciireq_tdata[12:10]	Output	axi_lite_clk	func_num: The PF number in the received TLP header on the CII.
ss_app_st_ciireq_tdata[23:13]	Output	axi_lite_clk	vf_num: The child VF number of parent PF in the received TLP header on the CII.
ss_app_st_ciireq_tdata[24]	Output	axi_lite_clk	vf_active: Indicates VF number is valid in the received TLP header on the CII.
ss_app_st_ciireq_tdata[25]	Output	axi_lite_clk	wr: Indicates a configuration write request detected in the received TLP header on the CII. Also, indicates that st_app_st_ciireq_tdata[67:36] is valid.
ss_app_st_ciireq_tdata[35:26]	Output	axi_lite_clk	addr: The double word register address in the received TLP header on the CII.
ss_app_st_ciireq_tdata[71:36]	Output	axi_lite_clk	dout: Received TLP payload data from the link partner to your application client. The data is in little endian format. The first received payload byte is in [43:36].

The following figure shows timing diagram for configuration write request indication to the application when intercept feature is not enabled.

The first command is a configuration write to PF1 byte-1 and byte-0 at address=0x200. The tvalid is high for 1 clock cycle as the application is ready to accept the packet.

The second command is a full dword configuration write to VF=26 of PF5 at address=0x3F0. As the application is not ready to accept the packet, sub-system holds the information till tready is seen

Figure 40. Configuration Write Request Indication



6.5.2. Configuration Intercept Response Interface (st_ciiresp)

The application must return the response for request received on "st_ciireq" interface using "st_ciiresp" interface. The IP is always ready to accept responses from the application. The application will provide response data with valid qualifier.

This interface is applicable only when operating as Endpoint in Power User mode. It is mutually exclusive with the Configuration Extension Bus Response Interface.

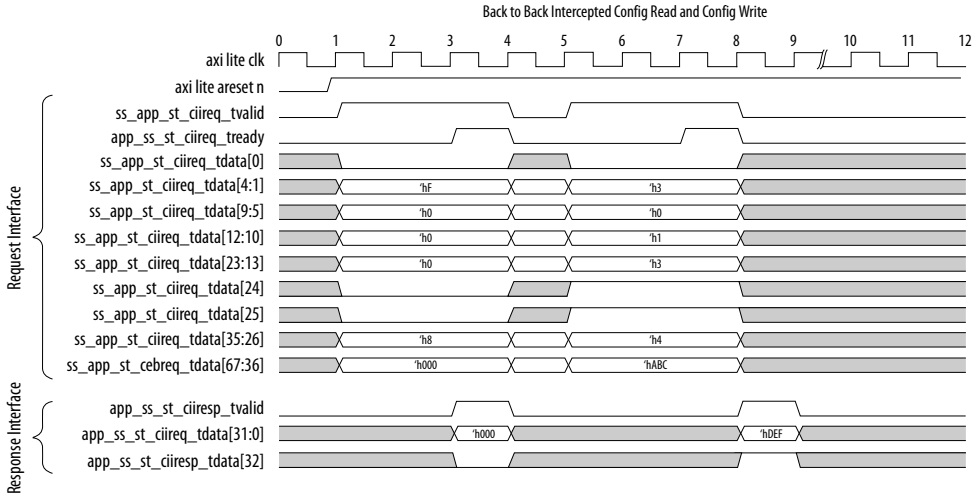
Table 47. Configuration Intercept Response Interface

Signal Name	Direction	Clock Domain	Description
app_ss_st_ciiresp_tvalid	Input	axi_lite_clk	Application asserts this signal for one clock to indicate that valid data is driven on app_ss_st_ciiresp_tdata bus.
app_ss_st_ciiresp_tdata[31:0]	Input	axi_lite_clk	Override data from application for the intercepted configuration request on "st_ciireq interface". For CfgWr: override the write data to the Configuration register with data supplied by the application logic. For CfgRd: override the data payload of the completion TLP with data supplied by the application logic.
app_ss_st_ciiresp_tdata[32]	Input	axi_lite_clk	Override Data Enable: Application assert this signal to override the CfgWr payload or CfgRd completion using the data supplied by the application logic on app_ss_st_ciiresp_tdata[31:0] bus.

The following figure shows timing diagram for back-to-back read and write request; the first request sends configuration read on st_ciireq interface for all four bytes of register located at address=0x8. Application decides not to intercept this configuration read and hence return tvalid=1 together with tdata[32]=0 on the st_ciiresp interface.

After receiving the first response on the st_ciiresp interface, the second request sends configuration write for byte0, byte1 and byte2 of register located at address=0x4 with data value of 0xABC. Application decides to intercept this configuration write and hence return tvalid=1 together with tdata[32]=1 on the st_ciiresp interface. Additionally, application provides the data (i.e., 0xDEF) to be used for the intercepted configuration write on the st_ciiresp interface through tdata[31:0].

Figure 41. Timing Diagram for Configuration Intercept Interface



6.6. Function Level Reset Interface

Each of the functions in the IP can be individually reset through the function level reset interface. The IP provides st_flrrcvd and st_flrcmpl interfaces for FLR handshake.

6.6.1. Function Level Reset Received Interface (st_flrrcvd)

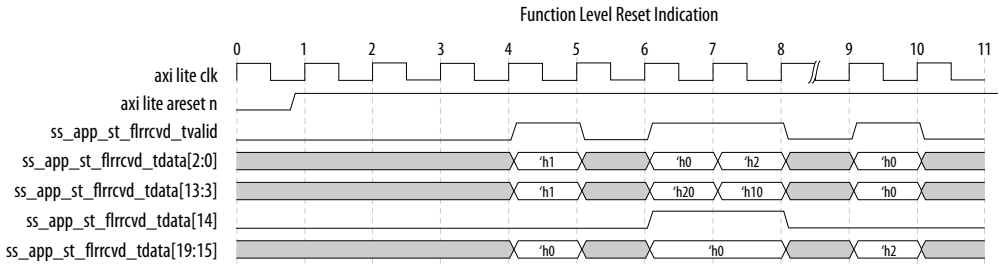
The st_flrrcvd interface provides indication to the application for which function FLR is received from HOST. The application responds with FLR completion using st_flrcmpl interface, indicating FLR process is completed. The IP does not implement any timeout for this handshake, so it is important that application sends FLR completion for FLR request received on st_flrrcvd interface.

Table 48. Function Level Reset Received Interface

Signal Name	Direction	Clock Domain	Description
ss_app_st_flrrcvd_tvalid	Output	axi_lite_clk	When asserted, indicates a FLR request received from HOST. The signal is valid for one clock cycle.
ss_app_st_flrrcvd_tdata[19:0]	Output	axi_lite_clk	[2:0] - The PF Number of FLR Request [13:3] - Indicates child VF Number of parent PF indicated by PF Number [14] - Indicates request is for Virtual Function implemented in slot's physical function [19:15] - The slot Number of FLR Request

The following figure shows timing diagram for function level reset indication to application. The first command indicates FLR for Physical Function =1 on slot = 0. The second and third back-to-back indications are for VF, the ss_app_st_flrrcvd_tdata[14] high indicates FLR is received for Virtual function. The fourth command signals FLR for PF=0 and slot=2.

Figure 42. Timing Diagram for Function Level Reset Receive Interface



6.6.2. Function Level Reset Completion Interface (st_flrcmpl)

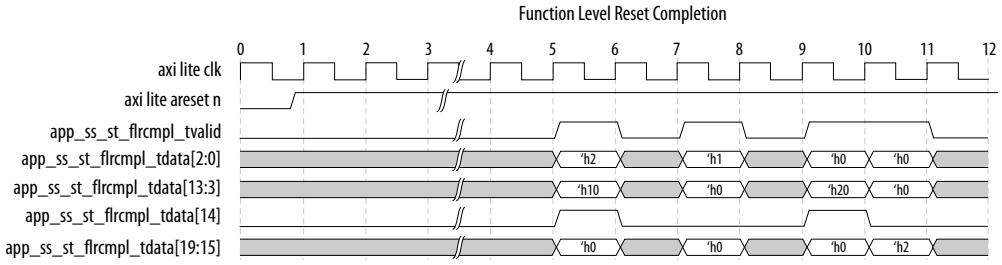
The st_flrrcvd interface provides indication to the application for which function FLR is received from HOST. The application responds with FLR completion using st_flrcmpl interface, indicating FLR process is completed. The IP does not implement any timeout for this handshake, so it is important that application sends FLR completion for FLR request received on st_flrrcvd interface.

Table 49. Function Level Reset Completion Interface

Signal Name	Direction	Clock Domain	Description
app_ss_st_flrcmpl_tvalid	Input	axi_lite_clk	When asserted, indicates a FLR request completed by application. The signal is valid for one clock cycle.
app_ss_st_flrcmpl_tdata[19:0]	Input	axi_lite_clk	[2:0] - The PF Number of FLR Completion [13:3] - Indicates child VF Number of parent PF indicated by PF Number [14] - Indicates completion is from Virtual Function implemented in slot's physical function [19:15] - The slot Number of FLR completion

The following figure shows timing diagram for function level reset completion from application. The first completion indicates FLR completion for Virtual Function =0x10, the app_ss_flrcmpl_tdata[14] high indicates FLR completion from Virtual function. The second completion indicates FLR completion for Physical Function =0x1. The third completion indicates FLR completion for Virtual Function =0x20, the app_ss_flrcmpl_tdata[14] high indicates FLR completion from Virtual function. The fourth completion indicates FLR completion for Physical Function =0x0 in "slot = 2"

Figure 43. Timing Diagram for Function Level Reset Completion Interface



6.7. Control Shadow Interface (st_ctrlshadow)

The control shadow interface is used to bring out the settings of the various configuration register fields of the function. These fields are often required in designing the control path of the application layer logic.

The application logic decodes information provided on this interface to create a shadow copy. The interface provides update to primary control signals only. The application logic must read extra information required through lite_csr interface by reading configuration register of interest. The sections below explain additional information required by application.

Bus Master Enable

The application logic requires the BME information to determine if it can generate request for a particular function. Each function in the application logic cannot generate bus master requests unless its corresponding BME is set. The application logic monitors control shadow interface for BME event for this purpose. Since PCIe SS does not autonomously generate bus master request by itself, it will not qualify the transmit path with BME settings and solely relies on application.

MSI Enable

The application logic requires MSI Address and MSI Data information from MSI capability to generate MSI interrupt. The application logic monitors control shadow interface for MSI Enable event to read this additional information from MSI capability.

VF Enable

The Virtual Function in application logic cannot generate any traffic unless they are enabled by HOST. The number of VFs enabled can be different than number of VFs advertised as initial VFs. The application logic can find number of VFs visible by reading NumVFs register in SRIOV capability. The read of this register must be triggered after VF Enable bit is set by HOST.

TPH Req Enable

The function can support all operational modes of TPH, no ST mode, interrupt vector mode or device specific mode. The HOST communicates mode of operation by writing ST Mode Select bits in TPH requester control register. The application reads this register and generates traffic only when TPH requester enable bit is set.

ATS Enable

The function can read Smallest Translation Unit field from ATS control register when ATS enable bit is set.

Table 50. Control Shadow Interface

Signal Name	Direction	Clock Domain	Description
ss_app_st_ctrlshadow_tvalid	Output	axi_lite_clk	The IP asserts this output for one clock cycle when there is an update to the register fields being monitored, because of a Configuration Write performed by the Root Comple.g., The user can copy the new settings of the register fields from the tdata bus.
ss_app_st_ctrlshadow_tdata[39:0]	Output	axi_lite_clk	When app_ctrl_shadow_tvalid has been asserted, this output provides the current settings of the register fields of the associated Function. [2:0] - Identifies the physical function Number of configuration register [13:3] - Identifies the virtual function Number of configuration register [14] - Indicates information is for Virtual Function implemented in slot's physical function [19:15] - Identifies the slot Number of configuration register [20] - Bus Master Enable [21] - MSI-X Mask [22] - MSI-X Enable [23] - Mem Space Enable [24] - ExpRom Enable [25] - TPH Req Enable [26] - ATS Enable [27] - MSI Enable [28] - MSI Mask [29] - Extended Tag [30] - 10 Bit Tag Req Enable [31] - PTM Enable

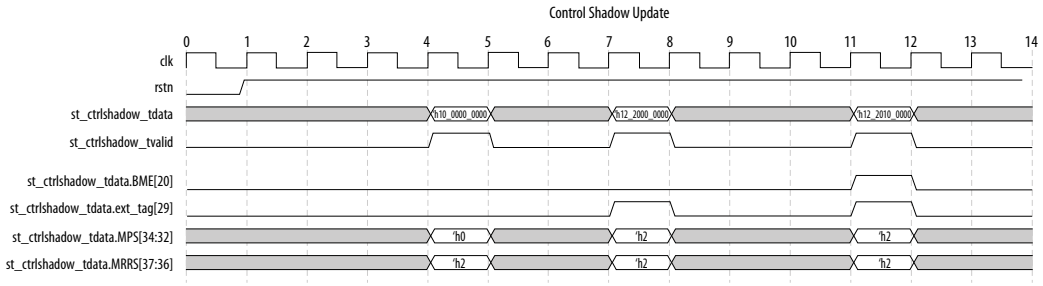
continued...

Signal Name	Direction	Clock Domain	Description
			[34:32] - MPS Size [37:35] - MRRS Size [38] - VF Enable [39] - Page Request Enable

ATS Enable Timing Diagram

The following figure shows output on the Control Shadow interface when there is an update to the control shadow bits in the HIP configuration register.

Figure 44. ATS Enable Timing Diagram



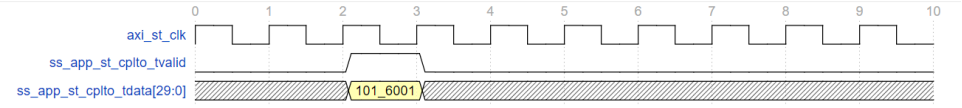
6.8. Completion Timeout Interface (st_cplto)

The completion timeout interface indicates completion timeout event to application. The interface provides the function number and tag number of the outstanding request timed out.

Table 51. Completion Timeout Interface

Signal Name	Direction	Clock Domain	Description
ss_app_st_cplto_tvalid	Output	axi_lite_clk	tvalid indicates that the completion timeout received for outstanding NP request.
ss_app_st_cplto_tdata[29:0]	Output	axi_lite_clk	Carries completion Timeout Information [9:0] - Tag Number [12:10] - PF Number, indicates parent PF number of VF when VF Active is high else PF Number of function [23:13] - VF Number, indicates VF number when VF Active js high [24] - VF Active, Indicates timeout is for VF [29:25] always 0. Reserved

Figure 45. Completion Timeout Interface Timing Diagram



6.9. Miscellaneous Signals

Table 52. Miscellaneous Signals

Signal Name	Direction	Clock Domain	Description
ss_app_serr	Output	axi_st_clk	Indicates System Error is detected. In TLP Bypass Mode indicates PL/DL/TL layer error detected by HardIP.
ss_app_linkup	Output	axi_st_clk	When asserted, this signal indicates the link is up.
ss_app_dlup	Output	axi_st_clk	Indicates Data Link Layer is UP.
ss_app_int_status	Output	axi_st_clk	This signal drives legacy interrupts to the Application Layer. The source of the interrupt will be logged in the Root Port Interrupt Status registers in the Port Configuration and Status registers. *Note:* Applicable only in Root Port Mode.
ss_app_surprise_down_err	Output	Async	Indicates that a surprise down event is occurring in the HardIP controller.
ss_app_ltssmstate	Output	axi_st_clk	Indicates the LTSSM state.
ss_app_rx_par_err	Output	axi_st_clk	Indicates a parity error detected at the input of the HIP'S RX buffer. Asserts for a single cycle. Note: Application must reset the HardIP if this occurs because parity errors can leave the HardIP in an unknown state.
ss_app_tx_par_err	Output	axi_st_clk	Indicates a parity error during TX TLP transmission at the HIP. Asserts for a single cycle.

6.10. Control and Status Register Responder Interface (lite_csr)

The IP provides a Control and Status Register Interface to access registers implemented in its modules. You can access PCI/PCIe Configuration Registers of all Functions through this interface as well as SS soft register space registers implemented in design. The interface follows AXI4-Lite protocol.

The IP does not differentiate between non-secure and secure accesses. All accesses are considered secure.

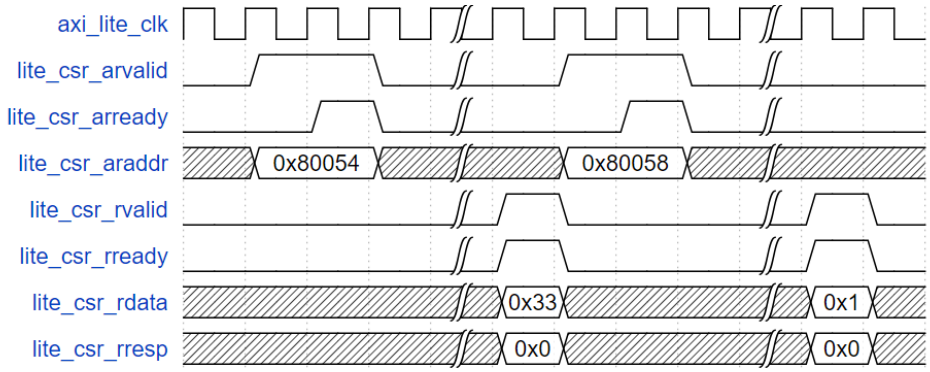
Table 53. Control and Status Register Responder Interface

Signal Name	Direction	Clock Domain	Description
Write Address Channel			
app_ss_lite_csr_awvalid	Input	axi_lite_clk	Indicates that the write address channel signals are valid.
ss_app_lite_csr_awready	Output	axi_lite_clk	Indicates that a transfer on the write address channel can be accepted.
app_ss_lite_csr_awaddr[LiteSlvAWD-1:0]	Input	axi_lite_clk	The address of the first transfer in a write transaction. The default value of LiteSlvAWD = 18

continued...

Signal Name	Direction	Clock Domain	Description
Write Data Channel			
app_ss_lite_csr_wvalid	Input	axi_lite_clk	Indicates that the write data channel signals are valid.
ss_app_lite_csr_wready	Output	axi_lite_clk	Indicates that a transfer on the write data channel can be accepted.
app_ss_lite_csr _wdata[LiteSlvDWD-1:0]	Input	axi_lite_clk	Write Data The default value of LiteSlvDWD=32
app_ss_lite_csr_wstrb[LiteSlvDWD/8-1:0]	Input	axi_lite_clk	Write strobes, indicate which byte lanes hold valid data.
Write Response Channel			
ss_app_lite_csr_bvalid	Output	axi_lite_clk	Indicates that the write response channel signals are valid.
app_ss_lite_csr_bready	Input	axi_lite_clk	Indicates that a transfer on the write response channel can be accepted.
ss_app_lite_csr_bresp[1:0]	Output	axi_lite_clk	Write response, indicates the status of a write transaction.
Read Address Channel			
app_ss_lite_csr_arvalid	Input	axi_lite_clk	Indicates that the read address channel signals are valid.
ss_app_lite_csr_arready	Output	axi_lite_clk	Indicates that a transfer on the read address channel can be accepted.
app_ss_lite_csr_araddr[LiteSlvAWD-1:0]	Input	axi_lite_clk	The address of the first transfer in a read transaction. The default value of LiteSlvAWD = 18
Read Data Channel			
ss_app_lite_csr_rvalid	Output	axi_lite_clk	Indicates that the read data channel signals are valid.
app_ss_lite_csr_rready	Input	axi_lite_clk	Indicates that a transfer on the read data channel can be accepted.
ss_app_lite_csr_rdata[LiteSlvDWD:0]	Output	axi_lite_clk	Read data The default value of LiteSlvDWD=32
ss_app_lite_csr_rresp[1:0]	Output	axi_lite_clk	Read response, indicates the status of a read transfer.

Figure 46. Control and Status Register Responder Interface Timing Diagram



6.11. VF Error Flag Interface (vf_err/sent_vfnonfatalmsg)

When SRIOV is enabled, the PCIe IP provides a passage for the HIP's VF Error Flag Interface to application logic. In the absence of AER and Error Message Generation support for VF in the HIP, the generation of VF's Non-Fatal Error messages relies on the user application logic. It is up to the user application logic to generate appropriate PCIe error messages when specific error conditions occur (as indicated by this interface).

Note: VF Non-Fatal errors reported through this interface would have their error status logged in the HIP registers already.

This interface exists when SRIOV is enabled only. N/A to PCIe device type is Root Port.

Table 54. VF Error Flag Interface

Signal Name	Direction	Clock Domain	Description
ss_app_vf_err_poisonedwrreq_<w>	Output	axi_lite_clk	Indicates a Poisoned Write Request is received.
ss_app_vf_err_poisonedcompl_<w>	Output	axi_lite_clk	Indicates a Poisoned Completion is received.
ss_app_vf_err_ur_postedreq_<w>	Output	axi_lite_clk	Indicates the IP core received a Posted UR request.
ss_app_vf_err_ca_postedreq_<w>	Output	axi_lite_clk	Indicates the IP core received a Posted CA request.
ss_app_vf_err_vf_num[10:0]_<w>	Output	axi_lite_clk	Indicates the VF number for which the error is detected.
ss_app_vf_err_func_num[2:0]_<w>	Output	axi_lite_clk	Indicates the physical function number associated with the VF that has the error.
/ss_app_vf_err_overflow	Output	axi_lite_clk	Indicates a VF error FIFO overflow and a loss of an error report. The overflow can happen when axi_lite_clk is slower than the coreclkout_hip clock. It can also overflow internally in the HIP.
app_ss_sent_vfnonfatalmsg	Input	axi_lite_clk	Indicates the user application sent a non-fatal error message in response to an error detected.
app_ss_vfnonfatalmsg_vf_num[10:0]	Input	axi_lite_clk	Indicates the VF number for which the error message was generated. This bus is valid when app_ss_sent_vfnonfatalmsg is high.
app_ss_vfnonfatalmsg_func_num[2:0]	Input	axi_lite_clk	Indicates the PF number associated with the VF with the error. This bus is valid when app_ss_sent_vfnonfatalmsg is high.

6.12. VIRTIO PCI Configuration Access Interface

The VIRTIO PCI Configuration Access Interface is provided to allow application to implement the VIRTIO PCI Configuration Access Data register functionality. The VIRTIO specification allows software to use the VIRTIO PCI Configuration Access capability register as an alternative method to access VIRTIO device region. When this interface is enabled, the PCIe IP provides a passage for the HIP's VIRTIO PCI

Configuration Access Interface to application logic. When this interface is disabled, the PCIe IP internally drops writes from HIP's VIRTIO PCI Configuration Access Interface and returns 0's for reads (per the requested byte length).

Note:

1. Only the first 3 bits of QHIP's virtio_pcicfg_length_o[31:0] will be used since length is restricted by VIRTIO specification to be 1, 2 or 4 only.
2. The QHIP's virtio_pcicfg_appvfnum_i and virtio_pcicfg_apppfnum_i are not used by QHIP. The PCIe SS can tie-off these to 0's.
3. The QHIP's virtio_pcicfg_rdbe_i[3:0] needs to be internally driven by PCIe SS based on the pending read length on the st_virtio_pcicfgreq interface.
4. When the QHIP is not instantiated, the VirtIO capability structure will be included in the SS.

6.12.1. VIRTIO PCI Config Access Request Interface (st_virtio_pcicfgreq)

Table 55. VIRTIO PCI Configuration Access Request Interface

Signal Name	Direction	Clock Domain	Description
ss_app_virtio_pcicfgreq_tvalid	Output	axi_lite_clk	When asserted, indicates a VIRTIO PCI Configuration Access Request received from HOST. The signal is valid for one clock cycle.
ss_app_virtio_pcicfgreq_tdata[95:0]	Output	axi_lite_clk	[0] - When set, the request is a write request. Else, the request is a read request. [1] - Indicates request is for Virtual Function implemented in slot's physical function [12:2] - Indicates child VF Number of parent PF indicated by PF Number [15:13] - The PF Number of the Request [20:16] - The Slot Number of the Request [28:21] - The BAR value to be used for the Request [60:29] - The BAR Offset value to be used for the Request [63:61] - The Length value to be used for the Request [95:64] - The Data value to be used for the Write Request. N/A for Read Request.

6.12.2. VIRTIO PCI Config Access Completion Interface (st_virtio_pcicfgcmpl)

Table 56. VIRTIO PCI Configuration Access Completion Interface

Signal Name	Direction	Clock Domain	Description
app_ss_virtio_pcicfgcmpl_tvalid	Input	axi_lite_clk	When asserted, indicates a VIRTIO PCI Configuration Access Completion to be returned to HOST. The signal is valid for one clock cycle.
app_ss_virtio_pcicfgcmpl_tdata[31:0]	Input	axi_lite_clk	[31:0] - The completion Data value.

6.13. Serial Data Signals

The AXI Streaming Intel FPGA IP for PCI Express for PCI Express natively supports 4, 8, or 16 PCIe lanes. Each lane includes a TX differential pair and an RX differential pair. Data is striped across all available lanes. Refer to the table Variables Used in the Bus Indices for more details on bus indices.

The following table shows the signals of the Serial Interface of the AXI Streaming Intel FPGA IP for PCI Express.

Table 57. Serial Data Signals

Signal Name	Direction	Clock Domain	Description
tx_p_out[<b>-1:0], tx_n_out[<b>-1:0]	Output	N/A	Transmit serial data outputs using the High-Speed Differential I/O standard.
rx_p_in[<b>-1:0], rx_n_in[<b>-1:0]	Input	N/A	Receive serial data inputs using the High-Speed Differential I/O standard.

7. Register Descriptions

The subsequent sections describe the PCIe IP registers in detail. The following table lists definitions for the acronyms used in the "Attribute User Side" column.

Table 58. Register Attribute Definition

Attribute	Definition
RW	Read Write
RWS	Read Write Sticky
RO	Read Only
ROS	Read Only Sticky
WO	Write Only
RW1S	Read Write 1 to Set. Clear by Hardware
RW1C	Read Write 1 to Clear. Set by Hardware
RW1CS	Read Write 1 to Clear sticky
RsvdZ	Reserved, Return 0
Hwinit	Hardware Initiate

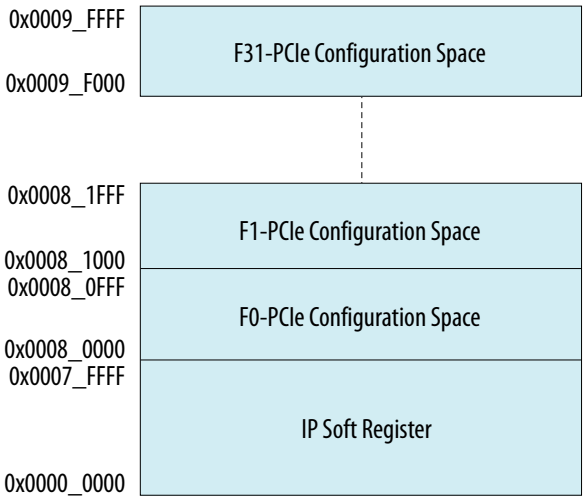
7.1. Register Address Map

The application can access registers of the IP as well as PCIe configuration space registers of physical functions present in the design.

The following figure shows the address map of registers when accessing them from the AXI Lite CSR Interface (lite_csr).

The following sections describe the register addresses and bit mappings for each register space.

Figure 47. IP Address Map



All AXI-Lite accesses are completed with appropriate response (BRESP, RRESP) so that the bus does not stall.

- AXI-Lite access to address ranges defined in Register Address Map shall be completed successfully with BRESP/RRESP="OK". This includes access to unimplemented, i.e., Reserved, register offsets within the valid address range.
  - Read to reserved register shall return value of all zeroes.
  - Write to register location containing any number of RO or RO/V bit, design shall return write response BRESP=OKAY. Write is dropped for the RO or RO/V bit location(s). This is not an error condition.
- AXI-Lite access to address beyond the register map should be completed gracefully with BRESP/RRESP="DECERR". Read returns all zeroes and write is dropped.

7.2. PCIe Configuration Space

7.2.1. PCIe Configuration Space Registers

Refer to the *Appendix A – Table PCIe Configuration Space registers for x16/x8/x4 controllers* of the *P-Tile Avalon Streaming Intel FPGA IP for PCI Express User Guide* for register details.

Refer to the *Appendix A – Configuration Space Registers* of the *F-Tile Avalon Streaming Intel FPGA IP for PCI Express User Guide* for register details.

Refer to the *Appendix A – Configuration Space Registers* of the *R-Tile Avalon Streaming Intel FPGA IP for PCI Express User Guide* for register details.

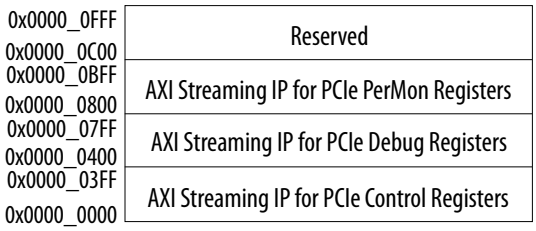
Related Information

- P-Tile Avalon Streaming Intel FPGA IP for PCI Express User Guide
- F-Tile Avalon Streaming Intel FPGA IP for PCI Express User Guide
- R-Tile Avalon Streaming Intel FPGA IP for PCI Express User Guide

7.3. AXI Streaming Intel FPGA IP for PCI Express Soft Register Address Map

The AXI Streaming Intel FPGA IP for PCI Express address map is shown in the following figure.

Figure 48. AXI Streaming Intel FPGA IP for PCI Express Soft Register Address Map



7.3.1. AXI Streaming Intel FPGA IP for PCI Express Control Registers

The following table lists the control registers implemented by the IP. The IP Control registers start from Base Address = 0x0.

Table 59. AXI Streaming Intel FPGA IP for PCI Express Control Registers Address Map

Register Name	Offset
IP Version	0x0000_0000
IP Features	0x0000_0004
IP Interface Attributes	0x0000_0008
Reserved	0x0000_000C
ERROR GEN CTRL	0x0000_0010
ERROR GEN ATTR	0x0000_0014
ERROR TLP Header DW0	0x0000_0018
ERROR TLP Header DW1	0x0000_001C
ERROR TLP Header DW2	0x0000_0020
ERROR TLP Header DW3	0x0000_0024
ERROR TLP Prefix	0x0000_0028
Hot Plug Gen Control	0x0000_002C
Power Management Control	0x0000_0030
Legacy Interrupt Control	0x0000_0034
CFG REG IA Control	0x0000_00C8
CFG REG IA FN NUM	0x0000_00CC
CFG REG IA FN WRDATA	0x0000_00D0
CFG REG IA FN RDDATA	0x0000_00D4
PRS Control	0x0000_00D8

continued...

Register Name	Offset
MSI Pending Control	0x0000_00DC
MSI Pending	0x0000_00E0
D-State STS	0x0000_00E4
CFG Retry Control	0x0000_00E8

7.3.1.1. AXI Streaming Intel FPGA IP for PCI Express Version

The register indicates the AXI Streaming Intel FPGA IP for PCI Express major and minor versions.

Default Value: 0x0100_0000

Table 60. AXI Streaming Intel FPGA IP for PCI Express Version Register

Register Name	Bit	Attribute User Side	Description
IP Version	7-0	RO	Reserved
	15-8	RO	Indicates Minor Version Number
	31-16	RO	Indicates Major Version Number

7.3.1.2. AXI Streaming Intel FPGA IP for PCI Express Features

The register indicates features enabled in the AXI Streaming Intel FPGA IP for PCI Express during compile time.

Default Value: Set As per Parameter Settings

Table 61. AXI Streaming Intel FPGA IP for PCI Express Feature Registers

Register Name	Bit	Attribute User Side	Description
IP Features 1	1-0	RO	Reflects Functional Mode parameter value 00 - Power User mode 01 - Reserved 10 - Reserved 11 - Reserved
	2	RO	Reserved
	3	RO	Indicates presence of Debug Toolkit block in a design 0 - Debug Toolkit not Present 1 - Debug Toolkit Present
	4	RO	Reserved
	5	RO	Reserved
	8-6	RO	Multiple AXI Stream Support 000 - Single Stream Present 001 - Two Stream Present All Others - Reserved
	10-9	RO	AXI-ST Header and Data Packing Scheme

continued...

Register Name	Bit	Attribute User Side	Description
			00 - Simple Packing 10 - Compact Packing
	31-11	RO	Reserved

7.3.1.3. AXI Streaming Intel FPGA IP for PCI Express Interface Attributes

The register indicates the AXI Streaming Intel FPGA IP for PCI Express interface attributes settings during compile time.

Default Value: Set As per Parameter Settings

Table 62. AXI Streaming Intel FPGA IP for PCI Express Interface Attributes

Register Name	Bit	Attribute User Side	Description
IP Interface Attributes	3-0	RO	Reflects AXI-ST Initiator Interface ready_latency setting
	7-4	RO	Reserved
	11-8	RO	Reflects AXI-MM Initiator Interface ready_latency setting
	14-12	RO	Indicates AXI-ST Interface Width 000 - 32 bits 001 - 64 Bits 010 - 128 Bits 011 - 256 Bits 100 - 512 Bits All others - Reserved
	17-15	RO	Indicates AXI-Lite Interface Width 000 - 32 bits 001 - 64 Bits Reserved
	20-18	RO	Reserved
	31-21	RO	Reserved

7.3.1.4. ERROR GEN CTRL

The following table lists details of ERROR GEN CTRL from Application Error Reporting Registers. If header and prefix logging is required, then you must set the ERROR GEN CTRL register after ERROR TLP Header DWn and ERROR TLP Prefix registers are populated with required information.

Default Value: 0x0000_0000

Register Name	Bit	Attribute User Side	Description
ERROR GEN CTRL	0	RW1S	Error Gen Trigger This bit will trigger Error Handling based on application error attributes at ERROR GEN ATTR, ERROR TLP Header DWn and ERROR TLP Prefix registers. Hence, this bit should only be set once the application error attributes are programmed with the correct values.

continued...

Register Name	Bit	Attribute User Side	Description
			Once this bit is set, the application error attributes should not be changed until the operation is completed. The IP will report these error(s) by utilizing the application error reporting mechanism. The IP will clear this bit when the requested operation is complete.
	1	RW	Log header If header logging is required, this bit must be set. The header must be supplied using Header Register *Note:* TLP Header error logging feature is not available for VF related errors (since AER is not supported for VF)
	2	RW	Log Prefix If prefix logging is required, this bit must be set. The prefix must be supplied using Prefix Register *Note:* TLP Prefix error logging feature is not available for VF related errors (since AER is not supported for VF)
	6-3	RsvdZ	Reserved
	7	RW	VF Active Indicates error reporting function is for a Virtual Function *Note:* Not supported in the current Quartus release
	12-8	RW	PF Number Indicates PF Number of function reporting an error. *Note:* Current Quartus release supports up to 8 PFs only.
	13	RsvdZ	Reserved
	24-14	RW	VF Number Indicates VF Number of function reporting an error *Note:* 1. Valid only when VF Active is set. 2. Not supported in the current release.
	25	RsvdZ	Reserved
	30-26	RW	Slot Number Indicates Slot Number of function reporting an error
	31	RsvdZ	Reserved

7.3.1.5. ERROR GEN ATTR

The following table lists details of ERROR GEN ATTR from Application Error Reporting Registers. When ERROR GEN CTRL operation is triggered and pending, the ERROR GEN ATTR should not be programmed with new error values, otherwise newly updated error(s) might not be treated as new errors. It might also lead to pending error(s) being reported incorrectly.

Default Value: 0x0000_0000

Register Name	Bit	Attribute User Side	Description
ERROR GEN ATTR	0	RW	Advisory Indicates application error (if applicable) is an advisory error. Examples when application can assert this are: • Treating a poisoned TLP as normal TLP • Detected Completion Timeout but intends to resend the request • Returning UR completion for a request that is treated as Unsupported Request • Returning CA completion for a request that is treated as Completer Abort • Receiving an unexpected completion
	1	RW	Unexpected Completion Error This bit should be set when an Application Layer master block detects an unexpected Completion. The IP responds to this by reporting a Fatal or Correctable Error to the Root Complex, depending on the severity of the Unexpected Completion Received error programmed in the AER Uncorrectable Error Severity Register of the Function (if not masked).
	2	RW	Completer Abort Error This bit should be set when Application Layer has treated a request as a Completer Abort (CA). The IP responds to this by reporting a Fatal or Non-Fatal/Correctable Error to the Root Complex, depending on the severity of the Completer Abort error programmed in the AER Uncorrectable Error Severity Register of the Function (if not masked).
	3	RW	Completion Timeout Error This bit should be set when a master-like interface has transmitted a Non-Posted request that never receives a corresponding Completion from the link and the error is not correctable. The IP responds to this by reporting a Fatal or Non-Fatal/Correctable Error to the Root Complex, depending on the severity of the Completion Timeout error programmed in the AER Uncorrectable Error Severity Register of the Function (if not masked).
	4	RW	Unsupported Request Error This bit should be set when Application Layer has treated a request as an Unsupported Request. The IP responds to this by reporting a Fatal or Non-Fatal/ Correctable Error to the Root Complex, depending on the severity of the Unsupported Request error programmed in the AER Uncorrectable Error Severity Register of the Function (if not masked).
	5	RW	Poisoned TLP Received Error This bit should be set when Application Layer has treated a request as poisoned. The IP responds to this by reporting a Fatal or Non-Fatal/Correctable Error to the Root

continued...

Register Name	Bit	Attribute User Side	Description
			Complex, depending on the severity of the Poisoned TLP Received error programmed in the AER Uncorrectable Error Severity Register of the Function (if not masked).
	6	RW	ECRC Check Failed Error This bit should be set when Application Layer has detected ECRC Check Failed error. The IP responds to this by reporting a Fatal or Non-Fatal/Correctable Error to the Root Complex, depending on the severity of the ECRC Check Failed error programmed in the AER Uncorrectable Error Severity Register of the Function (if not masked). Note: Not supported in the current Quartus release.
	7	RW	AtomicOp Egress Blocked Error This bit should be set when Application Layer has encountered AtomicOp Egress Blocked error. The IP responds to this by reporting a Fatal or Non-Fatal/Correctable Error to the Root Complex, depending on the severity of the AtomicOp Egress Blocked error programmed in the AER Uncorrectable Error Severity Register of the Function (if not masked). *Note:* Not supported in the current Quartus release
	8	RW	Uncorrectable Internal Error This bit should be set when Application Layer has encountered Uncorrectable Internal error. The IP responds to this by reporting a Fatal or Non-Fatal Error to the Root Complex, depending on the severity of the Uncorrectable Internal error programmed in the AER Uncorrectable Error Severity Register of the Function (if not masked). *Note:* Not supported in the current Quartus release.
	9	RW	Corrected Internal Error This bit should be set when Application Layer has corrected an internal error. The IP responds to this by reporting a Correctable Error to the Root Complex (if not masked). *Note:* Not supported in the current Quartus release.
	10	RW	TLP Prefix Blocked Error This bit should be set when Application Layer has encountered TLP Prefix Blocked error. The IP responds to this by reporting a Fatal or Non-Fatal/Correctable Error to the Root Complex, depending on the severity of the TLP Prefix Blocked error programmed in the AER Uncorrectable Error Severity Register of the Function (if not masked). *Note:* Not supported in the current Quartus release.
	11	RW	ACS Violation Error

continued...

Register Name	Bit	Attribute User Side	Description
			This bit should be set when Application Layer has encountered ACS Violation error. The IP responds to this by reporting a Fatal or Non-Fatal/Correctable Error to the Root Complex, depending on the severity of the ACS Violation error programmed in the AER Uncorrectable Error Severity Register of the Function (if not masked). *Note:* Not supported in the current Quartus release.
	12	RW	MC Blocked TLP Error This bit should be set when an Application layer detected a MC Blocked TLP error. The IP responds to this by reporting a Fatal or Non-Fatal Error to the Root Complex, depending on the severity of the MC Blocked TLP error programmed in the AER Uncorrectable Error Severity Register of the Function (if not masked). *Note:* Not supported in the current Quartus release.
	13	RW	Poisoned TLP Egress Blocked Error This bit should be set when an Application layer detected a Poisoned TLP Egress Blocked error. The IP responds to this by reporting a Fatal or Non-Fatal Error to the Root Complex, depending on the severity of the Poison TLP Egress Blocked error programmed in the AER Uncorrectable Error Severity Register of the Function (if not masked). *Note:* Not supported in the current Quartus release.
	31-14	RsvdZ	Reserved

7.3.1.6. ERROR TLP Header DW0-3

This register holds PCIe TLP Header information of the error packet reported using ERROR GEN CTRL register. When ERROR GEN CTRL operation is triggered and pending, the ERROR TLP Header DWn should not be programmed with new values, otherwise wrong TLP Header values might be used.

Default Value: 0x0000_0000

Register Name	Bit	Attribute User Side	Description
Header DWn	31 - 0	RW	Holds TLP Header DWn of the reported error packet. n indicates Header DWORD index

7.3.1.7. ERROR TLP Prefix

This register holds PCIe TLP Prefix information of the error packet reported using ERROR GEN CTRL register. When ERROR GEN CTRL operation is triggered and pending, the ERROR TLP Prefix should not be programmed with new values, otherwise wrong TLP Prefix values might be used.

Default Value: 0x0000_0000

Table 63. Error TLP Prefix Register

Register Name	Bit	Attribute User Side	Description
Prefix	31 - 0	RW	Holds TLP Prefix of the reported error packet.

7.3.1.8. HOT PLUG GEN CTRL

The following table lists details of HOT PLUG GEN CTRL Register.

Default Value: 0x0000_0000

Table 64. Hot Plug Control Register

Register Name	Bit	Attribute User Side	Description
HOT PLUG GEN CTRL	0	RW	Attention Button Pressed Indicates that the system attention button was pressed. The IP passes on this information to HIP block and clears this bit indicating requested operation complete.
	1	RW	Power Fault Detection Indicates the power controller detected a power fault at this slot The IP passes on this information to HIP block and clears this bit indicating requested operation complete.
	2	RW	MRL Sensor Changed Indicates that the state of the MRL sensor has changed. The IP passes on this information to HIP block and clears this bit indicating requested operation complete.
	3	RW	Presence Detect Changed. Indicates that the state of the card presence detector has changed. The IP passes on this information to HIP block and clears this bit indicating requested operation complete.
	4	RW	Command Completed Indicates that the Hot Plug controller completed a command. The IP passes on this information to HIP block and clears this bit indicating requested operation complete.
	5	RW	Data Link Layer State Change Indicates the state of Data Link Layer Link Active bit of the Link. Status register is changed The IP passes on this information to HIP block and clears this bit indicating requested operation complete.

continued...

Register Name	Bit	Attribute User Side	Description
	6	RW	MRL Sensor State This bit reports the status of the MRL sensor 0 - MRL Closed 1 - MRL Open
	7	RW	Presence Detect State 0 - Slot Empty 1 - Adaptor Present
	8	RW	Electromechanical Interlock Status Indicates whether the system electromechanical interlock is engaged 0 - Electromechanical Interlock Disengaged 1 - Electromechanical Interlock Engaged
	13-9	RW	Slot Number Indicates Slot Number of Function Generating Hot Plug Even
	31 - 14	RsvdZ	Reserved

7.3.1.9. POWER MANAGEMENT CTRL

Default Value: 0x0000_0000

Table 65. Power Management Control Register

Register Name	Bit	Attribute User Side	Description
POWER MANAGEMENT CTRL	0	RW	Generate PME Message Wake Up. If PME is enabled and PME support is configured for current PMCSR D-state asserting this signal will cause the controller to wake from either L1 or L2 state. When the controller has transitioned back to the L0 state it will transmit a PME message and set the PME_Status. Upon receiving the PME message the root complex should clear the PME_Status and change the D-state back to D0. The IP passes on this information to HIP block and clears this bit indicating requested operation complete.
	1	RW	Generate PME Turnoff Message Only Available in RC Mode The IP passes on this information to HIP block and clears this bit indicating requested operation complete.
	2	RW	Delay PM_Enter_L23 response

continued...

Register Name	Bit	Attribute User Side	Description
			Indication from application that it is ready to enter the L23 state. The controller sends PM_Enter_L23 in response to PM_Turn_Off when this bit is set. Application that do not require this feature hardware initialize bit[3]. If bit[3] is set this bit is do not care from hardware point of view. The IP passes on this information to HIP block and clears this bit indicating requested operation complete.
	3	Hwinit	Autonomous PM_Enter_L23 response The controller sends PM_Enter_L23 in response to PM_Turn_Off.
	15-4	RsvdZ	Reserved
	20-16	RW	PF Number Indicates PF Number of Function generating PME. *Note:* Current Quartus release limits to max 8 PFs only.
	31-21	RsvdZ	Reserved

7.3.1.10. LEGACY INTERRUPT CTRL

Default Value : 0x0000_0000

Table 66. Legacy Interrupt Control Register

Register Name	Bit	Attribute User Side	Description
LEGACY INTERRUPT CTRL	0	RW	Assert Message The application sets this bit when it wants to send assert message. The IP passes on this information to HIP block and clears this bit indicating requested operation complete.
	1	RW	Deassert Message The application sets this bit when it want to send deassert message. The IP passes on this information to HIP block and clears this bit indicating requested operation complete.
	15-4	RsvdZ	Reserved
	20-16	RW	PF Number Indicates PF Number of Function generating Assert or Deassert message. *Note:* Current Quartus release limits to max 8 PFs only.
	31-21	RsvdZ	Reserved

7.3.1.11. CFG REG IA CTRL

The following table lists details of configuration register indirect access control register.

Default Value: 0x0000_0000

Table 67. Configuration Register Indirect Access Control

Register Name	Bit	Attribute User Side	Description
CFG REG IA CTRL	0	RW	Initiate Access This bit should be set when a master-like interface wants to read from or write to PCIe configuration space register. The IP performs read or write operation to a function pointed to by IA_FN_NUM register when this bit is set and clears this bit indicating requested operation is complete. Master cannot initiate new transaction if this bit is set.
	1	RW	Access Type Indicates access type of operation. 0 - Read Operation 1 - Write Operation
	5-2	RW	Byte Enables Indicates Byte Enables of Write Operations. 4'b0001: Write byte 0 4'b0010: Write byte 1 4'b0100: Write byte 2 4'b1000: Write byte 3 4'b1111: Write all bytes. Any Combinations of byte enables are valid, e.g., 1010, 1011 etc.
	15-6	RW	Register Address DWORD Address of Register
	31-16	RsvdZ	Reserved

7.3.1.12. CFG REG IA FN NUM

This register points to the function number of the configuration register the master is accessing through indirect access mechanism.

Default Value: 0x0000_0000

Table 68. Configuration Register Indirect Access Function Number

Register Name	Bit	Attribute User Side	Description
CFG REG IA FN NUM	2-0	RW	Function Type 000 - Indicates Access Physical Function 001 - Indicates Access Virtual Function All others - Reserved
	7-3	RW	PF Number

continued...

Register Name	Bit	Attribute User Side	Description
			The PF Number of register access. *Note:* Current Quartus release limits to max 8 PFs only.
	8	RsvdZ	Reserved
	19-9	RW	VF Number The VF Number of register access.
	20	RsvdZ	Reserved
	25-21	RW	Slot Number The slot Number of register access.
	31-26	RsvdZ	Reserved

7.3.1.13. CFG REG IA WRDATA

This register points to the function number of the configuration register the master is accessing through indirect access mechanism.

Default Value: 0x0000_0000

Table 69. Configuration Register Indirect Access Write Data

Register Name	Bit	Attribute User Side	Description
CFG REG IA WRDATA	31-0	RW	Write Data Data to be written into configuration register with write access. Master writes this register with required Data before Initiating Write Access.

7.3.1.14. CFG REG IA RDDATA

This register holds read data from read operation initiated by the master using indirect access mechanism.

Default Value: 0x0000_0000

Table 70. Configuration Register Indirect Access Write Data

Register Name	Bit	Attribute User Side	Description
CFG REG IA RDDATA	31-0	RO	Read Data Data read from configuration register with read access. Master reads this register after read operation completion indicated by Initiate Access bit in CFG IA CTRL register.

The PF/VF and Miscellaneous HIP registers can be accessed through Indirect Access to the CFG REG IA registers. Selective registers are user accessible for specific features. Below are a few examples to show accesses to the CFG REG IA register fields:

- Access PF1 Type 0 Configuration Header: Vendor ID:
  - CFG REG IA CTRL - Register Address (DW address)= 0x0
  - CFG REG IA FN NUM - Function Type = 'h0, PF Number = 'h1
- Access VF2 of PF1 Type 0 Configuration Header: Command Register:
  - CFG REG IA CTRL - Register Address (DW address) = 0x1, (Byte address = 0x4)
  - CFG REG IA FN NUM - Function Type = 'h1, PF Number = 'h1, VF Number = 'h2

7.3.1.15. PRS CTRL

The IP generates Page Request Service (PRS) events to the QHIP based on the settings of this PRS CTRL register.

Usage of this control register is applicable only when operating as Endpoint and with TLP Bypass disabled.

Default Value: 0x0000_0000

Table 71. Page Request Service (PRS) Control Register

Register Name	Bit	Attribute User Side	Description
PRS CTRL	0	RW	Generate Page Request Service (PRS) Event Writing '1' to this bit triggers PRS event. Write to this bit is ignored if bit is already set. The IP generates PRS event and clears this bit indicating requested operation complete.
	5-1	RW	PF Number Indicates Physical Function Number of the PRS event. *Note:* Current Quartus release limits to max 8 PFs only.
	7-6	RsvdZ	Reserved
	8	RW	Response Failure: Indicate that the function has received a PRG response failure.
	9	RW	Unexpected Page Request Group Index: Indicate that the function has received a response with Unexpected Page Request Group Index.
	10	RW	Stopped: Indicate that the function has completed all previously issued page requests and that it has stopped requests for additional pages. Only valid when the PRS enable bit is clear.

continued...

Register Name	Bit	Attribute User Side	Description
	19-11	RsvdZ	Reserved
	24-20	RW	Slot Number Indicates Slot Number of function generating Interrupt.
	31-25	RsvdZ	Reserved

7.3.1.16. MSI PENDING CTRL

The application layer uses both MSI PENDING CTRL and MSI PENDING registers to update the MSI Pending Bits for each function.

Default Value: 0x0000_0000

Table 72. MSI Pending Control Register

Register Name	Bit	Attribute User Side	Description
MSI PENDING CTRL	0	RW	Update MSI Pending Bits Writing '1' to this bit causes an update to the MSI Pending. Bits based on MSI PENDING value. Write to this bit is ignored if bit is already set. The IP clears this bit indicating the requested update is complete.
	5-1	RW	PF Number Indicates Physical Function Number of the update request. *Note:* Current Quartus release limits to max 8 PFs only.
	6	RsvdZ	Reserved
	17-7	RW	VF Number Indicates Virtual Function Number of the update request.
	18	RsvdZ	Reserved
	19	RW	VF Active Indicates Virtual Function is the target of the update request.
	24-20	RW	Slot Number Indicates Slot Number of the update request.
	31-25	RsvdZ	Reserved

7.3.1.17. MSI PENDING

Default Value: 0x0000_0000

Table 73. MSI Pending Register

Register Name	Bit	Attribute User Side	Description
MSI PENDING	31 - 0	RW	Indicates MSI Pending Bits value to be updated.

7.3.1.18. D-STATE STS

The application layer can use this register to obtain the D-State values of each function. It should write appropriate values to the PF Number, VF Number, VF Active and Slot Number fields first before issuing a read to obtain the D-State value of the corresponding function.

Default Value: 0x0000_0000

Register Name	Bit	Attribute User Side	Description
D-STATE STS	4-0	RW	PF Number Indicates Physical Function Number of Interrupt *Note:* Current Quartus release limits to max 8 PFs only.
	5	RsvdZ	Reserved
	16-6	RW	VF Number Indicates Virtual Function Number of Interrupt.
	17	RsvdZ	Reserved
	18	RW	VF Active Indicates Virtual Function is generating Interrupt.
	23-19	RW	Slot Number Indicates Slot Number of function generating Interrupt.
	27-24	RsvdZ	Reserved
	31-28	RO	D-State Value Power Management D-State for each function per PF, VF, VF Active and Slot Number settings above. 0000: Uninitialized or Invalid 0001: D0 0010: D1 0100: D2 1000: D3

7.3.1.19. CFG RETRY CTRL

The application layer can use this register to update the per PF Configuration Retry Status Enable controls (CRS En Controls) driven to the Hard IP Controller. All VFs will share the same control as their parent PF. When the corresponding PF's CRS En Control is asserted, HardIP Controller will respond to configuration TLPs with a CRS (Configuration Retry Status) if it has not already responded to a Configuration TLP with non-CRS status since the last reset. You can use this to hold off on enumeration.

Note: This register control allows the application layer to update 8 PFs at one time. If more than 8 PFs are used, the application layer needs to perform multiple updates by changing the PF Index field to point to the respective 8 PFs.

Default Value: 0x0000_0000

Register Name	Bit	Attribute User Side	Description
CFG RETRY CTRL	0	RW	Update CRS En Control Writing '1' to this bit causes the IP to update the corresponding CRS En Controls indicated by "Slot Number", "PF Index" and "PF Number". The IP clears this bit when the update is complete. Write to this bit is ignored if bit is already set
	5-1	RW	Slot Number Indicates Slot Number of the CRS En Controls to be updated
	7-6	RsvdZ	Reserved
	9-8	RW	PF Index Indicates which 8 Physical Functions of the CRS En Controls to be updated. 00 - PF7:PF0 01 - PF15:PF8 10 - PF23:PF16 11 - PF31:PF24 *Note:* Current Quartus release limits to max 8 PFs only.
	15-10	RsvdZ	Reserved
	23-16	RW	PF Number Indicates up-to 8 Physical Functions (one-hot) of the CRS En. Controls to be updated
	31-24	RsvdZ	Reserved

7.3.2. IP Debug Registers

The following table lists the debug registers implemented by the IP.

The IP debug registers start from Base Address = 0x400

Table 74. IP Debug Registers Address Map

Register Name	Offset
HIP Status	0x0000_0000
Reserved	0x0000_0004
Reserved	0x0000_0008
Reserved	0x0000_000C
Reserved	0x0000_0010
Reserved	0x0000_0014
Reserved	0x0000_0018
Reserved	0x0000_001C
Reserved	0x0000_0020
Reserved	0x0000_0024

continued...

Register Name	Offset
Reserved	0x0000_0028
HIP BP CYCLES	0x0000_002C
HIA BP CYCLES	0x0000_0030
APP BP CYCLES	0x0000_003C
HIA RX BP CYCLES	0x0000_0040

7.3.2.1. HIP Status

Default Value: 0x0000_0000

Table 75. Hard IP Status Registers

Register Name	Bit	Attribute User Side	Description
HIP Status	0	ROS	Link Up Indication 0 - Link Down 1 - Link up
	1	ROS	Data Link Layer Active Indication 0 - DL not Active 1 - DL Active
	7:2	ROS	Indicates LTSSM State 6'h00: S_DETECT_QUIET 6'h01: S_DETECT_ACT 6'h02: S_POLL_ACTIVE 6'h03: S_POLL_COMPLIANCE 6'h04: S_POLL_CONFIG 6'h05: S_PRE_DETECT_QUIET 6'h06: S_DETECT_WAIT 6'h07: S_CFG_LINKWD_START 6'h08: S_CFG_LINKWD_ACCEPT 6'h09: S_CFG_LANENUM_WAIT 6'h0A: S_CFG_LANENUM_ACCEPT 6'h0B: S_CFG_COMPLETE 6'h0C: S_CFG_IDLE 6'h0D: S_RCVRY_LOCK 6'h0E: S_RCVRY_SPEED 6'h0F: S_RCVRY_RCVRCFG 6'h10: S_RCVRY_IDLE 6'h11: S_L0 6'h12: S_L0S 6'h13: S_L123_SEND_EIDLE 6'h14: S_L1_IDLE 6'h15: S_L2_IDLE 6'h16: S_L2_WAKE 6'h17: S_DISABLED_ENTRY 6'h18: S_DISABLED_IDLE 6'h19: S_DISABLED 6'h1A: S_LPBK_ENTRY 6'h1B: S_LPBK_ACTIVE 6'h1C: S_LPBK_EXIT 6'h1D: S_LPBK_EXIT_TIMEOUT 6'h1E: S_HOT_RESET_ENTRY

continued...

Register Name	Bit	Attribute User Side	Description
			6'h1F: S_HOT_RESET 6'h20: S_RCVRY_EQ0 6'h21: S_RCVRY_EQ1 6'h22: S_RCVRY_EQ2 6'h23: S_RCVRY_EQ3
	8	ROS	User Mode 0 - HIP is in non-user mode 1 - HIP in User Mode
	9	ROS	HIP PLD Interface Ready Indication 0 - HIP PLD Interface not Ready 1 - HIP PLD Interface Ready
	10	ROS	HIP Entered in Error Mode 0 - Normal Operation 1 - RAM ECC Error Detected by HIP
	11	RW1C	HIP Buffer Overflow 1 - Indicates a HIP Buffer Overflow issue.
	12	RW1C	Reordering Buffer Overflow 1 - Indicates an Overflow condition of the Reordering Buffer.
	13	RW1C	P/NP FIFO Overflow 1 - Indicates an Overflow condition of the P/NP FIFO.
	31-14	–	Reserved

7.3.2.2. HIP BP CYCLES

The register indicates back pressure cycles observed because HIP transmit interface was not ready to accept transaction.

Figure 49. HIP BP Cycles

Default Value: 0x0000_0000

Table 76. IP Interface attributes

Register Name	Bit	Attribute User Side	Description
HIP BP CYCLES	30-0	RW1C	Back Pressure Cycle Count
	31	RW1C	Indicates Overflow, cycle count reached 31'h7FFFFFFF

7.3.2.3. HIA BP CYCLES

The register indicates back pressure cycles observed because HIP interface adaptor transmit interface was not ready to accept transactions.

Default Value: 0x0000_0000

Table 77. HIA BP Registers

Register Name	Bit	Attribute User Side	Description
HIA BP CYCLES	30-0	RW1C	Back Pressure Cycle Count
	31	RW1C	Indicates Overflow, cycle count reached 31'h7FFFFFFF

7.3.2.4. APP BP CYCLES

The register indicates back pressure cycles observed because application logic connected to H2C block was not ready to accept transactions.

Default Value: 0x0000_0000

Table 78. APP BP Registers

Register Name	Bit	Attribute User Side	Description
APP BP CYCLES	30-0	RW1C	Back Pressure Cycle Count
	31	RW1C	Indicates Overflow, cycle count reached 31'h7FFFFFFF

7.3.2.5. HIA RX BP CYCLES

The register indicates back pressure cycles observed because HIP interface adaptor receive interface was not ready to accept transactions.

Default Value: 0x0000_0000

Table 79. HIA RX BP Registers

Register Name	Bit	Attribute User Side	Description
HIA RX BP CYCLES	30-0	RW1C	Back Pressure Cycle Count
	31	RW1C	Indicates Overflow, cycle count reached 31'h7FFFFFFF

7.3.3. IP Performance Monitor Registers

The following table lists the performance registers implemented by the IP.

The IP performance monitor registers start from Base Address = 0x800

Figure 50. IP Performance Register Address Map

Register Name	Offset
PERFMON CTRL	0x0000_0000
TX MRD TLP	0x0000_0004
TX MWR TLP	0x0000_0008
TX MSG TLP	0x0000_000C
TX CFGWR TLP	0x0000_0010
TX CFGRD TLP	0x0000_0014
RX MRD TLP	0x0000_0018
RX MWR TLP	0x0000_001C
RX MSG TLP	0x0000_0020
RX CFGWR TLP	0x0000_0024
RX CFGRD TLP	0x0000_0028
TX MEM DATA	0x0000_002C
TX CPL DATA	0x0000_0030
RX MEM DATA	0x0000_0034
RX CPL DATA	0x0000_0038

7.3.3.1. PERFMON CTRL

The register controls various performance monitor counters in the IP.

Default Value: 0x0000_0000

Table 80. PERFMON CTRL Registers

Register Name	Bit	Attribute User Side	Description
PERFMON CTRL	0	RW	Global Enable 1 - Turn On All Performance Counters 0 - Turn Off All Performance Counters
	1	RW	Clear Counters 1 - Clears all performance counters in a system 0 - No effect
	10-2	RW	Counting Duration in seconds 9'h000 - No Limit 9'h001 - 1 seconds 9'h002 - 2 seconds ..... 9'h1FF - 511 seconds
	31-11	RsvdZ	Reserved

7.3.3.2. TX MRD TLP

The register indicates number of memory read TLPs transmitted by the IP.

Default Value: 0x0000_0000

Table 81. TX MRD TLP Registers

Register Name	Bit	Attribute User Side	Description
TX MRD TLP	31-0	RW1C	Number of Memory Read TLPs

7.3.3.3. TX MWR TLP

The register indicates number of memory write TLPs transmitted by the IP.

Default Value: 0x0000_0000

Table 82. TX MWR TLP Registers

Register Name	Bit	Attribute User Side	Description
TX MWR TLP	31-0	RW1C	Number of Memory Write TLPs

7.3.3.4. TX MSG TLP

The register indicates the number of message TLPs transmitted by the IP.

Default Value: 0x0000_0000

Table 83. TX MSG TLP Registers

Register Name	Bit	Attribute User Side	Description
TX MSG TLP	31-0	RW1C	Number of Message Write TLPs

7.3.3.5. TX CFGWR TLP

The register indicates the number of configuration write TLPs transmitted by the IP.

Default Value: 0x0000_0000

Table 84. TX CFGWR TLP Registers

Register Name	Bit	Attribute User Side	Description
TX CFGWR TLP	31-0	RW1C	Number of Configuration Write TLPs

7.3.3.6. TX CFGRD TLP

The register indicates the number of configuration read TLPs transmitted by the IP.

Default Value: 0x0000_0000

Table 85. TX CFGRD TLP Registers

Register Name	Bit	Attribute User Side	Description
TX CFGRD TLP	31-0	RW1C	Number of Configuration Read TLPs

7.3.3.7. RX MRD TLP

The register indicates the number of memory read TLPs received by the IP.

Default Value: 0x0000_0000

Table 86. RX MRD TLP Registers

Register Name	Bit	Attribute User Side	Description
RX MRD TLP	31-0	RW1C	Number of Memory Read TLPs

7.3.3.8. RX MWR TLP

The register indicates the number of memory write TLPs received by the IP.

Default Value: 0x000AXI0_0000

Table 87. RX MWR TLP Registers

Register Name	Bit	Attribute User Side	Description
RX MWR TLP	31-0	RW1C	Number of Memory Write TLPs

7.3.3.9. RX MSG TLP

The register indicates the number of message TLPs received by the IP.

Default Value: 0x0000_0000

Table 88. RX MSG TLP Registers

Register Name	Bit	Attribute User Side	Description
RX MSG TLP	31-0	RW1C	Number of Message Write TLPs

7.3.3.10. RX CFGWR TLP

The register indicates the number of configuration write TLPs received by the IP.

Default Value: 0x0000_0000

Table 89. RX CFGWR TLP Registers

Register Name	Bit	Attribute User Side	Description
RX CFGWR TLP	31-0	RW1C	Number of Configuration Write TLPs

7.3.3.11. RX CFGRD TLP

The register indicates the number of configuration read TLPs received by the IP.

Default Value: 0x0000_0000

Table 90. RX CFGRD TLP Registers

Register Name	Bit	Attribute User Side	Description
RX CFGRD TLP	31-0	RW1C	Number of Configuration Read TLPs

7.3.3.12. TX MEM DATA

The register indicates data transmitted by the IP for memory write operation.

Default Value: 0x0000_0000

Table 91. TX MEM Registers

Register Name	Bit	Attribute User Side	Description
TX MEM DATA	31-0	RW1C	Bytes Transferred 32'h00000000 - No bytes 32'h00000001 - 1 KB 32'h00000002 - 2 KB ..... 32'hFFFFFFFF - 4 TB

7.3.3.13. TX CPL DATA

The register indicates completion data transmitted by the IP.

Default Value: 0x0000_0000

Table 92. TX CPL DATA Registers

Register Name	Bit	Attribute User Side	Description
TX CPL DATA	31-0	RW1C	Bytes Transferred 32'h00000000 - No bytes 32'h00000001 - 1 KB 32'h00000002 - 2 KB ..... 32'hFFFFFFFF - 4 TB

7.3.3.14. RX MEM DATA

The register indicates data received by the IP for memory write operation.

Default Value: 0x0000_0000

Table 93. RX MEM DATA Registers

Register Name	Bit	Attribute User Side	Description
RX MEM DATA	31-0	RW1C	Bytes Transferred 32'h00000000 - No bytes 32'h00000001 - 1 KB 32'h00000002 - 2 KB ..... 32'hFFFFFFFF - 4 TB

7.3.3.15. RX CPL DATA

The register indicates completion data received by the IP.

Default Value: 0x0000_0000

Table 94. RX CPL DATA Registers

Register Name	Bit	Attribute User Side	Description
RX CPL DATA	31-0	RW1C	Bytes Transferred 32'h00000000 - No bytes

Register Name	Bit	Attribute User Side	Description
			32'h00000001 - 1 KB 32'h00000002 - 2 KB ……. 32'hFFFFFFFF - 4 TB

8. Document Revision History for the AXI Streaming Intel FPGA IP for PCI Express User Guide

Document Version	Intel Quartus Prime Version	Changes
2024.02.12	23.4	Initial release.

A. Specifications

A.1. P-Tile Specifications

A.1.1. P-Tile Completion Buffer Size

P-tile PCIe Hard IP implements Completion Buffers for Header and Data for each PCIe core/port. In Endpoint mode, when Completion credits are infinite, user application needs to manage the number of outstanding requests according to the buffer size to prevent overflow and lost Completions packets.

Table 95. P-tile Completion Buffer Size

Completion Buffer	Depth	Width
Port 0 Cpl header	1144	NA
Port 0 Cpl data	1444	256
Port 1 Cpl header	572	NA
Port 1 Cpl data	1444	128
Port 2 Cpl header	286	NA
Port 2 Cpl data	1444	64
Port 3 Cpl header	286	NA
Port 3 Cpl data	1444	64

Refer to section 4.4.8.1 of the P-Tile Avalon Streaming Intel FPGA IP for PCI Express User Guide.

A.1.2. Power Management

Refer to section 4.9 of the P-Tile Avalon Streaming Intel FPGA IP for PCI Express User Guide.

A.1.3. MSI and MSI-X

Refer to sections 4.6.2 and 4.6.3 of the P-Tile Avalon Streaming Intel FPGA IP for PCI Express User Guide.

A.2. F-Tile Specifications

A.2.1. F-Tile Completion Buffer Size

For F-Tile, the PCIe Hard IP implements Completion Buffers for Header and Data for each PCIe core/port. In Endpoint mode, when Completion credits are infinite, user application needs to manage the number of outstanding requests according to the buffer size to prevent overflow and lost Completions packets.

Table 96. F-Tile Completion Buffer Size

Completion Buffer	Depth	Width
Port 0 Cpl Header	1144	128
Port 0 Cpl Data	1444	256
Port 1 Cpl Header	572	128
Port 1 Cpl Data	1444	128
Port 2 Cpl Header	286	128
Port 2 Cpl Data	1444	64
Port 3 Cpl Header	286	128
Port 3 Cpl Data	1444	64

A.2.2. Power Management

Refer to section 5.10 of the F-Tile Avalon Streaming Intel FPGA IP for PCI Express User Guide.

A.3. R-Tile Specifications

A.3.1. R-Tile Completion Buffer Size

For R-Tile, the PCIe Hard IP implements Completion Buffers for Header and Data for each PCIe core/port. In Endpoint mode, when Completion credits are infinite, the user application needs to manage the number of outstanding requests according to the buffer size to prevent overflow and lost Completion packets.

Table 97. R-Tile Completion Buffer Size

Completion Buffer	Depth	Width
Port 0 Cpl header	1024	NA
Port 0 Cpl data	2048	256
Port 1 Cpl header	256	NA
Port 1 Cpl data	1024	128
Port 2 Cpl header	128	NA
Port 2 Cpl data	512	64
Port 3 Cpl header	128	NA
Port 3 Cpl data	512	64

A.3.2. Power Management

Refer to section 4.3.8 of the R-tile Avalon Streaming Intel FPGA IP for PCI Express User Guide.

A.3.3. MSI and MSI-X

Refer to section 4.3.3.2 and 4.3.3.3 of the R-Tile Avalon Streaming Intel FPGA IP for PCI Express User Guide.

B. Simulating the Design Example

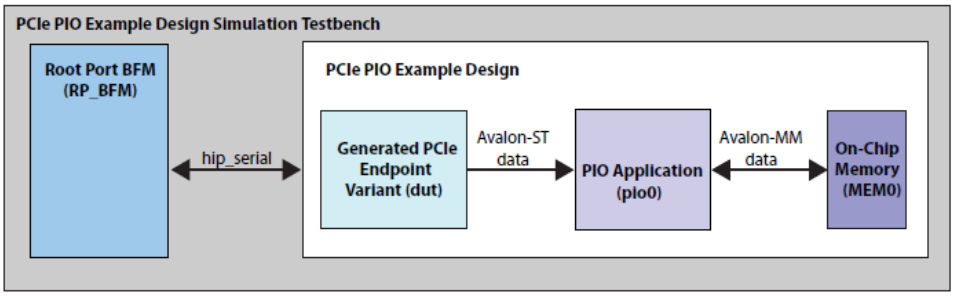
B.1. Testbench

The testbench uses a test driver module, `altpcietb_bfm_rp_gen5_x16.sv`, to initiate the configuration and memory transactions. At startup, the test driver module displays information from the Root Port and Endpoint Configuration Space registers, so that you can correlate to the parameters you specified using the Parameter Editor.

The example design and testbench are dynamically generated based on the configuration that you choose for the AXI Streaming Intel FPGA IP for PCI Express. The testbench uses the parameters that you specify in the Parameter Editor in Intel Quartus Prime.

This testbench simulates a Gen5 ×16 PCI Express link using the serial PCI Express interface.

Figure 51. PIO Design Example Simulation Testbench



When configured as an Endpoint variation, the testbench instantiates a design example with an AXI Streaming IP Endpoint and a Root Port BFM containing a second AXI Streaming IP (configured as a Root Port) to interface with the Endpoint. For more details on this Root Port BFM, refer to *Root Port BFM*. The Root Port BFM provides the following functions:

- A configuration routine that sets up all the basic configuration registers in the Endpoint. This configuration allows the Endpoint application to be the target and initiator of PCI Express transactions.
- A Verilog HDL procedure interface to initiate PCI Express transactions to the Endpoint.

This testbench simulates the scenario of a single Root Port talking to a single Endpoint.

The testbench uses a test driver module, altpcietb_bfm_rp_gen5_x16.sv, to initiate the configuration and memory transactions. At startup, the test driver module displays information from the Root Port and Endpoint Configuration Space registers, so that you can correlate to the parameters you specified using the Parameter Editor.

Note: The Intel testbench and Root Port BFM provide a simple method to do basic testing of the Application Layer logic that interfaces to the variation. This BFM allows you to create and run simple task stimuli to exercise basic functionality of the Intel example design. The testbench and Root Port BFM are not intended to be a substitute for a full verification environment. Corner cases and certain traffic profile stimuli are not covered. Refer to the items listed below for further details. To ensure the best verification coverage possible, Intel suggests strongly that you obtain commercially available PCI Express verification IP and tools, in combination with performing extensive hardware testing.

Your Application Layer design may need to handle at least the following scenarios that are not possible to create with the Intel testbench and the Root Port BFM, or are due to the limitations of the example design:

- It is unable to generate or receive Vendor Defined Messages. Some systems generate Vendor Defined Messages. The Hard IP block simply passes these messages on to the Application Layer. Consequently, you should make the decision, based on your application, whether to design the Application Layer to process them.
- It can only handle received read requests that are less than or equal to 256 bits or 8 DW. Many systems are capable of handling larger read requests that are then returned in multiple completions.
- It always returns a single completion for every read request. Some systems split completions on every 64-byte address boundary.
- It always returns completions in the same order the read requests were issued. Some systems generate the completions out-of-order.
- It is unable to generate zero-length read requests that some systems generate as flush requests following some write transactions. The Application Layer must be capable of generating the completions to the zero-length read requests.
- It does not support parity.
- It does not support multi-function designs.
- It does not support Single Root I/O Virtualization (SR-IOV).

B.1.1. Testbench Modules

The top-level of the testbench instantiates the following main modules:

- `altpcietb_bfm_rp_gen5x16.sv` —This is the Root Port PCIe BFM.

```
//Directory path
<project_dir>/intel_rtile_pcie_ast_0_example_design/pcie_ed_tb/ip/
pcie_ed_tb/dut_pcie_tb_ip/intel_rtile_pcie_tbed_<ver>/sim
```

- `pcie_ed_dut.ip`: This is the Endpoint design with the parameters that you specify.

```
//Directory path
<project_dir>/intel_rtile_pcie_ast_0_example_design/ip/pcie_ed
```

- `pcie_ed_pio0.ip`: This module is a target and initiator of transactions for the PIO design example.

```
//Directory path
<project_dir>/intel_rtile_pcie_ast_0_example_design/ip/pcie_ed
```

In addition, the testbench has routines that perform the following tasks:

- Generates the reference clock for the Endpoint at the required frequency.
- Provides a PCI Express reset at start up.

B.1.2. Test Driver Module

The test driver module, `intel_rtile_pcie_tbed_hwtcl.v`, instantiates the top-level BFM,`altpcietb_bfm_top_rp.v`.

The top-level BFM completes the following tasks:

1. Instantiates the driver and monitor.
2. Instantiates the Root Port BFM.
3. Instantiates the serial interface.

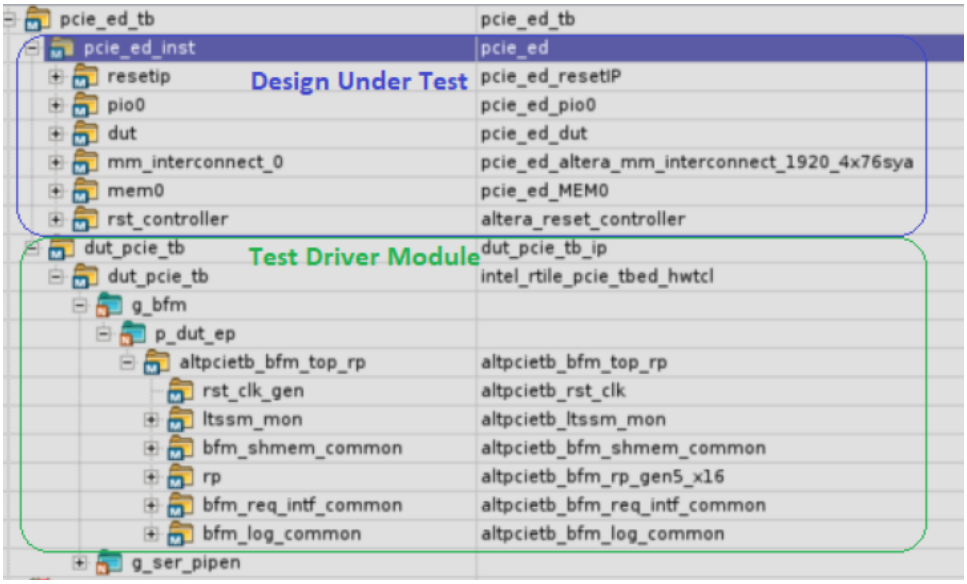
The configuration module, `altpcietb_g3bfm_configure.v`, performs the following tasks:

1. Configures and assigns the BARs.
2. Configures the Root Port and Endpoint.
3. Displays comprehensive Configuration Space, BAR, MSI, MSI-X, and AER settings.

B.1.3. PIO Design Example Testbench

The figure below shows the PIO design example simulation design hierarchy. The tests for the PIO design example are defined with the `apps_type_hwtcl` parameter set to 3. The tests run under this parameter value are defined in `ebfm_cfg_rp_ep_rootport`, `find_mem_bar` and `downstream_loop`.

Figure 52. PIO Design Example Simulation Design Hierarchy



The testbench starts with link training and then accesses the configuration space of the IP for enumeration. A task called `downstream_loop` (defined in the Root Port PCIe BFM `altpcietb_bfm_rp_gen5_x16.sv`) then performs the PCIe link test. This test consists of the following steps:

1. Issue a memory write command to write a single dword of data into the on-chip memory behind the Endpoint.
2. Issue a memory read command to read back data from the on-chip memory.
3. Compare the read data with the write data. If they match, the test counts this as a Pass.
4. Repeat Steps 1, 2 and 3 for 10 iterations.

The first memory write takes place around 219 us. It is followed by a memory read at the Avalon-ST RX interface of the Hard IP for PCIe. The Completion TLP appears shortly after the memory read request at the Avalon-ST TX interface.

Note: In the 2x8 design example, memory read and memory write transactions are simulated on Port 0 only.

B.1.4. Root Port BFM

For more details on the Root Port BFM instantiated by the testbench, refer to Root Port BFM.