UG0685
User Guide
PolarFire FPGA PCI Express

Microsemi
Power Matters.™

Microsemi Corporate Headquarters
One Enterprise, Aliso Viejo,
CA 92656 USA
Within the USA: +1 (800) 713-4113
Outside the USA: +1 (949) 380-6100
Fax: +1 (949) 215-4996
Email: sales.support@microsemi.com
www.microsemi.com

© 2017 Microsemi Corporation. All rights reserved. Microsemi and the Microsemi logo are trademarks of Microsemi Corporation. All other trademarks and service marks are the property of their respective owners.

Microsemi makes no warranty, representation, or guarantee regarding the information contained herein or the suitability of its products and services for any particular purpose, nor does Microsemi assume any liability whatsoever arising out of the application or use of any product or circuit. The products sold hereunder and any other products sold by Microsemi have been subject to limited testing and should not be used in conjunction with mission-critical equipment or applications. Any performance specifications are believed to be reliable but are not verified, and Buyer must conduct and complete all performance and other testing of the products, alone and together with, or installed in, any end-products. Buyer shall not rely on any data and performance specifications or parameters provided by Microsemi. It is the Buyer's responsibility to independently determine suitability of any products and to test and verify the same. The information provided by Microsemi hereunder is provided “as is, where is” and with all faults, and the entire risk associated with such information is entirely with the Buyer. Microsemi does not grant, explicitly or implicitly, to any party any patent rights, licenses, or any other IP rights, whether with regard to such information itself or anything described by such information. Information provided in this document is proprietary to Microsemi, and Microsemi reserves the right to make any changes to the information in this document or to any products and services at any time without notice.

About Microsemi

Microsemi Corporation (Nasdaq: MSCC) offers a comprehensive portfolio of semiconductor and system solutions for aerospace & defense, communications, data center and industrial markets. Products include high-performance and radiation-hardened analog mixed-signal integrated circuits, FPGAs, SoCs and ASICs; power management products; timing and synchronization devices and precise time solutions, setting the world's standard for time; voice processing devices; RF solutions; discrete components; enterprise storage and communication solutions, security technologies and scalable anti-tamper products; Ethernet solutions; Power-over-Ethernet ICs and midspans; as well as custom design capabilities and services. Microsemi is headquartered in Aliso Viejo, California, and has approximately 4,800 employees globally. Learn more at www.microsemi.com.

50200685. 2.0 6/17

Contents

Figures

Tables

1 Revision History

The revision history describes the changes that were implemented in the document. The changes are listed by revision, starting with the most current publication.

1.1 Revision 2.0

The following is a summary of the changes in revision 2.0 of this document.

- Information about Transfer Parameters was added. See Transfer Parameters, page 62.
- Information about how to use VIP models was added. See PCIe Simulation, page 38.
- Information about Interrupt and Address translation register was added. See Interrupts and MSI Settings, page 42 and Address Translation Registers, page 60.
- Information about DMA Descriptors was added. See Scatter-Gather Types, page 16.
- Updated Table 2, page 19. Table 5, page 30, Table 6, page 32, Table 8, page 35, Table 9, page 37, and Table 10, page 38.
- Updated Figure 14, page 24, Figure 16, page 27, Figure 17, page 28, Figure 18, page 29, Figure 19, page 31, Figure 20, page 33, Figure 21, page 35, Figure 22, page 36, and Figure 23, page 37.

1.2 Revision 1.0

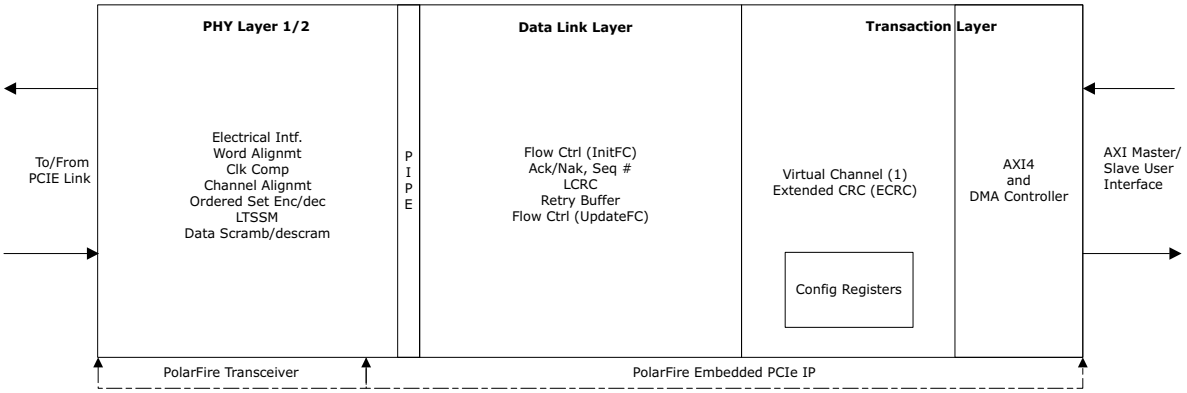
Revision 1.0 was the first publication of this document.

2 Overview

PCI Express (PCIe) is a scalable, high-bandwidth serial interconnect technology that maintains compatibility with existing PCI systems. Microsemi's PolarFire™ FPGAs contains fully integrated PCIe endpoint and root port subsystems with optimized embedded controller blocks that use the physical layer interface of the PolarFire transceiver for the PCI Express (PIPE) interconnection within the transceiver block.

Each PolarFire device includes two PolarFire embedded PCIe subsystem (PCIESS) blocks that can be configured either separately, or as a pair using the PCIESS configurator in the Libero® System-on-Chip (SoC) PolarFire software. The PCIESS encompasses features from the physical layer of the transceiver and physical coding sublayer (PCS) block to the application interface to the FPGA fabric.

Figure 1 • **PCIe Functional Layers of PolarFire PCIESS**



The PCIESS is compliant with *PCI Express Base Specification, Revision 2.1*. The PCIESS implements the memory-mapped advanced microcontroller bus architecture (AMBA) advanced extensible interface 4 (AXI4) access to the PCIe space as well as the PCIe access to the memory-mapped AXI4 space. The PCIESS is compatible with the *PCI Express Card Electromechanical (CEM) 2.0* and the *PCI Industrial Computer Manufacturers Group (PICMG) 3.4* specifications.

The embedded PCIESS can be configured using a simple GUI within the Libero SoC software flow that walks you through the steps to create an endpoint or root port application design. The PCIESS GUI allows configuration of parameters such as lane width, lane rate, reference clock frequency, and power management settings.

For more information on the specification standards referenced in this document, see:

- *UG0677: PolarFire FPGA Transceiver User Guide*
- *AMBA AXI Protocol Specification, Version 2.0 – ARM, March 2010*
- *PCI Express Base Specification, Revision 2.1*
- *PCI Express Card Electromechanical Specification, Revision 2.0 – April 2007*
- *PCI Power Management Specification, v1.2*

2.1 Features

The PCIESS is a hard PCI Express protocol stack embedded within every PolarFire device. It includes the transaction layer, data link layer, and physical layer. The PolarFire PCIESS includes both the physical media attachment (PMA) and the PCS that supports x1, x2, and x4 endpoints and root port configurations at up to five Gbps (Gen 2) speeds compliant with the *PCI Express Base Specification, Revision 2.1*.

The PCIESS supports advanced microcontroller bus architecture AMBA AXI4 master/slave user interface functionality between the AXI4 and PCIe systems. The interface, which has embedded full-bridge functionality, includes scatter-gather (SG) direct memory access (SGDMA) engines with controller and bridging logic encompassed within the PCIESS.

Two PCIESS blocks are available to support both endpoint and root port functionalities. Within each device, one PCIESS is capable of x1 and x2, and the other PCIESS is capable of x1, x2, and x4. Only the four lanes within the transceiver quad have access to the PCIESS blocks, and no lanes from adjacent quads may be used in conjunction with either PCIESS. This permits two x1 PCIESS or two x2 PCIESS that can run simultaneously, or one PCIESS that can run as x4 only if the other PCIESS is left unused. The PCIESS includes the following features:

- PolarFire transceivers for 2.5 and 5.0 Gbps line speeds
- Native x1, x2, and x4 lane-support PCIe block (down-configurable/downgradable)
- Endpoint support for up to six 32-bit or three 64-bit base address register (BAR)
- Root point support for up to two 32-bit or one 64-bit BAR.
- PCI Express Base specification 1.1- and 2.1- compliant
- One virtual channel (VC)
- Single-function capability
- Maximum payload size (MPS) of up to 256 bytes
- Advanced error reporting (AER) support
- Integrated clock domain crossing (CDC) to support user-selected frequency
- Lane reversal support/polarity inversion
- Legacy PCI power management support
- Supports end point hot-plug (not supported for root port)
- Native active-state power management L0 and L1 support
- Power management event (PME) message
- Latency tolerance reporting (LTR)
- Fully compliant physical layer PCS
- 64-bit AXI master and slave interface to the FPGA fabric
- End-to-end data integrity

2.1.1 PCIe DMA

Each PCIe controller supports the following built-in DMA features, enabling low-power and efficient data transfer to the FPGA fabric:

- Two fully-independent DMA engines
- Eight outstanding read and write requests
- Completion re-ordering support
- Reconfigurable source and destination, enabling targeting of PCIe and AXI4 master interfaces
- Flexible SGDMA modes, including dynamic DMA control per descriptor
- DMA engine that reports to the descriptor for easy software management
- Fetches up to three descriptors to optimize throughput

2.1.2 PCIe Address Translation

There are six address table registers (ATRs) that perform address translation from the PCIe address space (BAR) to the AXI master, and six ATRs for the slave, which addresses translation from the AXI4 slave to the PCIe address space.

2.1.3 Physical Layer Functions

The physical layer is an electrical PMA required to connect to a PCIe system, and supports the following features:

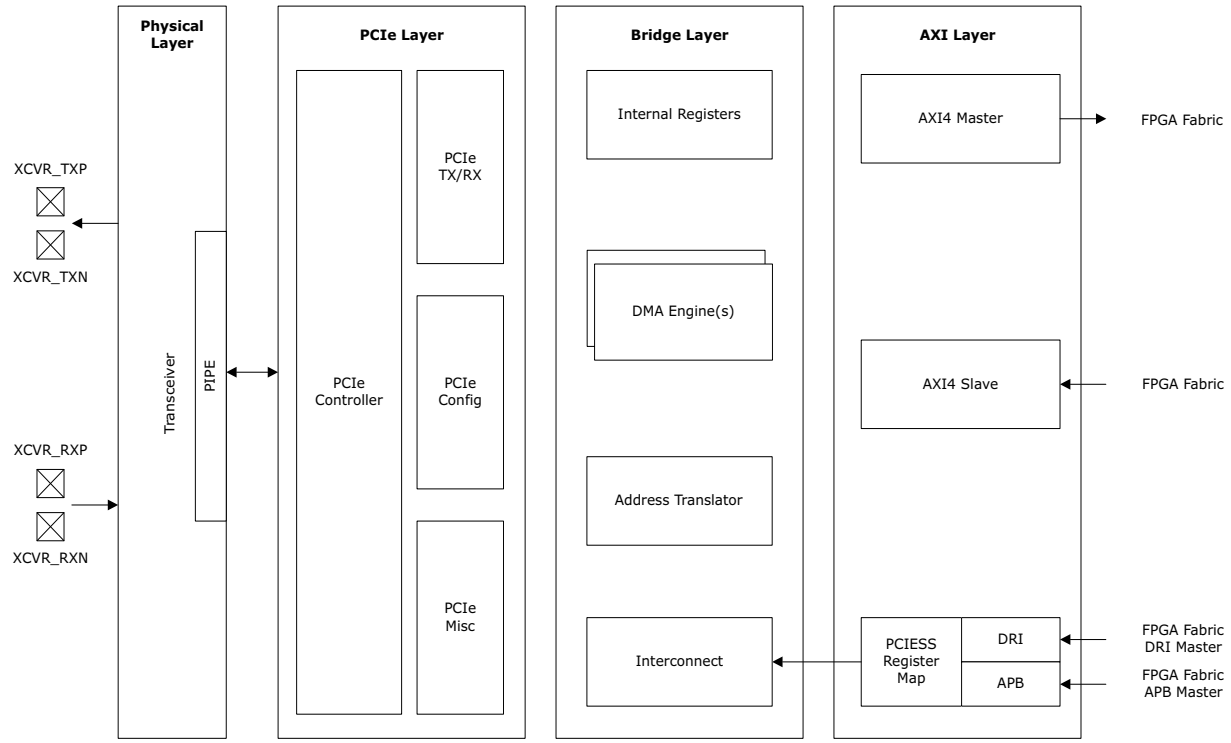
- PCI Express 2.0 electrical compliance
- 2.5 and 5 Gbps common-mode logic (CML) electrical interface
- Signal integrity programmability, including differential output voltage, transmitter de-emphasis, and receiver-side continuous-time linear equalization (CTLE)
- ± 300 ppm clock-tolerance compensation
- Serializer and de-serializer
- 8b/10b symbol encoding/decoding
- Data scrambling and de-scrambling
- Symbol alignment
- Clock tolerance compensation ± 300 ppm
- Framing and application of symbols to lanes
- Lane to lane Tx deskew

2.1.4 PCIe Embedded Block

As shown in the following figure, the PCIESS is composed of the following four sub-modules:

- Physical layer
- PCIe layer
- Bridge layer
- AXI layer

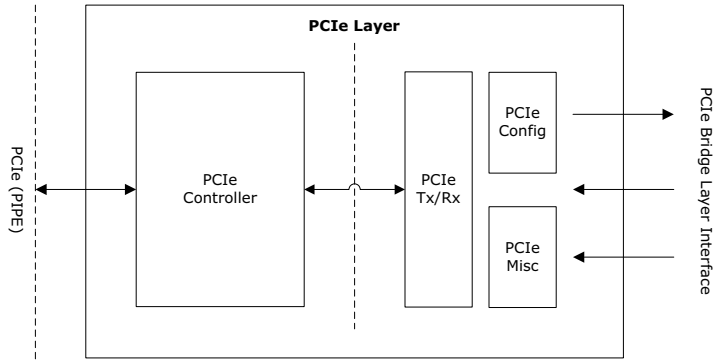
Figure 2 • **PolarFire Embedded PCIESS Architecture**



The PCIe layer includes:

- PCIe controller
- PCIe transmit/receive interface between the PCIe bridge and PCIe controller
- PCIe configuration interface providing the bridge access to the PCIe configuration space
- PCIe miscellaneous interface to allow the bridge access to manage low-power, and interrupts
- Lane reversal
- Link training and status state machine (LTSSM)
- Electrical idle generation
- Receiver detection
- TS1/TS2 generation/detection

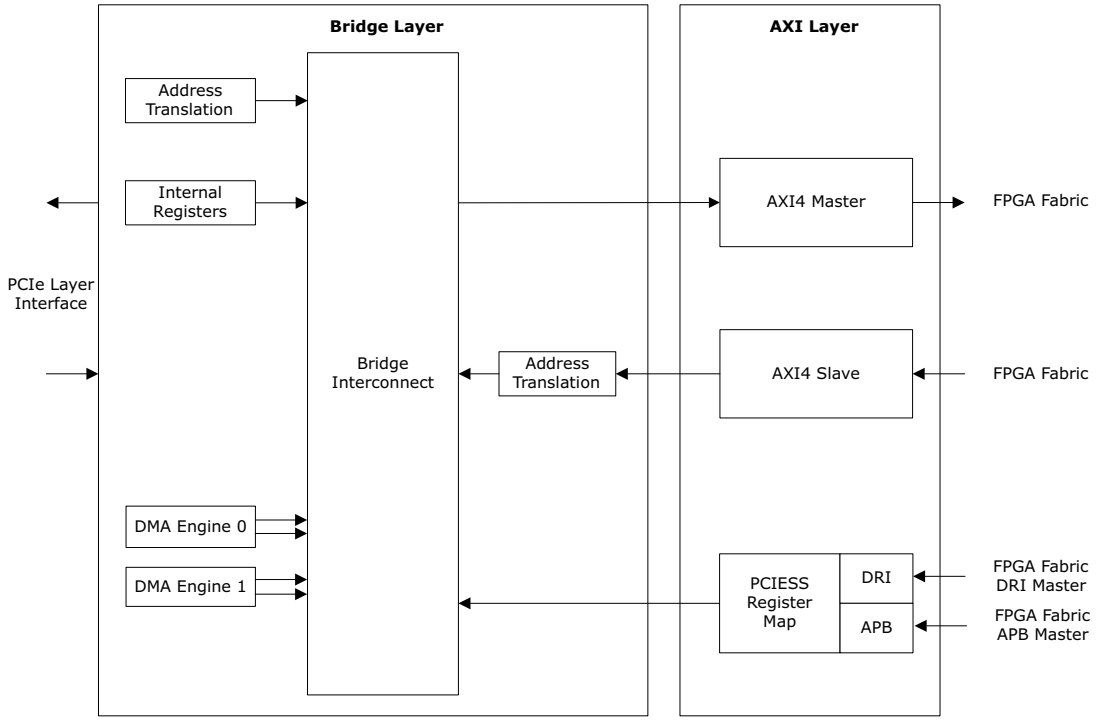
Figure 3 • **PCIe Layer**



The bridge layer includes:

- Bridge internal registers
- Two independent DMA engines
- An address translator to convert between the AXI and PCIe interfaces
- A bridge interconnect module to interconnect and arbitrate between input and output flows

Figure 4 • **Bridge Layer**



The AXI layer includes the AXI4 master and slave interfaces.

2.1.5 Configuration Interface

The registers required for the initial configuration of the PCIESS are loaded based on the options selected in the Libero PCIESS Configurator wizard. On device power-up, these values are automatically loaded into the configuration registers from the on-chip non-volatile memory during the design initialization process. The PCIESS and transceiver block are associated with several flash bits. These bits contain the settings of registers that need to be initialized quickly. The design initialization process initializes the remaining registers using design-specific firmware included in design generation, when the device powers up such as PLL and clock configurations, PCIe configuration space, and resets. The design initialization uses the dedicated resources to bring up the PCIESS features at power-up or device reset. This does not require any programmable FPGA user resources. The PCIESS then reads and writes the configuration space registers as the link comes up and the endpoint device is enumerated into the host system.

The bridge configuration space consists of internal memories including the bridge internal registers and the PCIe configuration space. The configuration space accessible to the PCIe controller through the dynamic reconfiguration interface (DRI). The DRI is a slave APB interface that can update the configuration space either at startup using firmware, or at run-time through a fabric-based APB master to the DRI. For more information on DRI, see *UG0677: PolarFire FPGA Transceiver User Guide*.

The DRI reads the configuration space and sets up the DMA controller, if required. In root port mode, the DRI can target the PCIe configuration space through an enhanced configuration access mechanism (ECAM) supported by the bridge. In endpoint mode, a remote PCIe root port can read or write to the PCIe configuration space using the DRI interface targeting the bridge configuration space. For more information on dynamic reconfiguration Interface, see *UG0677: PolarFire FPGA Transceiver User Guide*.

2.1.6 Other Signals

The PCIESS also has several status signals, interrupt signals, and power management signals available to the FPGA fabric. These signals are listed in the pin list (Table 2, page 19) and can be connected to the fabric to control and monitor the PCIESS.

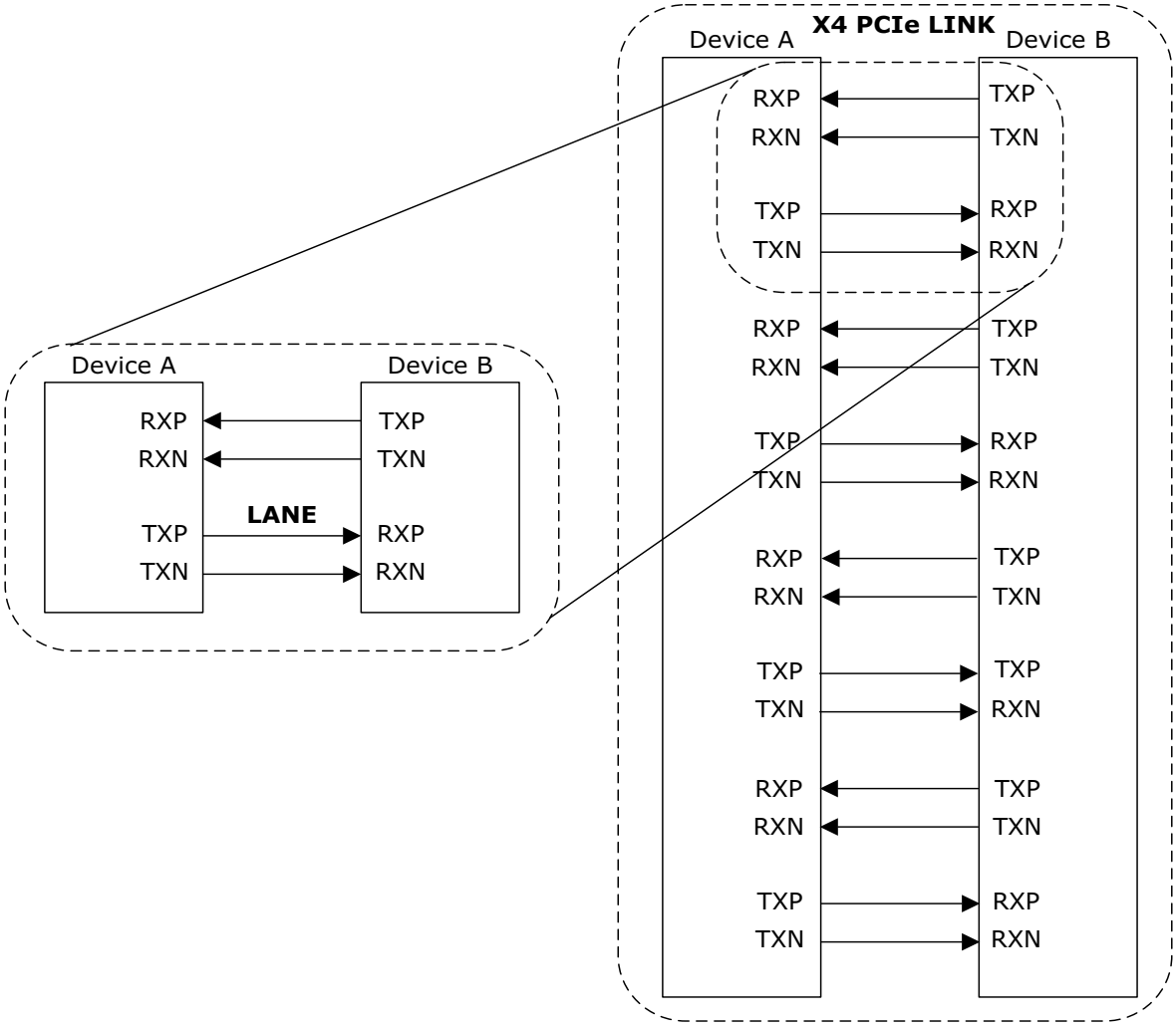
3 Functional Descriptions

The PCIe subsystem uses several built-in features such as PolarFire transceivers, embedded PCIe controller, and programmable FPGA resources. The functional details of the PCIe subsystem are described in this chapter.

3.1 Physical Layer Interface

The PCIe link consists of PolarFire differential transceivers organized in multiple lanes. A PCIe lane consists of a pair of differential transmit signals and a pair of differential receive signals. The lanes are organized where lane 0 is used for an x1 link configuration; lane 0 and lane 1 and/or lane 2 and 3 are used for an x2 link configuration; and lanes 0, 1, 2, and 3 together are used for an x4 link configuration.

Figure 5 • **Physical Layer Interface**



3.1.1 Receiver

The PolarFire transceiver input includes all the features required to build a PCIe interface, such as input level sensitivity, signal detection, and termination. When PCIESS is used within a PolarFire device, the Libero software configures the receiver with all the necessary input features. For more information, see *UG0677: PolarFire FPGA Transceiver User Guide*.

3.1.2 Transmitter

The PolarFire transceiver output includes features such as output swing, termination, and de-emphasis. When PCIESS is used within a PolarFire device, the Libero software configures the transmitter with all the necessary output features. For more information, see *UG0677: PolarFire FPGA Transceiver User Guide*.

PCIe specifications mandate a boosted post-cursor (PCIe de-emphasis) to improve data eye on a PCIe channel. The PolarFire transceiver has transmit de-emphasis capability for channel correction. This integrated feature minimizes large variations in instantaneous amplitude feeding into the receiver, improves inter-symbol interference (ISI) and receiver harmonics, and manages amplitude-to-phase noise jitter as required by PCI-SIG specifications completion with data.

The receiver detection circuitry within the transmitter depends on capacitive loading from the AC-coupled capacitors placed on the system printed circuit board (PCB). When a receive signal appears at the output terminals along with the AC-coupling capacitor, the response time of the common-mode voltage is changed. Each transmitter, at the start of linkup, produces a low-frequency pulse on each of the differential transmit outputs. The transmitter includes a simple detection circuit to monitor the line response to this pulse. The edge rate and amplitude of the line change are much higher with no receiver attached, than with a receiver attached.

3.1.3 Reference Clock

For PCIe applications, a differential 100 or 125 MHz reference clock is used by the transceiver transmit PLL and CDR PLL to generate the required 62.5 or 125 MHZ output clock (depending on the lane speed settings) which is passed to the embedded PCIESS. The settings for the transmit PLL and the CDR PLL are automatically determined by the Libero SoC software.

The transceiver reference clock inputs accept LVDS/CML/HCSL input clock signals according to PCIe specifications. Proper termination is included as required by the specification. For more information, see *UG0677: PolarFire FPGA Transceiver User Guide*.

The PCIe standard specifies a 100 or 125 MHz clock reference clock (Refclk) with greater than ± 300 ppm frequency stability in both the transmitting and receiving devices.

According to PCIe specifications, upstream and downstream PCIe devices must transmit data at a rate within 600 ppm of each other at all times. This specification allows a bit-rate clock source with a ± 300 ppm tolerance. To ensure that the minimum clock period is not violated, the PCIESS uses a spread-spectrum technique that does not permit modulation above the nominal frequency.

The data rate can be modulated from 0% to 0.5% of the nominal data rate frequency at a modulation rate ranging from 30 to 33 KHz. Along with the ± 300 ppm tolerance limit, both ports require the same bit-rate clock when the data is modulated using SSC.

PolarFire devices support two distinct clocking topologies defined by the PCIe specifications: common Refclk and separate Refclk.

3.1.3.1 Common Refclk

This topology is the most widely supported clocking method in open systems where the root port or root complex provides a clock to the endpoint. An advantage of this clocking architecture is that it supports spread-spectrum clocking (SSC), which is useful in reducing electromagnetic interference (EMI). PolarFire devices support incoming SSC used in common clock systems with both transceiver transmit PLLs.

3.1.3.2 Separate Refclk

This topology uses two independent clock sources: one for the root and the other for the endpoint. The clock sources must still maintain ± 300 ppm frequency accuracy and cannot use SSC.

3.1.4 PIPE

The PolarFire PCIESS relies on a standardized PIPE interface within the PolarFire transceiver. The PIPE interface is transparent within the user's PCIESS design.

For more information on the capabilities and configuration of the PIPE, see *UG0677: PolarFire FPGA Transceiver User Guide*.

3.1.5 Low-Power States

The PolarFire PCIESS supports PCIe low-power operation states known as L2 and P2 states. The PCIe hot reset and L2 and P2 compatibility settings can be selected when customizing the PCIESS design. L2 states are available in both, root port and endpoint configurations.

An endpoint can send a power management event to the host when the endpoint is in a low-power state and needs to be restored to a fully-functional power state. The endpoint automatically enters L2 state when directed to do so by its link partner. It can exit this state only when directed by the link partner. The endpoint can send a power management event to request that the link partner re-enable the link.

When the root port detects a power management event, it sets an interrupt in the PM_MSI_INT bit of the ISTATUS_LOCAL, page 50 register. The application must determine the device responsible for the event and restore it to full power.

Entry to L2 state turns off the PCIe link. The link can only be initiated by a root port. The root port enters L2 state and disables the link as soon as its link partner is ready. The application sets the "Turn Off Link" bit in the ICMD_PM, page 52 register when the root port disables the link.

The root port exits L2 state and enables the link when the application clears the "Turn Off Link" bit in the ICMD_PM register.

When the application requests a power management event and the endpoint is in L2 state, the PCIESS can request the host to wake up and re-enable the link by sending a wake-up event. An endpoint can send a wake-up event through either of the following methods:

- A beacon signal transmitted by the physical layer on the link and controlled through PIPE interface signals
- A WAKE# pin controlled by the PCIESS output signal PCIE_#_WAKE#

A root port can detect a wake-up event through either of the following methods: A beacon signal detected by the physical layer and reported through PIPE interface signals

When a root port detects a wake-up event, it automatically enables the link. As soon as the link is enabled, the application receives a power management interrupt.

3.2 Data and Transaction Layers

The PCIESS includes the data path from the transceiver to the user-defined application layer of the FPGA fabric. The AXI4 bridges the application layer to the transaction layer. The transaction layer communicates with the data link layer through FIFOs. The data link layer communicates with the physical layer through FIFOs. The data is then passed to the transaction layer blocks that manage read and write requests from the software. Finally, the data is passed to the application layer hosted in the FPGA fabric (Figure 6, page 10).

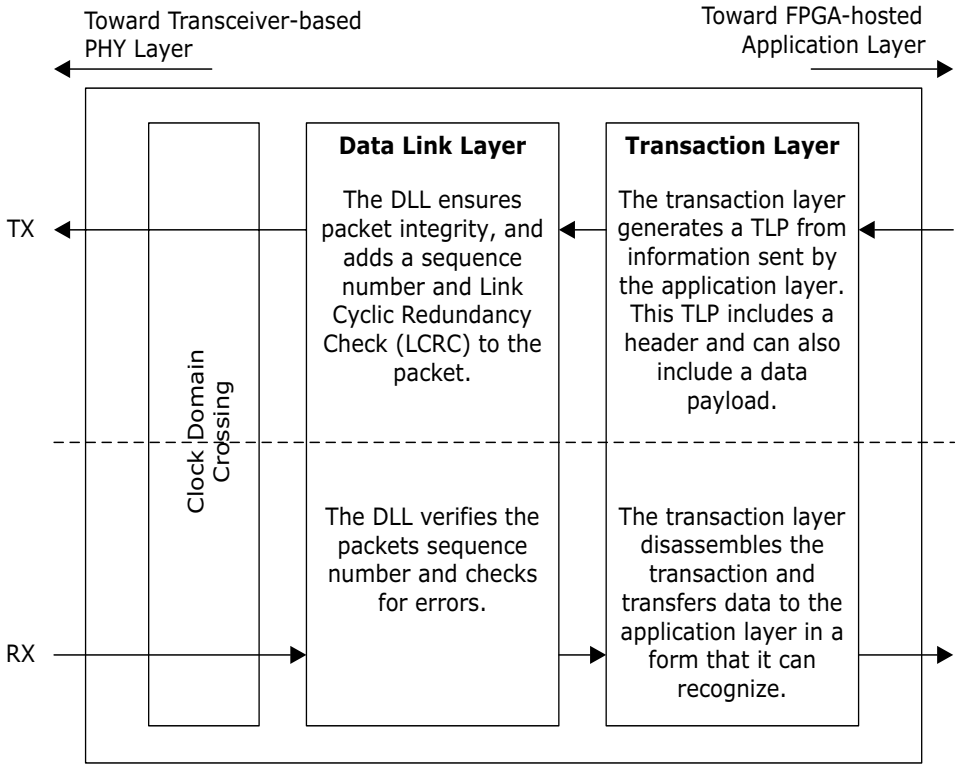
3.2.1 Data Link Layer

The data link layer (DLL) is responsible for link management including transaction layer packet (TLP) acknowledgment (a retry mechanism in case of a non-acknowledged packet), flow control across the link (transmission and reception), power management, CRC generation and checking, error reporting, and logging.

3.2.2 Transaction Layer

The transaction layer is responsible for the transfer of transaction layer packets (TLP). On the application side, it has 64-bit AXI4 master and slave interfaces. The PCIe link initiates transactions to the fabric through the AXI4 master, and the fabric initiates transactions towards the PCIe link through the AXI4 slave. The following figure shows the PCIe data and transaction layers.

Figure 6 • **PCIe Data and Transaction Layers**



3.3 System Credits

PCIe is a flow control-based protocol. Receivers advertise the supported number of receive buffers, and transmitters are not allowed to send TLPs without ensuring that sufficient receive buffer space is available. This control is provided by the 8 KB dedicated buffers included in the PCIESS receive and transmit channels.

3.3.1 Maximum Payload Size

The size of the TLP depends on the capabilities of both link partners. After the link is trained, the root complex sets the maximum payload size register (MAX_PAYLOAD_SIZE) value in the device control register. The maximum allowable payload size is 256 bytes. This limit can support up to 16 outstanding read requests in both, PCIe-to-bridge and bridge-to-PCIe directions.

After the link is trained, the PCIe host sets the MAX_PAYLOAD_SIZE value in the device control register. The replay buffer located in the data link layer stores a copy of a transmitted TLP until the transmitted packet is acknowledged by the receive side of the link. Each stored TLP includes the header, an optional data payload (the maximum size of which is determined by the maximum payload size parameter), an optional end-to-end cyclic redundancy check (ECRC), the sequence number, and the link cyclic redundancy check (LCRC) field for transaction and data integrity. The replay buffer stores the read data payload from the AXI4 master and write data payload from the AXI4 slave.

3.3.2 Credit Queue

PCIe credit queues are categorized into posted, non-posted, and completion queues. When the host sends writes to the PolarFire PCIESS, this data consumes posted and non-posted credits. When the host needs to send a completion to a read, it consumes completion credits. As is the case for most PCIe endpoints, the completion credit count is infinite in the PCIESS.

A completion timeout is a condition where the host blocks completion due to the lack of non-posted credits. When the PCIESS does a read, if completion data is not returned, the PCIESS issues a completion timeout and the transaction is voided, triggering the AXI slave to terminate the read to avoid a deadlock.

All credit settings are automatically set according to the buffer sizes fixed in the PCIESS. As per PCIe standards, when the PCIe controller issues a read request, it ensures that the controller has the ability to sync the associated read data (completion TLP).

3.3.3 Receive Buffer

The transaction layer contains an 8-KB receive buffer to accept incoming TLPs from the link and send them to the application layer for processing. The receive buffer stores TLPs based on the transaction type (posted, non-posted, or completion). A transaction always has a header but does not always have data. The receive buffer accounts for this distinction, maintaining separate resources for the headers and data for each type of transaction. For completion transactions, data (CPLD) TLPs are stored in the received buffer in the 64-bit addressing format, with each outstanding AXI4 slave read request consuming 16 credits (totaling 128 bytes), and the headers and data consuming one credit each (totaling 16 bytes).

3.3.4 Replay Buffer

In the transmit direction, the AXI4 bridge layer first checks the credits available on the far end before sending the TLP. There must be enough credits available for the entire TLP to be sent. The AXI4 bridge layer must then check that the PCIESS is ready to send the TLP. If there is enough credit, it can proceed with the sending data. If the credit is insufficient, the TLP must wait until enough credit is available.

The replay buffer is used to fetch TLPs that are:

- Issued by the AXI before their transmission on the PCIe link
- Transmitted on the PCIe link, as long as they are not acknowledged by the opposite PCIe device, in case a replay is required

The replay buffer is 8 KB, allowing the transmit buffer to support up to 16 outstanding transmit replay data packets with a maximum payload of 256 bytes each. A maximum of 32 TLPs can be supported by the replay buffer.

3.4 AXI4 Interface

The AXI4 interface of the PCIESS provides a transaction-level translation of AXI4 commands to PCIe packets and PCIe requests to AXI4 commands. The user application in the FPGA fabric must implement an AXI4 master interface to transfer data to the PCIe link and an AXI4 slave interface to receive data from the PCIe link.

3.4.1 PCIe-to-AXI Window

Address requests received by the PCIESS are provided to the AXI4 master. The PCIe-to-AXI4 address window manages read and write requests from the PCIe link and translates 32- or 64-bit PCIe base address transactions to 64-bit AXI4 base address transactions.

3.4.2 AXI Master

A typical PCIe application interface uses an AXI master interface to respond to data read requests and an AXI slave interface to initiate requests. The AXI master performs the following functions:

- Conveys PCIe read and write transactions to the fabric in the form of AXI4 reads and writes.
- Accesses the DMA controller through the bridge and issues reads and writes that allow data pulled from the fabric to be sent over PCIe and data pulled from the PCIe to be written out to the fabric.

3.4.3 AXI Master Write Transaction Handling

Write transactions are handled in big-endian order as required by the PCI Express Base Specification. The master path does not reorder transactions, but arbitrates between transactions at the AXI4 master interface. If a transaction is currently waiting for a response phase, the transaction is allowed to complete before the read transaction is forwarded to the AXI4 master interface. PCIe transactions sizes may vary up to the configurable maximum payload size (256 bytes):

- AXI4 transactions are limited to 256 bytes, and the received TLP is divided into several AXI4 transactions.
- AXI4 master receives a write transaction, processing it in 256-byte segments.
- TLP is de-constructed from the PCIe system and sent to the AXI4 interface in little-endian format.

3.4.4 AXI Master Read Transaction Handling

Read transactions are handled similar to write transactions, except that before transferring the transaction to the AXI4 master read channel, the PCIESS checks the transmit buffer for available space. If the transmit replay buffer does not have sufficient place to store the PCIe completions, the PCIESS does not transfer the read transaction. The number of outstanding AXI4 master read transactions is, therefore, limited by the size of the transmit buffer.

The AXI4 master read channel can receive transactions in any order, and data can be completely interleaved. However, the PCIESS generates completions in the order they are initiated on the link.

3.4.5 AXI Slave Block

The AXI4 slave interface forwards AXI read and write requests from the FPGA fabric to the PCIe link. The fabric application initiates PCIe transactions (memory write TLP and memory read TLP transactions) using the slave interface. The data on a read request comes back to the same interface. The slave path does not reorder transactions, but arbitrates between transactions if they occur simultaneously with master read completions. The order of priority for arbitrations is as follows:

1. Master read completions
2. Slave write requests
3. Slave read requests

3.4.6 AXI Slave Write Transaction Handling

Slave write transactions support incrementing address bursts, fixed bursts, wrapping bursts, and narrow type transfers. Data interleaving, however, is not supported. Data packets of a maximum of 2 K bytes can be created. Wait states are used if the buffer is full, or has less than 128 bytes of available space. Write responses are generated as soon as the last phase is over with support for up to four outstanding write transactions.

3.4.7 AXI Slave Read Transaction Handling

PCIESS generates a PCIe tag, arbitrates between write requests and completions, and then checks for available credits. An error response is generated if a timeout occurs or if a “completion with error” status is received.

3.4.8 Base Address Register Settings

The PCIe implementation supports up to six 32-bit base address registers (BARs). Each BAR is 32 bits, but two BARs can be combined to make a 64-bit BAR. For example, BAR0 (address offset 010h) and BAR1 (address offset 014h) define the type and size of BAR01 of the PCIe native endpoint.

An endpoint requesting memory resources through a BAR must set the BAR’s prefetchable bit, unless the range contains locations with read side-effects or locations where the device does not tolerate write merging. It is strongly recommended that memory-mapped resources be designed as prefetchable whenever possible.

3.4.9 Address Translation on AXI4 Interface

The address space for PCIe is different from the AXI4 address space. To access one address space from another address space, an address translation process is required. The PCIe embedded block can receive both 32-bit and 64-bit PCIe address requests.

The PCIESS uses address translation to convert:

- PCI Express read and write requests to AXI4 master interface read and write transactions
- AXI4 slave interface read and write transactions to PCI Express read and write requests

3.4.9.1 Master Windows

The PCIESS Configurator in Libero SoC software provides a GUI to configure the BAR settings for an endpoint application.

When the PCIESS is in endpoint mode, address translation moves the address from PCIe domain to AXI domain. There is direct mapping between the BAR and the address tables, as shown in the following figure.

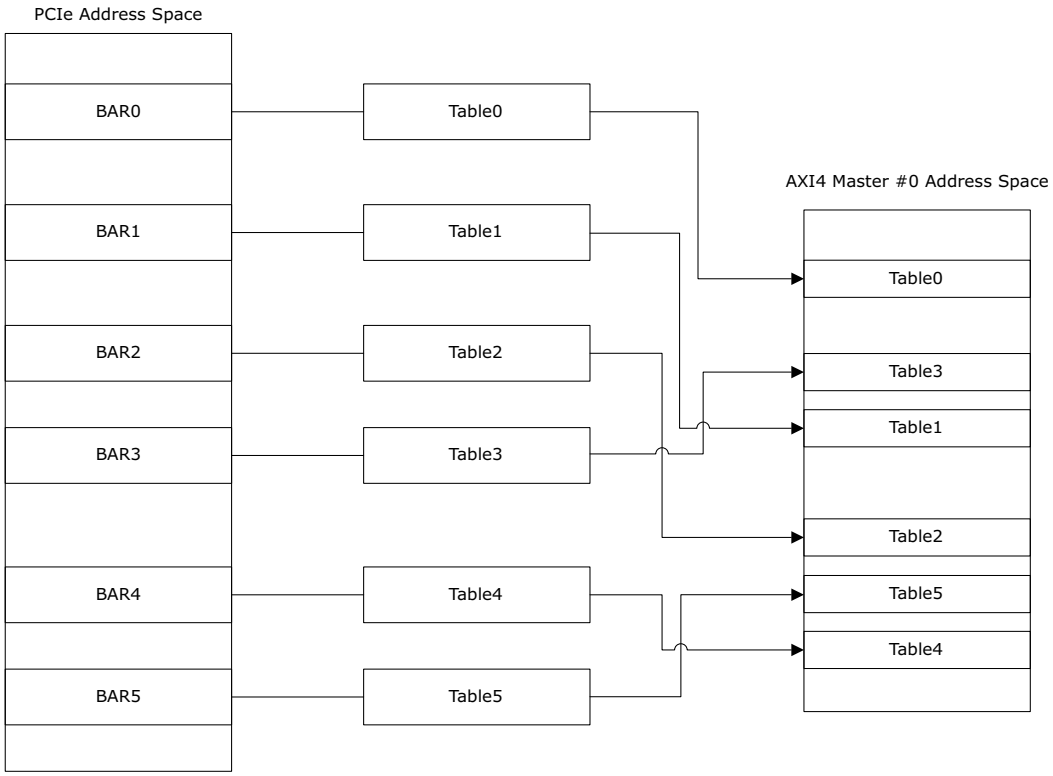
BAR0-5 can be of any type and width, with each BAR having one of the six corresponding address translation tables (ATR).

For 64-bit BARs, the even register is selected to be 64 bits wide. The subsequent (odd) register then serves as the upper half of 64 bits.

For example, a prefetchable 64-bit BAR or a non-prefetchable 64-bit BAR can be memory mapped to BAR01. With the 64-bit BAR mapping, address tables are used as follows:

- BAR01 targets ATR0 and ATR1 remains unused
- BAR23 targets ATR2 and ATR3 remains unused
- BAR45 targets ATR4 and ATR5 remains unused

Figure 7 • **PCIe to AXI4 Master Address Translation Endpoint Mode**

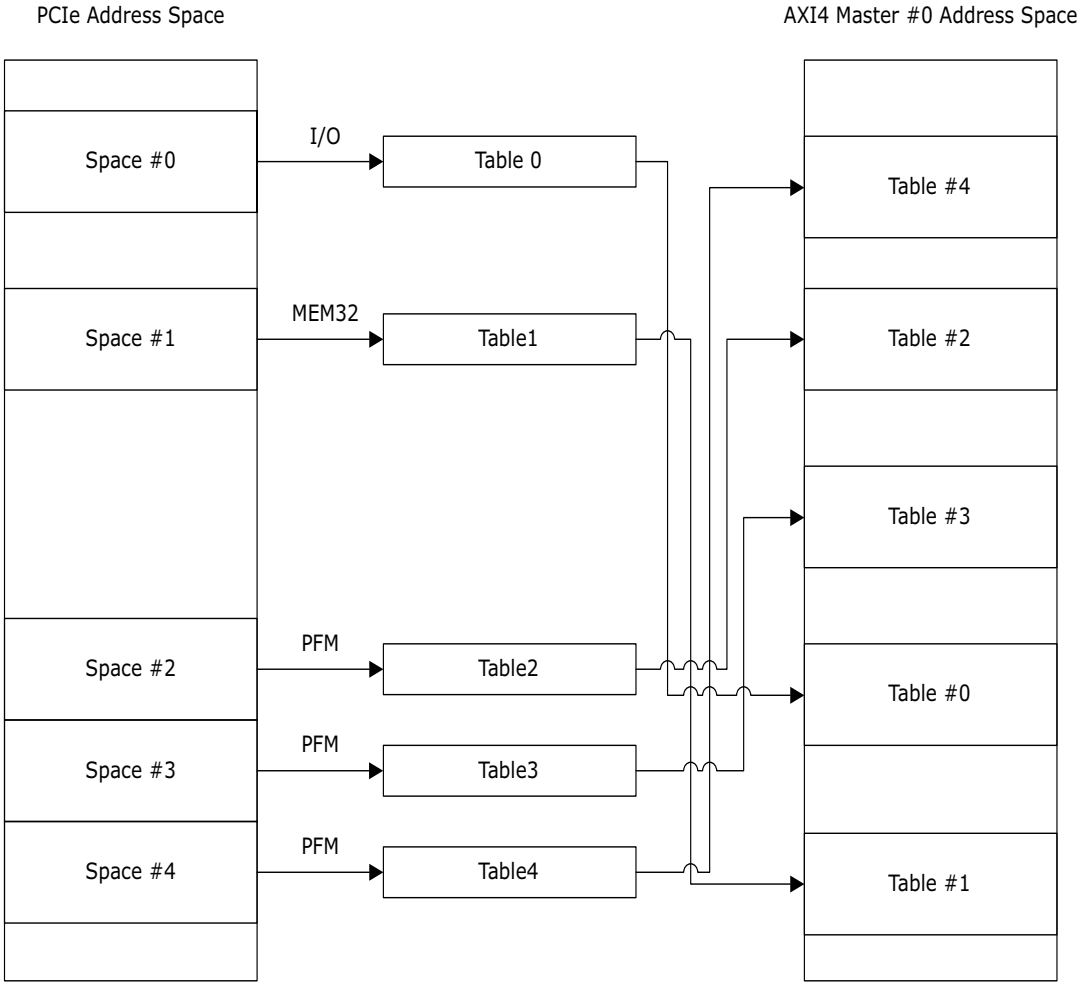


When the PCIESS is in root port mode (Figure 8, page 14), up to six translation tables can be implemented. When transferring PCI Express receive requests to the AXI master, the bridge performs a windows match using the PCIe 64-bit address. If a match is found, the bridge forwards the request to the desired AXI4 master interface and adds the corresponding AXI base address.

When the PCI Express BAR0/1 are configured as a 64-bit prefetchable memory space of 16 K bytes. PCIe read and write requests targeting BAR0 or BAR1 are routed to the bridge configuration space. This memory space operates in the following two ways:

- When I/O or prefetchable memory windows are implemented, PCIe read and write requests targeting I/O or prefetchable memory windows are routed to the PCIe window address translation module.
- When I/O or prefetchable memory windows are not implemented, PCIe read and write requests that do not target BAR0 or BAR1 are routed to the PCIe window address translation module.

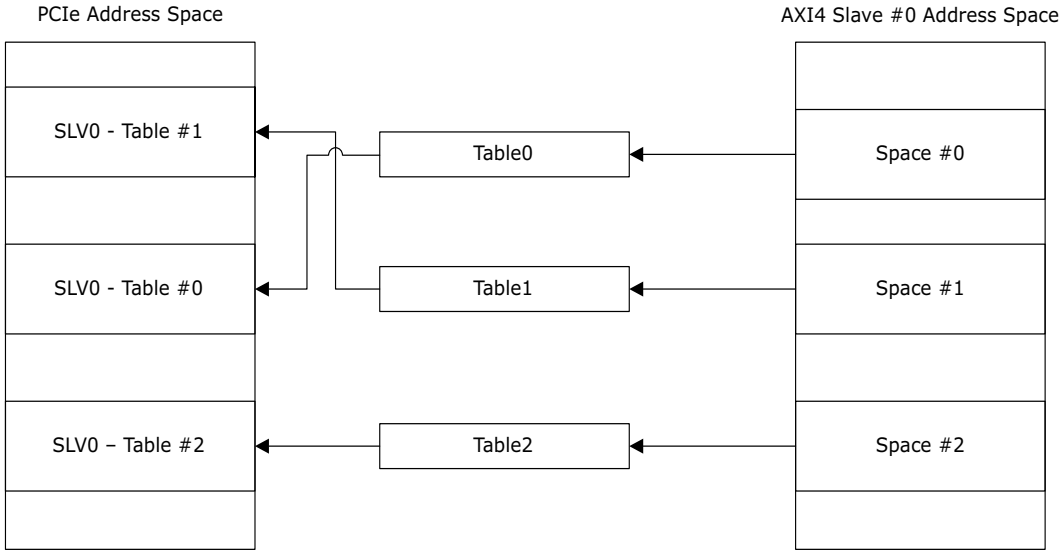
Figure 8 • **PCIe to AXI4 Master Address Translation Root Port Mode**



3.4.9.2 Slave Windows

The address translation method used to transfer AXI slave receive requests to the PCIe interface is similar to that used in the endpoint mode. Up to eight translation tables can be implemented for each AXI4 slave interface.

Figure 9 • **AXI4 Slave to PCIe Address Translation**



3.5 ECRC Handling

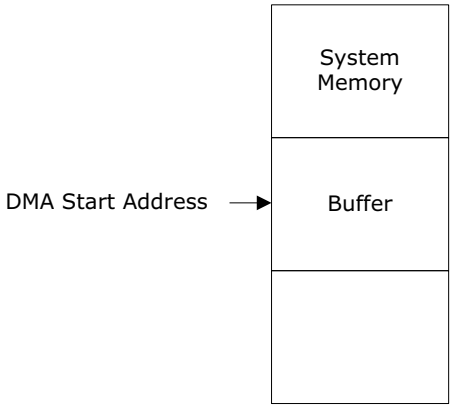
The PCIESS optionally performs automatic ECRC to ensure data integrity. The PCIe implementation transmits a TLP with ECRC from the transmit port of the application layer. When using ECRC forwarding mode, ECRC check is done in the application layer. The PCIE_PEX_SPC2, page 47 bridge configuration register controls the ECRC settings.

3.6 DMA Transfers

The DMA engines in the PCIESS help deliver higher performance on both, write and read functions. The PCIESS can implement up to two fully-independent DMA engines that enable programming of direct or SGDMA transfers. For more information about DMA registers, see DMA Engine Registers, page 54.

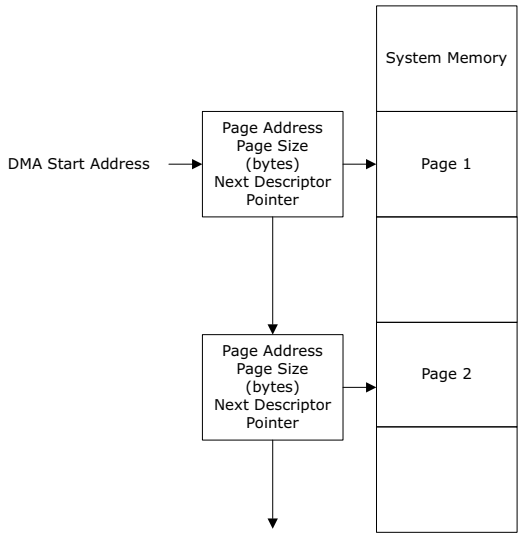
In direct DMA transfer mode, the DMA start address is a pointer to a contiguous data buffer mapped in the PCI bus address space. Data is read from and written to the buffer sequentially.

Figure 10 • **Direct SMA Transfer**



In SG transfer mode, the DMA source and/or destination start address is a pointer to a chained list of page descriptors. Each descriptor contains the address and size of a data block (page), as well as a pointer to the next descriptor block to enable circular buffers.

Figure 11 • **SGDMA Chained List**



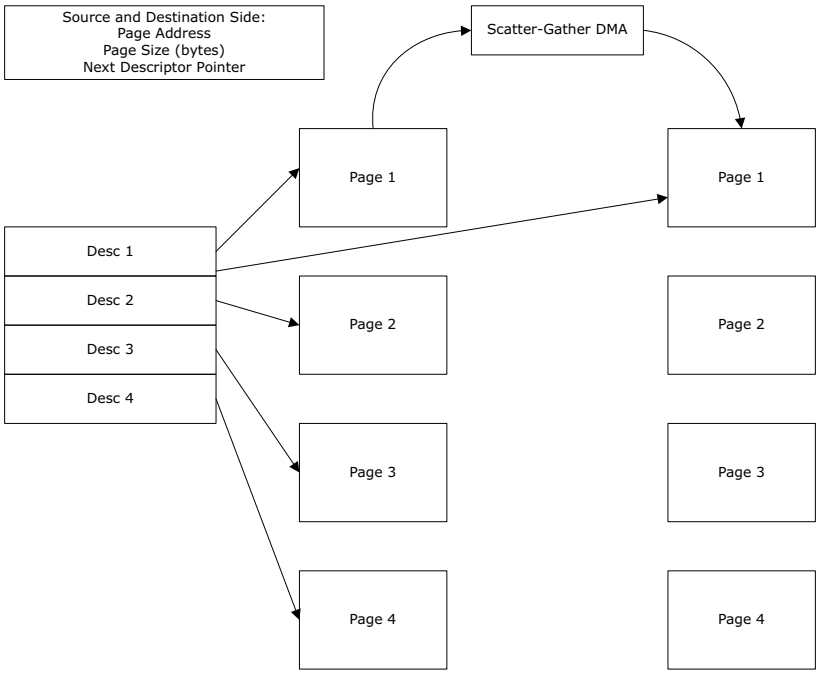
3.7 Scatter-Gather Types

The following figures show the different types of Scatter-Gather defined for the DMA.

3.7.1 DMA0

Source and Destination addresses are set according to the Descriptor. DMA source is always PCIe and destination is reconfigurable.

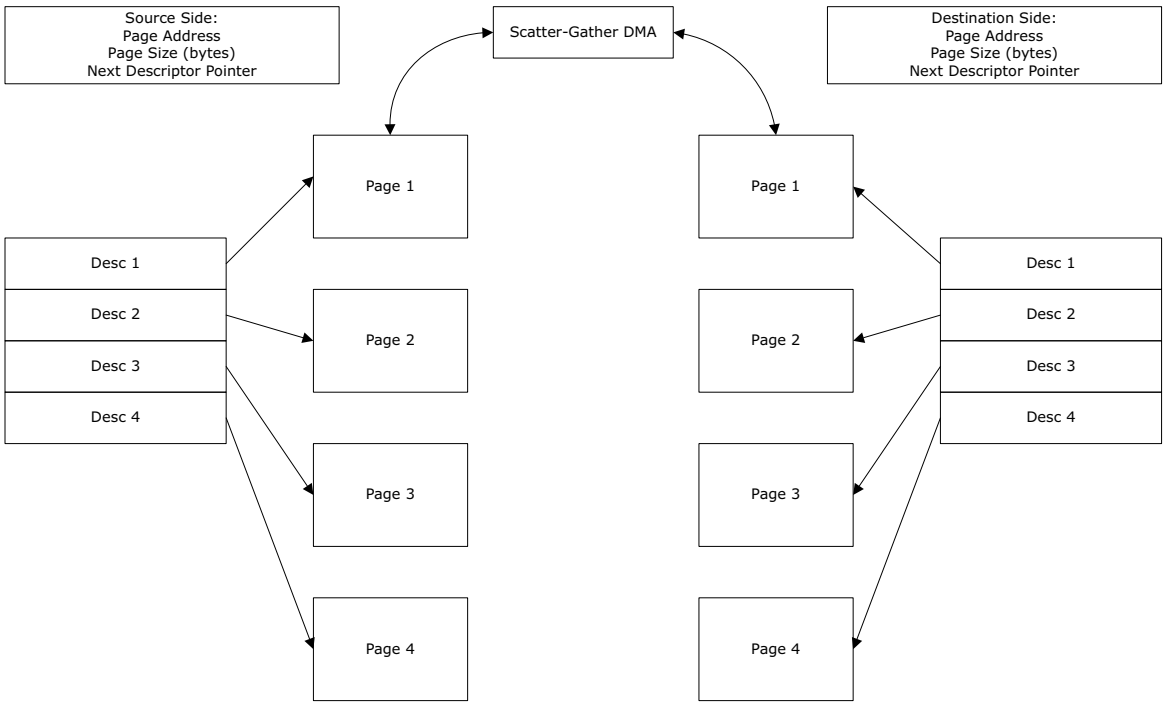
Figure 12 • **DMA0 Descriptor**



3.7.2 DMA1

Independent Scatter-Gather for both Source and Destination, DMA destination is always PCIe and source is reconfigurable.

Figure 13 • **DMA1 Descriptor**



3.7.3 Scatter-Gather DMA Descriptors

The following table lists the Scatter-Gather DMA Descriptors.

Table 1 • **Scatter-Gather DMA Descriptors**

Name	Byte Offset	R/W	Description
DESC_STATUS	0x00 - 0x03	RW	It enables dynamic monitoring of the SG-DMA transfer (when 0th bit of DESC_CONTROL register is set) Bits [3:0]: Provides the status number of the DMA engine. This number is incremented, enabling the application to determine the last processed descriptor, which is key in streaming flow between asynchronous devices. Bits [7:4]: Bit 4: Indicates SG-DMA descriptor has been processed. Bit 5: Indicates an error occurred during the processing of the current SG-DMA descriptor. Bit 6: Indicates an End of Packet (EOP) condition has been reported by the source of the SG-DMA transfer. Bit 7: Reserved Bits [31:8]: Indicates the Processed Page Size, which is the actual written or read page size.

Table 1 • **Scatter-Gather DMA Descriptors *(continued)***

Name	Byte Offset	R/W	Description
DESC_CONTROL	0x04 - 0x07	RO	DESC_CONTROL enables dynamic control of the SG-DMA transfer. Bit [0]: Defines whether the DMA engine provides a status report by writing to DESC_STATUS when the current SG-DMA descriptor has been processed. Bits [3:1]: Reserved Bits [7:4]: Defines when an interrupt should be issued. Bit 4: Indicates an IRQ is issued when this SG-DMA descriptor has been processed. Bit 5: Indicates an IRQ is issued if an error occurs Bit 6: Indicates an IRQ is issued if the source of the transfer reports an EOP condition. Bit 7: Reserved Bits [31:8]: Provides the page size in bytes, from 1 to 16 M bytes.
DESC_NEXT_ADDR[63:32]	0x0C - 0x0F	RO	Indicates next descriptor address. This field must be aligned on a 32-byte boundary.
DESC_NEXT_ADDR[31:5]	0x08 - 0x0B	RO	Indicates next descriptor address.
DESC_NEXT_RDY			Indicates if the next SG-DMA Descriptor is ready and fetchable.
DESC_SE_COND[3:0]			Defines the Start and End conditions for SG-DMA descriptor processing. Bit 0: Indicates end the DMA transfer after this SG-DMA descriptor has been processed (equivalent to an End Of Chain). Bit 1: Indicates to abort this SG-DMA descriptor processing if an error occurs. Bit 2: Reserved Bit 3: Indicates to start this SG-DMA descriptor processing when the source of the transfer reports a SOP reception.
DESC_SRC_ADDR[31:0]	0x10 - 0x13	RO	Indicates the source address of the descriptors.
DESC_SRC_ADDR[63:32]	0x14 - 0x17	RO	
DESC_DEST_ADDR[31:0]	0x18 - 0x1B	RO	Indicates the destination address of the descriptors.
DESC_DEST_ADDR[63:32]	0x1C - 0x1F	RO	

3.8 PCIESS Pin List

The PCIESS block is generated using the Libero PCIe Configurator. The generation of the PCIESS block includes pins based on the PCIe Configurator settings. The following table lists the pin descriptions.

Table 2 • **PCIESS Pin List**

Port Name	Direction	Description
AXI_CLK	Input	Global AXI clock. Shared for both PCIESS interfaces. For information on minimum operating frequency, see *DS0141: PolarFire FPGA Datasheet*.
AXI_CLK_STABLE	Input	Clock lock signal. Indicates that the AXI_CLK source from the fabric is locked and ready for use.
Master		
PCIESS_AXI_#_M_ARADDR[31:0]	Output	Read address. The address of the first transfer in a read burst transaction.
PCIESS_AXI_#_M_ARBURST	Output	Read burst type. The burst type and the size of information determine how the address for each transfer within the burst is calculated.
PCIESS_AXI_#_M_ARID[3:0]	Output	Read address ID. Identification tag for the read address group of signals.
PCIESS_AXI_#_M_ARLEN[7:0]	Output	Burst length. Indicates the exact number of transfers in a burst.
PCIESS_AXI_#_M_ARREADY	Input	Read address ready. Indicates that the slave is ready to accept an address and associated control signals.
PCIESS_AXI_#_M_ARSIZE[1:0]	Output	Burst size. Indicates the size of each transfer in the burst.
PCIESS_AXI_#_M_ARVALID	Output	Read address valid. Indicates that the channel is signaling valid read address and control information.
PCIESS_AXI_#_M_AWADDR[31:0]	Output	Write address. The address of the first transfer in a write burst transaction.
PCIESS_AXI_#_M_AWBURST	Output	Write burst type. The burst type and the size of information determine how the address for each transfer within the burst is calculated.
PCIESS_AXI_#_M_AWID[1:0]	Output	Write address ID. Identification tag for the write address group of signals.
PCIESS_AXI_#_M_AWLEN[4:0]	Output	Burst length. Indicates the exact number of transfers in a burst, which determines the number of data transfers associated with the address.
PCIESS_AXI_#_M_AWREADY	Input	Write address ready. Indicates that the slave is ready to accept an address and associated control signals.
PCIESS_AXI_#_M_AWSIZE	Output	Burst size. Indicates the size of each transfer in the burst.
PCIESS_AXI_#_M_AWVALID	Output	Write address valid. Indicates that the channel is signaling valid write address and control information.
PCIESS_AXI_#_M_BID[3:0]	Input	Response ID tag. Identification tag for the write response.
PCIESS_AXI_#_M_BREADY	Output	Response ready. Indicates that the master is ready to accept a write response.
PCIESS_AXI_#_M_BRESP	Input	Write response. Indicates the status of the write transaction.
PCIESS_AXI_#_M_BVALID	Input	Write response valid. Indicates that the channel is signaling a valid write response.

Table 2 • **PCIESS Pin List *(continued)***

Port Name	Direction	Description
PCIESS_AXI_#_M_RDATA[63:0]	Input	Read data.
PCIESS_AXI_#_M_RID[3:0]	Input	Read ID tag. Identification tag for the read data group of signals generated by the slave.
PCIESS_AXI_#_M_RLAST	Input	Read last. Indicates the last transfer in a read burst.
PCIESS_AXI_#_M_RREADY	Output	Read ready. Indicates that the master can accept the read data and associated control signals, along with response information.
PCIESS_AXI_#_M_RRESP	Input	Read response. Indicates the status of the read transfer.
PCIESS_AXI_#_M_RVALID	Input	Read valid. Indicates that the channel is signaling the required read data.
PCIESS_AXI_#_M_WDATA[63:0]	Output	Write data.
PCIESS_AXI_#_M_WLAST	Output	Write last. Indicates the last transfer in a write burst.
PCIESS_AXI_#_M_WREADY	Input	Write ready. Indicates that the slave can accept the write data.
PCIESS_AXI_#_M_WSTRB[7:0]	Output	Write strobes. Indicates the byte lanes that hold valid data. There is one write strobe bit for every eight bits of the write data bus.
PCIESS_AXI_#_M_WVALID	Output	Write valid. Indicates that valid write data and strobes are available.
Slave		
PCIESS_AXI_#_S_ARADDR[31:0]	Input	Read address. The address of the first transfer in a read burst transaction.
PCIESS_AXI_#_S_ARBURST[1:0]	Input	Burst type. The burst type and the size of information determine how the address for each transfer within the burst is calculated.
PCIESS_AXI_#_S_ARID[3:0]	Input	Read address ID. Identification tag for the read address group of signals.
PCIESS_AXI_#_S_ARLEN[7:0]	Input	Burst length. Indicates the exact number of transfers in a burst.
PCIESS_AXI_#_S_ARREADY	Output	Write address ready. Indicates that the slave is ready to accept an address and the associated control signals.
PCIESS_AXI_#_S_ARSIZE[1:0]	Input	Burst size. Indicates the size of each transfer in the burst.
PCIESS_AXI_#_S_ARVALID	Input	Read address valid. Indicates that the channel is signaling valid read address and control information.
PCIESS_AXI_#_S_AWADDR[31:0]	Input	Write address. Address of the first transfer in a write burst transaction.
PCIESS_AXI_#_S_AWBURST[1:0]	Input	Burst type. The burst type and the size information determine how the address for each transfer within the burst is calculated.
PCIESS_AXI_#_S_AWID[3:0]	Input	Write address ID. Identification tag for the write address group of signals.
PCIESS_AXI_#_S_AWLEN[7:0]	Input	Burst length. Indicates the exact number of transfers in a burst, which determines the number of data transfers associated with the address.
PCIESS_AXI_#_S_AWREADY	Output	Write address ready. Indicates that the slave is ready to accept an address and associated control signals.
PCIESS_AXI_#_S_AWSIZE[1:0]	Input	Burst size. Indicates the size of each transfer in the burst.
PCIESS_AXI_#_S_AWVALID	Input	Write address valid. Indicates that the channel is signaling valid write address and control information.

Table 2 • **PCIESS Pin List *(continued)***

Port Name	Direction	Description
PCIESS_AXI_#_S_BID[3:0]	Output	Response ID tag. Identification tag for the write response.
PCIESS_AXI_#_S_BREADY	Input	Response ready. Indicates that the master can accept a write response.
PCIESS_AXI_#_S_BRESP[1:0]	Output	Write response. Indicates the status of the write transaction.
PCIESS_AXI_#_S_BVALID	Output	Write response valid. Indicates that the channel is signaling a valid write response.
PCIESS_AXI_#_S_RDATA[63:0]	Output	Read data.
PCIESS_AXI_#_S_RID[3:0]	Output	Read ID tag. Identification tag for the read data group of signals generated by the slave.
PCIESS_AXI_#_S_RLAST	Output	Read last. Indicates the last transfer in a read burst.
PCIESS_AXI_#_S_RREADY	Input	Read ready. Indicates that the master can accept the read data and response information.
PCIESS_AXI_#_S_RRESP[1:0]	Output	Read response. Indicates the status of the read transfer.
PCIESS_AXI_#_S_RVALID	Output	Read valid. Indicates that the channel is signaling the required read data.
PCIESS_AXI_#_S_WDATA[63:0]	Input	Write data.
PCIESS_AXI_#_S_WLAST	Input	Write last. Indicates the last transfer in a write burst.
PCIESS_AXI_#_S_WREADY	Output	Write ready. Indicates that the slave can accept the write data.
PCIESS_AXI_#_S_WSTRB[7:0]	Input	Write strobes. Indicates the byte lanes that hold valid data. There is one write strobe bit for each eight bits of the write data bus.
PCIESS_AXI_#_S_WVALID	Input	Write valid. Indicates that valid write data and strobes are available.
PCIe		
PCIE_#_M_RDERR	Input	Master read data error. Allows the application to report data error.
PCIE_#_M_WDERR	Output	Master write data error. Asserted when an uncorrectable error is detected by the memory's ECC logic when reading data from the buffer. The error is reported on the same clock cycle as the affected data.
PCIE_#_S_RDERR	Output	Slave read data error. Asserted when an uncorrectable error is detected by the memory's ECC logic when reading data from the buffer.
PCIE_#_S_WDERR	Input	Slave write data error. Allows the application to report data error.
PCIE_#_L2_EXIT	Output	L2 exit. Asserted for one clock cycle when the link training and status state machine (LTSSM) exits the L2 state. Prompts the application layer to perform a global reset.
PCIE_#_HOT_RST_EXIT	Output	Hot reset exit. Asserted for one clock cycle when the LTSSM exits hot reset state. Prompts the application layer to perform a global reset.
PCIE_#_DLUP_EXIT	Output	DL-up exit. Indicates transition from DL_UP to DL_DOWN.
PCIE_#_INTERUPT[7:0]	Input	Local interrupt input ports.The fabric logic can drive up to eight interrupt sources by generating a pulse (high) on the ports. [7:0] can be used for MSI. [0] is also available for INTx. Used only for endpoints. MSI offsets are [negotiated interrupt-1:negotiated interrupt-8]

Table 2 • **PCIESS Pin List *(continued)***

Port Name	Direction	Description
PCIE_#_INTERUPT_OUT	Output	Local interrupt output port. Indicates that one of the possible interrupt sources was detected and the user can read the interrupt through the DRI. It is a level sensitive signal whenever an interrupt described in ISTATUS_LOCAL, SEC_ERROR_INT register, DED_ERROR_INT register, and PCIE_EVENT_INT register is active, the PCIE_#_INTERUPT_OUT signal gets asserted and remains high. It is low when the corresponding interrupts in the registers are cleared.
PCIE_#_LTSSM[4:0]	Output	LTSSM state encoding: LTSSM_DET_QUIET : h0x0 LTSSM_DET_ACT : h0x1 LTSSM_POL_ACT : h0x2 LTSSM_POL_COMP : h0x3 LTSSM_POL_CFG : h0x4 LTSSM_CFG_LWSTR : h0x5 LTSSM_CFG_LWACC : h0x6 LTSSM_CFG_LWAIT : h0x7 LTSSM_CFG_LNACC : h0x8 LTSSM_CFG_CPLT : h0x9 LTSSM_CFG_IDLE : h0xa LTSSM_RCV_RLOCK : h0xb LTSSM_RCV_EQL : h0xc LTSSM_RCV_SPEED : h0xd LTSSM_RCV_RCFG : h0xe LTSSM_RCV_IDLE : h0xf LTSSM_L0 : h0x10 LTSSM_L0S : h0x11 LTSSM_L1_ENTRY : h0x12 LTSSM_L1_IDLE : h0x13 LTSSM_L2_IDLE : h0x14 LTSSM_L2_XMIT/WAKE : h0x15 LTSSM_DISABLED : h0x16 LTSSM_LOOPBACK_ENTRY : h0x17 LTSSM_LOOPBACK_ACTIVE : h0x18 LTSSM_LOOPBACK_EXIT : h0x19 LTSSM_HOTRESET : h0x1a
PCIE_#_WAKE#	Output	Wake-up signal. L2/P2 implementation: L2 exit request to RP. Wake (WAKE#) is connected to any I/O on the PolarFire device. It is required on any add-in card or system board that supports wake-up functionality compliant with the PCIE CEM specification. WAKE# can use any general purpose IO.
PCIE_#_PERST_N	Input	Asynchronous. Input signal used for L2/P2 implementation: L2 exit request from RP. PERST_N can use any general purpose IO.
PCIE_#_TL_CLK_125MHz	Input	125 MHz (maximum) clock input. Continuous running clock is required for PCIe core transaction layer. Connects to the DIV_CLK output from the TX_PLL.

Table 2 • **PCIESS Pin List *(continued)***

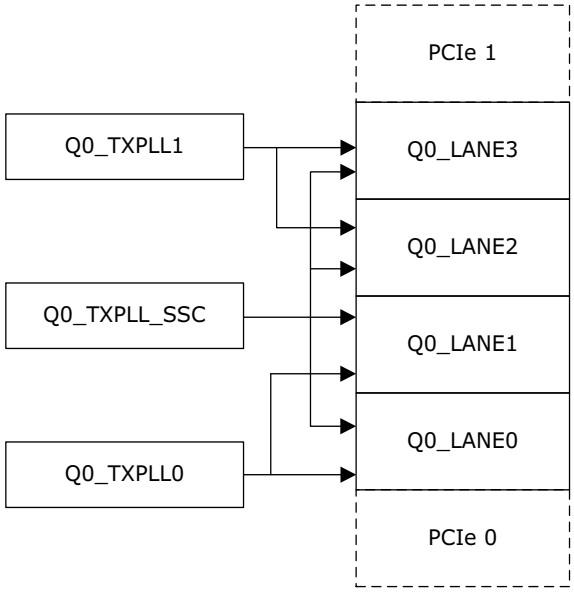
Port Name	Direction	Description
Transceiver		
CLKS_FROM_TXPLL_TO_PCIE_#	Input	PCIESS_LANE#_TX_BIT_CLK_#: This port must be driven by the BIT_CLK output of the Tx PLL. Gen1 1.25 G, Gen2 2.5 G, and mix of Gen 1 and Gen 2 is 2.5 G.
		PCIESS_LANE#_CDR_REF_CLK_#: Reference clock to lane CDR. Connects to the REF_CLK output of the TX_PLL.
		PCIESS_LANE_#_TX_PLL_LOCK_#: Lock status input to PCIESS. Connects to the lock output of the TX_PLL.
PCIESS_LANE_TX_P[0:3]	Output	Transceiver differential output transmit data.
PCIESS_LANE_TX_N[0:3]		
PCIESS_LANE_RX_P[0:3]	Output	Transceiver differential input receive data.
PCIESS_LANE_RX_N[0:3]		

Note: Unless otherwise indicated, all signals are active high.

4 Implementation

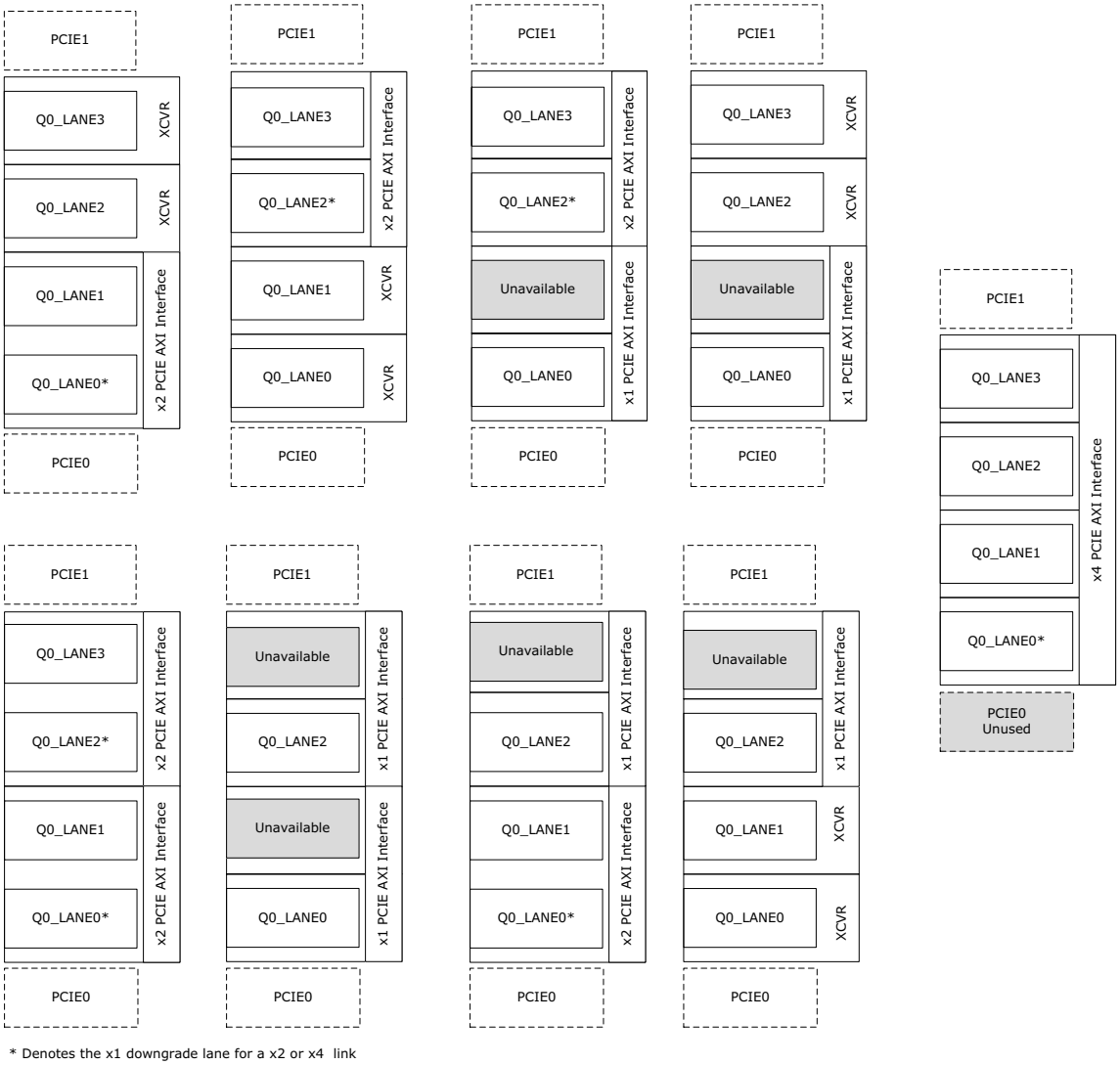
The PolarFire PCIe core uses several embedded blocks that are built using Libero configurators. PCIESS functionality is reserved for the Quad0 transceiver block, and this functionality allows up to two x1 or x2 PCIe endpoint/rootports links or one x4 PCIe endpoint/rootport link. PCIe 0 and PCIe 1 blocks can be used in any combination of x1 and x2 links within Quad0. A PCIe x4 link is only supported using PCIe 1, PCIe 0 is unused. The Libero configurator allows the setting of the reference clock and data rates for the PCIe block. This information is used to generate the configuration settings for the PMA as well as associated interface logic. The configurators build the components that are used to instantiate/configure specific hardware macros including the PMA and PCS blocks using the Libero SmartDesign software.

Figure 14 • **Transceiver Quad0 PCIESS Overview**



Lane[0:1] and Lane[2:3] share on-chip hardware resources that create inter-dependency between the physical lanes. The possible combinations for implementing and mixing the PCIe controllers on four physical XCVR lanes within QUAD0 are shown in the following figure.

Figure 15 • **Legal Combinations of PCIe and XCVR Protocols**



* Denotes the x1 downgrade lane for a x2 or x4 link

4.1 Libero Configurators

The PolarFire FPGA transceiver configurator is the preferred tool for the wrapper generation needed to instantiate the transceiver primitive macros called PF_XCVR_REF_CLK, PF_TXPLL, and PF_PCIE. The configurator is part of the Libero SoC design tools and is available when the PolarFire macros are downloaded into the Libero catalog. The following table provides details of three Libero configurator modules used by the Libero FPGA design when the blocks are implemented in the design. These three blocks must be instantiated and configured in the PCIe design.

Table 3 • **Libero Configurators in Libero Software**

Libero Configurator	Macro	Details
Transceiver Reference Clock	PF_XCVR_REF_CLK	Generates the reference clock based on the input to the GUI—differential or single-ended input buffer and single or dual-clock input to the transmit PLL clock network. Reference clocks for PCIESS systems use differential HCSL/LVDS. However, this can vary according to the system application.
Transmit PLL[1]	PF_TXPLL	Generates the TxPLL/TxPLL_SSC based on the input to the GUI. Typically a 100 MHz clock (Refclk) with greater than ± 300 ppm frequency stability is used for PCIe applications. For Refclk flexibility, the PCIESS block accepts 100 MHz or 125 MHz input and translates for PCIe Gen1 or Gen2 speeds.
PCI Express	PF_PCIE	Configures the requested number of lanes with the same PMA and PCS settings—location of each lane and CDRPLL settings. The configurator has presets for all the supported protocols.

1. It is not advisable to share the TxPLL with other serial protocols that have a tight transmit jitter specification.

Each transceiver module configurator automatically guides the user through a sequential selection of choices and defaults. Each configurator maintains a macro diagram that displays module properties. When all of the choices are made, the configurator generates a macro specific to the requirements of the design. Only the relevant ports appear in the generated macro.

This section describes how to enter these configuration parameters in the Libero configurator GUIs.

A PCIe design requires the transceiver reference clock and transceivers transmit PLL blocks to be configured and instantiated in the design. For more information on TX_REF_CLK and TX_PLL block configurators, see *UG0677: PolarFire FPGA Transceiver User Guide*.

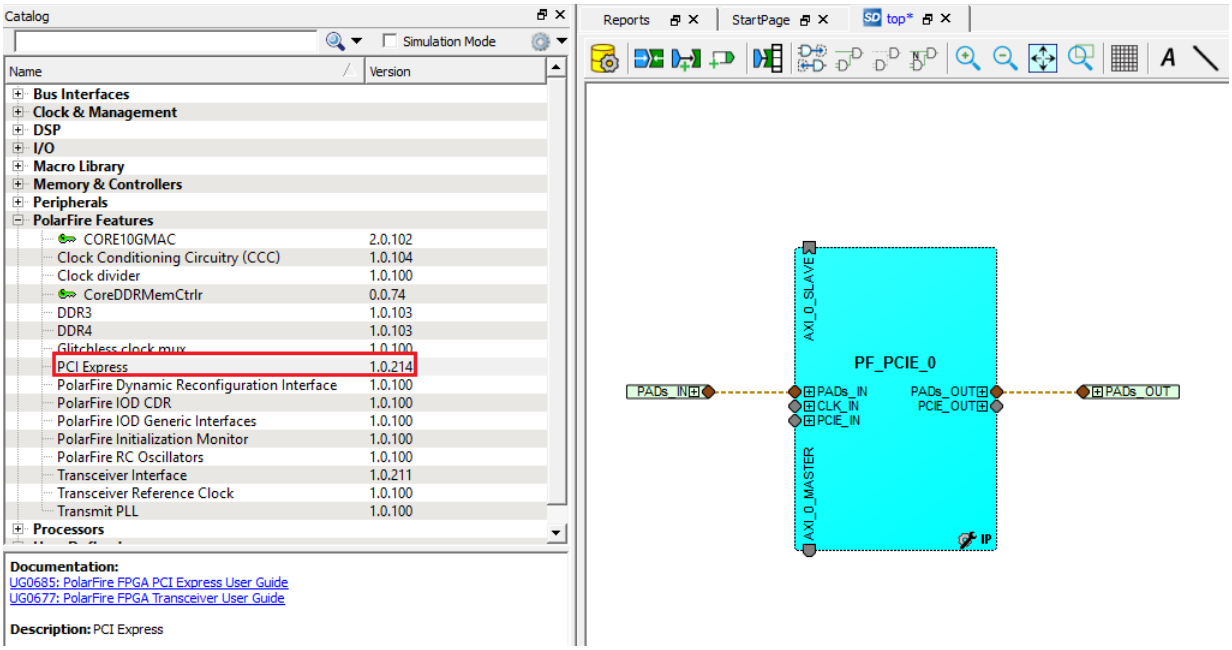
4.2 PCIe Configurator

The PCIe configurator is used to build a PCIe endpoint or rootport PCIe block. The configurator sets up the correct PCIESS registers and ports based on the user inputs.

To initiate the PCIe configurator follow these steps:

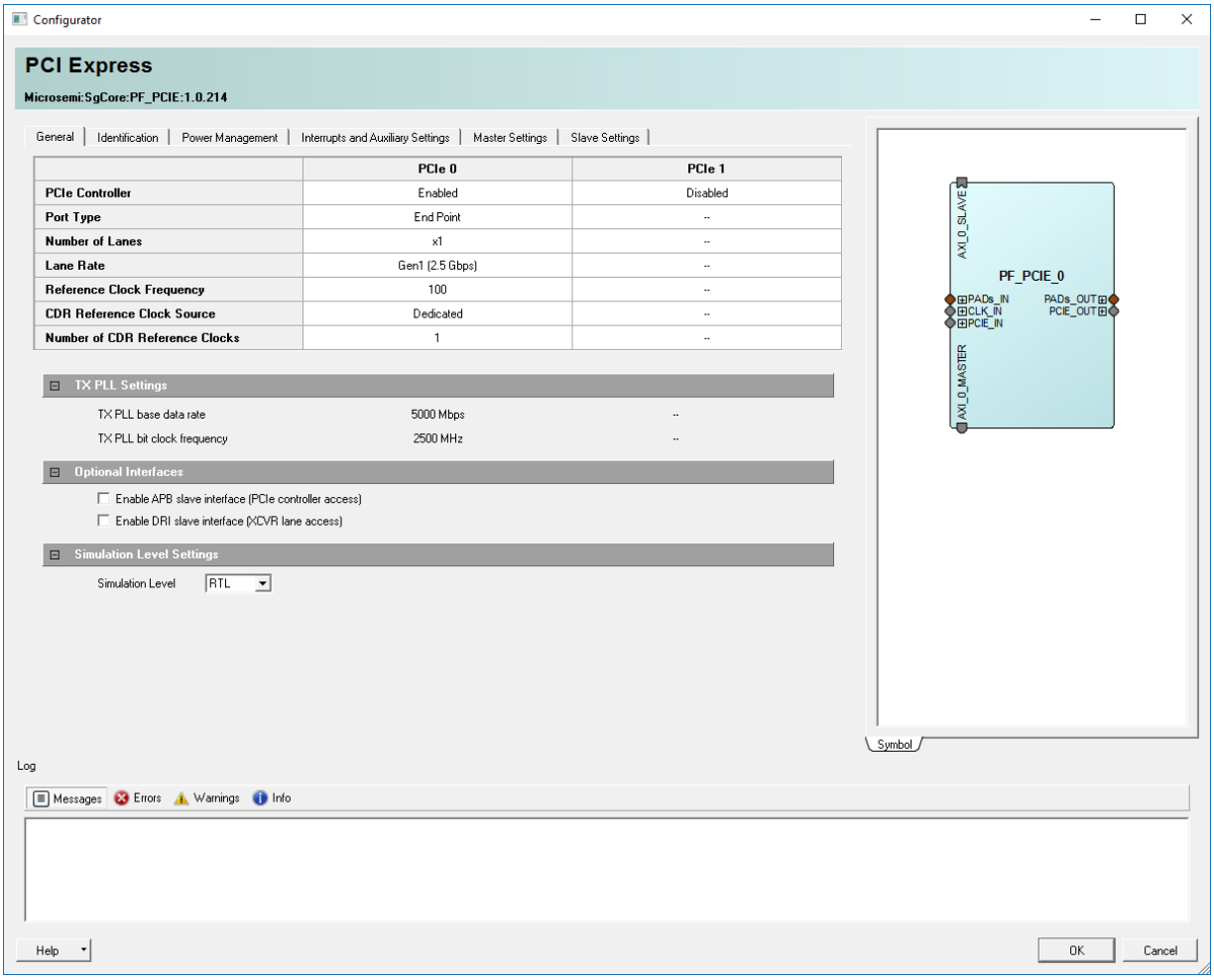
1. Access the PCIe module in the PolarFire Features from the catalog as shown in the following figure.

Figure 16 • **PCIe Selection from Catalog**



2. Double-click on each PF_PCIE block on the SmartDesign canvas to launch the configurator, as shown in the following figure. The GUI allows the user to select the related PCIe properties.

Figure 17 • **PCIe General Settings**



The following table lists the options available in the General tab.

Table 4 • **PCIe General Settings**

PCIE General Settings (PCIe 0 and PCIe 1)	**Options**	**Default**
PCIe Controller	Enabled and Disabled	PCIe 0 = Enabled PCIe 1 = Disabled
Port Type	End Point and Root Port	End Point
Number of Lanes	x1, x2, and x4	x1
Reference Clock Frequency (MHz)	100, 125, and 156.25	100
CDR Reference Clock Source	Dedicated and Fabric	Dedicated
Number of CDR Reference Clocks	1 and 2	1
Optional Interfaces		
APB Slave Interface[1]	Enabled and Disabled	Disabled
DRI Slave Interface[2]	Enabled and Disabled	Disabled

1. APB interface is used to configure PCIe control registers.
2. DRI interface is used to configure lane related registers.

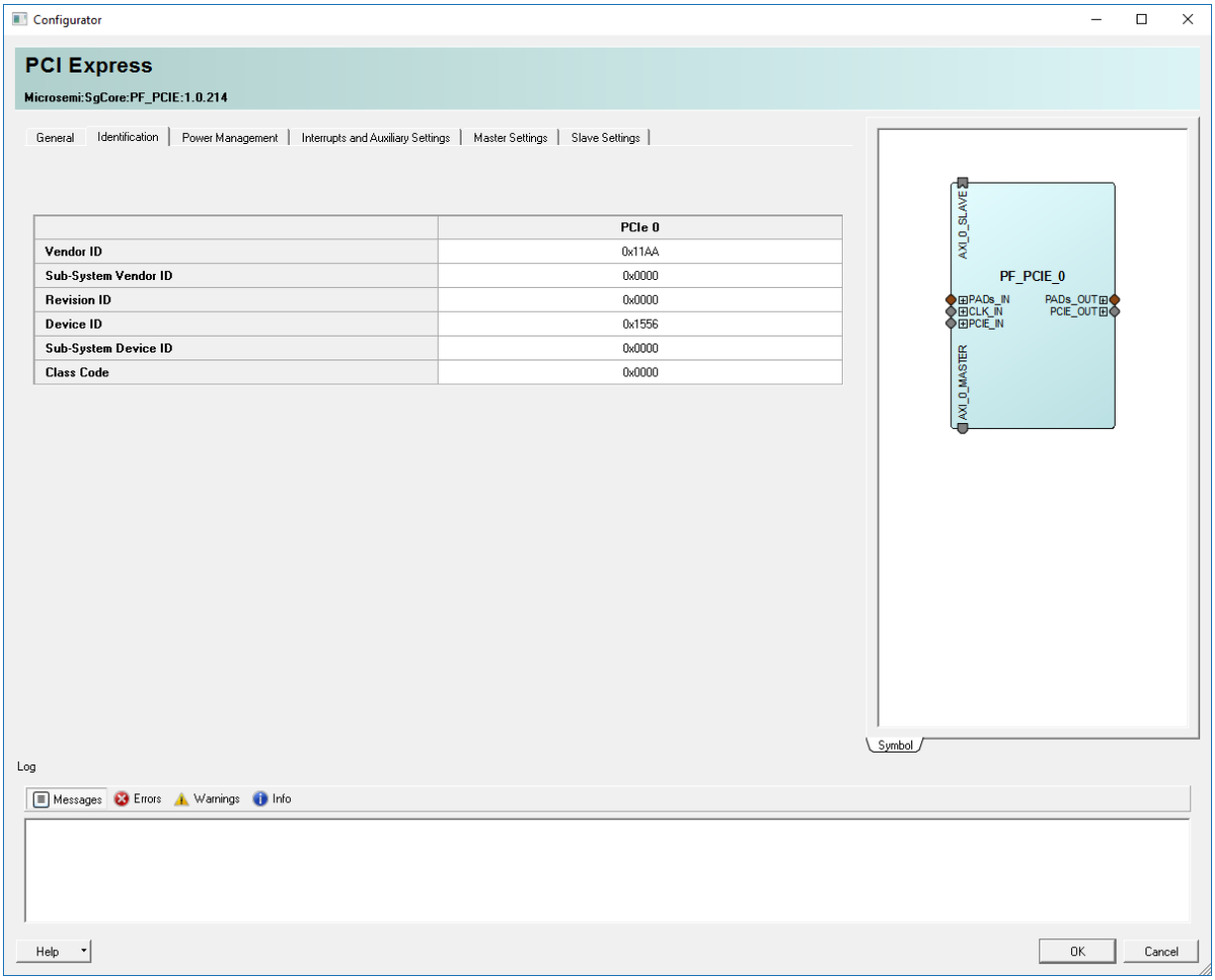
Number of Lanes: PCIe requires selection of the initial lane width. Wider lane-width cores are capable of training down to smaller lane widths if attached to smaller lane-width devices. The configurations, x2 and x4 support automatic lane reversal, allowing the PCIe link to permit board interconnections with reversed lane numbers, and the PCIESS continues to link train successfully and operate normally.

Reference Clock Frequency: PCIESS requires a 100 MHz or 125 MHz or 156.25 MHz clock input. The specified clock frequency must match with the TXPLL clock frequency.

Optional Interfaces (APB Slave/DRI Slave): Enabling these options, exposes the particular bus on the PCIESS component for connecting to the FPGA fabric of the APB and DRI.

The following figure shows the options available in the Identification tab.

Figure 18 • **PCIe Identification Settings**



The following table lists the options available in the Identification tab.

Table 5 • **PCIe Identification Settings**

PCIE Identification Settings (PCIe 0 and PCIe 1)	Options	Default
Vendor ID	User Input	0x11AA
Sub-System Vendor ID	User Programmable	0x0000
Revision ID	User Input	0x0000
Device ID	User Input	0x1556
Sub-System Device ID	User Input	0x0000
Class Code	User Input	0x0000

Vendor ID: It identifies the manufacturer of the device or application. The default value (0x11AA) is the vendor ID of Microsemi and is registered with PCI-Sig. Customized vendor identification IDs can also be used.

Sub-System Vendor ID: This ID further qualifies the manufacturer of the device or application. The default value is 0x0000 matching the vendor ID. Customized vendor identification IDs can also be used.

Note: 0x0000 is not recommended for vendor IDs or sub-system vendor ID.

Revision ID: This indicates the revision of the device or application; an extension of the device ID. The default value is 0x0000. Customized revision IDs can also be used.

Device ID: A unique identifier for the application. This can be any value based on the input.

Sub-System Device ID: This is similar to sub-system vendor ID and further qualifies the device application.

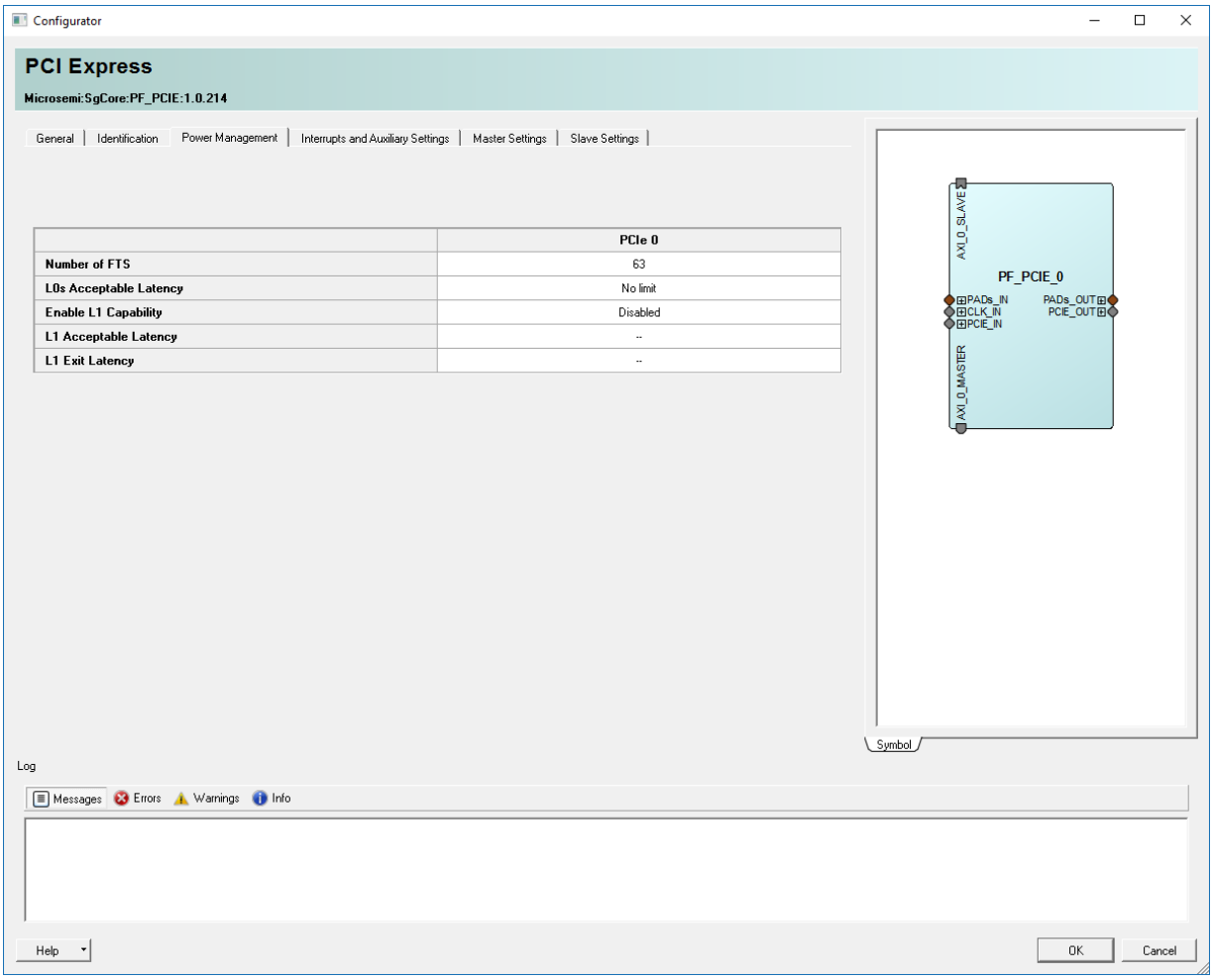
Class Code: The class code identifies the general function of a device, and is divided into three byte-size fields:

- Base Class: Broadly identifies the type of the function performed by the device.
- Sub-Class: More specifically identifies the device function.
- Interface: Defines a specific register-level programming interface.

Class code encoding details can be found at *www.pcisig.com*.

The following figure shows the options available in the Power Management tab. By selecting the power management option, allows loading settings to the PCIe config space headers.

Figure 19 • **PCIe Power Management Settings**



The following table lists the options available in the Power Management tab.

Table 6 • **PCIe Power Management Settings**

PCIE Power Management Settings (PCIe 0 and PCIe 1)	Options	Default
Number of Fast Training Sequences (FTS)	User entered	63
L0 Standby (L0s) Acceptable Latency	No Limit, Maximum of 64 ns, Maximum of 128 ns, Maximum of 256 ns, Maximum of 512 ns, Maximum of 1 µs, Maximum of 2 µs, and Maximum of 4 µs	No Limit
Enable L1 Capability	Disabled and Enabled	Disabled
L1 Acceptable Latency	No Limit, Maximum of 1 µs, Maximum of 2 µs, Maximum of 4 µs, Maximum of 8 µs, Maximum of 16 µs, Maximum of 32 µs, and Maximum of 64 µs	No Limit
L1 Exit Latency	Less than 1 µs, 1µs less than 2 µs, 2µs less than 4 µs, 4µs less than 8 µs, 8µs less than 16 µs, 16µs less than 32 µs, 32µs to 64µs, and more than 64 µs	16 µs to less than 32 µs

The PCIe base specification defines two levels of active state power management (ASPM) that are designed to provide options for trading off increased power conservation with rapid recovery to the L0 state.

Number of fast training sequences (FTS): The specific number to be repeated is defined by the receiving device and broadcast during training sequences at the link up time. The more FTS transmitted, the easier it is to obtain a receiver lock on the transmitted signal. The user can specify an input value for the number of FTS required.

L0s Acceptable Latency: This state is required by all the PCIe devices and applies to a single direction on the link. The latency to return to L0 from L0s is specified to be very short. When entering L0s, the device moving into the power saving state sends an electrical idle ordered set (EIOS) to the receiving device, and then turn off the power to its transmitter. When returning from L0s to L0, the device must first generate a specific number of small ordered known as FTS. However, the purpose of L0s is to regain receiver lock and be able to receive traffic as quickly as possible, so the receiving device selects the lowest number of FTS that ensure clock recovery based on its specific design. This selection is used to choose a time interval to achieve L0s.

Enable L1 Compatibility: The L1 ASPM is optionally enabled and can be entered to achieve a greater degree of power conservation. In this state, both directions of the link are placed in the L1 state. Return to L0 requires both devices to go through the link recovery process which results in a greater latency to return to L0, so that the power state can typically be used when activity on the link is not expected for some significant time period.

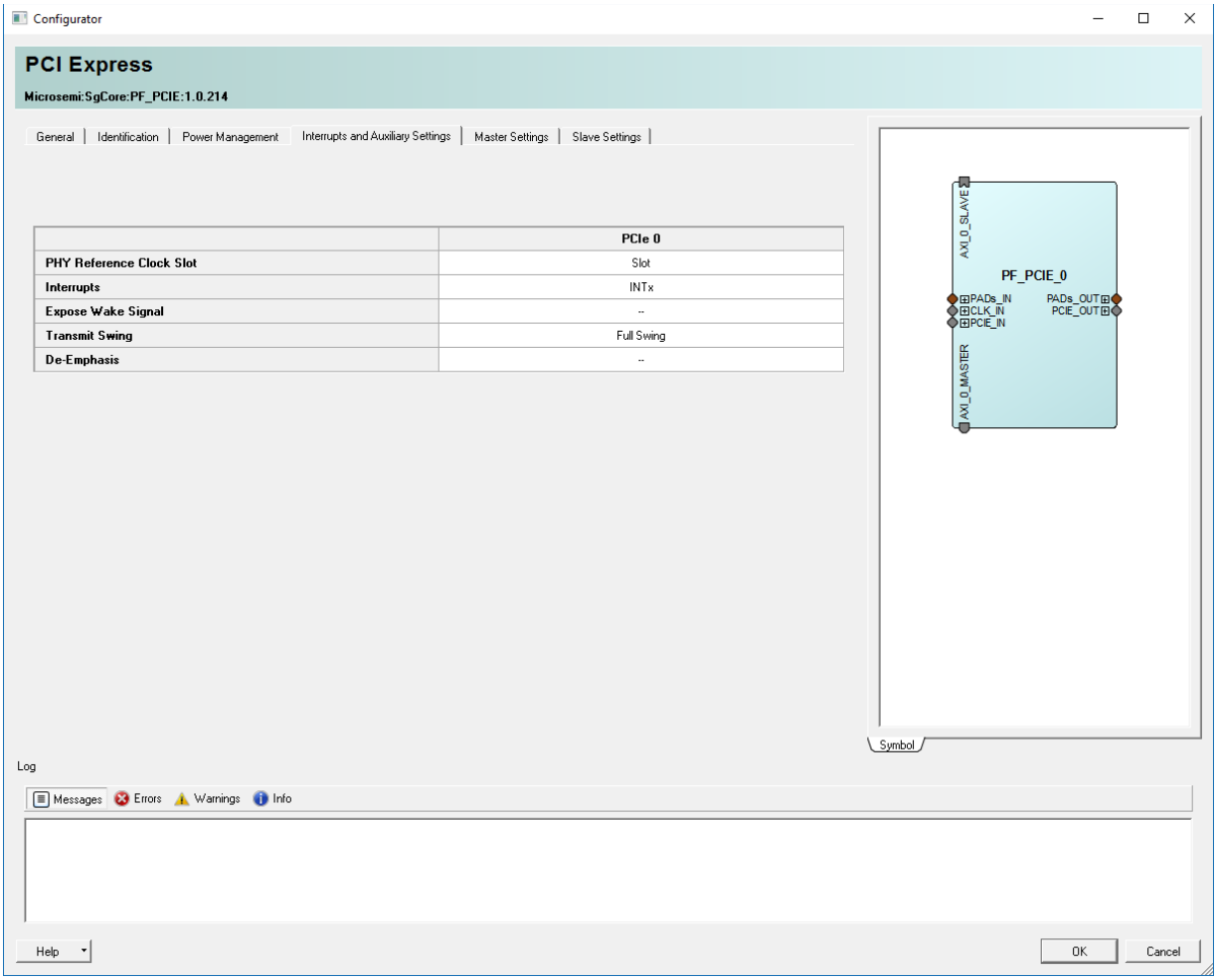
L1 Acceptable Latency: To enter the L1 state, the downstream device must first request permission from the upstream device for entering in to a deeper power conservation state. Upon acknowledgement, both devices turn off their transmitters and enter an electrical idle state. This settings gives the allowable time to wait to achieve L1.

L1 Exit Latency: Returning from L1 requires, that both devices must now go through the link recovery process. The link recovery process uses TS1 and TS2 standard ordered sets as opposed to the smaller FTSs used by L0s. This setting selects the time interval to exit from L1.

Note: When Enable L1 capability is enabled.

The following figure shows the options available in the Interrupts and Auxiliary Settings tab.

Figure 20 • **PCIe Interrupts and Auxiliary Settings**



The following table lists the options available in the Interrupt and Auxiliary Settings tab.

For description of the PCIe Identification Settings, see Table 11, page 41.

Table 7 • **PCIe Interrupts and Auxiliary Settings**

PCIE Interrupts and Auxiliary Settings (PCIe 0 and PCIe 1)	Options	Default
Physical layer reference clock slot	Slot and Independent	Slot
Interrupts	INTx, MSI 1, MSI 2, MSI 4, MSI 8, MSI 16, and MSI 32	INTx
Expose wake signals	Disabled and Enabled	Disabled
Transmit swing	Full Swing and Half Swing	Full swing
De-Emphasis	−3.5 dB and −6 dB	−3.5dB

PHY Reference Clock Slot: Select this option, if the PHY reference clock is either from a PCIe slot or is generated separately. Slot is a clock source shared in the PCIe system between the host and endpoint link. An Independent slot is used in a system that uses the independent clock sources on either side of the link. This setting changes the PCIe configuration space register, to advertise the used clocked topology to the system root. It makes no other functional changes to the endpoint.

Interrupts: The PCIe EP implementation supports 32 MSI interrupt and INTx interrupts. It cannot simultaneously support both the interrupts.

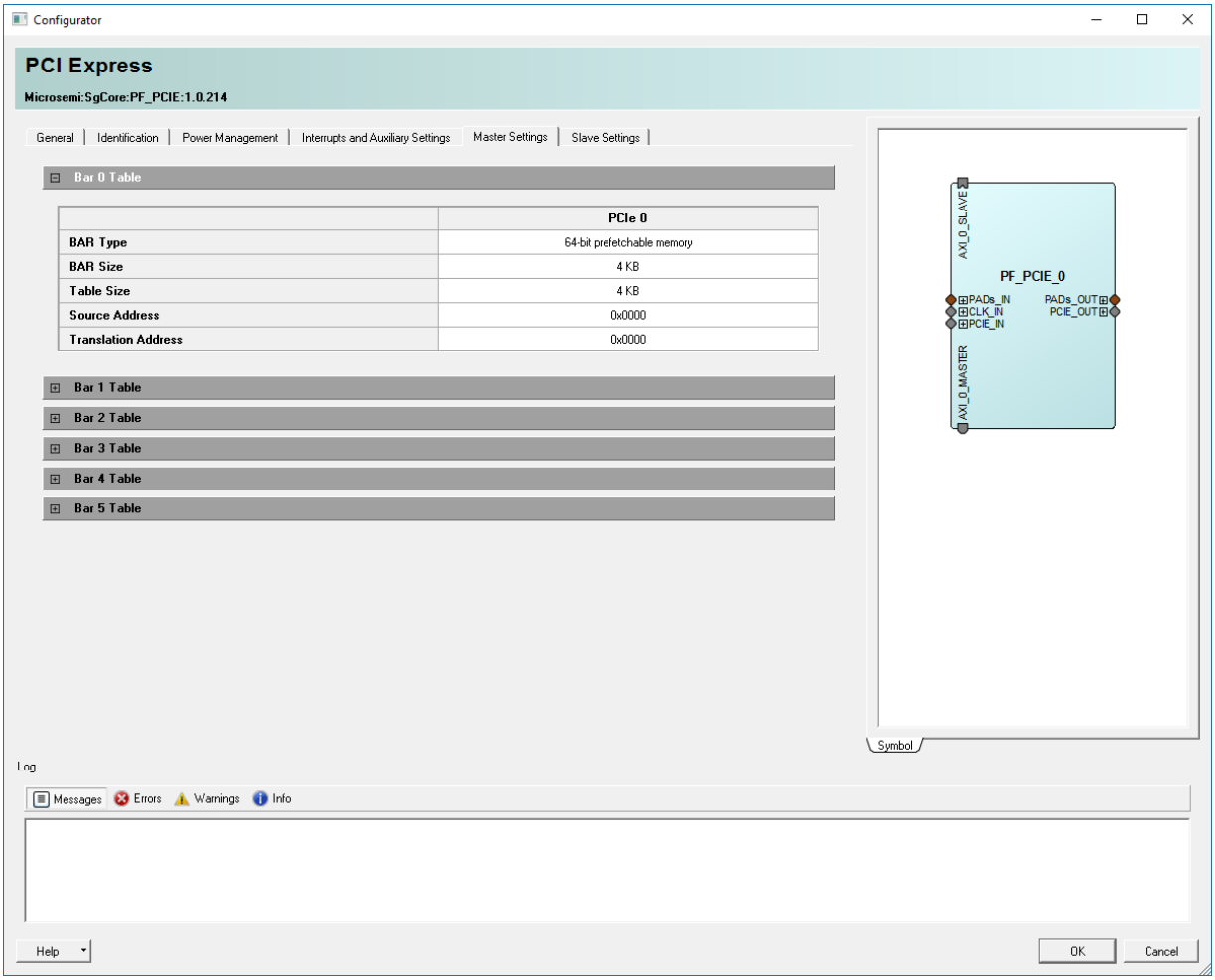
Expose Wake Signals: Enabling this option, exposes the WAKE# input signal on the PCIESS component allowing connection to the FPGA fabric.

Transmit Swing: This sets the transmit swing for PCIe GEN 2 speed.

De-Emphasis: This sets the de-emphasis (3.5 dB and 6.0 dB) for PCIe GEN 2 speed.

The following figure shows the options available in the Master Settings tab.

Figure 21 • **PCIe Master Settings**



The following table lists the options available in the Master Settings tab.

Table 8 • **PCIe Master Settings**

PCIE Master Settings (PCIe 0 and PCIe 1)	Options	Default
BAR Type	Disabled, 32-bit memory, 32-bit prefetchable memory, and 64-bit prefetchable memory	32-bit memory
BAR Size	4 KB, 8 KB, 16 KB, 32 KB, 64 KB, 128 KB, 256 KB, 512 KB, 1 MB, 2 MB, 4 MB, 8 MB, 16 MB, 32 MB, 64 MB, 128 MB, 256 MB, 512 MB,1 GB, and 2 GB.	4 KB
Table Size	4 KB, 8 KB, 16 KB, 32 KB, 64 KB, 128 KB, 256 KB, 512 KB, 1 MB, 2 MB, 4 MB, 8 MB, 16 MB, 32 MB, 64 MB, 128 MB, 256 MB, 512 MB, 1 GB, 2 GB, 4 GB, 8 GB, 16 GB, 32 GB, 64 GB, 128 GB, 256 GB, 512 GB, 1 TB, 2 TB, 4 TB, 8 TB, 16 TB, 32 TB, 64 TB, 128 TB, 256 TB, 512 TB, 1 PB, 2 PB, 4 PB, 8 PB, 16 PB, 32 PB, 64 PB, 128 PB, 256 PB, 512 PB,1 EB, 2 EB, 4 EB, 8 EB, and 16 EB	4 KB
Source Address[63:12]	User input, lower 12 bits are zero	0x0000

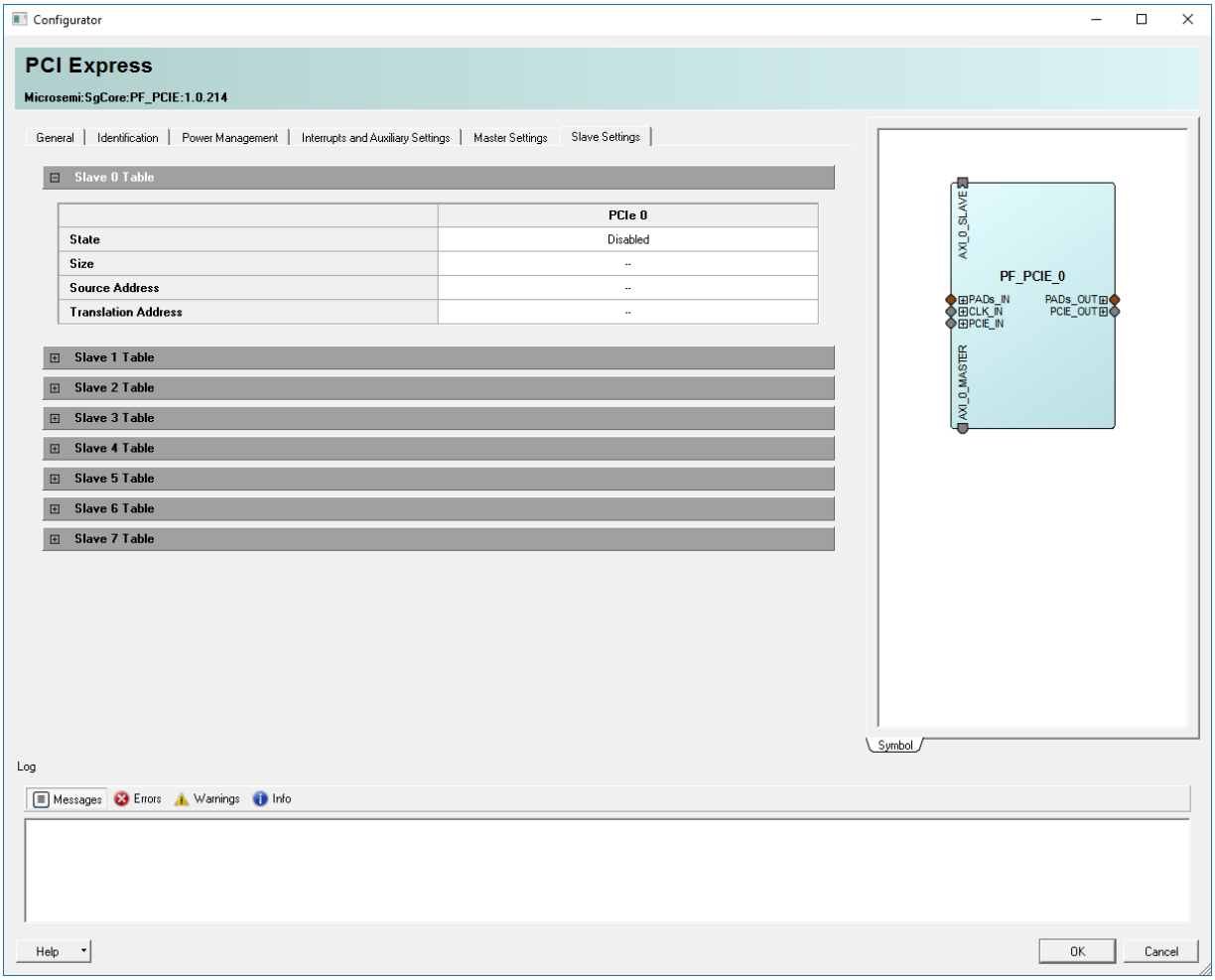
Table 8 • **PCIe Master Settings *(continued)***

PCIE Master Settings (PCIe 0 and PCIe 1)	Options	Default
Translation Address[63:12]	User Input, lower 12 bits are zero	0x0000

Note: At least one Master Bar must be enabled for PCIe 0 and PCIe 1 controllers.

The following figure shows the options available in the Slave Settings tab.

Figure 22 • **PCIe Slave Settings**



The following table lists the options available in the Slave Settings tab.

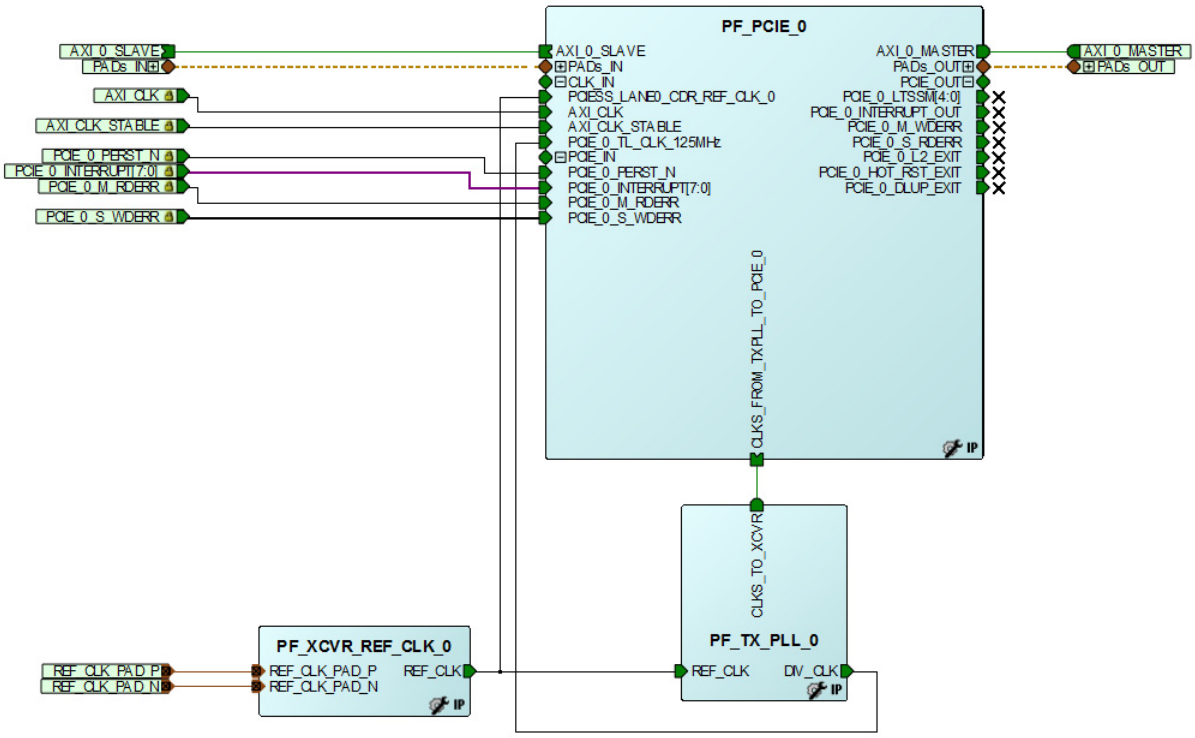
Table 9 • **PCIe Slave Settings**

PCIE Slave Settings (PCIe 0 and PCIe 1)	Options	Default
State	Disabled Enabled	Disabled
Size	4 KB, 8 KB, 16 KB, 32 KB, 64 KB, 128 KB, 256 KB, 512 KB, 1 MB, 2 MB, 4 MB, 8 MB, 16 MB, 32 MB, 64 MB,128 MB, 256 MB, 512 MB, 1 GB, 2 GB, 4 GB, 8 GB, 16 GB, 32 GB, 64 GB, 128 GB, 256 GB, 512 GB, 1 TB, 2 TB, 4 TB, 8 TB, 16 TB, 32 TB, 64 TB, 128 TB, 256 TB, 512 TB, 1 PB, 2 PB, 4 PB, 8 PB, 16 PB, 32 PB, 64 PB, 128 PB, 256 PB, 512 PB, 1 EB, 2 EB, 4 EB, 8 EB, and 16 EB	4 KB
Source Address[63:12]	User input, lower 12 bits are zero	0x0000
Translation Address[63:12]	User Input, lower 12 bits are zero	0x0000

Note: When State is enabled, the size, source address, and translation address settings are available

3. After making all selections in the PCIe configurator, complete the generation by clicking OK.
4. The next step is to create the XCVR_REFCLK and TX_PLL modules to be instantiated and connected to the PCIe block. Typically, the REF_CLK output of the PF_XCVR_REF_CLK is connected to the respective inputs of the PF_PCIE as well as the input REF_CLK of the PF_TX_PLL. For information on XCVR block generation, see *UG0677: PolarFire FPGA Transceiver User Guide*.

Figure 23 • **Completed PCIe Interface Example**



The following table lists the key connections of the SmartDesign PCIe example.

Table 10 • **Key Connections of SmartDesign**

Source	Destination
PF_XCVR_REF_CLK_0:REF_CLK	PF_TX_PLL_0:REFCLK
	PF_PCIE_0:PCIESS_LANE0_CDR_REF_CLK_0
PF_TX_PLL_0:CLKS_TO_XCVR	PF_PCIE_0: CLKS_FROM_TXPLL_TO_PCIE_0
PF_TX_PLL_0:DIV_CLK	PF_PCIE_0:PCIE_0_TL_CLK_125MHz

4.3 Design Constraints

No physical (PDC) constraints are required for PCIe. Constraints are required for PF_TXPLL and PF_XCVR_REF_CLK. The Libero software automatically places the PCIESS blocks. The PCIESS overlays the related transceiver quad 0 lanes. Therefore, when only PCIe 0 is used, it can be used as a x1 or x2 link. When both PCIe 0 and PCIe 1 are used through the DRI, they can be used as two x1 links, two x2 links, or one x4 link. One x4 link located in PCIE1 PCIESS and PCIE0 PCIESS will be unused. For configurations, see Figure 15, page 25.

Timing constraints are automatically generated by Libero SoC PolarFire. Timing constraints for the designs are required to meet performance goals of the PCIESS. Specific timing constraints are generated into a SDC-based constraint file based on the related timing of the PCIESS. Designers can adjust the timing constraints either directly within the SDC file or by using the timing constraints editor.

4.4 PCIESS Configuration Settings

The PCIe settings can be reconfigured through 32 bit-wide configuration space registers, which include:

- Information registers, which provide device, system, and bridge identification information.
- Bridge configuration registers, which enable configuration of the bridge functionality. These include:
  - Read-only registers that report control and status registers to the AXI4 bus
  - Bridge settings that must be configured at power-up, such as local interrupt mapping.
- Control/status registers, which can be used by the AXI4 bus to control bridge behavior during an operation.
- Power management registers, which configure the power management capabilities of the bridge.
- Address mapping registers, which provide address mapping for AXI4 master and slave windows used for address translation.
- Root port and endpoint interrupt registers.
- PCIe control and status registers, which enable the local processor to check the PCIe interface status. These read-only registers enable the local processor to detect the initialization of the bridge's PCIe interface and monitor PCI link events.

Some registers are hardwired to a fixed value within the embedded PCIESS block. For more information, see Configuration Registers, page 41.

4.5 PCIe Simulation

The PCIe is simulated using the bus functional model (BFM) for the PCIESS. In simulation mode, data transfer does not go off-chip. The simulation mode is available only with the PCIESS block. In the PCIe BFM simulation mode, the user can transmit/receive data from/to the fabric using the AXI4 of the PCIESS. The BFM simulation mode is selected from the Libero PCIESS configurator GUI. Libero generates the required files for BFM simulation.

The PCIe simulation mode uses the BFM commands to emulate the data that is transferred through the PCIESS block across the AXI4 bus interface to the fabric. The physical layer of the PCIe protocol is not implemented in this simulation mode and does not witness data transfer on the serial PMA interface of the PCIESS block. This mode is intended to validate the fabric interfaces to the PCIESS block, and the physical interface of the XCVR PMA block remains inactive.

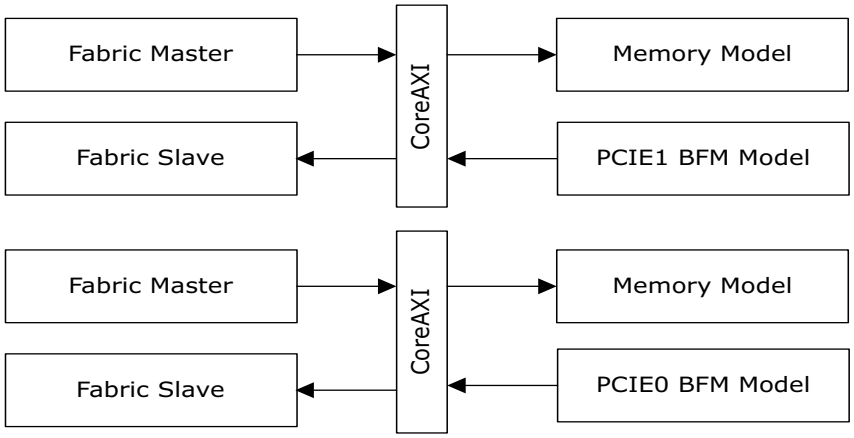
The AXI4 bus master in the PCIe BFM simulation mode enables emulating 64-bit AXI master transactions. Libero SoC PolarFire generates user-customizable BFM files that instruct the model to start transactions to the fabric. The BFM allows the user to use a text file to issue the transactions from the PCIe AXI master interface to the fabric, to exercise the design. The user must include BFM instructions in the `<project>/simulation/PCIE_<0:1>_user.bfm` file. The BFM model interprets these instructions and initiates AXI transactions in sequence. The PCIE_init.bfm model is not user-editable.

The AXI bus slave available in the BFM_PCIe simulation mode provides a 64-bit slave interface for fabric communication. The user can interact with the slave by initiating write and read bus transactions using the appropriate bus master. The slave acts as a memory model, so whatever is written to the slave can be read back from the same address.

The BFM commands used in the PCIESS BFM files are similar to the BFM commands used by the bus masters.

The following figure shows the PCIESS BFM structure.

Figure 24 • **PCIESS BFM Structure**



Note: There are additional BFM commands that are only used for the PCIe AXI BFM simulation, to emulate 64-bit AXI transactions:

The command, `write64 w <base_address> <base_address_offset> <32-bit MSB> <32-bit LSB>`, makes the bus master start a 64-bit write transaction on the external bus for a slave with address given by the <base_address> and <base_address_offset>, using the data generated by <32-bit MSB> and <32-bit LSB >.

For example: `write64 w 0x00000000 0x0 0xA0A1A2A3 0xB0B1B2B3;`

The command, `readcheck64 w <base_address> <base_address_offset> <32-bit MSB> <32-bit LSB>`, makes the bus master start a 64-bit read transaction for the address given by the <base_address> and <base_address_offset>. It compares the 64-bit read data to the data.

Note: The BFM commands for other PCIESS features will be added in future releases of the software.

The PCIe BFM is the recommended method to simulate the PCIe designs. An alternative method is to use a third-party verification IP (VIP) model for PCIe. In this setup, the register-transfer level (RTL) model of the PCIESS is used, and the entire data path through the PCIESS is exercised. The alternative method is supported, but the user is responsible for the VIP model for the PCIe. This is a complex setup, but can be powerful when system-level integration or throughput, needs to be obtained.

When using VIP models, ensure the following:

1. Verification IP must be configured properly.
   - BFM type: indicate type of BFM (0 – Root port and 1 – End point).
   - Number of lanes: indicates number of connected lanes
   - I/O size: specifies the size of internal I/O space. The values range from 12 to 24.
   - MEM32_SIZE: specifies the size of internal 32-bit addressing memory space. The values range from 12 to 24.
   - MEM64_SIZE: specifies the size of internal 64-bit addressing memory space. The values range from 12 to 24.
   - PCLK: PIPE clock frequency depends on the signaling rate and PIPE interface width configuration.
2. The receiver pin for XCVR should not be in an unused state. The following example code snippet is used in the testbench to prevent the transmitter pin from going into an unused state.
   ```
   rxp[i] <=(tx_1b[i]===1'bX || tx_1b[i]===1'bZ) ? 1'b0 :   tx_1b[i];
   rxn[i] <=(tx_1b[i]===1'bX || tx_1b[i]===1'bZ) ? 1'b0 :  ~tx_1b[i];
   ```
3. The receiver pin for VIP model should not be in an unused state. The following example code snippet is used in the testbench to prevent the transmitter pin from going into an unused state.
   ```
   rx_1b[i] <=(txp[i]==txn[i] || txp[i]===1'bX) ? 1'bZ : txp[i];
   ```
   Where,
   i – Number of BFM lanes.
   txp and txn – Transmitter pins from XCVR
   rxp and rxn – Receiver pins from XCVR
   tx_1b – transmitter pin from VIP model
   rx_1b – receiver pin from VIP model

4.6 Register Initialization

When the device powers up, some registers contained in the PCIESS are automatically initialized using data stored in the non-volatile storage of the device. The values associated with these registers are set either using flash bits or based on fixed values. Some registers are also loaded at power-up through an initialization mechanism that is programmed into devices implementing PCIESS blocks. This initialization is autonomously generated by Libero and is transparent. For information on initialization, see *UG0725: PolarFire FPGA Device Power-Up and Resets User Guide*.

5 Configuration Registers

Several PCIESS configuration registers are programmed using the PCIESS Configurator in the Libero SoC software.

5.1 PCIe Identification Registers

These registers are used to set the six identification registers for the subsystem. When not configured, they default to Microsemi-assigned values.

Table 11 • **PCIE_PCI_IDS Register Map**

Bit Locations	Field Name	Comments
[15:0]	Vendor ID	0×11AA is the default vendor ID assigned to Microsemi by PCI-SIG. Contact Microsemi to allocate subsystems under the Microsemi vendor ID.
[31:16]	Device ID	The manufacturer's assigned part number by the vendor.
[39:32]	Revision ID	Revision number, if applicable.
[63:40]	Class code	PCIe device's generic function.
[79:64]	Subsystem vendor ID	Card manufacturer's ID.
[95:80]	Subsystem vendor ID	Assigned by the subsystem vendor.

Note: Default value is 96'h11001556_FF000001_11001556.

Address offset: 0x0098
Physical address
PCIE0:0x300 4098
PCIE1:0x300 8098

5.2 Legacy Power Management Settings

The following table lists the PCIe PCI legacy power management settings.

Table 12 • **PCIe PCI Legacy Power Management Settings**

Bit	Description
[15:0]	Reserved
[16]	Address translation service support
[17]	Page request interface support
[20:18]	Reserved
[21]	DSI
[24:22]	Auxiliary current
[25]	D1 support
[26]	D2 support
[31:27]	PME support

Note: Default value is 32'hFE000000.

Address offset: 0x004A
Physical address
PCIE0:0x300 40A4
PCIE1:0x300 80A4

5.3 Power Management Data Configuration Settings

Power management data registers provide the scaling factor and state-dependent data related to the power state associated with the Data_Select field of the power management control and status register.

- PM_CONF
- Default value: 96'h00000000_00000000_00000000

5.4 Interrupts and MSI Settings

The PolarFire PCIESS supports 32 MSI or INTx interrupts. Only one interrupt model (either MSI or INTx) can be used at a time and is specified using the Libero PCIESS Configurator.

For MSI, up to 32 vectors are available for selection in the PCIESS Configurator. When using MSI, the first eight interrupts can be sent using the PCIE_INTERRUPTS[7:0] port of the transceiver: [0] for MSI [negotiated interrupts-8], [1] for MSI [negotiated interrupt-7], and so on. If more than eight interrupts are needed, they must be sent as a memory write transaction to a specific address in the AXI4 slave set by the root port during interrupt negotiation.

MSI-X is also available in the PolarFire PCIESS. The MSI-X table settings can be configured using the PCIE_PCI_IRQ[95:16] port in the bridge configuration space.

5.4.1 PCIE_PCI_IRQ[15:0]

The following table lists the PCIe PCI legacy power management settings.

Table 13 • **PCIe PCI Legacy Power Management Settings**

Bit	Description
[2:0]	Interrupt port: 000: MSI 1 001: MSI 2 010: MSI 4 011: MSI 8 100: MSi 16 101: MSI 32
[3]	Reserved
[6:4]	Number of MSI messages (0001 to 10132)
[7]	MSI per-vector masking support
[15:8]	Reserved

Note: Default value is 16'h0054.

Address offset: 0x00A8
Physical address
PCIE0: 0x300 40A8
PCIE1: 0x300 80A8

5.4.2 MSI-X Settings

5.4.2.1 PCIE_PCI_IRQ[95:16]

The following table lists the PCIe MSI-X settings.

Table 14 • **PCIe MSI-X Settings**

Bit	Description
[26:16]	Table size
[30:27]	Reserved
[31]	Implement MSI-X capability
[34:32]	Table BIR
[63:35]	Table offset
[66:64]	PBA BIR
[95:67]	PBA offset

Note: Default value is 80'h00000000_00000000_8000.

5.4.2.2 PCIE_VC_CRED

The following table lists the available credits settings.

Table 15 • **Available Credits Settings**

Bit	Description
[7:0]	Posted header credits
[19:8]	Posted data credits
[27:20]	Non-posted header credits
[35:28]	Non-posted data credits
[43:36]	Completion header credits
[55:44]	Completion data credits
[63:56]	Reserved

Note: Default value is 64'h0000000_1010_0B818.

Address offset: 0x0090
Physical address
PCIE0: 0x300 4090
PCIE1: 0x300 8090

5.4.2.3 PCIE_PEX_DEV

The following table lists the device capabilities settings.

Table 16 • **Device Capabilities Settings**

Bit	Description
[2:0]	Maximum payload size
[4:3]	Reserved for phantom functions support
[5]	Reserved
[8:6]	Endpoint L0s acceptable latency

Table 16 • **Device Capabilities Settings** ***(continued)***

Bit	Description
[11:9]	Endpoint L1 acceptable latency
[27:12]	Reserved
[28]	Function-level reset capability
[31:29]	Reserved

Note: Do not set the PCIe maximum payload size to a value greater than the bridge maximum payload size.

Note: Default value is 32'h00000001.

Address offset: 0x00C0
Physical address
PCIE0: 0x300 40C0
PCIE1: 0x300 80C0

5.4.2.4 PCIE_PEX_DEV2

The following table lists the device2 capabilities settings.

Table 17 • **Device2 Capabilities Settings**

Bit	Description
[3:0]	Completion timeout ranges
[4]	Completion timeout disable supported
[10:5]	Reserved
[11]	Latency tolerance reporting mechanism support
[31:12]	Reserved

Note: Default value is 32'h0000081F.

Address offset: 0x00C4
Physical address
PCIE0: 0x300 40C4
PCIE1: 0x300 80C4

5.4.2.5 PCIE_PEX_LINK

The following table lists the link capabilities settings.

Table 18 • **Link Capabilities Settings**

Bit	Description
[3:0]	SRIS lower SKP OS generation-supported speeds vector bits 3:0.
[7:4]	SRIS lower SKP OS reception-supported speeds vector bits 3:0.
[9:8]	Reserved.
[10]	ASPM L0s support.
[11]	ASPM L1 support.
[14:12]	L0s exit latency.
[17:15]	L1 exit latency.
[18]	Reserved for clock power management support.
[19]	Reserved for surprise down error reporting capable support.

Table 18 • **Link Capabilities Settings *(continued)***

Bit	Description
[20]	Reserved for DLL active reporting capable support.
[23:21]	Reserved.
[31:24]	Port number.

Note: Default value is 32'h01000C00.

Address offset: 0x00C8
Physical address
PCIE0: 0x300 40C8
PCIE1: 0x300 80C8

5.4.2.6 PCIE_PEX_SLOT

The following table lists the slot capabilities settings.

Table 19 • **Slot Capabilities Settings**

Bit	Description
[0]	Attention button present
[1]	Power controller present
[2]	MRL sensor present
[3]	Attention indicator present
[4]	Power indicator present
[5]	Hot-plug surprise
[6]	Hot-plug capable
[14:7]	Slot power limit value
[16:15]	Slot power limit scale
[17]	Electromechanical interlock present
[18]	No command complete support
[31:19]	Physical slot number

Note: Default value is 32'h0000007F.

Address offset: 0x00CC
Physical address
PCIE0: 0x300 40CC
PCIE1: 0x300 80CC

5.4.2.7 PCIE_PEX_ROOT_VC[15:0]

The following table lists the root capabilities settings.

Table 20 • **Root Capabilities Settings**

Bit	Description
[0]	CRS software visibility support
[15:1]	Reserved for root capabilities support

Note: Default value is 16'h0001.

Address offset: 0x00D0
Physical address
PCIE0: 0x300 40D0
PCIE1: 0x300 80D0

5.4.2.8 PCIE_PEX_SPC

The following table lists the specific capabilities settings.

Table 21 • **Specific Capabilities Settings**

Bit	Description
[11:0]	Reserved.
[12]	Slot register implemented.
[13]	Slot clock configuration: – 0: independent. Select this option if used in a system with independent clock sources on either side of the link. This setting changes the PCIe configuration space register to advertise to the system root whose clocking topology is used. It makes no other functional changes to the endpoint. – 1: refclk. Select this option if the PCIe reference clock is coming from a PCIe slot or is generated separately. A slot refers to a clock source that is shared in the PCIe system between both ends of the link.
[14]	Link selectable de-emphasis.
[15]	Root port RCB.
[20:16]	Device number for root port.
[30:21]	Reserved.
[31]	AER implemented.

Note: Default value is 32'h80005000.

Address offset: 0x00D4
Physical address
PCIE0: 0x300 40D4
PCIE1: 0x300 80D4

5.4.2.9 PCIE_PEX_SPC2

The following table lists the specific2 capabilities settings.

Table 22 • **Specific2 Capabilities Settings**

Bit	Description
[0]	Reserved
[1]	ECRC generation support
[2]	ECRC checking support
[7:3]	AER MSI message number
[12:8]	PCI Express MSI message number
[17:13]	ASPM L0s entry delay (in steps of 256 ns)
[22:18]	ASPM L1 entry delay (in steps of 256 ns)
[31: 23]	Reserved

Note: Default value is 32'h000C6006.

Address offset: 0x00D8
Physical address
PCIE0: 0x300 40D8
PCIE1: 0x300 80D8

5.4.2.10 PCIE_BAR_WIN

The following table lists the windows configuration settings.

Table 23 • **Windows Configuration Settings**

Bit	Description
[0]	I/O window implemented
[1]	I/O window 32-bit addressing support
[2]	Prefetchable memory window implemented
[3]	Prefetchable memory window 64-bit addressing support
[31:4]	Reserved

Note: Default value is 4'b1111.

Address offset: 0x00FC
Physical address
PCIE0: 0x300 40FC
PCIE1: 0x300 80FC

5.4.2.11 PCIE_PEX_NFTS

The following table lists the NFTS settings.

Table 24 • **NFTS Settings**

Bit	Description
[7:0]	Number of fast training sequences at 2.5 Gbps
[15:8]	Number of fast training sequences at 5.0 Gbps
[23:16]	Number of fast training sequences at 8.0 Gbps
[31:24]	Reserved

Note: Default value is 32'h00202020.

Address offset: 0x00DC
Physical address
PCIE0: 0x300 40DC
PCIE1: 0x300 80DC

5.4.2.12 PCIE_PEX_L1SS

The following table lists the L1 sub-states capabilities settings.

Table 25 • **L1 Sub-States Capabilities Settings**

Bit	Description
[0]	PCI-PM L1.2 supported
[1]	PCI-PM L1.1 supported
[2]	ASPM L1.2 supported
[3]	ASPM L1.1 supported
[4]	L1 PM substates supported (the L1 PM substates capability is only implemented if this bit is set)
[7:5]	T_POWEROFF value in units of 256 ns (000 = 256 ns, 110 = 7 × 256 ns)
[15:8]	Port common-mode restore time
[17:16]	Port T_POWER_ON scale
[18]	Reserved
[23:19]	Port T_POWER_ON value
[31:24]	Reserved

Note: Default value is 32'h00280A1D.

Address offset: 0x00E0
Physical address
PCIE0: 0x300 40E0
PCIE1: 0x300 80E0

5.4.2.13 IMASK_LOCAL

Setting a bit enables the associated interrupt source, and clearing a bit masks the interrupt source. The following table lists the bit definitions for the IMASK_LOCAL register.

Table 26 • **IMASK_LOCAL**

Bits	Description
0	Mask or enable the interrupt source described in ISTATUS_LOCAL register bit 0
1	Mask or enable the interrupt source described in ISTATUS_LOCAL register bit 1
7:2	Reserved
8	Mask or enable the interrupt source described in ISTATUS_LOCAL register bit 8
9	Mask or enable the interrupt source described in ISTATUS_LOCAL register bit 9
15:10	Reserved
16	Mask or enable the interrupt source for AXI Post Error event described in ISTATUS_LOCAL register bit 16.
17	Mask or enable the interrupt source for AXI Fetch Error event described in ISTATUS_LOCAL register bit 17.
18	Mask or enable the interrupt source for AXI Discard Error Event described in ISTATUS_LOCAL register bit 18.
19	Mask or enable the interrupt source for AXI Doorbell Event described in ISTATUS_LOCAL register bit 19.
20	Mask or enable the interrupt source for PCIe Post Error event described in ISTATUS_LOCAL register bit 20.
21	Mask or enable the interrupt source for PCIe Fetch Error event described in ISTATUS_LOCAL register bit 21.
22	Mask or enable the interrupt source for PCIe Discard Error Event described in ISTATUS_LOCAL register bit 22.
23	Mask or enable the interrupt source for PCIe Doorbell Event described in ISTATUS_LOCAL register bit 23.
24	Mask or enable the interrupt source of PCI interrupt line A (RP only, reserved for EP).
25	Mask or enable the interrupt source of PCI interrupt line B (RP only, reserved for EP).
26	Mask or enable the interrupt source of PCI interrupt line C (RP only, reserved for EP).
27	Mask or enable the interrupt source of PCI interrupt line D (RP only, reserved for EP).
28	Mask or enable the interrupt source of MSI received (RP only, reserved for EP).
29	Mask or enable the interrupt source of AER Event (RP only, reserved for EP).
30	Mask or enable the interrupt source of PM/LTR/Hotplug event for Rootport, Legacy power management state change for Endpoint.
31	Mask or enable the system error interrupt source.

Note: Default value is 32'h0.

Address offset: 0x0180
Physical address
PCIE0: 0x300 4180
PCIE1: 0x300 8180

5.4.2.14 ISTATUS_LOCAL

Local Processor Interrupt Status—the register bits are set when the corresponding interrupt sources are activated. Each source is independent. The local processor monitors and clear status bits—writing 1 clears a bit, writing 0 has no effect. The following table lists the bit definitions for the ISTATUS_LOCAL register.

Table 27 • **ISTATUS_LOCAL**

Bits	Description
0	Reports that a DMA transfer is ended, which corresponds to DMA engine 0.
1	Reports that a DMA transfer is ended, which corresponds to DMA engine 1.
7:2	Reserved.
8	Reports that an error occurred during a DMA transfer corresponding to DMA Engine 0.
9	Reports that an error occurred during a DMA transfer corresponding to DMA Engine 1.
15:10	Reserved.
16	Asserted to indicate that an error occurred on an AXI write request.
17	Asserted to indicate that an error occurred on an AXI read request.
18	Asserted to signal a completion timeout on an AXI read request.
19	Asserted when an AXI request has successfully targeted an Address Translation Table.
20	Asserted to indicate that an error occurred on a PCIe write request.
21	Asserted to indicate that an error occurred on a PCIe read request.
22	Asserted to signal a completion timeout on a PCIe read request.
23	Asserted when a PCIe request has successfully targeted an Address Translation Table.
24	Asserted when PCI interrupt line A is asserted (RP only, reserved for EP).
25	Asserted when PCI interrupt line B is asserted (RP only, reserved for EP).
26	Asserted when PCI interrupt line C is asserted (RP only, reserved for EP).
27	Asserted when PCI interrupt line D is asserted (RP only, reserved for EP).
28	MSI received (RP only, reserved for EP).
29	AER Event (RP only, reserved for EP).
30	PM/LTR/Hotplug event for Rootport, Legacy power management state change for Endpoint.
31	System error signaled (Rootport only, reserved for Endpoint).

Note: Default value is 32'h0.

Address offset: 0x0184
Physical address
PCIE0: 0x300 4184
PCIE1: 0x300 8184

5.4.2.15 IMASK_HOST

The following table lists the bit definition for IMASK_HOST register.

Table 28 • **IMASK_HOST**

Bits	Description
31:0	Setting a bit enables the associated interrupt source and clearing a bit masks the interrupt source. For more information about register bits, see ISTATUS_HOST, page 51

Note: Reserved for Rootport.

Note: Default value is 32'h0.

Address offset: 0x0188
Physical address
PCIE0: 0x300 4188
PCIE1: 0x300 8188

5.4.2.16 ISTATUS_HOST

Host Processor Interrupt Status—This is reserved for Rootport. The register bits are automatically set when the corresponding interrupt source is activated. Each source is independent. The host processor monitors and clears status bits—writing 1 clears a bit, writing 0 has no effect.

Table 29 • **ISTATUS_HOST**

Bits	Description
7:0	Interrupt Sources as described in the corresponding bits of ISTATUS_LOCAL register
9:8	
19:10	
23:20	
31:24	Reports interrupt requests from the local processor to the Host Processor.

Note: Default value is 32'h0.

Address offset: 0x018C
Physical address
PCIE0: 0x300 418C
PCIE1: 0x300 818C

5.4.2.17 IMSI_ADDR

The following table lists the bit definitions for the IMSI_ADDR register.

Table 30 • **IMSI_ADDR**

Bits	Description
[31:0]	Specifies the address on which incoming MSI messages are received when the PCIe is Rootport. The Rootport captures all memory write operations at this address and treats them as MSI. MSI Capture Address (reserved for Endpoint)

Note: Default value is 32'h0.

Address offset: 0x0190
Physical address
PCIE0: 0x300 4190
PCIE1: 0x300 8190

5.4.2.18 ISTATUS_MSI

The following table lists the bit definitions for the ISTATUS_MSI register.

Table 31 • **ISTATUS_MSI**

Bits	Description
[31:0]	Asserted when an MSI with message number 31-0 is received by the Rootport. The local processor must monitor and clear these bits—writing 1 clears a bit and 0 has no effect. MSI messages with numbers greater than 31 are ignored and discarded. MSI Message (reserved for Endpoint)

Note: Default value is 32'h0.

Address offset: 0x0194
Physical address
PCIE0: 0x300 4194
PCIE1: 0x300 8194

5.4.2.19 ICMD_PM

This register enables the local processor to activate and send events to the PCIe bus. The following table lists the bit definitions for the ICMD_PM register.

Table 32 • **ICMD_PM**

Bits	Description
0	Send PME (EP only, reserved for RP): Local processor writes 1 to send the PME bit command to generate a PME# event on the PCI Express link. Requests PCI host processor to restore bridge to a fully functional legacy power state.
[3:1]	Reserved
4	RP only, reserved for EP. The local processor sends a “turn off link” command in order to start L2 state entry negotiation. If the endpoint device is ready to enter this state, then both devices enter L2 state and this link is turned off. Deasserting this signal forces the Core to exit L2 state and wakes the link.
[7:5]	Reserved
8	The application uses this field to indicate that the PCI Express reference clock is safely removed (when applicable). If 0: the application does not allow the reference clock to be removed. If 1: the application allows the reference clock to be removed. This field must not be used, and is set to 0, when Clock Power Management and L1 PM substates with CLKREQ# are not implemented.
[31:9]	Reserved

Note: Default value is 32'h0.

Address offset: 0x0198
Physical address
PCIE0: 0x300 4198
PCIE1: 0x300 8198

5.4.2.20 SEC_ERROR_INT

It is an interrupt contributor registers for SEC errors. The following table lists the bit definitions for the SEC_ERROR_INT register.

Table 33 • **SEC_ERROR_INT**

Bits	Description
[3:0]	Indicates a SEC_RAM_ERR occurred in the appropriate Tx buffer 0x0 - Indicates that the SEC error is not encountered in Tx buffer 0x1 - Indicates that the SEC error is encountered in Tx buffer
[7:4]	Indicates a SEC_RAM_ERR occurred in appropriate Rx buffer 0x0 - Indicates that the SEC error is not encountered in Rx buffer 0x1 - Indicates that the SEC error is encountered in Rx buffer
[11:8]	Indicates a SEC_RAM_ERR occurred in appropriate Rx buffer
[15:12]	Indicates a SEC_RAM_ERR occurred in appropriate PCIe2AXI buffer 0x0 - Indicates that the SEC error is not encountered in AXI2PCIe buffer 0x1 - Indicates that the SEC error is encountered in AXI2PCIe buffer
[31:16]	Reserved

Note: Default value is 32'h0.

Address offset: 0x028
Physical address
PCIE0: 0x300 6028
PCIE1: 0x300 A028

5.4.2.21 DED_ERROR_INT

It is an interrupt contributor registers for DED errors. The following table lists the bit definitions for the DED_ERROR_INT register.

Table 34 • **DED_ERROR_INT**

Bits	Description
[3:0]	Indicates a DED_RAM_ERR occurred in Tx buffer 0x0 - Indicates that the DED error is not encountered in Tx buffer 0x1 - Indicates that the DED error is encountered in Tx buffer
[7:4]	Indicates a DED_RAM_ERR occurred in Rx buffer 0x0 - Indicates that the DED error is not encountered in Rx buffer 0x1 - Indicates that the DED error is encountered in Rx buffer
[11:8]	Indicates an SEC_RAM_ERR occurred in PCIe2AXI buffer
[15:12]	Indicates a DED_RAM_ERR occurred in AXI2PCIe buffer 0x0 - Indicates that the DED error is not encountered in AXI2PCIe buffer 0x1 - Indicates that the DED error is encountered in AXI2PCIe buffer
[31:16]	Reserved

Note: Default value is 32'h0.

Address offset: 0x030
Physical address
PCIE0: 0x300 6030
PCIE1: 0x300 A030

5.4.2.22 PCIE_EVENT_INT

It is an Interrupt contributor registers for L2 exit, Hot Reset exit, and DLUP exit.The following table lists the bit definitions for the PCIE_EVENT_INT register.

Table 35 • **PCIE_EVENT_INT**

Bits	Description
0	Indicates L2 exit event occurred 0x0 - Indicates that the L2 exit event is not encountered in PCIe 1x0 - Indicates that the L2 exit event is encountered in PCIe
1	Indicates Hot Reset exit event occurred 0x0 - Indicates that the Hot Reset exit event is not encountered in PCIe 0x1 - Indicates that the Hot Reset exit event is encountered in PCIe
2	Indicates DLUP exit event occurred 0x0 - Indicates that the DLUP exit event is not encountered in PCIe 0x1 - Indicates that the DLUP exit event is encountered in PCIe
[15:3]	Reserved
16	Indicates whether L2 exit from PCIe should contribute to PCIe interrupt or not 0x0 - Indicates that the l2 exit event contributes to pcie_interrupt 0x1 - Indicates that the l2 exit event contribution to pcie_interrupt is masked
17	Indicates whether Hot Reset exit from PCIe should contribute to PCIe interrupt or not 0x0 - Indicates that the Hot Reset exit event contributes to pcie_interrupt 0x1 - Indicates that the Hot Reset exit event contribution to pcie_interrupt is masked
18	Indicates whether DLUP exit from PCIe should contribute to PCIe interrupt or not 0x0 - Indicates that the DLUP exit event contributes to pcie_interrupt 0x1 - Indicates that the DLUP exit event contribution to pcie_interrupt is masked
[31:19]	Reserved

Note: Default value is 32'h0.

Address offset: 0x14C
Physical address
PCIE0: 0x300 614C
PCIE1: 0x300 A14C

5.4.3 DMA Engine Registers

The DMA engine registers enable up to 2 fully independent DMA engines to be configured.

Table 36 • **DMA Registers**

Byte Address	R/W	Description
0x0400 - 0x043F	RW/RO	DMA engine 0 configuration space
0x0440 - 0x047F	RW/RO	DMA engine 1 configuration space

The following registers describe the configuration space.

5.4.3.1 DMAx_SRC_PARAM

DMAx_SRC_PARAM is used to configure the source of the DMA transfer and the transfer parameters.

Table 37 • **DMAx_SRC_PARAM**

Bits	Description
[3:0]	Defines the source interface ID of the DMA transfer. When these register fields are not hardwired by core constants, they are read/write and their default value after reset is 4'h0. 4'b0: PCIe Interface 4'b4: AXI4-Master Interface DMA0 source is fixed to PCIe DMA1 source is fixed to AXI
[15:4]	Reserved
[27:16]	Provides the transfer parameters. Its content depends on the value of source interface ID.
[31:28]	Reserved

Note: Default value is 32'h0.

Address offset: DMA0 – 0x0400 and DMA1 – 0x0440
Physical address
PCIE0: 0x300 4400
PCIE1: 0x300 8400

5.4.3.2 DMAx_DESTPARAM

DMAx_DESTPARAM is used to configure the destination of the DMA transfer, the transfer priority inside the Bridge IP Core, and the transfer parameters.

Table 38 • **DMAx_DESTPARAM**

Bits	Description
[3:0]	Defines the destination interface ID of the DMA transfer: When these register fields are not hardwired by core constants, they are read/write and their default value after reset is 4'h0. For some DMA1, it is fixed to PCIe. 4'b0: PCIe Interface 4'b4: AXI4-Master Interface
[15:4]	Reserved
[27:16]	Provides the transfer parameters. Its content depends on the value of destination interface ID.
[31:28]	Reserved

Note: Default value is 32'h0.

Address offset: DMA0 – 0x0404 and DMA1 – 0x0444
Physical address
PCIE0: 0x300 4404
PCIE1: 0x300 8404

5.4.3.3 DMAx_SRCADDR_LDW

This register provides the source address of the DMA transfer (LSB).

Table 39 • **DMAx_SRCADDR_LDW**

Bits	Description
[31:0]	LSB 32 bits of the starting source address of DMA transfer.[1]

1. When SG mode is enabled for the source, DMA_SRCADDR provides the address of the first source descriptor.

Note: Default value is 32'h0.

Address offset: DMA0 – 0x0408 and DMA1 – 0x0448
Physical address
PCIE0: 0x300 4408
PCIE1: 0x300 8408

5.4.3.4 DMAx_SRCADDR_UDW

This register provides the source address of the DMA transfer (MSB).

Table 40 • **DMAx_SRCADDR_UDW**

Bits	Description
[31:0]	MSB 32 bits of the starting source address of DMA transfer. If the actual source and destination address are less than 64-bits, MSB bits are ignored.[1]

1. When SG mode is enabled for the source, DMA_SRCADDR provides the address of the first source descriptor.

Note: Default value is 32'h0.

Address offset: DMA0 – 0x040C and DMA1 – 0x044C

Physical address
PCIE0: 0x300 440C
PCIE1: 0x300 840C

5.4.3.5 DMAx_DESTADDR_LDW

This register provides the destination address of the DMA transfer (LSB).

Table 41 • **DMAx_DESTADDR_LDW**

Bits	Description
[31:0]	LSB 32 bits of the starting destination address of DMA transfer.[1]

1. When SG mode is enabled for the destination, DMA_DESTADDR provides the address of the first destination descriptor.

Note: Default value is 32'h0.

Address offset: DMA0 – 0x0410 and DMA1 – 0x0450
Physical address:
PCIE0: 0x300 4410
PCIE1: 0x300 8410

5.4.3.6 DMAx_DESTADDR_UDW

This register provides the destination address of the DMA Transfer (MSB).

Table 42 • **DMAx_DESTADDR_UDW**

Bits	Description
[31:0]	MSB 32 bits of the starting destination address of DMA transfer. If the actual destination address is less than 64-bits, MSB bits are ignored.[1] Note: When SG mode is enabled for the destination, DMA_DESTADDR provides the address of the first Destination descriptor.

1. When SG mode is enabled for the destination, DMA_DESTADDR provides the address of the first destination descriptor.

Note: Default value is 32'h0.

Address offset: DMA0 – 0x0414 and DMA1 – 0x0454
Physical address:
PCIE0: 0x300 4414
PCIE1: 0x300 8414

5.4.3.7 DMAx_LENGTH

This register provides the amount of data in bytes that should be transferred from the source to the destination.

Table 43 • **DMAx_LENGTH**

Bits	Description
[31:0]	Number of bytes in DMA transfer.

Note: Default value is 32'h0.

Address offset: DMA0 – 0x0418 and DMA1 – 0x0458
Physical address:
PCIE0: 0x300 4418
PCIE1: 0x300 8418

5.4.3.8 DMAx_CONTROL

This register provides the basic controls of the DMA.

Table 44 • **DMAx_CONTROL**

Bits	Description
0	When set to 1, it launches the DMA transfer. Appropriate registers should have previously been set. This bit is automatically cleared by the DMA Engine at the end of the DMA transfer.
1	When set to 1, DMA transfer is paused (to temporarily give more bandwidth to a transfer with higher priority).
2	Reserved
3	If 1, enable SG mode. If 0, disable SG mode.
4	Reserved
5	1 – stop if DMA _LENGTH is reached.
[7:6]	Reserved
8	If 1, an IRQ is issued on a DMA end.

Table 44 • **DMAx_CONTROL *(continued)***

Bits	Description
9	If 1, an IRQ is issued if an error occurs.
10	If 1, an IRQ is issued if the source of the transfer reports an EOP condition.
11	Reserved
12	If 1, an interrupt is issued to the Local Processor (on AXI domain).
13	If 1, an interrupt is issued to the Host Processor (on PCIe domain).
[22:14]	Reserved
23	Set to 1 by the application to indicate to the DMA Engine that a Descriptor has been updated.
[25:24]	Defines the Scatter-Gather type for the DMA (only relevant if bit 3 of control register is set). 2'b00: independent SG for both Source and Destination (DMA1). 2'b11: Source and Destination addresses are set according to Descriptor. When this register field is not hardwired by Core Constants (DMA0).
[28:26]	SG_ID: Defines on which interface the descriptors should be read. Fixed to '0'. 0 - PCIe Interface 1 - AXI4-Master Descriptor Interface
[31:29]	SG2_ID: If Scatter-Gather type is set to 2'b00, SG_ID is connected to the Source and SG2_ID is connected to the Destination. Otherwise, SG_ID is connected to the Source and/or Destination, and SG2_ID is irrelevant. 0 - PCIe Interface 1 - AXI4-Master Descriptor Interface

Note: Default value is 32'h3000000.

Address offset: DMA0 – 0x041C and DMA1 – 0x045C
Physical address:
PCIE0: 0x300 441C
PCIE1: 0x300 841C

5.4.3.9 DMAx_STATUS

This register provides DMA status.

Table 45 • **DMAx_STATUS**

Bits	Description
0	If set, DMA Complete with DMA_LENGTH reached.
1	If set, DMA Complete with an EOP condition reported by the source of the transfer.
2	If set, DMA Complete with EOC received on the last descriptor.
3	If set, DMA Complete with Error.
4	If set, DMA Complete with more than 4GBytes of data transferred.
5	Reserved.
6	If set, DMA successfully stopped by user.
7	If set, DMA incorrectly ended (buffer or descriptor not released).
8	If set, SRC error Completion Timeout.
9	SRC error 1 – CA received if on PCIe domain. 0 – EXOKAY received if on AXI domain.

Table 45 • **DMAx_STATUS**

Bits	Description
10	SRC error 1 – EP or ECRC received if on PCIe domain. 0 – SLVERR response received if on AXI domain.
11	SRC error 1 – UR received PCIe if on PCIe domain. 0 – DECERR response received if on AXI domain.
15:12	Reserved.
16	DEST_ERR: Completion Timeout.
17	DEST_ERR 1 – CA received if on PCIe domain. 0 – EXOKAY received if on AXI domain.
18	DEST_ERR 1 – EP or ECRC received if on PCIe domain. 0 – SLVERR response received if on AXI domain.
19	DEST_ERR 1 – UR received PCIe if on PCIe domain. 0 – DECERR response received if on AXI domain.
31:20	Reserved.

Note: Default value is 32'h0.

Address offset: DMA0 – 0x0420 and DMA1 – 0x0460
Physical address:
PCIE0: 0x300 4420
PCIE1: 0x300 8420

5.4.3.10 DMAx_PRC_LENGTH

This register provides the amount of data in bytes actually transferred from the source to the destination.

Table 46 • **DMAx_PRC_LENGTH**

Bits	Description
[31:0]	Number of bytes transferred.

Note: Default value is 32'h0.

Address offset: DMA0 – 0x0424 and DMA1 – 0x0464
Physical address:
PCIE0: 0x300 4424
PCIE1: 0x300 8424

5.4.3.11 DMAx_SHARE_ACCESS

This register provides DMA share access information.

Table 47 • **DMAx_SHARE_ACCESS**

Bits	Description
0	DMA Access Locked: When setting to 1, write access to the DMA engine registers is restricted to the physical or virtual function identified by bits [10:4]. Otherwise, all functions are allowed write access.
1	DMA Access Granted: returns 1 when read by the physical or virtual functions identified by bits [10:4] or when DMA access locked is set to 0. Otherwise, it returns 0.
3:2	Reserved.
10:4	Virtual function number (1 - 64). If 0, then a physical function is targeted. These bits are only available if virtual functions are implemented.
31:11	Reserved.

Note: Default value is 32'h0.

Address offset: DMA0 – 0x0428 and DMA1 – 0x0468
Physical address:
PCIE0: 0x300 4428
PCIE1: 0x300 8428

5.5 Address Translation Registers

Address Translator uses six address translation tables per BAR for both master and slave as listed in the following table.

Table 48 • **Address Translation Registers**

ATR	Byte Address
Master	
ATR0	0x0600 - 0x061F
ATR1	0x0620 - 0x062F
ATR2	0x0630 - 0x063F
ATR3	0x0640 - 0x064F
ATR4	0x0650 - 0x065F
ATR5	0x0660 - 0x066F
Slave	
ATR0	0x0800 - 0x081F
ATR1	0x0820 - 0x082F
ATR2	0x0830 - 0x083F
ATR3	0x0840 - 0x084F
ATR4	0x0850 - 0x085F
ATR5	0x0860 - 0x086F

Each Address Translation Table is 32 bytes in length as described in the following tables.

Table 49 • **SRCADDR_PARAM**

Byte Address	Bits	Description
0x00 - 0x03	0	ATR_IMPL: If 1, it indicates that the Translation Address Table is implemented.
	6:1	ATR_SIZE: Defines the Address Translation Space Size. This space size in bytes is equal to 2^(ATR_SIZE +1). Allowed values for this field are from 6'd11 (2^12 = 4 KBytes) to 6'd63 (2^64 = 16 KBytes) only.
	11:7	Reserved
	31:12	SRC_ADDR[31:12]

Table 50 • **SRC_ADDR**

Byte Address	Bits	Description
0x04 - 0x07	63:32	SRC_ADDR defines the starting address of the address translation space.

Table 51 • **TRSL_ADDR_LSB**

Byte Address	Bits	Description
0x08 - 0x0B	11:0	Reserved
	31:12	Defines the LSB [31:12] bits starting address of the address translation space.

Table 52 • **TRSL_ADDR_MSB**

Byte Address	Bits	Description
0x0C - 0x0F	63:32	Defines the MSB [63:32] bits starting address of the address translation space.

Table 53 • **TRSL_PARAM**

Byte Address	Bits	Description
0x10 - 0x13	3:0	TRSL_ID: Defines the translated ID of the request. The completer ID field of a read or write request to an address that targets the specified translation space is converted to a TRSL_ID value. The values allowed for this field are: 4'd0: PCIe Tx/Rx Interface 4'd1: PCIe Configuration Interface 4'd4: AXI4 Master
	15:4	Reserved
	27:16	TRSF_PARAM: Provides the translated parameter of the request. The transfer parameter field of a read or write request to an address that targets this address translation space is converted to the Transfer Parameters, page 62 value.
	31:28	Reserved

Table 54 • **TRSL_MASK**

Byte Address	Bit	Description
0x18 - 0x1F	[31:0]	Defines the translation table mask address. It is equal to (0 - Table Size) Where, the table size is equal to 2^(ATR_SIZE +1). For example, if the table size is fixed to 256 KBytes, TRSL_MASK is equal to (0 - 256 KBytes) = 64'hFFFFFFFFFFFC0000.

5.6 Transfer Parameters

The DMAx_SRC_PARAM register or TRSL_PARAM register bit [27:16] is used to transfer the content of the transfer parameters when using DMA or Address Translation. The content of the transfer parameters depends on the targeted interface. The following table describes the transfer parameters for each targeted interface.

Table 55 • **Transfer Parameters**

Targeted Interface	Target ID	Bits	Description
PCIe	4'h0	2:0	TLP Type: 3'b000: Memory 3'b001: Memory Locked 3'b011: Translation Request 3'b101: Memory Translated Request 3'b010: IO 3'b100: Message Other values are reserved.
		3	Translation Request No Write (NW) Flag. (must be 0b when TLP Type is different from 3'b011)
		4	TLP Attributes: No Snoop
		5	TLP Attributes: Relaxed Ordering
		6	TLP Attributes: ID-Based Ordering
		7	ECRC Forward
		10:8	Traffic Class
		11	Reserved
AXI4-Master	4'h4	3:0	ACACHE
		4	ALOCK
		7:5	APROT
		11:8	AQOS

6 Board Design Recommendations

This chapter discusses board-level implementation details of a PCIe design using PolarFire FPGAs. Optimal performance requires understanding the functionality of the device pins and properly addressing issues such as device interfacing, protocol specifications, and signal integrity.

For more information, see *UG0726: PolarFire FPGA Board Design User Guide.*

Various specifications from PCI-SIG apply depending on the form factor of the design. This chapter focuses on a subset of these specifications centered on chip-to-chip and add-in card implementations.

For more information, see the *PCI Express Base Specification, Revision 2.0* and *PCI Express Card Electromechanical Specification (CEM) Revision 2.0* from PCI-SIG.

6.1 AC-Coupling

PCIe electrical signals require a 75–200 nF AC-coupling capacitor between the transmitter and receiver. All transmitters are AC-coupled, either within the media or within the transmitting component itself. If located within the media, the AC-coupling capacitors are placed close to the transmitter. The AC-coupling capacitor is used in conjunction with internal termination for PCIe link detection.

6.2 Lane Reversal

The PCIESS supports lane reversal when required, allowing the PCIe physical I/O to be reversed with the block's logical lanes for a more flexible PCB layout. Lane reversal functionality is incorporated into the PCIESS to be layout-agnostic with respect to lane ordering and lane polarity.Using lane reversal can ease routing congestion on the PCB, leading to a cleaner interface between the PCIe host and the endpoint or root port.

6.3 Polarity Inversion

The PolarFire transceiver block with PCIESS supports differential polarity inversion. Receiver polarity is automatically detected by the PCIESS during link training, as defined in the PCIe specification. The differential data received by the transceiver RX are reversed if RXP and RXN differential traces are swapped on the PCB accidentally. The transceiver RX inversion allows within the PCIESS to offset the reversed polarity of a serial differential pair.

6.4 PCIe Power-Up

The PCIe specification provides timing requirements for power-up. The PCIe connector specification specifies that the fundamental reset (PERST_N) be de-asserted at a minimum of 100 ms from the point of power being stable. The PCIe PERST_N signal release time (known as PCIe timing parameter TPVPERL) of 100 ms is used for the PCIe card electro-mechanical specification for add-in cards.

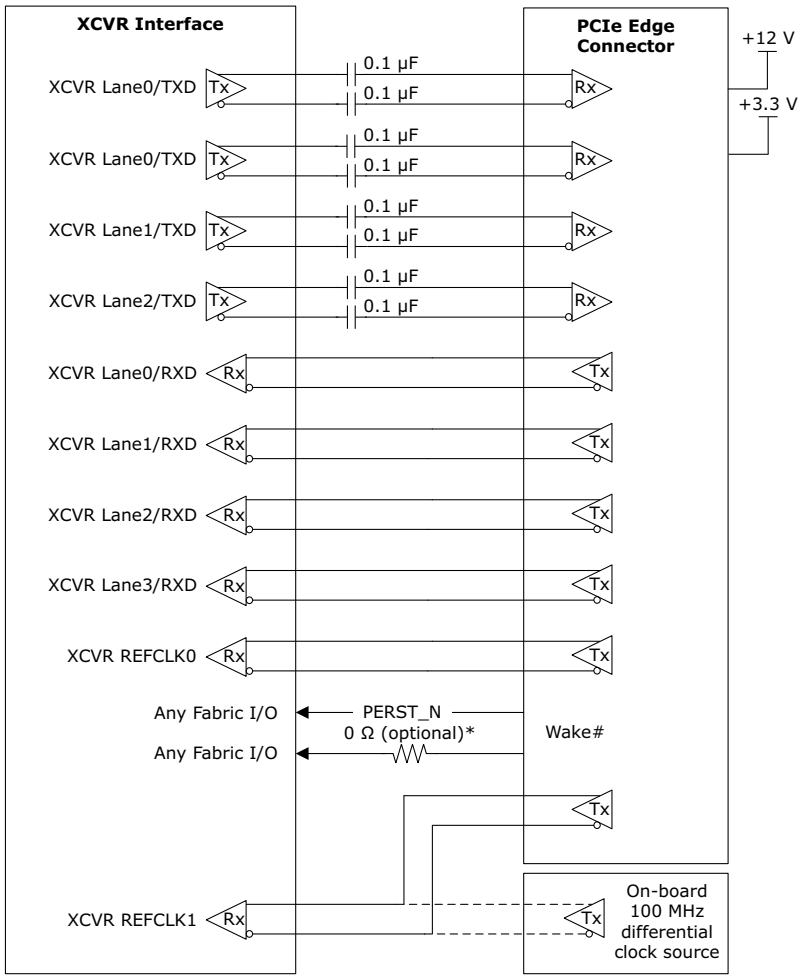
The semi-autonomous nature of the PCIESS in a PolarFire device allows the device to quickly move from power-up to link detect. The PolarFire transceiver initially terminates to 50 kΩ for hot-swap protection but quickly returns to 100 Ω termination so that link detection operates within the PCIe specifications. When the device is detected by the root, it proceeds to the polling state of the LTSSM. The link then cycles through the remaining LTSSM states. In cases where the root point and the endpoint power-up separately, the PERST_N signal must be used to handshake the link startups.

6.4.1 PCIe Edge Connector

PCIe is a point-to-point serial differential low-voltage interconnect supporting up to four channels. Each lane consists of two pairs of differential signals: transmit pair, receive pair, XCVR_x_TXy_P/N, and XCVR_x_RXy_P/N. Each signal has a 2.5 GHz embedded clock.

The following figure shows the connectivity between the PolarFire FPGA transceiver interface and the PCIe edge connector.

Figure 25 • **Connectivity Between XCVR Interface and PCIe Edge Connector**



Note: Between the fabric I/O and the host may require additional components to match 3.3 V levels.