PCIe x1 IP Core

IP Version: v3.0.0

User Guide

FPGA-IPUG-02091-2.2

December 2025

Disclaimers

Lattice makes no warranty, representation, or guarantee regarding the accuracy of information contained in this document or the suitability of its products for any particular purpose. All information herein is provided AS IS, with all faults, and all associated risk is the responsibility entirely of the Buyer. The information provided herein is for informational purposes only and may contain technical inaccuracies or omissions, and may be otherwise rendered inaccurate for many reasons, and Lattice assumes no obligation to update or otherwise correct or revise this information. Products sold by Lattice have been subject to limited testing and it is the Buyer's responsibility to independently determine the suitability of any products and to test and verify the same. LATTICE PRODUCTS AND SERVICES ARE NOT DESIGNED, MANUFACTURED, OR TESTED FOR USE IN LIFE OR SAFETY CRITICAL SYSTEMS, HAZARDOUS ENVIRONMENTS, OR ANY OTHER ENVIRONMENTS REQUIRING FAIL-SAFE PERFORMANCE, INCLUDING ANY APPLICATION IN WHICH THE FAILURE OF THE PRODUCT OR SERVICE COULD LEAD TO DEATH, PERSONAL INJURY, SEVERE PROPERTY DAMAGE OR ENVIRONMENTAL HARM (COLLECTIVELY, "HIGH-RISK USES"). FURTHER, BUYER MUST TAKE PRUDENT STEPS TO PROTECT AGAINST PRODUCT AND SERVICE FAILURES, INCLUDING PROVIDING APPROPRIATE REDUDANCIES, FAIL-SAFE FEATURES, AND/OR SHUT-DOWN MECHANISMS. LATTICE EXPRESSLY DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY OF FITNESS OF THE PRODUCTS OR SERVICES FOR HIGH-RISK USES. The information provided in this document is proprietary to Lattice Semiconductor, and Lattice reserves the right to make any changes to the information in this document or to any products at any time without notice.

Inclusive Language

This document was created consistent with Lattice Semiconductor’s inclusive language policy. In some cases, the language in underlying tools and other items may not yet have been updated. Please refer to Lattice’s inclusive language FAQ 6878 for a cross reference of terms. Note in some cases such as register names and state names it has been necessary to continue to utilize older terminology for compatibility.

Contents

Figures

Tables

Acronyms in This Document

A list of acronyms used in this document.

Acronym	Definition
AHB	Advanced High-Performance Bus
APB	Advanced Peripheral Bus
ASPM	Active State Power Management
AXI	Advanced Extensible Interface
AXI-MM	Advanced Extensible Interface – Memory Mapped
BAR	Base Address Register
CSR	Configuration and Status Register
DLLP	Data Link Layer Packet
DMA	Direct Memory Access
ECC	Error Correction Coding
EP	Endpoint
FIFO	First In First Out
LMMI	Lattice Memory Mapped Interface
LTSSM	Link Training and Status State Machine
MSI	Message Signaled Interrupt
RTL	Register Transfer Language
PCI	Peripheral Component Interconnect
PCIE	Peripheral Component Interconnect Express
PCS	Physical Coding Sublayer
PLL	Phase-Locked Loop
PM	Power Management
PMA	Physical Medium Attachment
RAM	Random Access Memory
RC	Root Complex
RP	Root Port
TLP	Transaction Layer Packet
UCFG	User Configuration Interface

1. Introduction

1.1. Overview of the IP

PCI Express® is a high performance, fully scalable, and well-defined standard for a wide variety of computing and communications platforms. As a packet-based serial technology, the PCI Express standard greatly reduces the number of required pins and simplifies board routing and manufacturing. PCI Express is a point-to-point technology, as opposed to the multi-drop bus in PCI. Each PCI Express device has the advantage of full duplex communication with its link partner to greatly increase overall system bandwidth. The basic data rate for a single lane is double that of the 32-bit/33 MHz PCI bus. A four-lane link has eight times the data rate in each direction of a conventional bus.

The Lattice PCIe x1 IP Core provides a flexible, high-performance, easy-to-use Transaction Layer Interface to the PCI Express Bus. The Lattice PCIe x1 IP Core implementation is a hardened IP with soft logic provided for interface conversion options. The hardened IP is an integration of PHY and Link Layer blocks.

The Lattice PCIe x1 IP Core is supported in the CrossLink™-NX, Certus™-NX, and MachXO5™-NX FPGA device families and is available in the Lattice Radiant™ software.

1.2. Quick Facts

Table 1.1. Summary of the PCIe x1 IP

IP Requirements	Supported Devices	CrossLink-NX, Certus-NX[1], MachXO5-NX (LFMXO5-35T and LFMXO5-65T only)
	IP Changes	Refer to the PCIe x1 IP Release Notes (FPGA-RN-02060).
Resource Utilization	Supported User Interface	APB, AXI-Stream, TLP, AXI
Design Tool Support	Lattice Implementation	IP Core v3.0.0 – Lattice Radiant Software 2025.2 or later
	Synthesis	Synopsys® Synplify Pro® for Lattice
	Simulation	• QuestaSim Lattice-Edition, QuestaSim Pro • Modelsim OEM and Modelsim Pro (supported in Radiant 2024.1 or earlier)

Notes:

1. CABGA484 package is only supported in LFD2NX-35 and LFD2NX-65 devices.
2. In some instances, the IP may be updated without changes to the user guide. This user guide may reflect an earlier IP version but remains fully compatible with the later IP version. Refer to the IP Release Notes for the latest updates.
3. Lattice Implementation indicates the IP version release coinciding with the software version release. Check the software for IP version compatibility with earlier or later software versions.

1.3. IP Support Summary

Table 1.2. PCIe x1 IP Support Readiness

Device Family	IP	Configuration	Radiant Timing Model	Hardware Validated
Certus-NX	PCIe x1	Gen1x1	Final	No
Certus-NX	PCIe x1	Gen2x1	Final	No
CrossLink-NX	PCIe x1	Gen1x1	Final	No
CrossLink-NX	PCIe x1	Gen2x1	Final	No
MachXO5-NX	PCIe x1	Gen1x1	Final	No
MachXO5-NX	PCIe x1	Gen2x1	Final	No

1.4. Features

The Hard IP PHY key features include:

- Transmitter
  - Configurable driver impedance, amplitude
  - Support for one lane
- Receiver
  - Configurable receiver impedance, Continuous Time Linear Equalizer (CTLE) gain, 1-Tap Decision Feedback
  - Baud rate Eye Monitoring capability to map eye density at receiver post equalization
  - Bit skip feature to allow adjusting of received byte clock alignment
- PCS
  - Rate negotiation support
  - Selectable parallel data widths such as 5, 10, and 16
  - 8b/10b encoding at 2.5 Gbps and 5.0 Gbps
  - Test support features such as near-end loopback and PLL bypass modes
  - Protocol-compatible features such as LOS, squelch, and power modes
  - L1-substates and special L1P2 support for PCIe 2.0

The Hard IP Link Layer key features include:

- PCI Express Base Specification Revision 3.0 compliant, including compliance with earlier PCI Express Specifications
  - Backward compatible with PCI Express 2.x, and 1.x
- x1 PCI Express Lane only
- 5.0 GT/s, and 2.5 GT/s line rate support
- Comprehensive application support – Endpoint
- Multi-Function support with 1-4 Physical Functions
- ECC RAM and Parity Data Path Protection
- Core Data Width
  - 32 bits for x1 lane
- Complete error-handling support
  - AER, ECRC generation/checking, recovery from Parity and ECC errors
  - Supports detection of numerous optional errors and embedded simulation error checks/assertions
  - Simulation and hardware error injection features enable error testing
- Flexible core options allow for design complexity/feature trade-offs:
  - Configurable Receive, Transmit, and Replay Buffer sizes
- Supports Polarity Inversion, Up/Down-configure, Autonomous Link Width/Speed changes
- Power Management
  - Supports L1, ASPM L0s, and ASPM L1
  - L1 PM Substates with CLKREQ
  - Power Budgeting
  - Dynamic Power Allocation
- Latency Tolerance Reporting
- Implements Type 0 Configuration Registers in Endpoint Mode
- Dual mode design supports EP or RP through the register changes
- The above features enable:
  - Decoding of received packets to provide key routing (BAR hits and Tag) information
  - Implementation of all aspects of the required PCIe Configuration Space
  - PCI Express Message TLPs to be consumed or left in a band
  - Interfaces have consistent timing and function over all modes of operation
  - A wealth of diagnostic information for superior system-level debug and link monitoring
- Implements all three PCI Express Layers (Transaction, Data Link, and Physical)

1.4.1. Soft IP

- Non-DMA
  - TLP Mode
  - AXI-Stream Data Interface
  - AXI-MM Data Interface (Bridge Mode)
  - AXI-Lite Data Interface (Bridge Mode)
- DMA
  - AXI-MM Data Interface
- LMMI Register Interfaces
- APB Register Interface[1]

Note:

1. Only supported in Non-DMA AXI-Steam Data Interface.

1.5. Licensing and Ordering Information

The PCIe x1 IP is available with the Lattice Radiant Subscription software. To purchase the Lattice Radiant Subscription license, contact Lattice Sales or go to the Lattice Online Store.

1.6. Hardware Support

Refer to the Example Design section for more information on the boards used.

1.7. Speed Grade Supported

The Lattice PCIe IP core supported speed grade is provided in this section. Different configurations may be supported using different speed grade due to fabric performance requirement.

- 9 – fastest speed grade

Table 1.3. Lattice PCIe IP Core Supported Speed Grade

PCIe Core Config	Device Family	Speed Grade
Gen2x1	CrossLink-NX	7/8/9
Gen2x1	Certus-NX	7/8/9
Gen2x1	MachXO5-NX	7/8/9

Note: For Speed Grade 7 and 8, the supported IP frequency is up to 100 MHz only.

1.8. Naming Conventions

1.8.1. Nomenclature

The nomenclature used in this document is based on Verilog HDL.

1.8.2. Signal Names

- *_n* are active low (asserted when value is logic 0)
- *_i* are input signals
- *_o* are output signals
- *[LINK]* index identifies which PCIe Link (0 or 1)

2. Functional Description

2.1. PCIe IP Architecture Overview

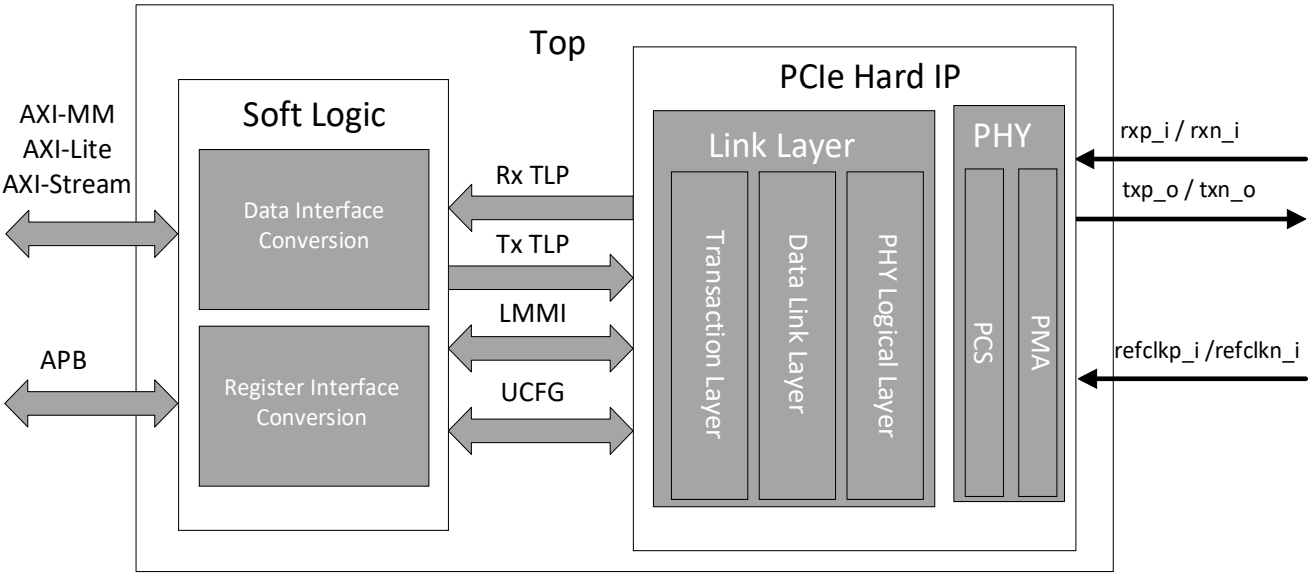

Figure 2.1. Lattice PCIe x1 IP Core Block Diagram

The Lattice PCIe x1 IP Core implements all three layers defined by the PCI Express Specification:

- Physical Layer
- Data Link Layer
- Transaction Layer

The soft logic is provided for optional interface conversion such as:

- AXI-Stream
- AXI-MM
- AXI-Lite
- APB for registers access

The Lattice PCIe x1 Hard IP has the following interfaces as shown in Figure 2.2. The details of each interface are discussed in the subsequent sections.

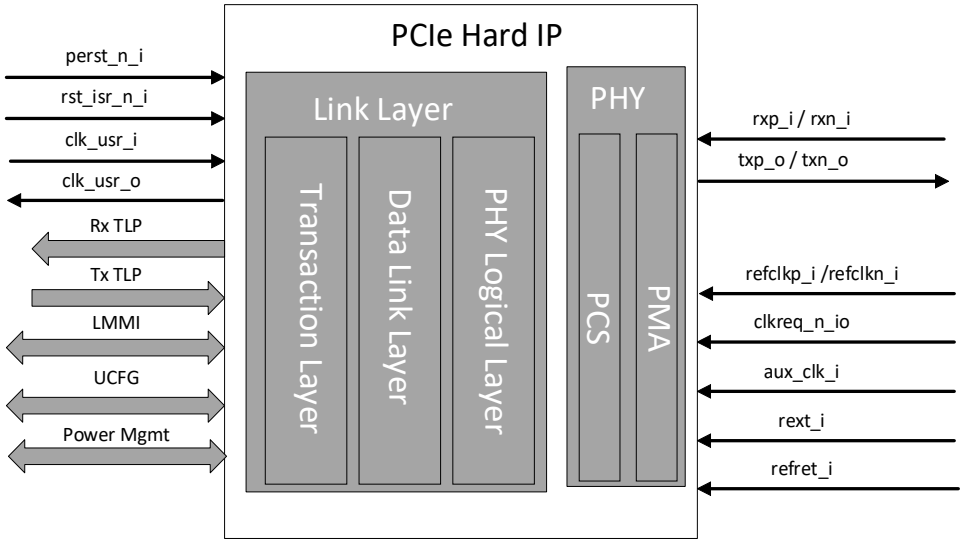

Figure 2.2. Lattice PCIe x1 Core Hard IP

- Clock and Reset Interface
  - The user domain interface can be clocked using the PHY PCLK output (sys_clk_i = clk_usr_o) or by the user generated clock using a PLL.
- Reset Interface
  - The fundamental reset (*perst_n_i*) resets the core (*PHY and Link Layer blocks*) except for the core configuration registers.
  - Another reset (*rst_usr_n_i*) is provided to reset only the Link Layer block.
- PHY Interface
  - High-Speed Serial Interface that supports a maximum rate of 5.0 GT/s
- TLP Receive Interface
  - Receive TLPs from the PCIe link partner
  - High bandwidth interface
- TLP Transmit Interface
  - Transmit TLPs to the PCIe link partner
  - High bandwidth interface
- Power Management Interface
  - Ports for implementing power management capabilities
- UCFG – User Configuration Space Register Interface
  - Enables access to the PCIe Configuration Space Registers
- LMMI – Configuration and Status Register (CSR) Interface
  - This interface is used to write to and read from the core configuration and status registers. This interface can also be used to read status registers such as PLL locked and LTSSM state and to turn off a capability register that is not configurable through the PCIe IP user interface.

2.2. Clocking

2.2.1. Clocking Overview

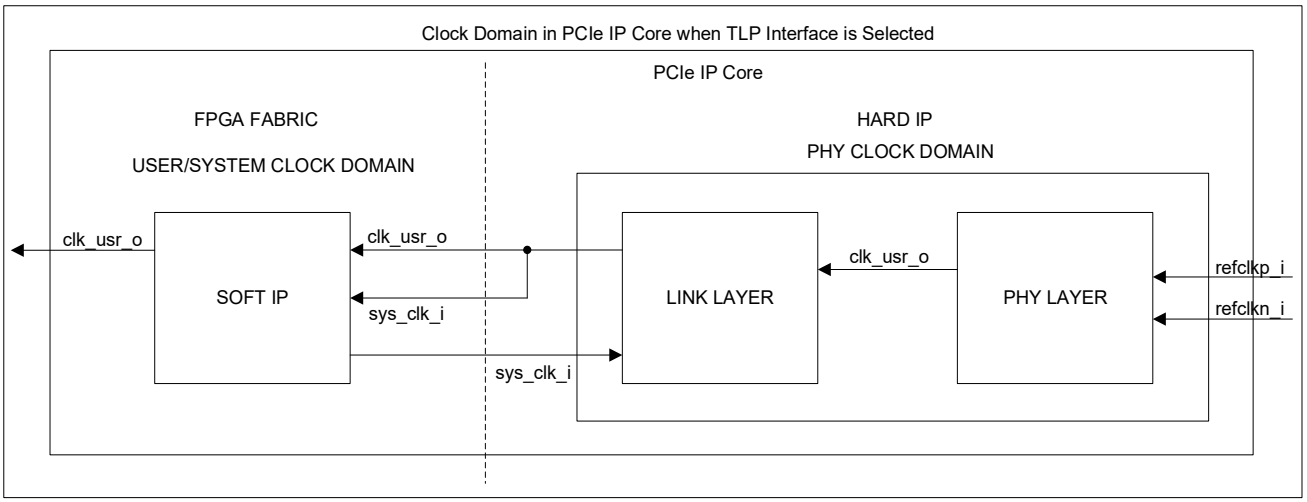

Figure 2.3. PCIe IP Clock Domain Block Diagram for TLP Interface

The PCIe x1 IP includes the following clock domains. The *sys_clk_i* is generated during the PLL IP instantiation. For the TLP interface, you can connect *link0_clk_usr_o* back to the *sys_clk_i*, as shown in Figure 2.3.

- refclkp_i/refclkn_i are differential PHY reference clocks at 100 MHz. The reference clock supports the following:
  - Common Clock (CC)
  - Common Clock with Spread (CCS) – Spread Spectrum Clocking can be modulated by +0% to -0.5% from nominal (5000 ppm). The modulation rate must be between 30 kHz and 33 kHz.
- sys_clk_i is the user clock domain input clock.
  - This clock is generated by the system PLL and shared to the Link layer blocks.
  - For the TLP interface variants, you can choose to connect *link0_clk_usr_o*back to the *sys_clk_i* as shown in Figure 2.3.
- link0_clk_usr_o is the user clock domain output clock.
  - This is the pclk output that comes from the PHY of the PCIe IP core.
  - By default, *clk_usr_o* uses the divide-by-2 version of the 125 MHz pclk from the PHY.

The clock frequency for each interface signal is described in the Signal Description section. Table 2.1 shows the clock frequency for each generation.

Table 2.1. PHY Clock and User Clock Frequencies

Link Speed	PHY Clock Domain	User Clock Domain
	refclkp_i/refclkn_i	sys_clk_i
Gen 1	100 MHz	62.5 MHz
Gen 2	100 MHz	125 MHz

2.3. Reset

2.3.1. Reset Overview

There are two fundamental reset events that can occur in PCI Express:

- Cold Reset – This is a fundamental reset applied during power cycling. The signal perst_n_i is asserted.
- Warm Reset – This is a fundamental reset triggered by hardware without the removal and re-application of power. The perst_n_i signal is asserted.

The fundamental reset *perst_n_i* resets the core (Link Layer and PHY Layer blocks) while another reset, which is the user clock domain Link Layer reset *rst_usr_n_i*, is used to reset the Link Layer block only.

Depending on the PCIe IP configuration, either DMA or non-DMA, either reset signal *c_apb_preset_n_i* or *usr_lmmi_resetn_i* is used.

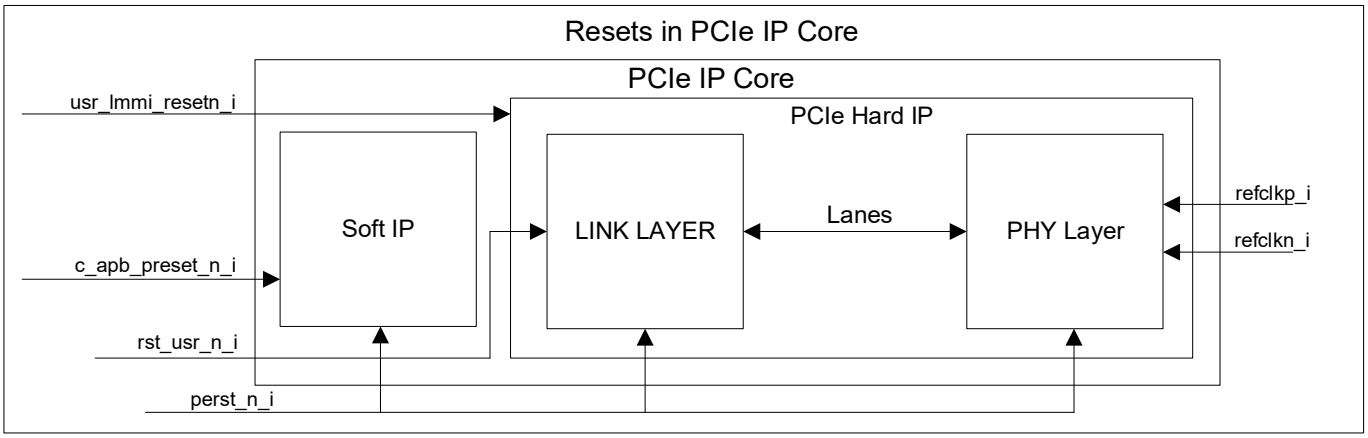

Figure 2.4. Reset Signals of Lattice PCIe IP Core

2.3.2. Clock and Reset Sequence

The PCIe IP clock and reset operation is shown in Figure 2.5.

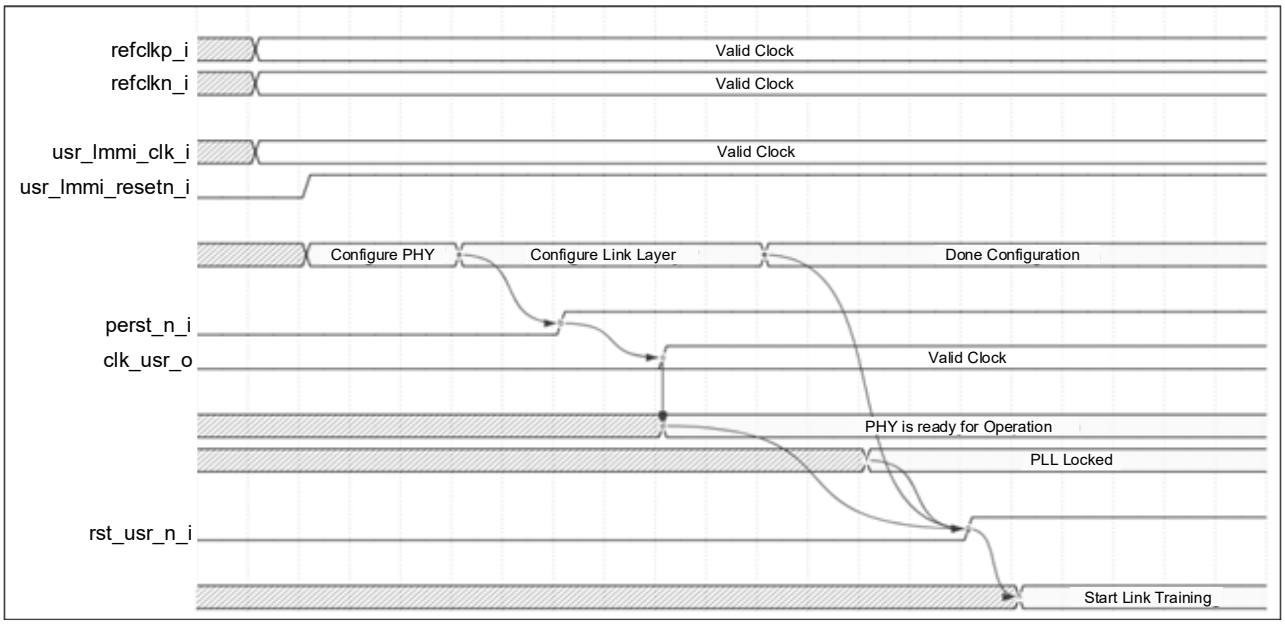

Figure 2.5. Clock and Reset Sequence Diagram

The Lattice PCIe x1 IP Core configuration register implementation has default values that are appropriate for most applications. You can change the register configuration through the LMMI or APB interface. When the LMMI or APB interface is used to configure the PHY layer registers, the configuration should be done before the deassertion of *perst_n_i* signal. The PHY Layer is released from reset and is ready for operation once it can generate the PIPE clock output (such as the clk_usr_o signal). The user domain reset (such as *rst_usr_n_i*) can be deasserted if the Link Layer register configuration is done or skipped.

To ensure that the clock is stable before the link training, you must wait for the PLL locked status of Tx PLL before de-asserting the user domain reset (*rst_usr_n_i*). For x1 link width, you may observe the one channel Tx PLL locked status. The TX PLL status (*bit-4, offset 0x7F in PHY PMA Status register*) can be read through the LMMI or APB.

2.4. Protocol Layers

There are three major classes of packets in PCIe devices: Transaction Layer Packets (TLP), Data Link Layer Packets (DLLP), and Ordered Sets (OS), which is also known as ordered sets. The function of the Protocol Layer is to generate and process these packets.

- Transaction Layer
  The Transaction Layer manages the TLPs to communicate request and completion data with other PCIe devices. The TLP packets are assembled at the *transmit side* of the link and disassembled at the *receive side* of the link. The TLP communicates through different formats either in I/O request format or in the memory request format.
- Data Link Layer
  The Data Link Layer transfers data from the Transaction Layer to the Physical Layer. It plays an important role in assuring good reception of the TLP packets. The DLLPs are used to convey the information about the link initialization, power management, flow control, and TLP acknowledgements.

- Physical Layer
  The Physical Layer converts the packets from the Data Link Layer into serialized bit streams and transfers it to the external physical link. The receive logic de-serializes the bits, reassembles the packets, and forwards it to the Data Link Layer. It conveys the communication between the Data Link Layer and the external physical link. The Physical layer is divided into the Logical sub-block and the Electrical sub-block. The Logical sub-block frames and deframes the packets and implements the LTSSM state machine. The scrambling, descrambling, and 8B/10B encoding and decoding of data are done in the logical sub-block. The Electrical sub-block provides the physical interface to the Link and contains the differential transmitters and receivers. The PLPs or ordered sets are exchanged during link training and link initialization.

2.4.1. ECC and Parity Data Path Protection

The Lattice PCIe x1 IP Core protects the TLP data path with Error Correction Coding (ECC) and Parity Protection. This is implemented in the Hard IP block.

ECC is used to protect TLP data in the following data path RAMs:

- Replay Buffer
- Receive Buffer
- Transmit Buffer

The ECC implementation enables correction for 1-bit errors and detection for 2-bit errors. The 8-bit of ECC information is included in the RAMs for each 64 (or fraction thereof) data bits.

Even (XOR) Parity (*parity[i] = ^(data[((i+1) × 8)-1:(i × 8)]*) is used to protect the data path. Parity provides detection for 1-bit errors (and other odd-bit errors). To enable continuous parity protection coverage, parity is passed through RAMs that are also protected by ECC.

The core includes the ability to enable/disable the reporting and handling of ECC/Parity errors. Correctable errors (ECC 1-bit errors) are fixed when correction is enabled. Uncorrectable ECC/Parity errors in the transmit data path result in the associated TLP being discarded or nullified when error handling is enabled. While error handling can be disabled, this is not recommended as passing a known TLP with bad contents can result in a more serious error condition than discarding the TLP.

2.4.1.1. Receive Data Path

For the receive data path, parity is generated for received TLPs prior to the removal and validation of the Link CRC (LCRC). Parity protection is thus overlapped with LCRC protection.

Received TLP parity is passed with the associated received TLP (header and payload) bytes through the Receive Buffer and onto the user Transaction Layer Receive interface. It is expected that parity is checked and errors are handled by the ultimate TLP consumer. Since TLP can have parity errors on any byte (toward the end of a longer TLP for instance), it is generally not possible to avoid processing the error TLP as the earlier portion of the TLP may already have been processed by the time that the error is detected.

Applications that do not want to process TLPs with errors need to store and forward the TLP for processing only after inspecting the parity of all data bytes. If the core Transaction Layer detects a parity error while it is consuming a received TLP (Type 0 Configuration Read/Write, Malformed TLP, and Message), the error is reported as Uncorrectable Error (in AER capability) and the core discards the TLP without processing it.

2.4.1.2. Transmit Data Path

For the transmit data path, parity is generated by the TLP source. For user TLPs (for example those transmitted on the core's Transaction Layer Transmit Interface), the parity is provided along with associated TLP (header and payload) bytes. The provided parity is kept with the associated data as it traverses the core. The parity is checked and discarded just after the TLP PCIe LCRC is generated.

Parity protection is thus overlapped with LCRC protection, including the associated PCIe replay mechanism. If the core detects a parity or uncorrectable ECC error during transmission of a TLP, the error is reported and the associated TLP is nullified (discarded) and not retransmitted. This is a serious error that must be handled by the software. The TLP is discarded to not propagate the error and risk potentially worse consequences in other components that receives TLPs with known bit errors.

2.4.1.3. Uncorrectable Error Recovery

PCI Express includes the ability to nullify or cancel a TLP transmission immediately after it is completed by inverting the LCRC and using End Bad (EDB) end framing instead of the normal TLP end framing. TLP can be nullified to reduce propagation, potentially multiplying the effects of the error. Nullified TLPs are not regenerated by the original TLP source as it is difficult for software to construct the missing TLP. As a result, there is a fatal system error condition regardless of whether the error TLP is nullified or not. When TLP is nullified due to errors, the core attempts to keep the transmit stream active so that the software can be notified of the error using the standard in-band mechanisms (for example, transmission of ERR_NFAT or ERR_FAT message).

TLPs are allocated a sequence number during transmission and the PCIe receiver only accepts TLPs in sequential order. When a TLP is nullified due to an uncorrectable error, the missing sequence number must be recovered before the link can continue to transmit TLPs.

TLPs are allocated Virtual Channel Flow Control Credits when they are transmitted by the Transaction Layer. The PCI Express device receiving the TLP over the PCI Express link frees the associated credits by sending Flow Control Update DLLPs. TLPs, which are nullified due to uncorrectable ECC and Parity errors, are allocated credits by the Transaction Layer, which is never freed since the TLP is nullified and not received by the Receiver. Nullified TLPs are discarded by the Receiver without affecting Flow Control Credits or Sequence Number.

Whenever a transmitted TLP is nullified due to an uncorrectable error, this causes the PCI Express link to be unable to process further TLPs. The sequence number and flow control credits that are allocated to the nullified TLP must be reclaimed before the link is repaired. The Lattice PCIe x1 IP Core contains logic to correct the link when TLPs are nullified due to uncorrectable errors.

Whenever an uncorrectable ECC of Parity error is detected, it is recommended for you to reset the link through the software to reset the link although the link is corrected for further transmission.

2.4.2. Error Handling

The Lattice PCIe x1 IP Core detects and implements the appropriate response to most error conditions without user intervention. You generally only need to detect and report errors that the core does not have enough information to detect.

2.4.2.1. PCIe-Defined Error Types

The following defines the error types in the PCIe. The *Type* column in Table 2.2 to Table 2.5 defines the PCI Express defined error severity:

- COR – Correctable
- NFAT – Uncorrectable – Non-Fatal
- FAT – Uncorrectable – Fatal

Table 2.2. General PCI Express Error List

Error	Type
Corrected Internal Error	COR
Uncorrectable Internal Error	FAT
Header Log Overflow	COR

Table 2.3. Physical Layer Error List

Error	Type
Receiver Error	COR

Table 2.4. Data Link Layer Error List

Error	Type
Bad TLP	COR
Bad DLLP	COR
Replay Timeout	COR
REPLAY_NUM Rollover	COR
Data Link Layer Protocol Error	FAT
Surprise Down	FAT

Table 2.5. Transaction Layer Error List

Error	Type
Poisoned TLP Received	NFAT
ECRC Check Failed	NFAT
Unsupported Request	NFAT
Completion Timeout	NFAT
Completer Abort	NFAT
Unexpected Completion	NFAT
ACS Violation	NFAT
MC Blocked TLP	NFAT
AtomicOp Egress Blocked	NFAT
Receiver Overflow	FAT
Flow Control Protocol Error	FAT
Malformed TLP	FAT

2.4.2.2. User Error Reporting

The User Hardware design must be able to detect and report the following errors:

- Uncorrectable Internal Error
  - Signals if AER Version 0x2 is enabled in the core and user hardware are detected and unable to correct an application-specific error that is not reported through another error mechanism.
  - If AER is supported by the core, the header of the first TLP associated with the error may optionally be logged.
- Poisoned TLP Received with Advisory Non-Fatal Severity
  - Signals if the core's default poison handling is disabled (*ignore_poison == 1*) and you receive a poisoned TLP that is considered as *Advisory Non-Fatal* severity. If the data payload of a poisoned packet is used or the poison can be recovered from the software or other mechanism, the poison should be treated as *Advisory Non-Fatal* since a non-fatal error often causes a system operation to crash.
  - If AER is supported by the core and the core is operating in Endpoint mode, an ERR_COR message is requested and transmitted if enabled.
  - If AER is supported by the core, the header of the poisoned packet must be logged.
- Poisoned TLP with Non-Fatal Severity
  - Signals if the core's default poison handling is disabled (*ignore_poison == 1*) and you receive a poisoned TLP that is considered as *Non-Fatal* severity. Handling poison as *Non-Fatal* severity should be avoided when possible as this is often fatal to the system operation.
  - If AER is supported by the core, the header of the poisoned packet must be logged.

- Unsupported Request
  - A Type0 Vendor-defined message that is received but not supported by user logic is an Unsupported Request. This is uncommon since only devices designed to receive Type0 Vendor-defined messages should receive these. However, compliance tests may require this error to be handled; hence, it is recommended to implement this check. Receiving a message with Message Code == 0x7E should cause Unsupported Request to be reported, unless the user design is designed to receive these messages.
  - Completions that are received with a Reserved Completion status must be handled as if the Completion status is an Unsupported Request.
- Completion Timeout
  - If you initiate a non-posted request (all reads, I/O Write, and Configuration Write), you are required to implement a completion timeout timer that fires if completions to a non-posted request are not received in the allotted time. This error check needs to be implemented by the user design that includes initiating non-posted requests.
- Completion Abort
  - Signals if permanently unable to process a request due to a device-specific error condition. Generally, this error is only signaled if you choose to implement a restricted programming model (that requires the software to always perform DWORD size transactions and not support burst transactions). This is not recommended unless that the only software that can access the user design is your own software, which is designed to conform with the restricted programming model.
  - If AER is supported by the core, the header of the aborted request must be logged.
- Unexpected Completion
  - You must signal if a completion is received but the tag does not match any outstanding requests.
  - If the core is enabled for Target-Only mode indicating that the user design does not initiate non-posted requests, the core considers all completions as Unexpected Completions, discards them, and generates the appropriate response. In this case, you do not handle this error.
  - If AER is supported by the core, then the header of the completion must be logged.

As a minimum, it is recommended to report the following errors:

- Completion Timeout if user logic initiates non-posted requests (for example, DMA read requests)
- Unsupported Requests for the cases described above
- Unexpected Completion of the case described above
- Poison, when the core's default poison handling is disabled (ignore_poison == 1)

2.4.3. LTSSM State

2.4.3.1. Main LTSSM

The Lattice PCIe x1 IP Core follows the PCI Express specification for the Link Training and Status State Machine. However, to help hit higher frequencies, the LTSSM is split into one Major State LTSSM state machine and several separate LTSSM sub-state machines, with one sub-state state machine for each major state.

The Lattice PCIe x1 IP Core implements additional LTSSM sub-states that are necessary to meet PCIe specification LTSSM operation but are not given an explicit sub-state in the PCIe specification. Table 2.6 lists each state.

Table 2.6. LTSSM State Definition

LTSSM Major State	LTSSM Sub-state	Description
0 – Detect	0 – DETECT_INACTIVE	The sub-state is *DETECT_INACTIVE* whenever the LTSSM major state is not *Detect*.
	1 – DETECT_QUIET	Detect.Quiet
	2 – DETECT_SPD_CHG0	Detect.Quiet – Sub-state to change speed change back to 2.5G if needed. Request PHY speed change.
	3 – DETECT_SPD_CHG1	Detect.Quiet – Sub-state to change speed change back to 2.5G if needed. Wait for speed change to complete.
	4 – DETECT_ACTIVE0	Detect.Active – First Rx Detection.
	5 – DETECT_ACTIVE1	Detect.Active – Wait 12 ms between Rx Detection attempts.
	6 – DETECT_ACTIVE2	Detect.Active – Second Rx Detection (if needed).

LTSSM Major State	LTSSM Sub-state	Description
	7 – DETECT_P1_TO_P0	Detect.Active – Change PHY power state from P1 to P0 (inactive to active) if needed (that is on Detect – Polling transition).
	8 – DETECT_P0_TO_P1_0	Change PHY power state from P0 to P1 (active to inactive) – Transmit Electrical Idle Ordered Sets to notify the link partner that the link is idle.
	9 – DETECT_P0_TO_P1_1	Change PHY power state from P0 to P1. Wait for TX Electrical Idle Ordered Set transit request made in DETECT_P0_TO_P1_0 to get transmitted at the output of the core.
	10 – DETECT_P0_TO_P1_2	Change PHY power state from P0 to P1. Wait for PHY to reach P1 state before continuing.
1 – Polling	0 – POLLING_INACTIVE	The sub-state is *POLLING_INACTIVE* whenever the LTSSM Major State is not Polling.
	1 – POLLING_ACTIVE_ENTRY	Polling.Active – Entry to *Polling.Active* State exists since in some cases, the LTSSM must exit Polling without Tx of TS OS.
	2 – POLLING_ACTIVE	Polling.Active
	3 – POLLING_CFG	Polling.Configuration
	4 – POLLING_COMP	Polling.Compliance – Transmitting compliance pattern.
	5 – POLLING_COMP_ENTRY	Polling.Compliance entry state – Directs a speed change through POLLING_COMP_EIOS, POLLING_COMP_EIOS_ACK, and POLLING_COMP_IDLE when necessary before going to POLLING_COMP.
	6 – POLLING_COMP_EIOS	Polling.Compliance – Transmit Electrical Idle Ordered Sets to notify the link partner that the link is idle.
	7 – POLLING_COMP_EIOS_ACK	Polling.Compliance – Wait for the Electrical Idle Ordered Sets transmitted in POLLING_COMP_EIOS to exit the core.
	8 – POLLING_COMP_IDLE	Polling.Compliance – Perform speed change now that link is idle.
2 – Configuration	0 – CONFIGURATION_INACTIVE	The sub-state is *CONFIGURATION_INACTIVE* whenever the LTSSM Major State is not Configuration.
	1 – CONFIGURATION_US_LW_START	Acting as Upstream Port – Configuration.Linkwidth.Start
	2 – CONFIGURATION_US_LW_ACCEPT	Acting as Upstream Port – Configuration.Linkwidth.Accept
	3 – CONFIGURATION_US_LN_WAIT	Acting as Upstream Port – Configuration.Lanenum.Wait
	4 – CONFIGURATION_US_LN_ACCEPT	Acting as Upstream Port – Configuration.Lanenum.Accept
	5 – CONFIGURATION_DS_LW_START	Acting as Downstream Port – Configuration.Linkwidth.Start
	6 – CONFIGURATION_DS_LW_ACCEPT	Acting as Downstream Port – Configuration.Linkwidth.Accept
	7 – CONFIGURATION_DS_LN_WAIT	Acting as Downstream Port – Configuration.Lanenum.Wait
	8 – CONFIGURATION_DS_LN_ACCEPT	Acting as Downstream Port – Configuration.Lanenum.Accept
	9 – CONFIGURATION_COMPLETE	Configuration.Complete
	10 – CONFIGURATION_IDLE	Configuration.Idle

LTSSM Major State	LTSSM Sub-state	Description
3 – L0	0 – L0_INACTIVE	The sub-state is *L0_INACTIVE* whenever the LTSSM Major State is not L0.
	1 – L0_L0	L0 – Link is in L0.
	2 – L0_TX_EL_IDLE	Tx_L0s.Entry, L1.Entry, or L2.Entry – Transmit Electrical Idle Ordered Sets to notify the link partner that the link is idle (that is for preparing to enter low power states such as Tx_L0s, L1, and L2).
	3 – L0_TX_IDLE_MIN	Tx_L0s.Entry, L1.Entry, or L2.Entry – Guarantee the minimum Tx Elec Idle time when entering electrical idle and require Rx EIOS to have been received when necessary.
4 – Recovery	0 – RECOVERY_INACTIVE	The sub-state is *RECOVERY_INACTIVE* whenever the LTSSM Major state is not *Recovery*.
	1 – RECOVERY_RCVR_LOCK	Recovery.RcvrLock
	2 – RECOVERY_RCVR_CFG	Recovery.RcvrCfg
	3 – RECOVERY_IDLE	Recovery.Idle
	4 – RECOVERY_SPEED0	Recovery.Speed – Transmit Electrical Idle Ordered Sets to notify the link partner that the link is idle.
	5 – RECOVERY_SPEED1	Recovery.Speed – Determine to which speed to change.
	6 – RECOVERY_SPEED2	Recovery.Speed – Wait for remote device to enter electrical idle and wait for the required minimum time.
	7 – RECOVERY_SPEED3	Recovery.Speed – Request PHY change speed and wait for PHY to finish changing speed.
	8 – RECOVERY_EQ_PH0	Recovery.Equalization – Phase 0
	9 – RECOVERY_EQ_PH1	Recovery.Equalization – Phase 1
	10 – RECOVERY_EQ_PH2	Recovery.Equalization – Phase 2
	11 – RECOVERY_EQ_PH3	Recovery.Equalization – Phase 3
5 – Disable	0 – DISABLE_INACTIVE	The sub-state is *DISABLE_INACTIVE* whenever the LTSSM Major state is not *Disable*.
	1 – DISABLE0	Disable – Transmit 16 to 32 TS1 Ordered Sets with Disable Link bit asserted.
	2 – DISABLE1	Disable – Transition to Electrical Idle.
	3 – DISABLE2	Disable – Wait to receive an Electrical Idle Ordered Set and min time of TX_IDLE_MIN afterwards.
	4 – DISABLE3	Disable – Wait until a Disable exit condition occurs.
6 – Loopback	0 – LOOPBACK_INACTIVE	The sub-state is *LOOPBACK_INACTIVE* whenever the LTSSM Major state is not *Loopback*.
	1 – LOOPBACK_ENTRY	Loopback.Entry – Loopback entry state – Loopback Leader may be required to Tx Loopback TS OS before continuing or speed may need to be changed before beginning loopback.
	2 – LOOPBACK_ENTRY_EXIT	Loopback.Entry – Prepare to enter Loopback.Active
	3 – LOOPBACK_EIOS	Loopback.Entry – Transmit Electrical Idle Ordered Sets to notify the link partner that the link is idle. (to change speed).
	4 – LOOPBACK_EIOS_ACK	Loopback.Entry – Wait for the Electrical Idle Ordered Sets transmitted in LOOPBACK_EIOS to exit the core.
	5 – LOOPBACK_IDLE	Loopback.Entry – Stay in Electrical Idle for required minimum time.
	6 – LOOPBACK_ACTIVE	Loopback.Active
	7 – LOOPBACK_EXIT0	Loopback.Exit – Tx Electrical Idle
	8 – LOOPBACK_EXIT1	Loopback.Exit – Stay in Electrical Idle for required minimum time.

LTSSM Major State	LTSSM Sub-state	Description
7 – Hot Reset	0 – HOT_RESET_INACTIVE	The sub-state is *HOT_RESET_INACTIVE* whenever the LTSSM Major state is not *Hot Reset*.
	1 – HOT_RESET_HOT_RESET	Hot Reset – as Follower
	2 – HOT_RESET_ LEADER _UP	Hot Reset – as Leader with Link Up
	3 – HOT_RESET_ LEADER _DOWN	Hot Reset – as Leader with Link Down
8 – TX L0s	0 – TX_L0S_INACTIVE	The sub-state is *TX_L0S_INACTIVE* whenever the LTSSM Major state is not *TX L0s*.
	1 – TX_L0S_IDLE	Tx_L0s.Idle – Idle
	2 – TX_L0S_TO_L0	Tx_L0s.Idle – Exiting TX L0s; wait for PHY to indicate exit from L0s complete
	3 – TX_L0S_FTS0	Tx_L0s.FTS – Transmit requested NFTS.
	4 – TX_L0S_FTS1	Tx_L0s.FTS – Transmit additional FTS required by Cfg Register Extended Sync.
9 – L1	0 – L1_INACTIVE	The sub-state is *L1_INACTIVE* whenever the LTSSM Major state is not *L1*.
	1 – L1_IDLE	L1.Idle
	2 – L1_SUBSTATE	L1.1 or L1.2 depending upon higher level Power Management State Machine control.
	3 – L1_TO_L0	L1.Idle – Exiting L1; wait for PHY to indicate exit from L1 complete.
10 – L2	0 – L2_INACTIVE	The sub-state is *L2_INACTIVE* whenever the LTSSM Major State is not *L2*.
	1 – L2_IDLE	L2.Idle – Idle
	2 – L2_TX_WAKE0	L2.TransmitWake – Transmit a Beacon until remote device exits electrical idle.
	3 – L2_TX_WAKE1	L2.TransmitWake – Assert Tx Electrical Idle before changing power state to P1.
	4 – L2_EXIT	L2.Idle – L2 exit; wait until PHY finishes power change out of L2.
	5 – L2_SPEED	L2.Idle – Change speed if required before going to L2.

2.4.3.2. RX L0s State Machine

The Rx_L0s State Machine follows the L0s state of the receiver. The Rx_L0s State Machine operates independently of the main LTSSM, which controls the state of the transmitter.

Table 2.7. RX L0s State Description

LTSSM Sub-state	Description
0 – RX_L0S_L0	The sub-state is “RX_L0S_L0” whenever the receiver is in L0 (that is not en route to or in Rx L0s).
1 – RX_L0S_ENTRY	Rx_L0s.Entry
2 – RX_L0S_IDLE	Rx_L0s.Idle
3 – RX_L0S_FTS	Rx_L0s.FTS
4 – RX_L0S_REC	Rx_L0s.FTS – Wait until LTSSM Major State == Recovery due to Rx L0s exit error

2.5. Multi-Function Support

The Lattice PCIe x1 IP Core supports 1 to 4 functions. The Multi-Function support can only be enabled for endpoints (functions implementing Type 0 Configuration Space). See section Function Register 0x08 for the register configuration.

When Multiple Function support is present, each function is assigned a static function number, starting at function number 0 and incrementing upwards. For ports that communicate function-specific information, port[0] applies to Function[0], port[1] applies to Function[1]. If a function is disabled, it does not affect the function number of the other enabled functions. Function [0] is always present and cannot be disabled.

Note: Refer to file LMMI_app.v inside the *Design File* of PCIe Multifunction Demo for Lattice Nexus-based FPGAs on how to enable and initial the configuration space registers of Function 1, Function 2, and Function 3 through the LMMI interface.

The demo design included the LMMI_app.v that overwrites the settings in the IP user interface at every reset.

Refer to FAQ 6900 on how the PCIe configuration space settings in the RTL of the demo for the Nexus Platform (Certus-NX, CrossLink-NX, and so on) can be interpreted.

2.6. Power Management

2.6.1. Power Management Supported by PCIe IP Core

The Lattice PCIe x1 IP Core supports L0, ASPM L0s, ASPM L1, L1 PM Substates, L1, and L3 link states. L0 (fully-operational state) and L3 (off) support is always enabled. The remaining link states may be enabled/disabled through the Core Configuration ports. If ASPM L0s, ASPM L1, and L1 PM Substates, or L1 support is enabled, then the user design must configure the power management capabilities of the core and for some link states, take additional action when link states are entered or exited. This section describes the recommended actions user logic should take to control and react to power management states ASPM L0s, ASPM L1, and L1.

The PCI Express Specification defines the following link states:

- L0 – Active
  - Powered
  - Clock and PLLs active; core clock active
  - All PCI Express Transactions and operations are enabled
- ASPM L0s – Low resume latency, energy saving *standby* state
  - Powered
  - Clock and PLLs active; core clock active
  - PHY transmitter in electrical idle
  - Remote PHY receiver must re-establish symbol lock during L0s exit
  - When L0s is enabled by power management software, the core autonomously enters L0s when the transmit side of the link is idle and exits L0s when there is pending information to transmit. The link management DLLPs are required to be transmitted periodically so when a link is otherwise idle, it still enters and exits L0s with regularity to transmit link management DLLPs.
- ASPM L1 – Low resume latency, energy saving *standby* state
  - Powered
  - Clock and PLLs active; core clock active
  - Significant portion of PHY powered down
  - PHY transmitter in electrical idle
  - PHY receiver in electrical idle
  - Deeper power savings but longer resume time than ASPM L0s
  - Remote and local PHY must re-establish symbol lock during L1 exit
  - When ASPM L1 is enabled by power management software, the core autonomously negotiates L1 entry with the link partner after an extended period of link inactivity. The link autonomously returns to L0 when either device in the link has TLPs to transmit.

- L1 – Higher latency, lower power *standby* state
  - Powered
  - Clock and PLLs active; core clock active
  - Significant portion of PHY powered down
  - PHY transmitter in electrical idle
  - PHY receiver in electrical idle
  - Remote and local PHY must re-establish symbol lock during L1 exit
  - The L1 state is entered both under control of power management software
- L3 – Off
  - Main power off; auxiliary power off
  - In this state, all power is removed and the core, PHY, and user logic are all non-operational
  - All state information is lost

2.6.2. Configuring Core to Support Power Management

The Lattice PCIe IP x1 Core allows user logic to implement a wide variety of power management functionality. The design's power management capabilities are primarily advertised and controlled using the core configuration ports.

2.6.3. APSM L0s

The Lattice PCIe x1 IP Core supports Active State Power Management (APSM) L0s. When L0s support is enabled, ASPM L0s TX Entry Time, the desired amount of time for TLP and DLLP transmissions to be idle before L0s TX is entered, is determined by mgmt_ptl_pm_aspm_l0s_entry_time. The Number of NFTS sets required by the local PHY to recover symbol lock when exiting L0s is determined by mgmt_tlb_ltssm_nfts_nfts. NFTS Timeout Extend, mgmt_tlb_ltssm_nfts_to_extend, controls how long the core waits after the expected L0s exit time before directing the link to Recovery to recover from a failed L0s exit. Due to high latencies between a PHY's Rx Electrical Idle output and the associated Rx Data it is normally necessary to choose a relatively high NFTS and NFTS Timeout Extend.

- Configuration Register Fields
  - The PCI Express Device Capabilities configuration register has the following L0s fields:
    - Bits [8:6] – Endpoint L0s Acceptable Latency – From PCI Express Base Specification, Rev 2.1 section 7.8.3 – Acceptable total latency that an Endpoint can withstand due to the transition from L0s state to the L0 state. It is essentially an indirect measure of the Endpoint's internal buffering. Power management software uses the reported L0s Acceptable Latency number to compare against the L0s exit latencies reported by all components comprising the data path from this Endpoint to the Root Complex Root Port to determine whether ASPM L0s entry can be used with no loss of performance. Note that the amount of buffering does not refer to the Lattice PCIe x1 IP Core buffering, but rather to user application buffering. You must set this field in accordance with how long a delay is acceptable for the application.
      - 000 – Maximum of 64 ns
      - 001 – Maximum of 128 ns
      - 010 – Maximum of 256 ns
      - 011 – Maximum of 512 ns
      - 100 – Maximum of 1 µs
      - 101 – Maximum of 2 µs
      - 110 – Maximum of 4 µs
      - 111 – No limit
      - Non-Endpoints must hard wire this field to 000

- The PCI Express Link Capabilities configuration register has the following L0s fields:
  - Bits[14:12] – L0s Exit Latency – Length of time required to complete transition from L0s to L0:
    - 000 – Less than 64 ns
    - 001 – 64 ns to less than 128 ns
    - 010 – 128 ns to less than 256 ns
    - 011 – 256 ns to less than 512 ns
    - 100 – 512 ns to less than 1 µs
    - 101 – 1 µs to less than 2 µs
    - 110 – 2 µs to 4 µs
    - 111 – More than 4 µs
    - Exit latencies may significantly increased if the PCI Express reference clocks used by the two devices in the link are common or separate.
  - Bits[11:10] – Active State Power Management (ASPM) Support is set to 01 or 11 if L0s support is enabled or 00 otherwise.

2.6.4. APSM L1

The Lattice PCIe x1 IP Core supports both software controller L1 entry (through the Power State Configuration Register) and hardware autonomous L1 entry (Active State Power Management (APSM) L1).

- Software-controlled L1 flow for Upstream Ports (Endpoint) is as follows:
  - Software initiates changing a link to L1 by writing the core's Power Management Capability: Power State Configuration Register to a value other than 00 == D0. Note that the component's Device driver participates in this process and must ensure that all traffic is idle before permitting the system to power down to L1.
  - When the core detects a change of Power State to a non-D0 value, the core's power management state machine, which is responsible for the higher-level power management protocol, follows the following sequence:
    - Block further TLP transmissions
    - Wait for all in process TLPs to complete transmission
    - Wait for the Replay Buffer to empty (all transmitted TLPs acknowledged)
    - Core transmits PM_ENTER_L1 DLLPs until receiving a PM_REQ_ACK DLLP from remote device
    - Core directs LTSSM state machine to L1
    - When a TLP is pending or the LTTSM state machine indicates L1 state has been exited due to link partner activity, the core returns to L0.
- Configuration Register Fields:
  - The PCI Express Device Capabilities configuration register has the following L1 fields:
    - Bits [11:9] – Endpoint L1 Acceptable Latency – From PCI Express Base Specification, Rev 2.1 section 7.8.3 – This field indicates the acceptable latency that an Endpoint can withstand due to the transition from L1 state to the L0 state. It is essentially an indirect measure of the Endpoint's internal buffering. Power management software uses the reported L1 Acceptable Latency number to compare against the L1 Exit Latencies reported (see below) by all components comprising the data path from this Endpoint to the Root Complex/Root Port to determine whether ASPM L1 entry can be used with no loss of performance. Note that the amount of buffering does not refer to Lattice PCIe x1 IP Core buffering, but rather to user application buffering. You must set this field in accordance with how long a delay is acceptable for the application.
      - 000 – Maximum of 1 µs
      - 001 – Maximum of 2 µs
      - 010 – Maximum of 4 µs
      - 011 – Maximum of 8 µs
      - 100 – Maximum of 16 µs
      - 101 – Maximum of 32 µs
      - 110 – Maximum of 64 µs
      - 111 – No limit
      - Non-Endpoints must hard wire this field to 000.

- PCI Express Link Capabilities configuration register has the following L1 fields:
  - Bits[17:15] – L1 Exit Latency – Length of time required to complete transition from L1 to L0:
    - 000 – Less than 1 µs
    - 001 – 1 µs to less than 2 µs
    - 010 – 2 µs to less than 4 µs
    - 011 – 4 µs to less than 8 µs
    - 100 – 8 µs to less than 16 µs
    - 101 – 16 µs to less than 32 µs
    - 110 – 32 µs-64 µs
    - 111 – More than 64 µs
    - Exit latencies may be significantly increased if the PCI Express reference clocks used by the two devices in the link are common or separate.
  - Bits[11:10] – Active State Power Management (ASPM) Support should be set to *10* or *11* if L1 support is enabled or *00* otherwise.
- Hardware-autonomous L1 (ASPM L1) entry is initiated only by Upstream Ports (Endpoint). The core ASPM L1 functionality must be enabled and advertised in PCIe Link Capabilities and software must enable ASPM L1 support for the hardware-autonomous L1 to be negotiated. When ASPM L1 support is present and enabled for an Upstream Port, the core requests the link to be directed to L1 using the ASPM L1 protocol, when the link is idle. The link idle refers to the no TLPs or ACK/NAL DLLPs being transmitted.
- A Downstream Port (Root Port) receives ASPM L1 requests from the remote device and may choose to accept or reject the request. The core accepts ASPM L1 requests when there are no TLP or ACK/NAK DLLP pending transmit and otherwise rejects the request.

2.7. DMA Support

The Direct Memory Access (DMA) support is an option provided by soft IP to enable a more efficient data transfer when the device acts as initiator.

2.7.1. DMA Overview

The Direct Memory Access is an efficient way of transferring data. In this, a DMA engine handles the data transaction process on behalf of the processor. Once the processor forms descriptors in host memory and programs DMA registers through memory write, the DMA engine handles the bus protocol and address sequencing on its own.

After the IP has its registers written with the total number of descriptor and the address of the first descriptor, it fetches the descriptors from host memory through the Memory Read TLP. When Completion(s) is received, the IP starts the transaction based on descriptor data.

Once a transaction with Interrupt bit is set in its descriptor is completed, the DMA IP transmits MSI as an interrupt to the host.

The IP supports data transfer for both Host-to-FPGA (H2F) and FPGA-to-Host (F2H). Each direction has a dedicated set of registers. Refer to DMA Registers for DMA registers.

2.7.2. DMA Descriptor

The descriptors are packets of data which contain information such as source address, destination address, length of DMA transfer, and other attributes such as the number of contiguous descriptors and interrupt. The descriptor data is stored in the host memory and to be fetched by the IP through Memory Read. The start address of the descriptor queue in the host memory and the total contiguous descriptor are given from *H2F Descriptor Fetching (0x0200)* and *F2H Descriptor Fetching (0x0300)* registers. Based on the start address of descriptor queue, the IP does the bulk fetching from the host memory.

Table 2.8. Descriptor Format

Offset	Fields				
0x00	RSVD[17:0]	CONT_DESC[5:0]	RSVD[5:0]	INT	EOP
0x04	RSVD[7:0]	LENGTH[23:0]			
0x08	NEXT_DESC_ADDR_LO[31:0]				
0x0C	NEXT_DESC_ADDR_HI[31:0]				
0x10	SRC_ADDR_LO[31:0]				
0x14	SRC_ADDR_HI[31:0]				
0x18	DEST_ADDR_LO[31:0]				
0x1C	DEST_ADDR_HI[31:0]				

Table 2.9. DESC_CTRL (0x00)

Field	Name	Width	Description
31:14	RSVD	18	Reserved
13:8	CONT_DESC	6	The number of contiguous Descriptor from the Descriptor address in NEXT_DESC_ADDR_LO and NEXT_DESC_ADDR_HI. All 0s mean 64 contiguous descriptors. This field is only valid when EOP at the last descriptor of a descriptor chunk is 0. In all other cases, this field is ignored.
2:7	RSVD	6	Reserved
1	INT	1	Interrupt trigger. Once the data transfer described by this descriptor is done, the interrupt is triggered by DMA engine to the Host. Interrupt type and vector mapping are configurable in DMA register.
0	EOP	1	Stop fetching the next descriptor. This bit can be 1 only at the last descriptor of a Descriptor Chunk. This field is ignored by the IP if it is not the last descriptor of a Descriptor Chunk.

Table 2.10. DMA_LEN (0x04)

Field	Name	Width	Description
31:24	RSVD	8	Reserved
23:0	LENGTH	24	DMA transfer length in Byte. 24'd1: 1 Byte transfer 24'd2: 2 Byte transfer and so on. All 0 means 16 Mega Byte transfer.

Table 2.11. NEXT_DESC_ADDR_LO (0x08)

Field	Name	Width	Description
31:0	NEXT_DESC_ADDR_LO	32	Lower 32 bit of the next Descriptor Address. This field is only valid when EOP at the last descriptor of a descriptor chunk is 0. In all other cases, this field is ignored.

Table 2.12. NEXT_DESC_ADDR_HI (0x0C)

Field	Name	Width	Description
31:0	NEXT_DESC_ADDR_HI	32	Upper 32 bit of the next Descriptor Address. This field is only valid when EOP at the last descriptor of a descriptor chunk is 0. In all other cases, this field is ignored.

Table 2.13. SRC_ADDR_LO (0x10)

Field	Name	Width	Description
31:0	SRC_ADDR_LO	32	Lower 32 bit of Source Address

Table 2.14. SRC_ADDR_HI (0x14)

Field	Name	Width	Description
31:0	SRC_ADDR_HI	32	Upper 32 bit of Source Address

Table 2.15. DEST_ADDR_LO (0x18)

Field	Name	Width	Description
31:0	DEST_ADDR_LO	32	Lower 32 bit of Destination Address

Table 2.16. DEST_ADDR_HI (0x1C)

Field	Name	Width	Description
31:0	DEST_ADDR_HI	32	Upper 32 bit of Destination Address

2.7.2.1. Descriptor Rules

The following shows the descriptor rules:

- NEXT_DESC_ADDR must be 8DW-aligned (bit[4:0] = 5'b00000) so that descriptors can end at RCB boundary.
- SRC_ADDR[63:0] and DEST_ADDR[63:0] must be 8DW-aligned (bit[4:0] = 5'b00000). There is no address translation for source address and destination address to the address in AXI-MM interface. The driver must have awareness of the exact physical addresses.
- EOP bit is only observed by the IP at the last descriptor of a descriptor chunk.

Note: Fail to comply to the descriptor rules may result in undefined behaviours.

2.7.2.2. Descriptor Example

In this example, the first descriptor chunk (starting address and number of contiguous descriptors are configured DMA register) has three contiguous descriptors.

The second descriptor chunk (starting address and number of contiguous descriptors from the last descriptor of the first chunk) has two contiguous descriptors.

The third descriptor chunk (starting address and number of contiguous descriptors from the last descriptor of the second chunk) has only one descriptor.

Table 2.17. First Descriptor Chunk Fetching through MRd TLP

Offset	Fields				
0x00	RSVD[17:0]	CONT_DESC[5:0] (Don't care)	RSVD[5:0]	INT	EOP (Don't care)
0x04	RSVD[7:0]	LENGTH[23:0]			
0x08	NEXT_DESC_ADDR_LO[31:0] (Don't care)				

Offset	Fields				
0x0C	NEXT_DESC_ADDR_HI[31:0] (Don't care)				
0x10	SRC_ADDR_LO[31:0]				
0x14	SRC_ADDR_HI[31:0]				
0x18	DEST_ADDR_LO[31:0]				
0x1C	DEST_ADDR_HI[31:0]				
0x20	RSVD[17:0]	CONT_DESC[5:0] (Don't care)	RSVD[5:0]	INT	EOP (Don't care)
0x24	RSVD[7:0]	LENGTH[23:0]			
0x28	NEXT_DESC_ADDR_LO[31:0] (Don't care)				
0x2C	NEXT_DESC_ADDR_HI[31:0] (Don't care)				
0x30	SRC_ADDR_LO[31:0]				
0x34	SRC_ADDR_HI[31:0]				
0x38	DEST_ADDR_LO[31:0]				
0x3C	DEST_ADDR_HI[31:0]				
0x40	RSVD[17:0]	CONT_DESC[5:0] = 2	RSVD[5:0]	INT	EOP= 0
0x44	RSVD[7:0]	LENGTH[23:0]			
0x48	NEXT_DESC_ADDR_LO[31:0] = 'hA0				
0x4C	NEXT_DESC_ADDR_HI[31:0] = 'h0				
0x50	SRC_ADDR_LO[31:0]				
0x54	SRC_ADDR_HI[31:0]				
0x58	DEST_ADDR_LO[31:0]				
0x5C	DEST_ADDR_HI[31:0]				

Table 2.18. Second Descriptor Chunk Fetching through MRd TLP

Offset	Fields				
0xA0	RSVD[17:0]	CONT_DESC[5:0] (Don't care)	RSVD[5:0]	INT	EOP (Don't care)
0xA4	RSVD[7:0]	LENGTH[23:0]			
0xA8	NEXT_DESC_ADDR_LO[31:0] (Don't care)				
0xAC	NEXT_DESC_ADDR_HI[31:0] (Don't care)				
0xB0	SRC_ADDR_LO[31:0]				
0xB4	SRC_ADDR_HI[31:0]				
0xB8	DEST_ADDR_LO[31:0]				
0xBC	DEST_ADDR_HI[31:0]				
0xC0	RSVD[17:0]	CONT_DESC[5:0] = 1	RSVD[5:0]	INT	EOP= 0
0xC4	RSVD[7:0]	LENGTH[23:0]			
0xC8	NEXT_DESC_ADDR_LO[31:0] = 'h1B0				
0xCC	NEXT_DESC_ADDR_HI[31:0] = 'h0				
0xD0	SRC_ADDR_LO[31:0]				
0xD4	SRC_ADDR_HI[31:0]				

Offset	Fields
0xD8	DEST_ADDR_LO[31:0]
0xDC	DEST_ADDR_HI[31:0]

Table 2.19. Third Descriptor Chunk Fetching through MRd TLP

Offset	Fields				
0x1B0	RSVD[17:0]	CONT_DESC[5:0] (Don't care)	RSVD[5:0]	INT	EOP= 1
0x1B4	RSVD[7:0]	LENGTH[23:0]			
0x1B8	NEXT_DESC_ADDR_LO[31:0] (Don't care)				
0x1BC	NEXT_DESC_ADDR_HI[31:0] (Don't care)				
0x1C0	SRC_ADDR_LO[31:0]				
0x1C4	SRC_ADDR_HI[31:0]				
0x1C8	DEST_ADDR_LO[31:0]				
0x1CC	DEST_ADDR_HI[31:0]				

2.7.3. DMA Registers

PCIe DMA registers are accessible by the Host when received MWr or MRd TLP has BAR 0 hit. Register read size is limited to maximum 1 DW per MRd TLP.

The Access Types of each register are defined in Table 2.20.

Table 2.20. Access Types

Access Type	Behavior on Read Access	Behavior on Write Access
RO	Returns register value	Ignores write access
WO	Returns 0	Updates register value
RW	Returns register value	Updates register value
RW1C	Returns register value	Writing 1'b1 on register bit clears the bit to 1'b0. Writing 1'b0 on register bit is ignored.
RW1S	Returns register value	Writing 1'b1 on register bit sets the bit to 1'b1. Writing 1'b0 on register bit is ignored.
RC	Returns register value Clear the register to 0 after read.	Ignores write access
RSVD	Returns 0	Ignores write access

Table 2.21. PCIe DMA Register Group

Register Base Offset	Register Group Name
0x0000	H2F DMA Control and Status
0x0100	F2H DMA Control and Status
0x0200	H2F Descriptor Fetching
0x0300	F2H Descriptor Fetching
0x0400	Interrupt Control and Status
Others	RSVD

2.7.3.1. H2F DMA Control and Status (0x0000)

Table 2.22. H2F_DMA_CTRL (0x0000)

Field	Name	Access	Width	Default	Description
31:1	RSVD	RO	31	0	Reserved
0	REQUEST	RW1S	1	0	Request to start DMA operation. Once this bit is 1, writing a 0 to clear it does not take effect. Once the field in "H2F Descriptor Fetching" is consumed by DMA Engine to trigger descriptor fetching, this bit is cleared to 0 by HW.

Table 2.23. H2F_DMA_STS (0x000C)

Field	Name	Access	Width	Default	Description
31:14	RSVD	RO	18	0	Reserved
13	RSVD	RO	1	0	Reserved
12	RSVD	RO	1	0	Reserved
11	DMA_LEN_ERR	RC	1	0	DMA Length Error 1: DMA Length is not DW-aligned. 0: No error
10	H2F_DESTADDR_ERR	RC	1	0	H2F Destination Address Error 1: H2F Destination Address is not 8DW-aligned. 0: No error
9	H2F_SRCADDR_ERR	RC	1	0	H2F Source Address Error 1: H2F Source Address is not 8DW-aligned. 0: No error
8	DESC_ADDR_ERR	RC	1	0	Descriptor Address Error 1: Desc Address is not 8DW-aligned. 0: No error
7	AXI_WRITE_ERR	RC	1	0	AXI Write Error 1: AXI Write Response is not OKAY (2'b00). 0: No error
6	H2F_CPLTO_ERR	RC	1	0	H2F Completion Timeout Error 1: Completion timeout at H2F DMA transfer. 0: No error
5	H2F_CPL_ERR	RC	1	0	H2F Completion Error 1: Completion Status is not Successful Completion. 0: No error
4	DESC_CPLTO_ERR	RC	1	0	Descriptor Completion Timeout Error 1: Completion timeout at H2F Descriptor fetching. 0: No error
3	DESC_CPL_ERR	RC	1	0	Descriptor Completion Error 1: Completion Status is not Successful Completion. 0: No error
2	DMA_INT_DONE	RC	1	0	DMA Interrupt Done 1: DMA transfer is done for descriptor with INT bit = 1. 0: No DMA Interrupt Done event.
1	DMA_EOP_DONE	RC	1	0	DMA EOP Done 1: DMA transfer is done for descriptor with EOP bit = 1. 0: No DMA EOP Done event.
0	BUSY	RO	1	0	DMA Engine busy 1 : DMA is busy. 0 : DMA in IDLE state, no operation pending.

Table 2.24. H2F_DMA_INT_MASK (0x0010)

Field	Name	Access	Width	Default	Description
31:12	RSVD	RO	20	0	Reserved
11	DMA_LEN_ERR_INTMASK	RW	1	0	DMA Length Error Interrupt Masking 1: Mask off interrupt generation caused by DMA_LEN_ERR. 0: No interrupt masking for DMA_LEN_ERR.
10	H2F_DESTADDR_ERR_INTMASK	RW	1	0	H2F Destination Address Error Interrupt Masking 1: Mask off interrupt generation caused by H2F_DESTADDR_ERR. 0: No interrupt masking for H2F_DESTADDR_ERR.
9	H2F_SRCADDR_ERR_INTMASK	RW	1	0	H2F Source Address Error Interrupt Masking 1: Mask off interrupt generation caused by H2F_SRCADDR_ERR. 0: No interrupt masking for H2F_SRCADDR_ERR.
8	DESC_ADDR_ERR_INTMASK	RW	1	0	Descriptor Address Error Interrupt Masking 1: Mask off interrupt generation caused by DESC_ADDR_ERR. 0: No interrupt masking for DESC_ADDR_ERR.
7	AXI_WRITE_ERR_INTMASK	RW	1	0	AXI Write Error Interrupt Masking 1: Mask off interrupt generation caused by AXI_WRITE_ERR. 0: No interrupt masking for AXI_WRITE_ERR.
6	H2F_CPLTO_ERR_INTMASK	RW	1	0	H2F Completion Timeout Error Interrupt Masking 1: Mask off interrupt generation caused by H2F_CPLTO_ERR. 0: No interrupt masking for H2F_CPLTO_ERR.
5	H2F_CPL_ERR_INTMASK	RW	1	0	H2F Completion Error Interrupt Masking 1: Mask off interrupt generation caused by H2F_CPL_ERR. 0: No interrupt masking for H2F_CPL_ERR.
4	DESC_CPLTO_ERR_INTMASK	RW	1	0	Descriptor Completion Timeout Error Interrupt Masking 1: Mask off interrupt generation caused by DESC_CPLTO_ERR. 0: No interrupt masking for DESC_CPLTO_ERR.
3	DESC_CPL_ERR_INTMASK	RW	1	0	Descriptor Completion Error Interrupt Masking 1: Mask off interrupt generation caused by DESC_CPL_ERR. 0: No interrupt masking for DESC_CPL_ERR.
2	DMA_INT_DONE_INTMASK	RW	1	0	DMA Interrupt Done Interrupt Masking 1: Mask off interrupt generation caused by DMA_INT_DONE. 0: No interrupt masking for DMA_INT_DONE.

Field	Name	Access	Width	Default	Description
1	DMA_EOP_DONE_INTMASK	RW	1	0	DMA EOP Done Interrupt Masking 1: Mask off interrupt generation caused by DMA_EOP_DONE. 0: No interrupt masking for DMA_EOP_DONE.
0	RSVD	RO	20	0	Reserved

Table 2.25. H2F_CPLT_DESC_COUNT (0x0018)

Field	Name	Access	Width	Default	Description
31:0	CPLT_DESC_CNT	RO	32	0	The number of competed descriptors since the last rising edge of REQUEST bit. Reset to 0 at the subsequent rising edge of REQUEST bit.

2.7.3.2. F2H DMA Control and Status (0x0100)

Table 2.26. F2H_DMA_CTRL (0x0100)

Field	Name	Access	Width	Default	Description
31:1	RSVD	RO	31	0	Reserved
0	REQUEST	RW1S	1	0	Request to start DMA operation. 1: Request to start DMA operation Once the field in “F2H Descriptor Fetching” is consumed by DMA Engine to trigger descriptor fetching, this bit is cleared to 0 by HW.

Table 2.27. F2H_DMA_STS (0x010C)

Field	Name	Access	Width	Default	Description
31:12	RSVD	RO	20	0	Reserved
11	DMA_LEN_ERR	RC	1	0	DMA Length Error 1: DMA Length is not DW-aligned. 0: No error
10	F2H_DESTADDR_ERR	RC	1	0	F2H Destination Address Error 1: F2H Destination Address is not 8DW-aligned. 0: No error
9	F2H_SRCADDR_ERR	RC	1	0	F2H Source Address Error 1: F2H Source Address is not 8DW-aligned. 0: No error
8	DESC_ADDR_ERR	RC	1	0	Descriptor Address Error 1: Desc Address is not 8DW-aligned. 0: No error
7	AXI_READ_ERR	RC	1	0	AXI Read Error 1: AXI Read Response is not OKAY (2’b00). 0: No error
6:5	RSVD	RO	2	0	Reserved
4	DESC_CPLTO_ERR	RC	1	0	Descriptor Completion Timeout Error 1: Completion timeout at F2H Descriptor fetching. 0: No error
3	DESC_CPL_ERR	RC	1	0	Descriptor Completion Error 1: Completion Status is not Successful Completion. 0: No error

Field	Name	Access	Width	Default	Description
2	DMA_INT_DONE	RC	1	0	DMA Interrupt Done 1: DMA transfer is done for descriptor with INT bit = 1. 0: No DMA Interrupt Done event.
1	DMA_EOP_DONE	RC	1	0	DMA EOP Done 1: DMA transfer is done for descriptor with EOP bit = 1. 0: No DMA EOP Done event.
0	BUSY	RO	1	0	DMA Engine busy 1: DMA is busy. 0: DMA in IDLE state, no operation pending.

Table 2.28. F2H_DMA_INT_MASK (0x0110)

Field	Name	Access	Width	Default	Description
31:12	RSVD	RO	20	0	Reserved
11	DMA_LEN_ERR_INTMASK	RW	1	0	DMA Length Error Interrupt Masking 1: Mask off interrupt generation caused by DMA_LEN_ERR. 0: No interrupt masking for DMA_LEN_ERR.
10	F2H_DESTADDR_ERR_INTMASK	RW	1	0	F2H Destination Address Error Interrupt Masking 1: Mask off interrupt generation caused by F2H_DESTADDR_ERR. 0: No interrupt masking for F2H_DESTADDR_ERR.
9	F2H_SRCADDR_ERR_INTMASK	RW	1	0	F2H Source Address Error Interrupt Masking 1: Mask off interrupt generation caused by F2H_SRCADDR_ERR. 0: No interrupt masking for F2H_SRCADDR_ERR.
8	DESC_ADDR_ERR_INTMASK	RW	1	0	Descriptor Address Error Interrupt Masking 1: Mask off interrupt generation caused by DESC_ADDR_ERR. 0: No interrupt masking for DESC_ADDR_ERR.
7	AXI_READ_ERR_INTMASK	RW	1	0	AXI Read Error Interrupt Masking 1: Mask off interrupt generation caused by AXI_READ_ERR. 0: No interrupt masking for AXI_READ_ERR.
6:5	RSVD	RO	2	0	Reserved
4	DESC_CPLTO_ERR_INTMASK	RW	1	0	Descriptor Completion Timeout Error Interrupt Masking 1: Mask off interrupt generation caused by DESC_CPLTO_ERR. 0: No interrupt masking for DESC_CPLTO_ERR.
3	DESC_CPL_ERR_INTMASK	RW	1	0	Descriptor Completion Error Interrupt Masking 1: Mask off interrupt generation caused by DESC_CPL_ERR. 0: No interrupt masking for DESC_CPL_ERR.
2	DMA_INT_DONE_INTMASK	RW	1	0	DMA Interrupt Done Interrupt Masking 1: Mask off interrupt generation caused by DMA_INT_DONE. 0: No interrupt masking for DMA_INT_DONE.

Field	Name	Access	Width	Default	Description
1	DMA_EOP_DONE_INTMASK	RW	1	0	DMA EOP Done Interrupt Masking 1: Mask off interrupt generation caused by DMA_EOP_DONE. 0: No interrupt masking for DMA_EOP_DONE.
0	RSVD	RO	20	0	Reserved

Table 2.29. F2H_CPLT_DESC_COUNT (0x0118)

Field	Name	Access	Width	Default	Description
31:0	CPLT_DESC_CNT	RO	32	0	The number of competed descriptors since the last rising edge of REQUEST bit. Reset to 0 at the subsequent rising edge of REQUEST bit.

2.7.3.3. H2F Descriptor Fetching (0x0200)

Table 2.30. H2F_DESC_ADDR_LOW (0x0200)

Field	Name	Access	Width	Default	Description
31:0	DESC_ADDR_LOW	RW	32	0	Lower 32-bit address of descriptor

Table 2.31. H2F_DESC_ADDR_HIGH (0x0204)

Field	Name	Access	Width	Default	Description
31:0	DESC_ADDR_HIGH	RW	32	0	Upper 32-bit address of descriptor

Table 2.32. H2F_CONT_REMAIN (0x0208)

Field	Name	Access	Width	Default	Description
31:6	RSVD	RO	26	0	Reserved
5:0	CONT_DESC	RW	6	0	The number of contiguous Descriptor from the Descriptor address in 0x00 and 0x04. All 0s mean 64 contiguous descriptors.

2.7.3.4. F2H Descriptor Fetching (0x0300)

Table 2.33. F2H_DESC_ADDR_LOW (0x0300)

Field	Name	Access	Width	Default	Description
31:0	DESC_ADDR_LOW	RW	32	0	Lower 32-bit address of descriptor

Table 2.34. F2H_DESC_ADDR_HIGH (0x0304)

Field	Name	Access	Width	Default	Description
31:0	DESC_ADDR_HIGH	RW	32	0	Upper 32-bit address of descriptor

Table 2.35. F2H_CONT_REMAIN (0x0308)

Field	Name	Access	Width	Default	Description
31:6	RSVD	RO	26	0	Reserved
5:0	CONT_DESC	RW	6	0	The number of contiguous Descriptor from the Descriptor address in 0x00 and 0x04. All 0s mean 64 contiguous descriptors.

2.7.3.5. Interrupt Control and Status (0x0400)

Table 2.36. INT_MODE (0x0400)

Field	Name	Access	Width	Default	Description
31:2	RSVD	RO	30	0	Reserved
1:0	INT_MODE_ENABLE	RO	2	0	Interrupt Mode Enable. 2'b00: Wire interrupt 2'b01: INTx 2'b10: MSI 2'b11: MSI-X Others: Reserved In the current release, only MSI is supported.

Table 2.37. H2F_MSI_VEC (0x0404)

Field	Name	Access	Width	Default	Description
31:5	RSVD	RO	27	0	Reserved
4:0	CHAN0_H2F_MSI_VEC	RW	5	0	Channel 0 H2F MSI Vector 5'd0: MSI Vector 0 5'd1: MSI Vector 1 And so on.

Table 2.38. F2H_MSI_VEC (0x0408)

Field	Name	Access	Width	Default	Description
31:5	RSVD	RO	27	0	Reserved
4:0	CHAN0_F2H_MSI_VEC	RW	5	0	Channel 0 F2H MSI Vector 5'd0: MSI Vector 0 5'd1: MSI Vector 1 And so on.

Table 2.39. USR_MSI_VEC_P1 (0x040C)

Field	Name	Access	Width	Default	Description
31:29	RSVD	RO	3	0	Reserved
28:24	USR3_MSI_VEC	RW	5	0	User 3 MSI Vector 5'd0: MSI Vector 0 5'd1: MSI Vector 1 And so on.
23:21	RSVD	RO	3	0	Reserved
20:16	USR2_MSI_VEC	RW	5	0	User 2 MSI Vector 5'd0: MSI Vector 0 5'd1: MSI Vector 1 And so on.
15:13	RSVD	RO	3	0	Reserved
12:8	USR1_MSI_VEC	RW	5	0	User 1 MSI Vector 5'd0: MSI Vector 0 5'd1: MSI Vector 1 And so on.
7:5	RSVD	RO	3	0	Reserved
4:0	USR0_MSI_VEC	RW	5	0	User 0 MSI Vector 5'd0: MSI Vector 0 5'd1: MSI Vector 1 And so on.

Table 2.40. USR_MSI_VEC_P2 (0x0410)

Field	Name	Access	Width	Default	Description
31:29	RSVD	RO	3	0	Reserved
28:24	USR7_MSI_VEC	RW	5	0	User 7 MSI Vector 5'd0: MSI Vector 0 5'd1: MSI Vector 1 And so on.
23:21	RSVD	RO	3	0	Reserved
20:16	USR6_MSI_VEC	RW	5	0	User 6 MSI Vector 5'd0: MSI Vector 0 5'd1: MSI Vector 1 And so on.
15:13	RSVD	RO	3	0	Reserved
12:8	USR5_MSI_VEC	RW	5	0	User 5 MSI Vector 5'd0: MSI Vector 0 5'd1: MSI Vector 1 And so on.
7:5	RSVD	RO	3	0	Reserved
4:0	USR4_MSI_VEC	RW	5	0	User 4 MSI Vector 5'd0: MSI Vector 0 5'd1: MSI Vector 1 And so on.

Table 2.41. USR_MSI_VEC_P3 (0x0414)

Field	Name	Access	Width	Default	Description
31:29	RSVD	RO	3	0	Reserved
28:24	USR11_MSI_VEC	RW	5	0	User11 MSI Vector 5'd0: MSI Vector 0 5'd1: MSI Vector 1 And so on.
23:21	RSVD	RO	3	0	Reserved
20:16	USR10_MSI_VEC	RW	5	0	User 10 MSI Vector 5'd0: MSI Vector 0 5'd1: MSI Vector 1 And so on.
15:13	RSVD	RO	3	0	Reserved
12:8	USR9_MSI_VEC	RW	5	0	User 9 MSI Vector 5'd0: MSI Vector 0 5'd1: MSI Vector 1 And so on.
7:5	RSVD	RO	3	0	Reserved
4:0	USR8_MSI_VEC	RW	5	0	User 8 MSI Vector 5'd0: MSI Vector 0 5'd1: MSI Vector 1 And so on.

Table 2.42. USR_MSI_VEC_P4 (0x0418)

Field	Name	Access	Width	Default	Description
31:29	RSVD	RO	3	0	Reserved
28:24	USR15_MSI_VEC	RW	5	0	User 15 MSI Vector 5'd0: MSI Vector 0 5'd1: MSI Vector 1 And so on.
23:21	RSVD	RO	3	0	Reserved
20:16	USR14_MSI_VEC	RW	5	0	User 14 MSI Vector 5'd0: MSI Vector 0 5'd1: MSI Vector 1 And so on.
15:13	RSVD	RO	3	0	Reserved
12:8	USR13_MSI_VEC	RW	5	0	User 13 MSI Vector 5'd0: MSI Vector 0 5'd1: MSI Vector 1 And so on.
7:5	RSVD	RO	3	0	Reserved
4:0	USR12_MSI_VEC	RW	5	0	User 12 MSI Vector 5'd0: MSI Vector 0 5'd1: MSI Vector 1 And so on.

2.7.3.6. General Status (0x0500)

Table 2.43. GENERAL_STS (0x0500)

Field	Name	Access	Width	Default	Description
31:6	RSVD	RO	26	0	Reserved
5:3	DMA_SUPPORT	RO	3	0	DMA Support 3'b000: Support F2H and H2F 3'b001: Support F2H Only 3'b010: Support H2F Only 3'b011: Do not support F2H and H2F Others: Reserved
2:0	DMA_TYPE	RO	3	0	DMA Type 3'b000: AXI-MM DMA Others: Reserved

2.7.4. DMA Transaction (AXI-MM)

The data transfer with the DMA support is illustrated in the following figures. Additional registers required by DMA are implemented as well as status registers and interrupt signals, which are discussed in the subsections below.

2.7.4.1. FPGA-to-Host (F2H) Transaction

In F2H transaction, the core reads the data from memory through AXI-MM Address Read and Read Data Channels and one or more Memory Write TLPs are generated and transmitted to the host through PCIe link until the transfer is completed.

Figure 2.6 shows an overall F2H data transfer.

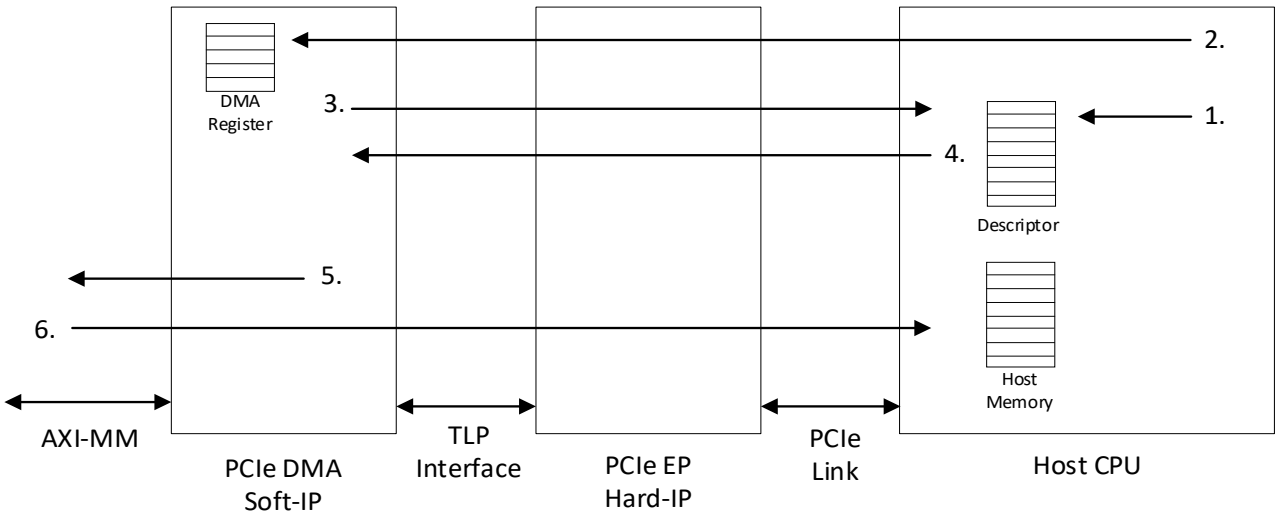

Figure 2.6. F2H Data Transfer

The numbers below are the sequence of F2H flow which corresponds to the numbers in Figure 2.6.

1. Driver forms Descriptors in the Host Memory. The format of Descriptor is available in the DMA Descriptor section.
2. Driver programs DMA registers through Memory Write TLP. It programs F2H Descriptor Fetching (0x0300) field followed by the Request bit in F2H_DMA_CTRL (0x0100) to kick start F2H data transfer.
3. When the Request bit is set, DMA Engine forms Memory Read TLP and transmits it to the Host targeting the address in F2H_DMA_CTRL registers for Descriptor fetching. Block Descriptor fetching can happen if CONT_DESC register is not 0. If CONT_DESC register shows the contiguous Descriptor is beyond MRRS or crossing 4 kB boundary, the descriptor fetching is split into multiple Memory Read TLPs.
4. Host returns Descriptor to the DMA Engine through Completion with Data TLP.

5. When the last received Descriptor has EOP bit = 0, Descriptor fetching through Memory Read moves on to the next Descriptor address.
6. DMA Engine decodes the received Completion with Data TLP to obtain Source Address, Destination Address, and Length information. It triggers memory read through AXI-MM Read Channel to the Source Address. The size of the read data does not cross the 4 kB boundary. Therefore, the memory read may be split to several AXI-MM Read.
7. FPGA application returns the data corresponding to the AXI-MM Read through AXI-MM Read Data Channel. Upon receiving the Read data, DMA Engine forms Memory Write TLP and transmits it to the Host targeting Destination Address. The IP guarantees the transmitted Memory Write TLP does not cross 4 kB boundary.

2.7.4.2. Host-to-FPGA (H2F) Transaction

In H2F transaction, the core transmits Memory Read TLP to the host. Incoming completions are matched with the read request entries and transferred to the specified destination through AXI4-MM Address Write and Write Data channels.

Figure 2.7 shows an overall H2F data transfer.

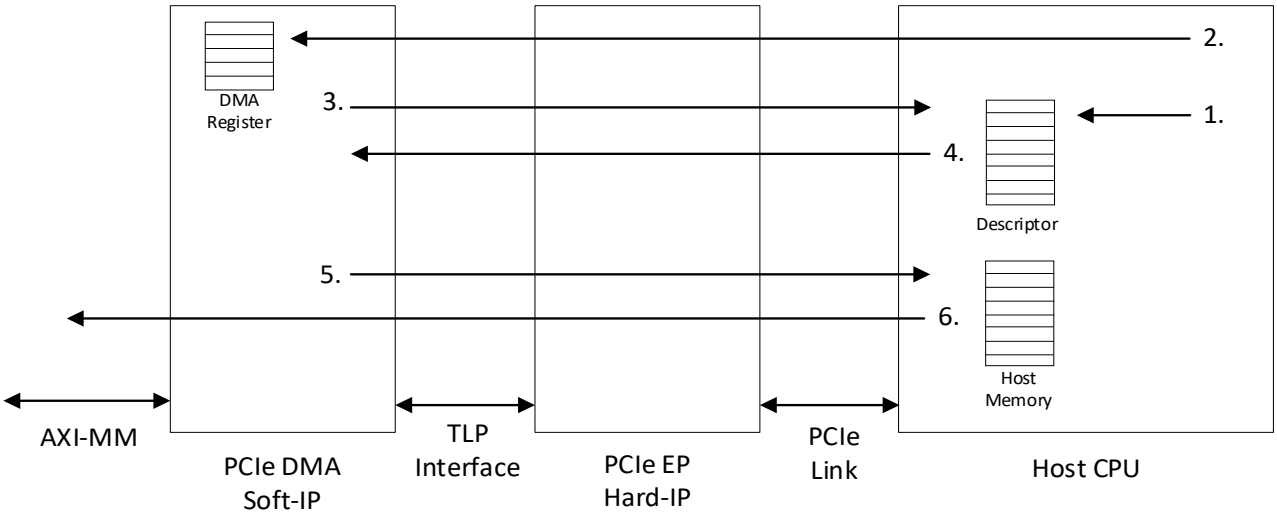

Figure 2.7. H2F Data Transfer

The numbers below are the sequence of H2F flow which is corresponding to the numbers in Figure 2.7.

1. Driver forms Descriptors in the Host Memory. The format of Descriptor is available in the DMA Descriptor section.
2. Driver programs DMA registers through Memory Write TLP. It programs H2F Descriptor Fetching (0x020) field followed by the Request bit in H2F_DMA_CTRL (0x0000) to kick start H2F data transfer.
3. When the Request bit is set, DMA Engine forms Memory Read TLP and transmits it to the Host targeting the address in H2F_DMA_CTRL registers for Descriptor fetching. Block Descriptor fetching can happen if CONT_DESC register is not 0. If CONT_DESC register shows contiguous Descriptor is beyond MRRS or crossing 4 kB boundary, the descriptor fetching is split into multiple Memory Read TLPs.
4. Host returns Descriptor to the DMA Engine through Completion with Data TLP.
5. When the last received Descriptor has EOP bit = 0, the descriptor fetching through Memory Read moves on to the next Descriptor address.
6. DMA Engine decodes the received Completion with Data TLP to obtain Source Address, Destination Address, and Length information. It generates Memory Read TLP and transmits it to the Host targeting the Source Address. If the Length exceeds MRRS or crossing 4kB boundary, the memory read splits into several Memory Read TLPs.
7. Host returns read data as Completion with Data TLP. It may split the read data into several Completion with Data TLPs depending on MPS and RCB. Upon receiving the TLPs, the DMA Engine writes it to the Destination Address through AXI-MM's Write Address Channel and Write Data Channel.

Note: The IP requires WLAST-to-BVALID latency to be within 40 clock cycles to prevent unexpected behaviour in terms of Memory Read TLPs' Tag number.

2.7.4.3. DMA Interrupt

In the current release, only MSI is supported.

Interrupt can be triggered when a DMA data transfer is completed, or an error occurs.

For DMA data transfer, interrupt is triggered when the last byte of data is transferred corresponding to a descriptor chunk (EOP = 1) or any descriptor with INT bit set to 1 (refer to the DMA Descriptor section). Interrupt can also be triggered by erroneous cases (refer to the DMA Registers section).

2.7.5. DMA Performance (AXI-MM)

The data will be available in future release.

2.7.6. DMA With Bridge Mode

DMA with Bridge Mode has an addition AXI (MM or Lite) interface which allows the received MWr and MRd TLP to be converted to AXI-MM/ AXI-Lite Manager Interface.

In the Lattice Radiant user interface, when DMA with Bridge Mode is selected in Configuration Mode, you can configure the BAR number that is associated to Bridge Mode. You can also configure the BAR size through the Radiant user interface.

When a received MWr/MRd TLP targets Bridge Mode BAR, the PCIe DMA IP converts the TLP to AXI-Lite Master Interface.

For read access, Read Data Channel from AXI-Lite is converted to CplD TLP by the IP and transmitted to the Host.

In the current version, only 1 DW MWr/MRd TLP is supported by Bridge mode. In addition, the IP supports DW-aligned address only. The read/write address must end with 0x0, 0x4, 0x8, or 0xC.

2.7.7. DMA User Interrupts

PCIe DMA IP supports up to 16 user interrupts. The number of user interrupts is configurable through the Radiant user interface.

Each user interrupt has a pair of request and acknowledgement signals at the IP interface, such as usr_intr_req_i and usr_intr_ack_o, respectively. When user logic asserts any usr_intr_req_i, the PCIe IP transmits MSI TLP to PCIe link partner. If more than one usr_intr_req_i are asserted, the IP arbitrates these requests with the round-robin arbitration scheme. The interrupt vector (MSI vector) associated with a user interrupt is configured through the USR_INT_VEC registers.

The following are the requirements of usr_intr_req_i and usr_intr_ack_o:

- usr_intr_req_i and usr_intr_ack_o come in a pair. Bit 0 of usr_intr_ack_o is associated with bit 0 of usr_intr_req_i, and so on.
- User application logic must assert usr_intr_req_i when it requires PCIe DMA IP to send interrupt (MSI) to the host.
- usr_intr_req_i and usr_intr_ack_o must comply with a full handshake relationship.

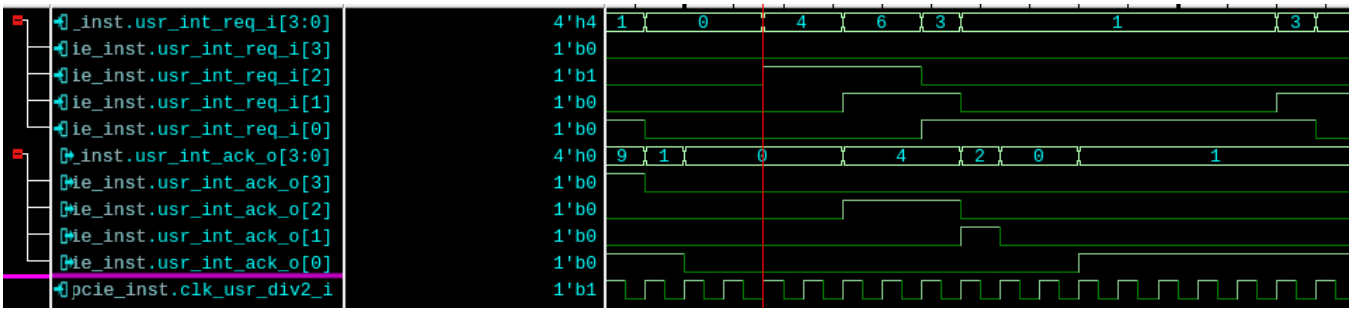

Figure 2.8. User Interrupt Request and User Interrupt ACK Relationship

When multiple user interrupt requests are asserted, the PCIe DMA IP services it in round robin manner starting with Bit 0. The waveform above shows three user interrupt requests. User interrupt Bit 2 is asserted, followed by the user interrupt Bit 1 and user interrupt Bit 0. As the PCIe DMA IP service the interrupt, it ack by asserting Bit 2 followed by Bit 1 and then Bit 0. As the ack is received, interrupt request can be de-asserted.

2.8. Non-DMA Support

2.8.1. Non-DMA Overview

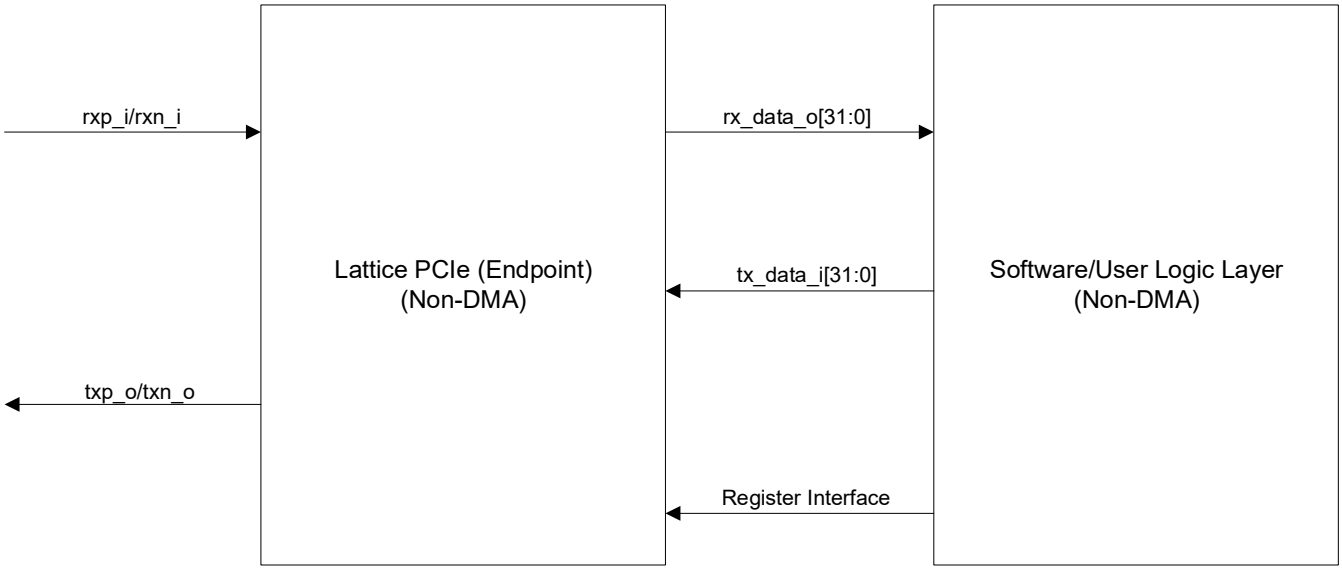

Figure 2.9. Non-DMA Application Data Flow – TLP Interface

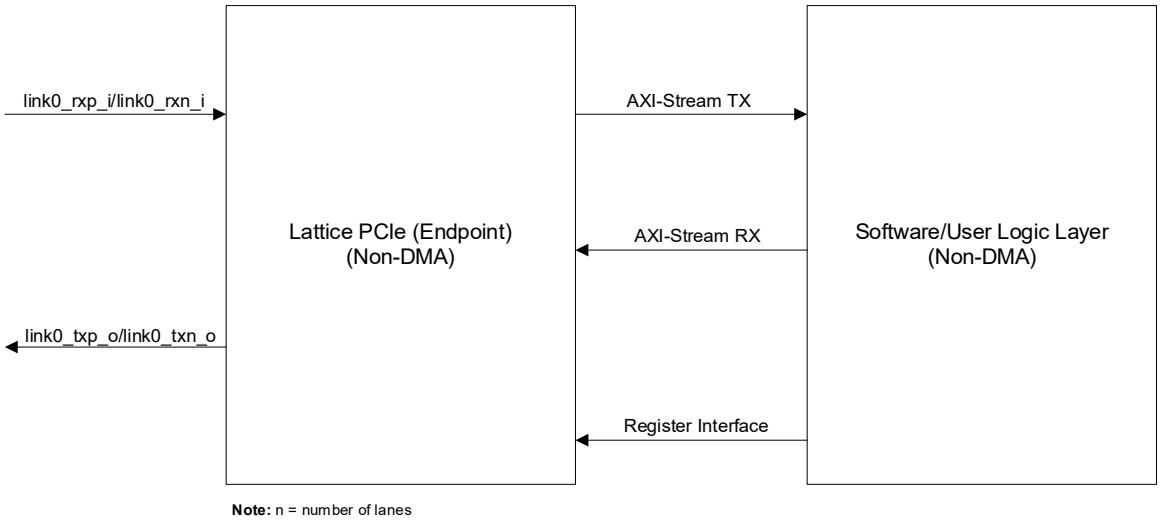

Figure 2.10. Non-DMA Application Data Flow – AXI-Stream Interface

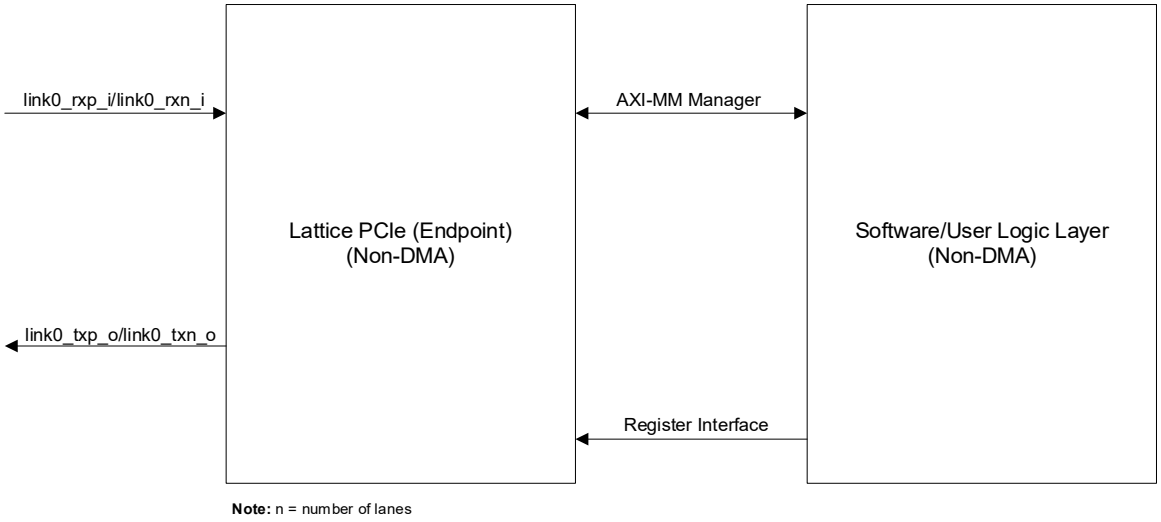

Figure 2.11. Non-DMA Application Data Flow – AXI-MM Interface (Bridge Mode)

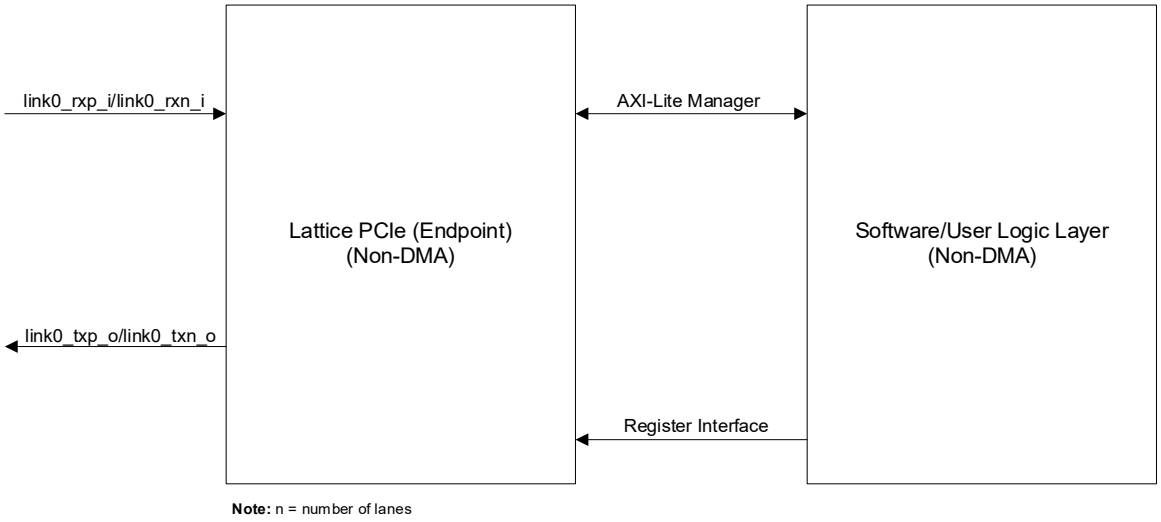

Figure 2.12. Non-DMA Application Data Flow – AXI-Lite Interface (Bridge Mode)

For the non-DMA design, the PCIe EP receives the data through the *rxp_i/rxn_i* serial lines from the Root Complex. The PCIe EP converts the serial data in the form of TLP packets. The TLP packets are sent to the non-DMA application layer through the *rx_data_o* signal. The TLP header info is decoded and the operation is decided whether the data is written or read. For the write operation, the data is written to the RAM present in the application layer. For the read operation, the data is read from the RAM and sends the encoded data to the PCIe EP in the form of TLP packets through the *tx_data_i* signal.

The register interface is enhanced as per the data interface selected in the user interface.

Table 2.44. Register Access for Different Data Interfaces

Data Interface	Register Interface
TLP	LMMI
AXI-Stream	APB
AXI-MM	LMMI
AXI-Lite	LMMI

2.8.2. Non-DMA Write

The PCIe EP sends the header data to the non-DMA application layer through the *rx_data_o* signal. The application layer initially verifies the operation by decoding the header information. Once the write operation is detected, the user data is received along with the valid signal from the PCIe IP. The valid data is stored in the RAM present in the application layer.

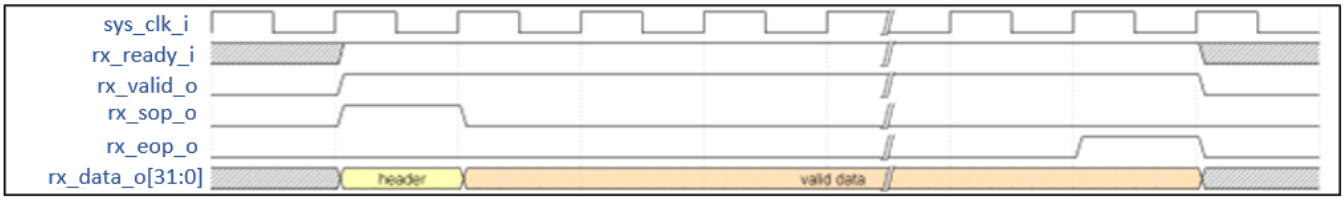

Figure 2.13. Non-DMA Write Operation (TLP Data Interface)

2.8.3. Non-DMA Read

The PCIe EP sends the header data to Non-DMA Application layer through the *rx_data_o* signal. The application layer initially verifies the operation by decoding the header information. Once the operation is detected as read, the application layer waits for the ready signal sent by the PCIe EP. Based on the ready signal and header address, the user data along with the valid signal is send to PCIe EP by the RAM present in the application layer through the *tx_data_i* signal.

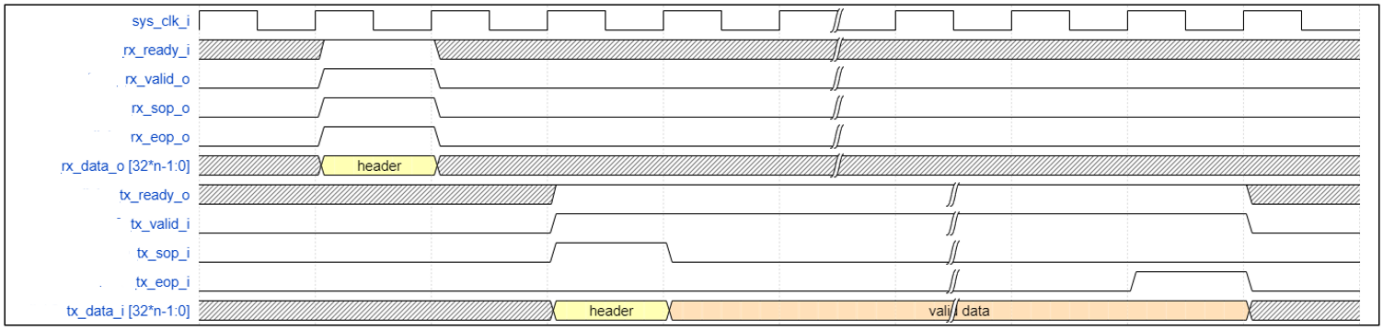

Figure 2.14. Non-DMA Read Operation (TLP Data Interface)

For more details on the TLP write and read data transaction, refer to the Register Interface Conversion section.

2.9. Interrupts

2.9.1. Generation of the Interrupts

The Lattice PCIe Core IP supports the Legacy Interrupts, Messaged Signaled Interrupts (MSI), and MSI-X interrupts.

For each function in the PCIe IP core, the system software configures the function to use MSI-X, MSI, or Legacy Interrupt mode as part of the PCI enumeration process.

The Legacy Interrupt is supported by the PCIe Core to support the backward compatibility by enabling the INTx pins.

To minimize the pin count, the function can generate the inband interrupt message packet to indicate the assertion and de-assertion of an interrupt pin. These are the MSI and MSI-X interrupts. This interrupt mechanism is used to conserve the pins because it does not use separate wires for interrupts.

In this mechanism a single Dword provides the information about the interrupt messages MSI-X/MSI interrupts are signaled using MSI-X/MSI Message TLPs, which you can generate and transmit in the Transmit Interface.

The MSI Interrupt is a posted memory write, which is distinguished from the other memory writes by the addresses they target, which are typically reserved by the system for interrupt delivery. The MSI Capability structure is stored in the Configuration Space and is programmed using Configuration Space accesses.

The MSI-X interrupt is the extended version for the MSI interrupts, supporting a greater number of MSI Vectors and the MSI-X capability structure points to an MSI-X table structure and an MSI-X Pending Bit Array (PBA) structure, which are stored in memory.

Enabling and Disabling of interrupts can be done through PCIe IP Core user interface or through Hard IP core configuration status registers.

Table 2.45 describes the register bits to enable and disable each of the interrupt.

Table 2.45. Base Address and Offset Address to Enable Interrupt

Base Address	Offset Address	Register Bits	Description
0x4000 (Function 0) 0x5000 (Function 1) 0x6000 (Function 2) 0x7000 (Function 3)	0x50	[0]	Support for Legacy Interrupts • 0 – Enable • 1 – Disable
	0xE8	[0]	Support for MSI Interrupts • 0 – Enable • 1 – Disable
	0xF0	[0]	Support for MSI-X Interrupts • 0 – Enable • 1 – Disable

Note: When using MSI interrupt, the Legacy interrupt register must also set to Enable, bit[0]=0 (Enable), either through IP user interface or write through LMMI interface.

2.9.2. Legacy Interrupt

When the legacy interrupts enabled, the PCIe IP core emulates the INTx Interrupts using virtual wire. The term INTx refers to the four legacy interrupts: INTA, INTB, INTC, and INTD.

The legacy_interrupt_i signal is used to generate Legacy interrupts on the PCI Express link. The legacy_interrupt_i has one input for each base (physical) function. When Legacy Interrupt Mode is enabled, legacy_interrupt_i implements one level-sensitive interrupt (INTA, INTB, INTC, or INTD) for each Base Function. Each function's interrupt sources must be logically ORed together and input as legacy_interrupt_i [i] for a given function. Each interrupt source must continue to drive a 1 until it has been serviced and cleared by software at which time it must switch to driving 0. The core ORs together INTA/B/C/D from all functions to create an aggregated INTA/INTB/INTC/INTD. The core monitors high and low transitions on the aggregated INTA/B/C/D and sends an Interrupt assert message on each 0 to 1 transition and an Interrupt de-assert message on each 1 to 0 transition of the aggregated INTA/B/C/D. Transitions, which occur too close together to be independently transmitted, are merged.

The core asserts the legacy_interrupt_o, signal, which is an active high level-based interrupt signal. This interrupt is asserted by the core, whenever an interrupt is generated by the core implemented PCI Express Capability and Advanced Error Reporting Capability. The legacy_interrupt_o should be merged with the legacy_interrupt_i signal to produce any user interrupt signal.

When a function has MSI-X or MSI Interrupt Mode enabled, legacy_interrupt_i is not used for that function.

The selection among the four interrupts can be done through the PCIe IP core user interface or through register interface as described in Table 2.46.

Table 2.46. Legacy Interrupt Register

Base Address	Offset Address	Register Bits	Description
0x4000 (Function 0) 0x5000 (Function 1) 0x6000 (Function 2) 0x7000 (Function 3)	0x50	[9:8]	Selects which Legacy Interrupt to be used: • 0 – INTA • 1 – INTB • 2 – INTC • 3 – INTD

2.9.3. MSI Interrupt

The Lattice PCIe IP core supports 32 MSI interrupts with a feature of enabling and disabling the vector masking. The MSI request can be either 32-bit addressable Memory Write TLP or a 64-bit addressable Memory Write TLP. There are two other registers called Mask Bits Register and Pending Bits Register. Since there is a support for 32 interrupts, the mask bit and pending register are 32-bit length, each bit represents the masking or pending status for each interrupt. The MSI-X capability structure values are programmed through the PCI Express configuration space register.

The address is taken from the Message Address and Message Upper Address fields of the MSI Capability Structure, while the payload is taken from the Message Data field.

The type of MSI TLP sent (32-bit addressable or 64-bit addressable) depends on the value of the Upper Address field in the MSI capability structure. By default, the MSI messages are sent as 32-bit addressable Memory Write TLPs. MSI messages use 64-bit addressable Memory Write TLPs only if the system software programs a non-zero value into the Upper Address register.

The message control register in the MSI capability Structure, disables and enables the various support in the MSI Interrupt.

Figure 2.15 and Figure 2.16 shows the MSI Capability Structure variant.

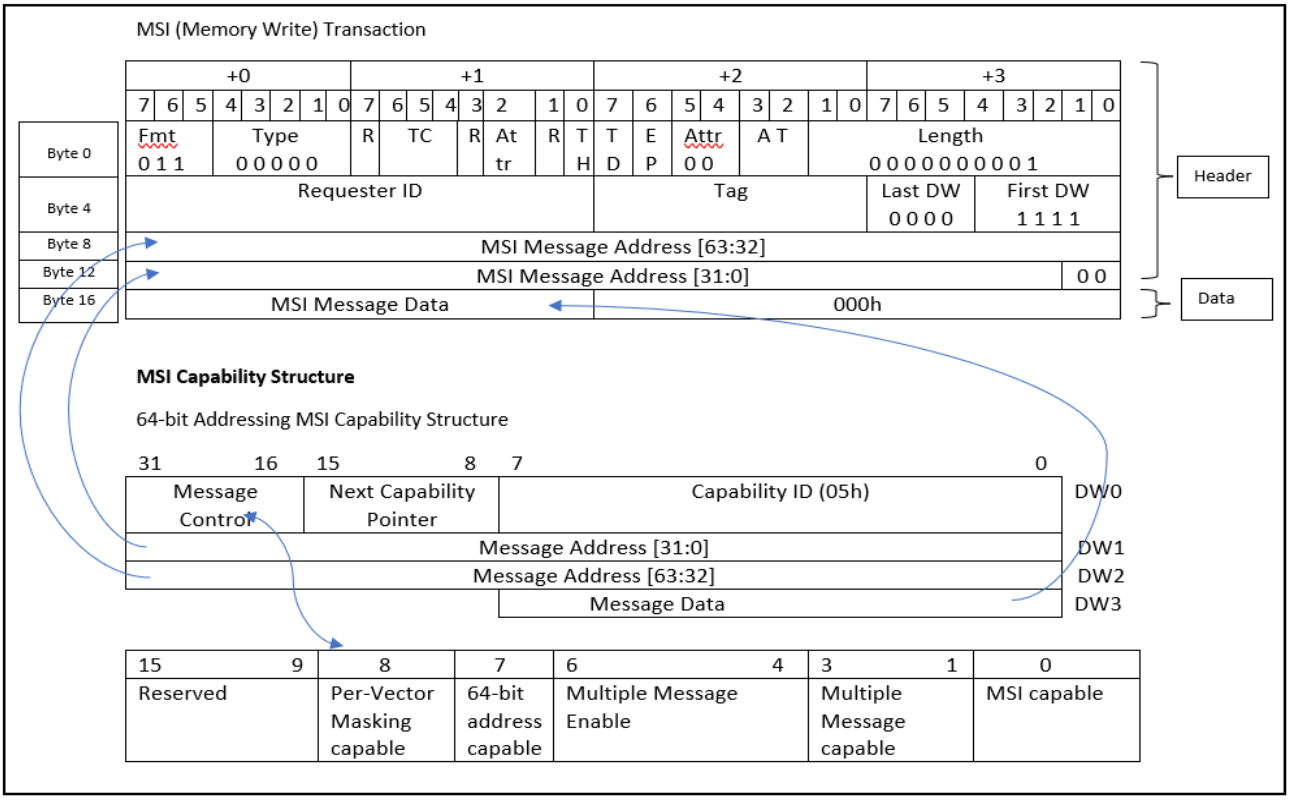

Figure 2.15. 64-bit Addressing MSI Capability Structure

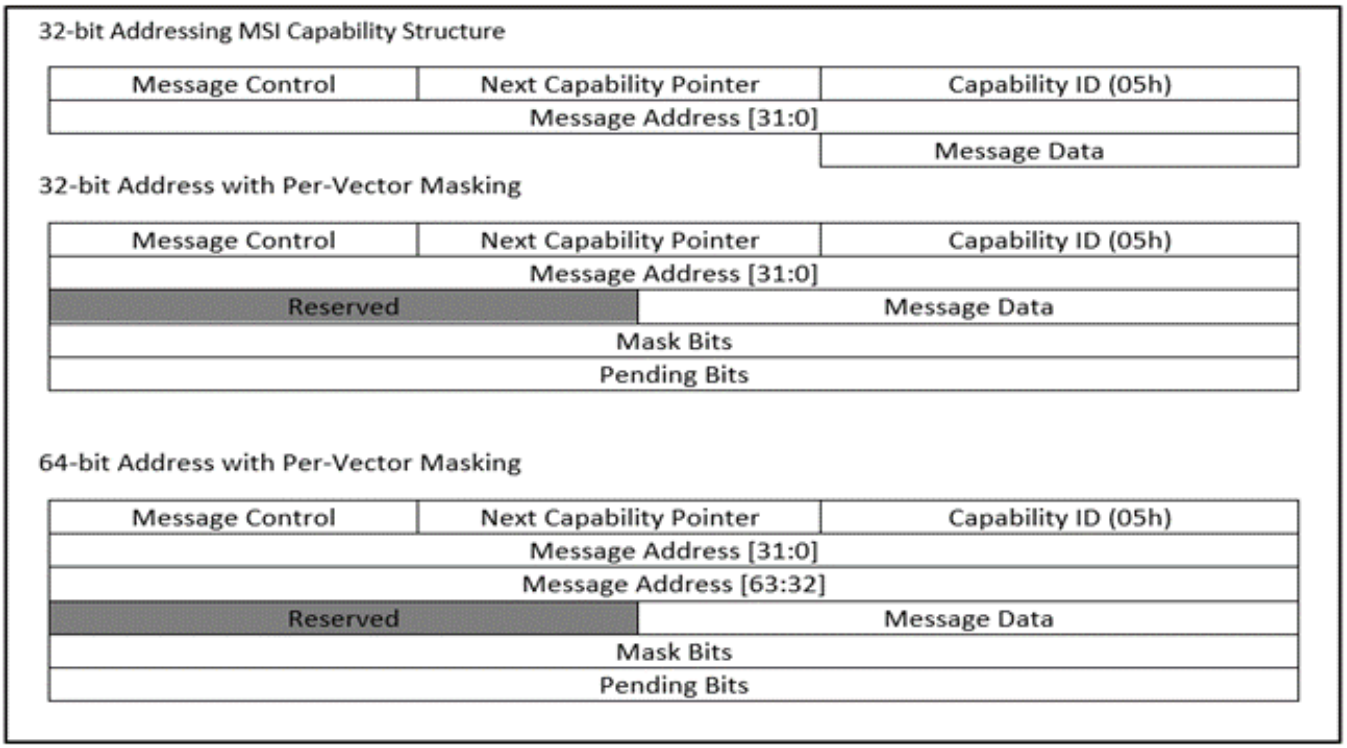

Figure 2.16. 32-bit Addressing MSI Capability

2.9.3.1. MSI Pending Register

The MSI Pending register is used to report MSI Interrupts that are appended in the user design. MSI Pending is a PCIe Configuration Register in the MSI Capability Structure that software uses to obtain status on pending MSI Interrupt vectors.

The MSI Pending register must be written whenever a MSI Interrupt Vector's pending status changes. A 1 must be written to the associated interrupt vector bit when an interrupt becomes pending and a 0 must be written to indicate that the interrupt is no longer pending.

The MSI Pending register must be updated whenever the status of your pending MSI interrupts changes. If MSI interrupts are not used, writing to the MSI Pending Register is not needed.

2.9.4. MSI-X Interrupt

2.9.4.1. MSI-X Capability Structure variant

MSI-X allows the support of large number of vectors with independent message data and address for each vector compared to the MSI Interrupts. It can support up-to 2048 vectors per function. The MSI-X Capability Structure points to an MSI-X table structure and an MSI-X Pending Bit Array (PBA) structure, which are stored in memory. In MSI-X interrupt the vector information is present in the memory location pointed by the Table Base address Indicator Register (BIR).

Figure 2.17 shows the MSI-X capability structure. The MSI-X interrupt configuration is done by the PCIe Configuration Space Registers.

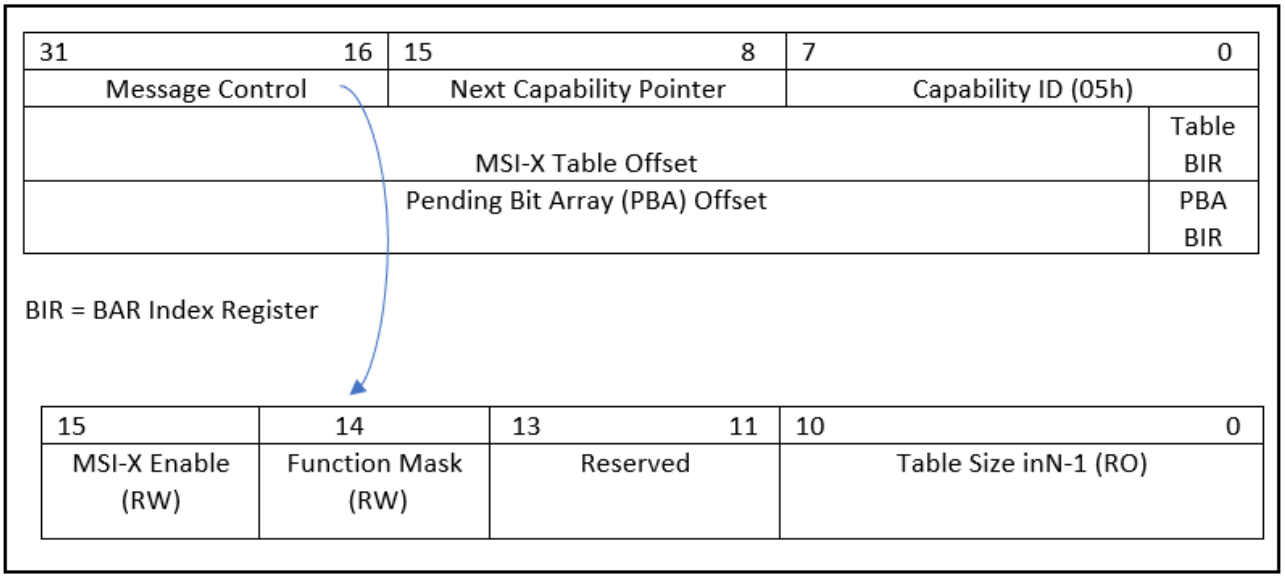

Figure 2.17. MSI-X Capability Structure Variant

The description of each bit in the Message controlled are explained in the section of the PCIe Configuration Space Register configuration for MSI-X Capability Structure. The MSI-X Capability Structure variant contains information about the MSI-X Table and the PBA structure, information such as pointers to the bases of the MSI-X Table and the PBA structure. The Table BIR in the MSI-X Capability Structure includes information about the BAR location that contains the MSI-X table.

2.9.4.2. MSI-X Table

The MSI-X table is an array of vectors and addresses. The MSI-X Table contains four Dwords. Each entry in the MSI-X table represents one vector. The DW0 and DW1 supply a unique 64-bit address for that vector and DW2 is the 32-bit data pattern for it. The DW3 is the mask bit for the vector and contains only 1 bit at present.

DW3	DW2	DW1	DW0	
Vector Control	Message Data	Upper Address	Lower Address	Entry 0
Vector Control	Message Data	Upper Address	Lower Address	Entry 1
Vector Control	Message Data	Upper Address	Lower Address	Entry 2
….	….	….	….	
….	….	….	….	
Vector Control	Message Data	Upper Address	Lower Address	Entry N-1

Figure 2.18. MSI-X Table Entries

2.9.4.3. Pending Bit Array

The Pending Bit Array (PBA) is located within the memory address. This can use the same MSI-X Table BIR value (that is the same BAR or a different BAR). The PBA can use either qword (64-bit) or Dword (32-bit) accesses. The PBA table contains the pending bit information for each interrupt used. Same as MSI interrupts, if the event that the interrupt triggers and if its mask bit is set, the MSI-X transaction is not sent and the corresponding pending bit is set. If the interrupt vector is unmasked and if the pending bit is still set, that interrupt is generated.

DW1 DW0	
Pending Bits 0 - 63	QW 0
Pending Bits 64 - 128	QW 1
Pending Bits 128 - 191	QW 2
……	
Pending Bits	QW (N-1)/64

Figure 2.19. Pending Bit Array

2.9.4.4. MSI-X Interrupts Operation

- When the MSI-X interrupts are supported, you need to mention the size and location of the MSI-X Table and Pending Bit Array (PBA) through the PCIe CSR and the MSI-X table and the PBA structure must be implemented at the application layer.
- When the MSI-X Interrupts are generated, it uses the contents of the MSI-X Table (Address and Data) and generate a Memory Write through the TLP interface.
- The Host reads the message control register to determine the MSI-X Table size. The maximum entry in the table is 2048 entries. The BAR mentioned in the table BIR can access the MSI-X table.
- The host sets up the MSI-X table by programming the address, data, and the mask bits for each entry in the table.
- When the application generates the interrupt, it reads the MSI-X table information and generates a MWR TLP data and the corresponding bits in the PBA is set.
- The generated TLP is sent to the corresponding address along with the data.
- When the MSI-X interrupt is sent, the application can clear the associated PBA bits.

2.10. PCIe Endpoint Core Buffers

The Lattice PCIe x1 IP Core contains three large RAM buffers:

- Transmit Buffer for transmitting TLPs.
- Receive Buffer for receiving TLPs.
- Replay Buffer for holding TLPs that were transmitted until positive acknowledgement of receipt is received.

The size of the Transmit Buffer, Receive Buffer, and Replay Buffer and the size of the corresponding buffers in the remote PCI Express Device have a fundamental impact on the throughput performance of the PCI Express link.

To achieve the highest throughputs, the buffers for both devices in the PCI Express link must be large enough that they can still accept more data while the oldest data begins to be freed from the buffer. If a buffer is too small, then the link stalls until the buffer has enough space to continue. The buffers must be large enough to overcome the expected latencies or the throughput is affected.

2.10.1. PCI Express Credits

The Flow Control DLLPs communicates the available buffer space in units of Header and Data Credits as defined in the PCI Express Specification. The amount of space required by a Header is 12-20 bytes or (3-5 DWORDs with 1 DWORD == 4 bytes). Each Header Credit represents the capability to store a maximum size packet header, which includes all the transaction control information (address, byte enables, and requester ID) and an optional End to End CRC (ECRC). Each Data Credit represents 16 bytes (4 DWORDs) of data payload. A transaction cannot be transmitted unless there is at least 1 header credit and enough data credits for the packet payload available in the remote device's Receive Buffer.

Credits are further divided into three categories for each of the main types of traffic:

- Posted (memory write requests and messages)
- Non-Posted (all reads, Configuration and I/O writes)
- Completion (responses to Non-Posted Requests) credit categories.

Each type of traffic must obey the PCI Express transaction ordering rules and is stored in its own buffer area. The Credit categories are annotated as:

- PH – Posted Request Header Credits
- PD – Posted Request Data Payload Credits
- NH – Non-Posted Request Header Credits
- ND – Non-Posted Request Data Payload Credits
- CH – Completion Header Credits
- CD – Completion Data Payload Credits

The PCI Express is inherently high-latency due to the serial nature of the protocol (clock rate matching and lane-lane de-skewing) and due to the latency induced by requiring packets to be fully received and robustly checked for errors before forwarding them for higher-level processing.

To achieve the best throughputs, both the Lattice PCIe x1 IP Core and the remote PCI Express device must be designed with a suitable number of credits and the capability to overlap transactions to bury the transaction latency.

The Lattice PCIe x1 IP Core Transmit, Receive, and Replay buffers are delivered with sufficient size to overcome the latencies of typical open system components.

2.10.2. Max Payload Size

The maximum payload size of any given packet is limited by the Max Payload Size field of the Device Control Configuration Register. The PCI Express Specification defines 128, 256, 512, 1024, 2048, and 4096-byte payload sizes. The maximum payload size that a device can support is limited by the size of its posted and completion TLP buffers. The Transmit Buffer and Receive Buffer Posted and Completion TLP storage and the Replay Buffer TLP storage needs to be able to hold at least four Max Payload Size TLPs to be reasonably efficient. Each device advertises the maximum payload size that it can support, and the OS/BIOS configures the devices in a link to use the lowest common maximum payload size. Thus, it is not advantageous to support a greater maximum payload size than the devices with which one is communicating.

The higher the TLP payload size, the lower the TLP header and framing overhead is compared to the data. Above 512-byte Max Payload Size the incremental throughput benefit of higher payload sizes is small and the design area and latency for using these larger payloads is expensive. Thus, it is generally recommended to design for ≤ 512 Max Payload Size.

The Lattice PCIe x1 IP Core supports up to 512 Bytes Max Payload Size and the internal buffers can hold about 3x of the max payload size. However, given that typical PCIe devices are currently available to communicate with support 256-byte maximum payloads, supporting greater than this amount is not likely to result in better performance and consumes more memory/logic resources.

2.11. Hard IP Interface

2.11.1. PHY Interface

The Link Layer is used in conjunction with a PCI Express PHY to implement a complete Lattice PCIe x1 IP Core PCI Express implementation. The PHY implements the high-speed serial and analog functions required to support PCI Express while the Link Layer implements most of the digital logic as well as the higher levels of the PCI Express protocol.

The PIPE PHY Interface that connects the Link Layer and PHY is not shown since the interface is only internal and is not visible to you.

The physical interface includes the differential receive and transmit signals along with the differential reference clock to the PCIe.

2.11.2. TLP TX/RX Interface

The Lattice PCIe core implements a complete PCI Express implementation including Physical, Data Link, and Transaction Layer functionality.

You transmit the PCI Express TLPs on the PCI Express link through the transmit interface. Also, you receive the PCI Express TLPs from the PCI Express link through the receive interface.

The PCIe core uses the Transaction Layer Interface as data interface to transmit/receive the data in the form of TLP Packets. Each TLP packet is a collection of a group of TLP frames, and each frame consists of 4DW (4X32 bit) data. A minimum of 4DW data is sent through a TLP. The Lattice PCIe core lane can access 32-bit (4 bytes) of data at a time. To transmit a single TLP frame, x1 configuration takes a duration of 4 clock cycles.

All TLPs on the Transaction Layer Receive and Transmit Interfaces, which are processed through rx_data_o/tx_data_i port(s) and must be transmitted in the TLP format. The rx_sel_o and rx_cmd_data_o ports provide useful information about the TLP through receive interface to enable you to determine the destination of the packet (BAR and tag), traffic class, and whether it is a write or a read without having to read and parse the TLP. This allows you to optimize the code to reduce latency and relieves necessity for you to decode the TLP header to determine the packet's destination.

2.11.2.1. TLP Header Description

The Lattice PCIe uses 3DW header for memory transactions to transfer the data in the form of TLP packets. The description of each field is described as shown in Figure 2.20.

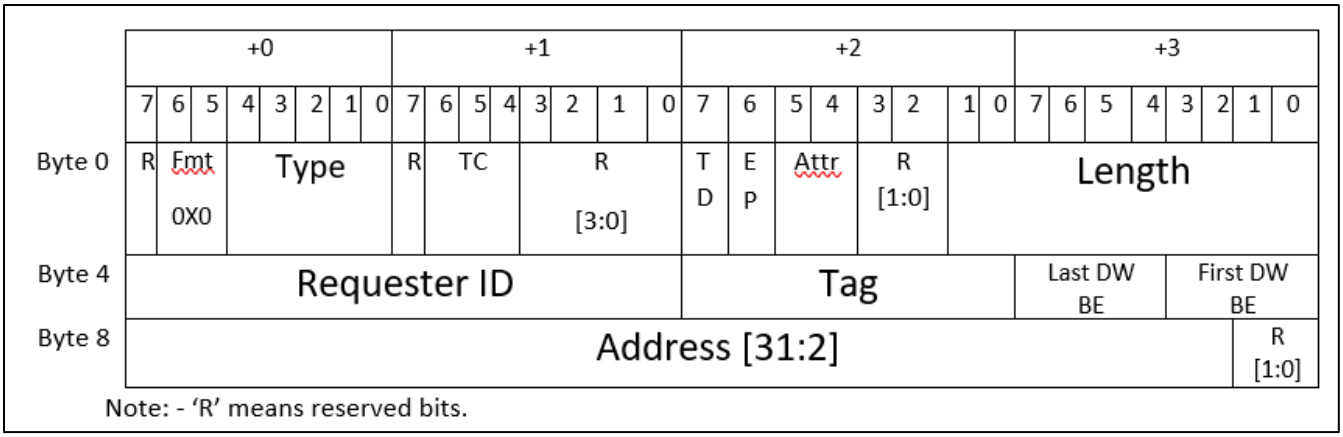

Figure 2.20. TLP Memory Request Header

Table 2.47 lists the description of each field.

Table 2.47. TLP Header Field

Field Name (with Size)	Header Byte/Bit	Function
Fmt [1:0] (Format)	Byte 0 Bit 6:5	Packet Formats: 00b = Memory Read 10b = Memory Write
Type [4:0]	Byte 0 Bit 4:0	TLP packet Type field: 00000b = Memory Read or Write 00001b = Memory Read Locked
TC [2:0] (Traffic Class)	Byte 1 Bit 6:4	These bits encode the traffic class to be applied to a Request and to any associated completion. 000b = Traffic Class 0 (Default)
TD (TLP Digest)	Byte 2 Bit 7	If 1, the optional TLP Digest field is included with this TLP.
EP (Poisoned Data)	Byte 2 Bit 6	If 1, the data accompanying this packet is considered to have an error although the transaction is allowed to complete normally.
Attr [1:0] (Attribute)	Byte 2 Bit5:4	Bit 5 = Relaxed ordering. When set = 1, PCI-X relaxed ordering is enabled for this TLP. Otherwise, strict PCI ordering is used. Bit 4 = No Snoop. If 1, system hardware is not required to cause processor cache snoop for coherency for this TLP. Otherwise, cache snooping is required.
Length [9:0]	Byte 2 Bit 1:0 Byte 3 Bit 7:0	TLP data payload transfer size, in DW. Maximum size is 1024 DW (4 kB).
Requester ID [15:0]	Byte 4 Bit 7:0 Byte 5 Bit 7:0	Identifies a requester's return address for a completion: Byte 4, 7:0 = Bus Number Byte 5, 7:3 = Device Number Byte 5, 2:0 = Function Number
Tag [7:0]	Byte 6 Bit 7:0	These identify each outstanding request issued by the Requester. By default, only bit 4:0 is used, allowing up to 32 requests to be in progress at a time. If the Extended Tag bit in the Control Register is set, then all 8 bits may be used (256 tags).

Field Name (with Size)	Header Byte/Bit	Function
Last DW BE [3:0] (Last DW Byte Enables)	Byte 7 Bit 7:4	These qualify bytes within the last DW of data transferred.
First DW BE [3:0] (First DW Byte Enables)	Byte 7 Bit 3:0	These qualify bytes within the first DW of the data payload.
Address [31:2]	Byte 8 Bit 7:0 Byte 9 Bit 7:0 Byte 10 Bit 7:0 Byte 11 Bit 7:2	The 32 bits start address for the memory transfer are used. The lower two bits of the address are reserved, forcing a DW-aligned start address.

2.11.2.2. TLP Transmit Interface

The Transmit Interface is the mechanism with which you transmit PCI Express TLPs over the PCI Express bus. You can send a complete TLP comprised of 3DW packet header, data payload, and optionally a TLP Digest. The core Data Link Layer adds the necessary framing (STP/END/EDB), sequence number, Link CRC (LCRC), and optionally computes and appends the ECRC (TLP Digest) when ECRC is not already present in the TLP.

You can transmit TLPs as completion packets in response to non-posted transaction packets sent by the Lattice PCIe IP core. If the remote device does not have sufficient space in the receive buffer for transmit TLPs, the Lattice PCIe IP core pauses the TLP transmission until space becomes available.

The Transmit Interface includes the option to nullify TLPs (instruct the Receiver to discard the TLP) to support you to cancel TLP transmissions when errors are detected after the TLP transmission has started. Nullified TLPs that target internal core resources (Root Port Configuration Registers and Power Management Messages) are discarded without affecting the internal core resources. Nullified TLPs that do not hit internal resources are discarded.

Transmit Credit Interface

The Transmit Credit Interface provides the means for flow control of non-posted transmit transactions between you and the core transmit buffer. This is important for allowing Posted and Completion TLPs to continue to make progress when non-posted TLPs are blocked (which can be necessary in some cases to avoid potential deadlock conditions). The amount of non-posted TLP storage in the core transmit buffer is communicated on the transmit credit Interface. You are expected to use this interface to limit simultaneously outstanding TLP transmission of non-posted TLPs, to the amount of non-posted TLPs that the core can absorb into the non-posted transmit buffer.

When the core Transaction Layer for the link is ready to accept TLP transmissions, the core asserts tx_credit init_o == 1 for one clock cycle and indicates the non-posted TLP Header storage capacity (NH) of the transmit buffer on tx_credit_nh_o[11:0] on the same cycle. You are expected to keep and initialize the non-posted TLP Header capacity (NH) available transmit credit counters on tx_credit init_o==1.

When a non-posted TLP is pending for transmission, you must check the currently available NH credit count for the associated link and hold the transmission until enough NH credits are available to transmit the TLP. Once the TLP is committed for transmit, the amount of NH credits required by the TLP are decremented from the NH credit count. The core forwards transmitted TLPs from the transmit buffer and thus makes room for new TLPs, the core asserts tx_credit_return_o==1 for one clock cycle and places the number of NH credits being returned on tx_credit_nh_o[11:0].

In this manner, you can manage sending only enough non-posted TLPs that the core can hold in its Transmit Buffer. This allows you to know when non-posted TLPs are blocked and thus sends posted and/or completed TLPs instead. This is important for avoiding deadlocks and keeps non-posted TLP blockage from reducing posted and completion throughput. When the core receives more non-posted TLPs than the core can store in its non-posted TLP transmit storage, the core pauses the TLP transmission rather than allow an overflow to occur. Thus, if you do not wish to use the Transmit Credit Interface, you may ignore this interface provided you are willing to permit blocked non-posted TLPs from also blocking following posted and completion TLPs.

Note that core/link partner transmit TLP flow control is not managed through this interface, the core manages transmit flow control between the core and the PCIe link partner Receive Buffer without user intervention.

Transmit Interface Example Transactions

As mentioned, the TLP data interface option is made available when non-DMA support is enabled through PCIe user interface. The following are the examples of the memory read transactions that you need to send in completion to the read requests.

In case of memory read transactions, the Lattice PCIe IP core sends the header packet, which contains information about the address and size of data that you need to send in completion to the received packet. You need to send the completion packet with the header followed by the data when the link0_tx_ready_o signal from the PCIe is high as the data you sent is validated only when PCIe is ready. Based on the number of lanes used, the packet header and data are transmitted accordingly as shown in the below figures for four lanes, two lanes, and one lane respectively.

Note that the header packet has a unknown(trash) value in MSB, because the TLP header is of 3DW (12 bytes) whereas the TLP frame is of 4DW (32 bytes) size. To send the complete TLP, some garbage data in Dword is appended with header data, which can be ignored.

The following are the notations used in the figures:

- *N* – size of the data packet in Dwords
- *data* – 1 Dword of unknown data attached in case of 3DW TLP Header
- H0, H1, and H2 – Header information
- D0, D1,…,D(N-1),D(N) – User data

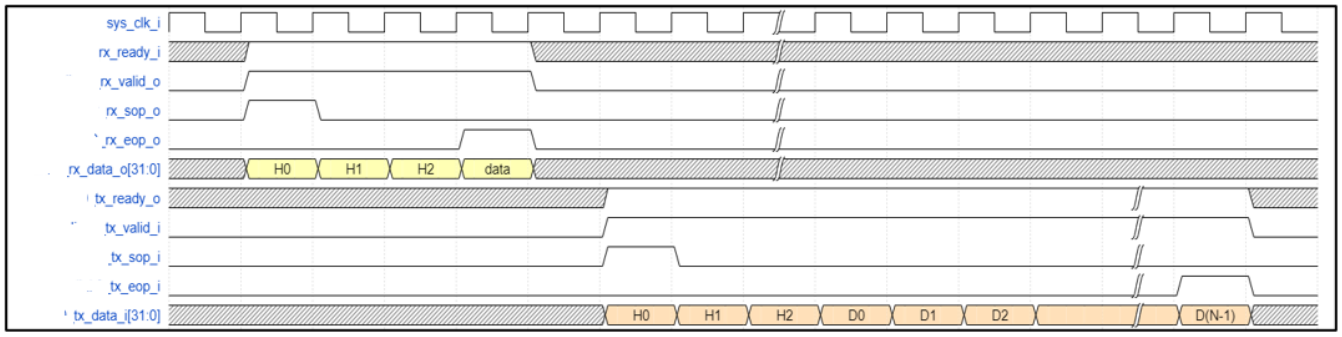

Figure 2.21. TLP Memory Read Operation (x1 Lane)

Figure 2.21 and Figure 2.22 show the TLP transaction according to the tx_ready_o behaviour based on the minimum timing of tx_ready_o:

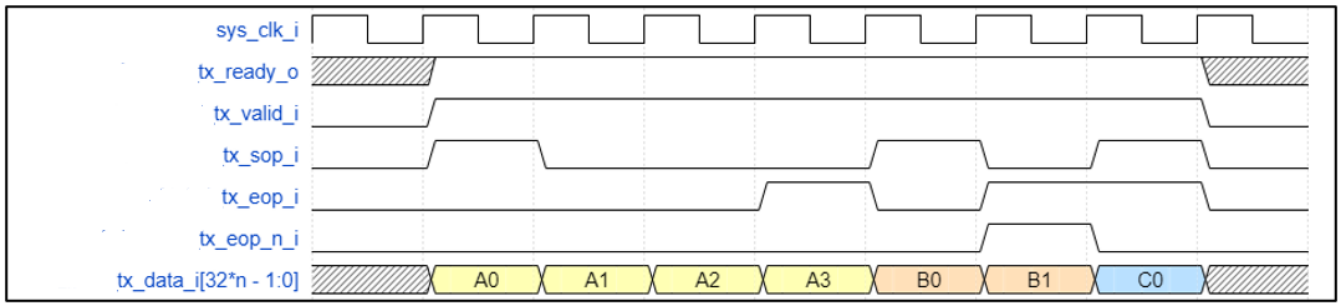

Figure 2.22. Minimum tx_ready_o Timing Diagram

Transaction A begins on cycle 2 with the assertion of tx_sop_i and ends on cycle 5 with the assertion of tx_eop_i==tx_valid_i==tx_ready_o==1. The packet transfers with minimum timing with tx_valid_i==tx_ready_o==1 on cycles 2-5.

Transaction B begins immediately after Transaction A on cycle 6 with the assertion of tx_sop_i and ends on cycle 7 with the assertion of tx_eop_i==tx_valid_i==tx_ready_o==1. The packet transfers with minimum timing with tx_valid_i==tx_ready_o==1 on cycles 6 to 7. Transaction B is nullified (dropped) during packet forwarding on cycle 7 because of the following conditions: tx_eop_i_n==1 happens when tx_eop_i==1.

Transaction C begins immediately after Transaction B on cycle 8 with the assertion of tx_sop_i and ends on the same cycle with the assertion of tx_eop_i==tx_valid_i==tx_ready_o==1. The packet transfers with minimum timing with tx_valid_i==tx_ready_o==1 on cycle 8 considering the wait state timing of tx_ready_o:

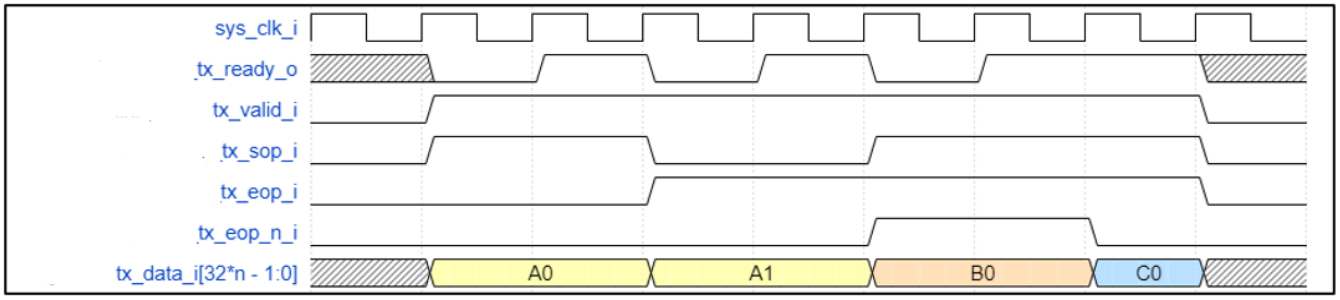

Figure 2.23. Wait State of tx_ready_o Timing Diagram

Transaction A begins on cycle 2 with the assertion of tx_sop_i and ends on cycle 5 with the assertion of tx_eop_i==tx_valid_i==tx_ready_o==1. The packet transfers only on cycles 3 and 5 when tx_valid_i==tx_ready_o==1.

Transaction B begins immediately after Transaction A on cycle 6 with the assertion of tx_sop_i and ends on cycle 7 with the assertion of tx_eop_i==tx_valid_i==tx_ready_o==1. The packet transfers only on cyle 7 when tx_valid_i==tx_ready_o==1. Transaction B is nullified (dropped) during packet forwarding on cycle 7 because of the following conditions: tx_eop_i_n==1 happens when tx_eop_i==1.

Transaction C begins immediately after Transaction B on cycle 8 with the assertion of tx_sop_i and ends on the same cycle with the assertion of tx_eop_i==tx_valid_i==tx_ready_o==1. The packet transfers with minimum timing (no wait states) with tx_valid_i==tx_ready_o==1 on cycle 8.

Transmit Interface Considerations

The following considerations are provided to simplify logic using the Transmit Interface and to address common problems, which must be avoided:

- For each TLP that you transmit, the core adds a minimum of 2-bytes of STP/END/EDB framing, a 2-byte Sequence Number, and a 4-byte Link CRC for a total of 8 bytes (64-bits). These additional 8 bytes, which the core transmits but do not appear on tx_data_i, allows you the flexibility of not using every clock cycle on the Transmit Interface. This flexibility can be useful to simplify user logic and improve design timing closure.
- Completions, which are transmitted in response to a previously received non-posted request, must reflect the Traffic Class, Requester ID, Tag, and Attributes of the original request. While most of these are obvious, it may not be obvious to reflect the attributes, and this is known to cause problems on some systems.
- When the link trains at less than full width or speed, tx_ready_o is gaped in relation to the number of lanes being used and the number of lanes available in the core. You must remember to include a simulation case which forces the link into lower than full-width and/or speed to test that the logic properly handles the gaping of tx_ready_o and the corresponding lower data transfer rate in this case.
- While TLPs are transmitted over PCI Express, these are placed into a replay buffer in case the TLPs need to be replayed due to transmission errors. The core negotiates the replay process in conjunction with the remote PCI Express Device and does not require any user intervention. You can monitor the frequency of replays, if desired, by monitoring the appropriate error status registers.
- The Lattice PCIe core interface is designed to support high throughput applications. Small interruptions in transmissions occurs, however, as the core periodically needs to transmit link management DLLPs and SKP Ordered Sets and may also need to transmit error messages, configuration write/read completions, and interrupt TLPs.
- The Lattice PCIe core handles all Data Link Layer functionality for you and handles most of the error cases for you as well. To accomplish these functions, the core occasionally delays your Local Transmit Interface requests while it completes its own TLP transmissions for these purposes. All the core's TLP transmissions are short, so it delays your request for only a few clock cycles. The core transmits DLLPs used for link maintenance, TLP messages to communicate errors, interrupt TLPs, and completions to notify the system of malformed or un-routable TLP requests.

If the user TLP transmit requests are delayed for extended periods of time, this may be due to insufficient link partner receive buffer space or local replay buffer space or due to the link having to wake from a lower power state or recover from an error before transmission can occur.

2.11.2.3. TLP Receive Interface

The Receive Interface is the mechanism with which receives the PCI Express TLPs from the PCI Express link partner. You receive complete Transaction Layer Packets (TLPs) comprised of a three DWORD TLP header, data payload (if present), and TLP Digest (ECRC, if present).

The TLPs, which were received without errors and were not nullified, are presented on the receive interface. Therefore, the user logic only needs to handle valid received TLPs.

The PCIe core transmits the TLPs only after considering the following checks:

- The core checks received TLPs for transmission errors (Sequence Number or LCRC error) and negotiates replay of TLPs with the link partner as required.
- The core discards TLPs which are nullified by the link partner during transmission (TLP is received without transmission errors and with EDB instead of END framing).
- The core checks received TLPs which were received without transmission errors and without being nullified for Malformed TLP due to length and content errors.
  - If the core determines that a received TLP is malformed due to length (TLP length calculated from the received TLP Header Format and Type, Length, and TLP Digest does not match the received TLP length), the core discards the TLP and report the error.
  - If the TLP fails to hit an enabled resource or is malformed due to its content (invalid Traffic Class, invalid Format and Type, and invalid Byte Enables), the core discards the TLP and reports the error.

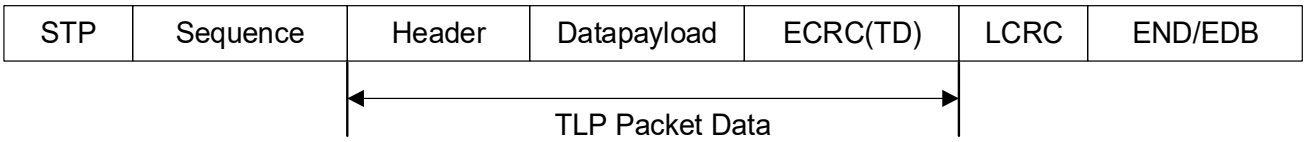

Figure 2.24. TLP Packet Formation by the Lattice PCIe IP Core

If the TLP passes all the above checks, it is considered a valid TLP and is forwarded to the receive interface for the user's logic to consume. The core strips the Physical Layer framing (STP/END/EDB) and Data Link Layer Sequence Number and Link CRC (LCRC) before presenting the TLPs to you on the Receive Interface. The core does not strip the received TLP ECRC (if present) as some user designs require forwarding the ECRC either to transmit the TLP out another PCIe port. The ECRC value is also checked at a later point in the user's data path to continue the ECRC error detection protection for a larger portion of the receive data path. If an ECRC is present in the TLP, the core checks the validity of the received ECRC and reports detected ECRC errors on the receive interface.

The core also decodes received TLPs against its Configuration Registers and provides the transaction decode information on the Receive Interface such that the TLP can be directed to the appropriate destination without the need for you to parse the TLP until its destination. For example, if the received TLP is an I/O or Memory write or read request, the Base Address Register (BAR) resource that is hit is indicated and if the TLP is a completion, the TLP's tag field is provided. The core also provides additional useful transaction attributes.

Receive Credit Interface

The Receive Credit Interface provides the means for flow control of non-posted receive transactions between the core receive buffer and user receive TLP logic. This is important for allowing Posted and Completion TLPs to continue to make progress when non-posted TLPs are blocked (which is necessary in some cases to avoid potential deadlock conditions). The amount of non-posted TLP storage in the user's design is communicated on the Receive Credit Interface. The core uses this interface to limit the simultaneously outstanding receive non-posted TLPs to the amount of non-posted TLPs that the user design advertises that it can absorb into the non-posted receive buffer.

When you are ready to accept non-posted TLP reception, assert the rx_credit init_i == 1 for one clock cycle and the non-posted TLP header storage capacity of the user design is indicated through rx_credit_nh_i[11:0] on the same clock cycle. Holding off credit initialization for an extended period can cause received non-posted TLP transactions to timeout in the source component which may cause to serious errors.

The core limits simultaneous outstanding non-posted receive TLPs on the receive interface to ensure no more than the initialized NH credits are simultaneously outstanding to user receive TLP logic.

Once the received non-posted TLPs are processed/forwarded such that more room is available to receive new non-posted TLPs, assert the rx_credit_return_i==1 for one clock cycle and place the number of NH credits being returned on rx_credit_nh_i[11:0]. In this manner, you can limit the outstanding core receive TLPs to the user design. This permits the core to know when non-posted TLPs are blocked and thus send posted and/or completion TLPs to the user design instead. This is important for avoiding deadlocks and keeps non-posted TLP blockage from reducing posted and completion throughput.

Note that the link partner/core receive TLP flow control is not managed through this interface; the core manages receive buffer flow control between itself and the PCIe link partner transmit gating function without user intervention.

Receive Interface Example Transactions

The Lattice PCIe core sends the data in the form of TLP packets when non-DMA option is enabled. The receive interface presents the TLP data through link0_rx_data_o signal. The data is validated only when link0_rx_valid_o signal is high and you are ready (for example, link0_rx_ready_i must be high to access the data sent by the core). As the Lattice PCIe core transmits TLP packet, which consists of 3DW header along with data (in TLP frames), the last DW of TLP packet is sent as trash value(X) to ensure the complete TLP is transmitted.

The timing diagrams below show the receive interface behavior when the PCIe core receives a Memory Write TLP.

The following are the notations used in the figures:

- *N* – size of the data packet in Dwords
- *data* – 1 Dword of unknown data attached in case of completion of TLP packet
- H0,H1, and H2 – Header information
- D0,D1,…,D(N-1) – Write data

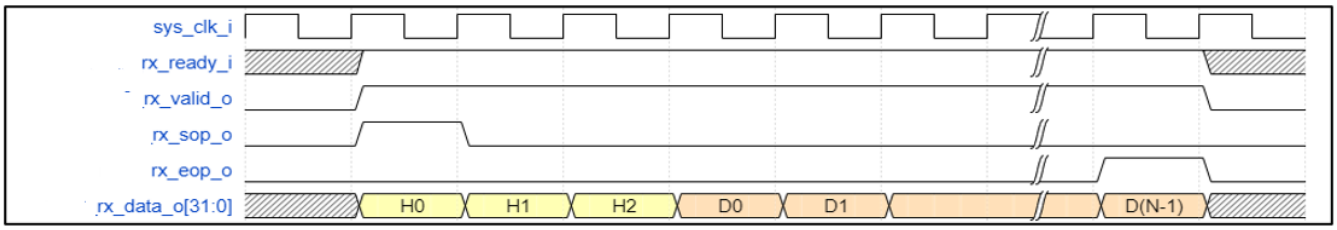

Figure 2.25. TLP Memory Write Operation (x1 Lane)

Figure 2.25 and Figure 2.26 shows the TLP transaction according to the rx_ready_i behaviour based on the minimum timing of rx_ready_i:

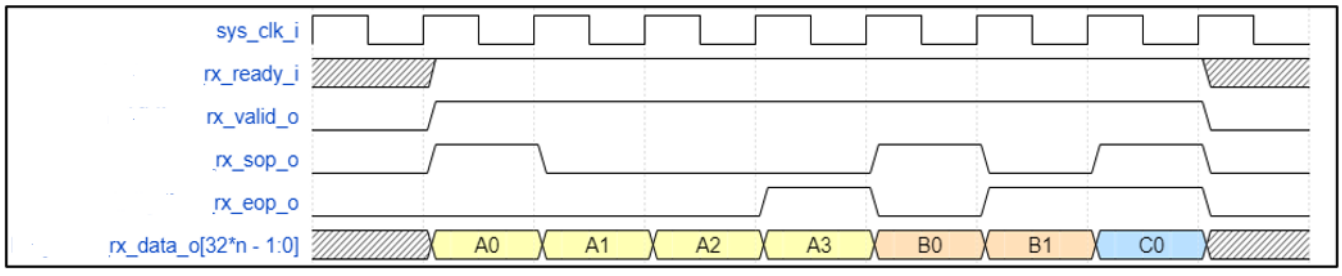

Figure 2.26. Minimum rx_ready_i Timing Diagram

Transaction A begins on cycle 2 with the assertion of rx_sop_o==rx_valid_o==1 and ends on cycle 5 with the assertion of rx_eop_o==rx_valid_o==rx_ready_i==1. The packet transfers with minimum timing since rx_valid_o == rx_ready_i == 1 on cycles 2-5. Data transfers on cycles 2-5.

Transaction B begins immediately after Transaction A on cycle 6 with the assertion of rx_sop_o==rx_valid_o==1 and ends on cycle 7 with the assertion of rx_eop_o==rx_valid_o==rx_ready_i==1. Data transfers on cycles 6-7.

Transaction C begins immediately after Transaction B on cycle 8 with the assertion of rx_sop_o==rx_valid_o==1 and ends on the same cycle since rx_eop_o==rx_ready_i==1 also asserted. Data transfers on cycle 8 considering the wait state timing of rx_ready_i:

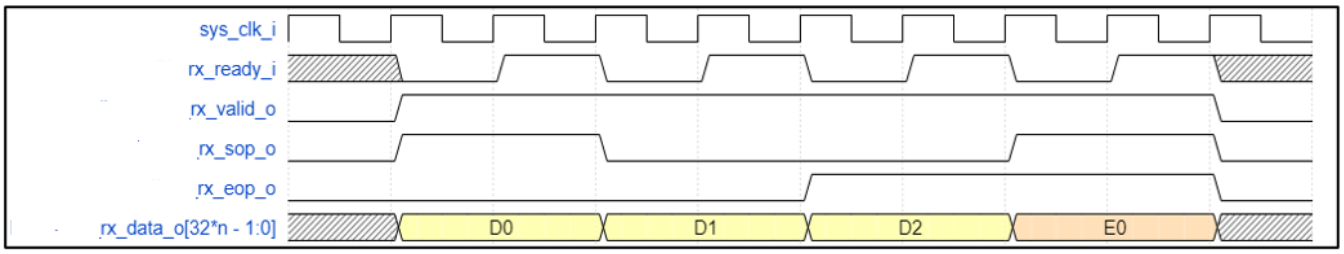

Figure 2.27. Wait State of rx_ready_i Timing Diagram

Transaction D begins on cycle 2 with the assertion of rx_sop_o==rx_valid_o==1 and ends on cycle 7 with the assertion of rx_eop_o==rx_valid_o==rx_ready_i==1. The packet transfer wait states due to rx_ready_i==0 on cycles 2, 4, and 6. Data transfers on cycles 3, 5, and 7.

Transaction E begins on cycle 8 with the assertion of rx_sop_o==rx_valid_o==1 but is wait stated due to rx_ready_i==1 on cycle 8. On cycle 9 the transaction completes with rx_sop_o==link[LINK]_rx_eop_o==rx_valid_o==rx_ready_i==1.

Receive Interface Considerations

The following considerations are provided to simplify logic using the receive interface and to address common problems, which must be avoided:

- For each TLP that you receive, the core strips a minimum 2-bytes of STP/END/EDB framing, a 2-bytes of Sequence Number, and a 4-bytes of Link CRC, for a total of 8 bytes (64-bits). These additional 8 bytes, which the core receives but which do not appear on rx_data_o, allows you the flexibility of not using every clock cycle on the receive interface. This flexibility can be useful to simplify user logic and improve design timing closure.
- TLPs that appear on the receive interface have passed the Physical Layer and Link Layer error detection and correction logic and can be assumed to be free of transmission errors. When the core receives a TLP with a STP/END/EDB framing, Sequence Number, or Link CRC error, the core coordinates re-transmission of the TLP with the remote PCI Express device and only forwards packets that pass transmission error checks onto to the receive interface.
- TLPs that are received from PCI Express are decoded for validity against the core's configuration registers and are only forwarded to the receive interface if they hit an enabled resource. Therefore, you only need to handle valid TLPs which target the user resources. TLPs, which do not hit user resources, are terminated by the core and the appropriate error message and response is handled by the core on your behalf.
- The Lattice PCIe core handles all Data Link Layer functionality for you and handles most of the Transaction Layer error cases as well. The core consumes Configuration Transactions, Messages, and TLPs which do not map to user resources and transmits the appropriate response. TLPs which are handled by the core do not appear on the Receive Interface.
- User logic that manages read requests (for DMA) and assigns a tag to each read request that is transmitted. The core provides the tag of each received completion on rx_cmd_data_o to allow user logic to route completions from different sources to the destination without having to parse the TLP for tag information. The core does not track the outstanding tags that are in use by the user. If a completion is received with a tag that does not correspond to an outstanding user read request, then you must report the error.

Data Byte Order

The core transmits the TLP data in the following byte order:

- tx_data_i[7:0], tx_data_i[15:8], tx_data_i[23:16],…

The core receives the TLP data in the following byte order:

- rx_data_o[7:0], rx_data_o[15:8], rx_data_o[23:16],…

For example, you transmit, or the core receives a 32-bit Memory Read Transaction Layer Packet in the following byte order as shown in Table 2.48.

Table 2.48. Data Byte Order

tx_data_i/ rx_data_o	First Data Word	Second Data Word	Third Data Word
[7:0]	{R, Fmt[1:0], Type[4:0]}	RequesterID[15:8]	Addr[31:24]
[15:8]	{R, TC[2:0], R[3:0]}	RequesterID[7:0]	Addr[23:16]
[23:16]	{TD, EP, Attr[1:0], Length[9:8]}	Tag[7:0]	Addr[15:8]
[31:24]	Length[7:0]	{LastDWBE[3:0], 1stDWBE[3:0]}	{Addr[7:2], R[1:0]}

2.11.2.4. Transaction Layer Interface Error Detection and Correction

The Lattice PCIe IP Core has built in error detection and correction mechanisms for both Transaction Layer Packets (TLPs) which are transferred between PCI Express and Transaction Layer Interface and Data Link Layer Packets (DLLPs) which are used by the core internally for link management.

The Lattice PCIe IP core adds the required Physical Layer framing (STP/END/EDB) and Data Link Layer error detection and correction information (Sequence Number/Link CRC) to the TLP packets transmitted on the Transmit Interface. Likewise, when TLP packets are received from PCI Express, the core validates that the packet is received correctly by checking the Physical Layer framing (STP/END/EDB) and Data Link Layer error detection and correction information (Sequence Number/Link CRC). Packets that are forwarded to you on the receive interface are sent after stripping the Physical Layer framing (STP/END/EDB) and Data Link Layer error detection and correction information (Sequence Number/Link CRC).

If transmission errors are detected in packet transmission or reception, the core coordinates with the remote PCI Express device to retry the transaction and recover from the error. This process occurs without any user intervention. The Lattice PCIe core logs both corrected and uncorrected errors. This error status information is made available through the status registers and is accessed by system software through the Configuration Registers. The core generates and transmits error Message TLPs to the remote PCI Express device in response to different types of errors detected.

ECRC (TLP Digest) generation and checking is a core option. When ECRC generation support is enabled by the software (AER Capability: ECRC Generation Enable == 1), the core generates and adds ECRC to all transmitted TLPs (except those that already contain an ECRC with TD bit set to 1). When ECRC checking support is enabled by software (AER Capability: ECRC Check Enable == 1), the ECRC fields present in received TLPs are checked for validity and any errors are noted on the Receive Interface and are reported in the AER Capability. The core does not modify the ECRC or TD (TLP Digest == ECRC indicator) fields on received TLPs and passes these fields onto the receive interface as received.

The Lattice PCIe core also handles TLPs that are Type 0 Configuration transaction requests, messages requests for link management, TLPs that do not hit an enabled resource and any requests that the core determines are malformed.

If the core found TLP having transmission errors, then that TLP is consumed by the core (and not forwarded) and then transmits any required completion packet(s), generates required error messages and logs any required errors.

The core has been designed in such a way that it is feasible for you to only consume and generate the TLPs and can make use of these TLPs for transferring data and control information between your application and the remote PCI Express devices.

2.11.3. LMMI Interface

When you select the TLP as data interface option in the PCIe IP user interface, the IP by default configures LMMI as register interface. The Core Configuration and Status Registers (CSR) are made accessible to the user design through the Lattice Memory Mapped interface (LMMI).

An example of the register configuration through the LMMI is shown below in the LMMI write and read timing diagrams.

The data transaction, through the LMMI, only starts when usr_lmmi_request_i==usr_lmmi_ready_o==1. Consecutive request must be done with at least one clock period wait cycle (for example, usr_lmmi_request_i should de-assert first after a successful transaction before making another request).

When usr_lmmi_request_i==usr_lmmi_ready_o==1, usr_lmmi_wr_rdn_i, and usr_lmmi_offset_i must be valid and describe the transaction to execute; if the transaction is a write as indicated by usr_lmmi_wr_rdn_i==1, usr_lmmi_wdata_i must also be valid.

Note: Only one request should be active at a given time.

2.11.3.1. LMMI Write Operations

You can write the data to PCIe core registers only when the ready signal is received from PCIe IP. For example, the usr_lmmi_ready_o signal must be 03(x1)/07(x2)/1f(x4). The data is written to PCIe registers only when PCIe IP gets a request from you; that is, usr_lmmi_request_i is configured as 01 and the usr_lmmi_wr_rdn_i signal must be high when usr_lmmi_ready_o signal is asserted as 03(x1)/07(x2)/1f(x4).

For example, you need to write 0X01 data into 0X0A register and then 0X02 data into 0X0B register. The 0X01 data is written into 0X0A register in one transaction only as ready signal is high when request is asserted. But to write 0X02 data into 0X0B register took two transactions because ready signal is low when request is asserted in first transaction. Therefore, the data is written to the register in the second transaction only when ready signal is high as shown in Figure 2.28.

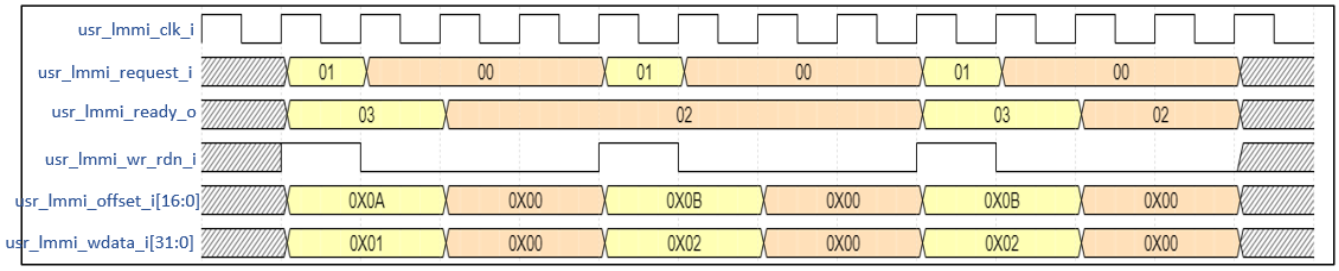

Figure 2.28. LMMI Write Operation

2.11.3.2. LMMI Read Operation

You can read the data from PCIe core registers only when the ready and read valid signals are received from PCIe IP. For example, the usr_lmmi_ready_o signal must be 03 and the usr_lmmi_rdata_valid_o signal must be 02. The data is read from PCIe registers only when PCIe IP gets a request from the user. For example, usr_lmmi_request_i is configured as 02 and the usr_lmmi_wr_rdn_i signal must be low when the usr_lmmi_ready_o signal is asserted as 03 and usr_lmmi_rdata_valid_o is asserted as 02.

For example, you want to read the data [0X0000001d00000000] from lane 0 PMA Status register offset 0x7F. The transaction follows the steps as shown in Figure 2.29.

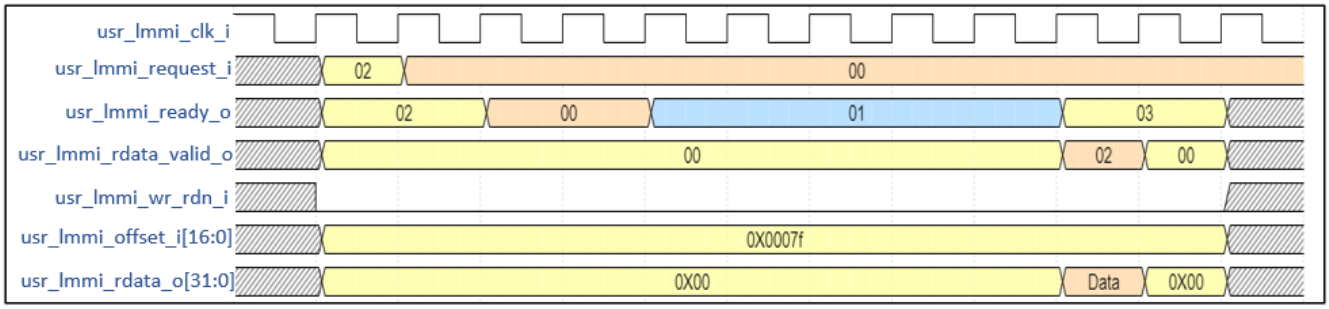

Figure 2.29. LMMI Read Operation

The following are the registers to be configured through the LMMI:

- Register Address – 0Xf004 [Base address: 0x0F000, Offset address: 0X4]
  This register is used to assert the PCIe core reset.
- Simulation Registers
  - Register address – 0x2000
    This register is used to reduce the ltssm ts_1 and timeouts to fasten the simulation when asserted as *1*.
  - Register address – 0x3000
    This register is used to reduce the Power Management State Machine timeouts to fasten the simulation when asserted as *1*.
  - Register address – 0x4000
    This register is used to reduce the timeouts to fasten the simulation when asserted as *1*.
- Lane (PLL status) register address – 0X0007f
  This register is used to read PLL status of each lane.

After configuring all the registers, configure the 0Xf004 register again to make the PCIe core out of reset mode.

2.11.4. UCFG Interface

The UCFG Interface is provided for users to read the current values of the Lattice PCIe x1 IP Core's PCIe Configuration Registers and to also obtain the status of the Lattice PCIe x1 IP Core that may be needed to implement the user's design.

2.11.4.1. UCFG Operation

In the Lattice PCIe IP Core, the UCFG Interface is provided for users to read the current values of the PCIe IP configuration. The UCFG Interface is a simple SRAM-like interface that accepts write/read transactions. The UCFG Interface supports multiple outstanding transaction requests to enable higher throughput on the interface. This interface is referenced to sys_clk_i clock domain. Writes and reads are executed out in the same order that they are accepted on the interface.

2.11.4.2. UCFG Transaction

The UCFG Interface is primarily intended to obtain the current values of the Lattice PCIe IP core configuration registers, that are needed by the user. The UCFG interface support both reading and writing, but the PCIe configuration registers should not be written by the user exception for the MSI and the Error Reporting functions. The PCIe Configuration registers are written by the Host OS/BIOS during the PCIe enumeration and writing them through this interface risks creating incompatibilities with the OS/BIOS that may cause serious errors.

PCIe Configuration Registers may change at any time as the Host OS/BIOS updates them with writes. Some registers also contain current PCIe Endpoint Core status which changes in response to link events such as speed changes, recovery cycles, width changes, and power state changes. The UCFG Interface thus should be used to poll the values needed for the user design as frequently as possible to maintain up-to-date values.

UCFG Write Transaction

The following are the write access registers of the UCFG Interface:

- Address 0x2D MSI Pending
- Address 0x3f0-3F3 Error Report Header
- Address 0x3F8 Error Report

Writes to these registers are not part of the core's PCIe Configuration Registers but instead enables you to implement the MSI Pending register and report errors detected in the user design to the core's AER Capability.

UCFG Read Transaction

The UCFG read operation can be started after the tl_link_up signal is asserted. The transaction is started only when the ucfg_valid_i and ucfg_ready_o signals are asserted. For reading the data the ucfg_wr_rd_n signal is driven low. Figure 2.30 shows the UCFG read transaction timing diagram of register offset 0x5F.

The offset 0x5F read access through the UCFG interface provides the following data from the Type 0 Configuration Write packets that the PCIe Endpoint Core received. This information is typically used as Requester ID for Memory TLPs, or Completer ID for Completion TLPs.

- bits[15:8] – Bus Number
- bits[7:3] – Device Number
- bits[2:0] – Function Number

For APB interface, example of address to read the Completer ID or Requester ID at offset 0x5F:

c_apb_paddr_i = 32'b1100_0101_0010_0_010_00_00_00_0101_1111_00 = 0xC522_017C

bit[31:19]: PCIe CSR base address configured in IP GUI. In this example, the base address is 0xC520_0000

bit[18:16]: 3'b010

bit[15:14]: 2'b00 for Link0

bit[13:12]: 2'b00 function 0

bit[11:2]: 0x5F

bit[1:0]: 2'b00

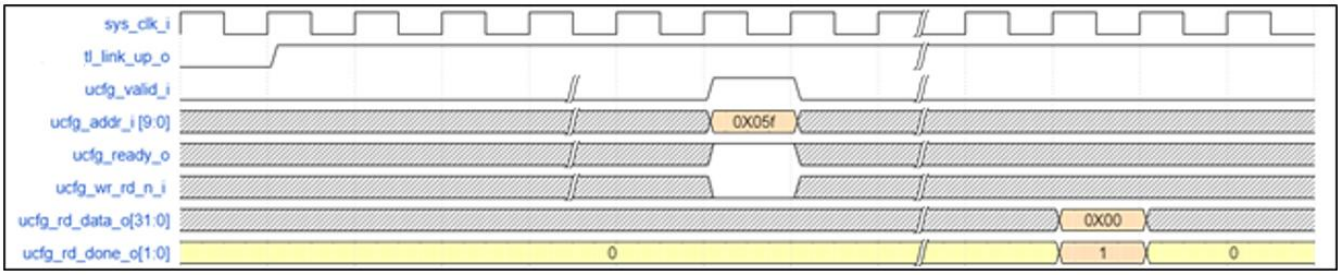

Figure 2.30. UCFG Read Transaction Timing Diagram

2.11.4.3. UCFG Address Space

The UCFG Interface may be used to access all the Lattice PCIe x1 IP Core's PCIe Configuration Registers.

In addition, the UCFG Interface implements a small number of registers to provide useful status that is not available in the PCIe Configuration Registers.

The PCIe Configuration Registers are accessed by ucfg_addr_i [11:2], which is a DWORD (32-bit) aligned address.

Table 2.49. UCFG Address Space

Capability/Data	usfg_addr_i[11:2]	Description
Configuration Header	0x00-0x0F	PCI Configuration Header The following fields are likely needed: Address 0x01 • Bit 0 – I/O Space Enable • Bit 1 – Memory Space Enable • Bit 2 – Bus Leader Enable • Bit 10 – Interrupt Disable
PCI Express Capability	0x10-0x1E	PCI Express Capability Structure The following fields are likely needed: Address 0x10 • Bits 29:25 – Interrupt Message Number Address 0x12 • Bit 4 – Enable Relaxed Ordering • Bits 7:5 – Maximum Payload Size • Bit 8 – Extended Tag Enable • Bit 11 – Enable No Snoop • Bits 14:12 – Maximum Read Request Size Address 0x14 • Bit 19:16 – Link Speed • Bits 25:20 – Negotiated Link Width Address 0x17 • Bit 4 – CRS Software Visibility Enable Address 0x1A • Bits 3:0 – Completion Timeout Value • Bit 4 – Completion Timeout Disable • Bit 6 – Atomic Op Requester Enable • Bit 8 – IDO Request Enable • Bit 9 – IDO Completion Enable • Bit 10 – LTR Mechanism Enable
Power Management Capability	0x20-0x21	Power Management Capability Structure The following fields are likely needed: Address 0x21 • Bits 1:0 – PM Power State
MSIX Capability	0x24-0x27	MSIX Capability Structure The following fields are likely needed: Address 0x24 • Bit 30 – MSIX Function Mask • Bit 31 – MSIX Enable Address 0x25 • Bits 31:3 – MSIX Table Offset Address 0x26 • Bits 31:3 – MSIX PBA Offset

Capability/Data	usfg_addr_i[11:2]	Description
MSI Capability	0x28-0x2D	MSI Capability Structure The following fields are likely needed: Address 0x28 • Bit 16 – MSI Enable • Bits 22:20 – MSI Multiple Message Enable • Bit 24 – MSI Per-Vector Mask Capable Address 0x29 • Bits 31:0 – MSI Address [31:0] Address 0x2A • Bits 31:0 – MSI Address [63:32] Address 0x2B • Bits 15:0 – MSI Data Address 0x2C • Bits 31:0 – MSI Mask Bits Address 0x2D • Bits 31:0 – MSI Pending Bits (writable)
AER Capability	0x40-0x51	AER Capability Structure The following fields are likely needed: Address 0x46 • Bit 6 – ECRC Generation Enable • Bit 8 – ECRC Check Enable
Vendor Specific Capability	0x54-0x5F	Vendor Specific Capability Structure These addresses contain important PCIe Endpoint Core status that is available through the UCFG Interface. Address 0x5D • Bits 15:0 – Number of CH Credits implemented by the Receive Buffer. • Bits 31:16 – Number of CD Credits implemented by the Receive Buffer. Address 0x5E • Bits 3:0 – Current LTSSM Major State • Bits 7:4 – Current LTSSM Minor State • Bits 10:8 – Current RX L0s State • Bits 15:12 – Current Lane Reverse Status • Bits 20-16 – Current PM State • Bits 31:24 – Current Function Enable Status Address 0x5F • Bits 15:0 – Current Completer ID or Requester ID for endpoint. The decoding: [15:8] = Bus number [7:3] = Device number [2:0] = Function number • Bit 16 – Current Port Type 1 = Downstream Port 0 = Upstream Port • Bit 17 – Current PCIe Cfg Register Type 1 = Reserved 0 = Type 0 (Endpoint)
ATS Capability	0x80-0x81	ATS Capability Structure The following fields are likely needed: Address 0x81 • Bits 20:16 – ATS Smallest Transaction Unit (STU) • Bit 31 – ATS Enable

Capability/Data	usfg_addr_i[11:2]	Description
LTR Capability	0xF8-0xF9	LTR Capability Structure The following fields are likely needed: Address 0xF9 • Bits 12:0 – LTR Max Snoop Latency • Bits 28:16 – LTR Max No-Snoop Latency
Error Report Header	0x3F0-0x3F3	Address Range used to Report TLP Error Headers Address 0x3F0 • Bits 31:0 – TLP Header [31:0] Address 0x3F1 • Bits 31:0 – TLP Header [63:32] Address 0x3F2 • Bits 31:0 – TLP Header [95:64] Address 0x3F3 • Bits 31:0 – TLP Header [127:96]
Error Report	0x3F8	This address is used to report errors to the AER capability. • Bits 7:0 – Error function Number • Bits 13:8 – Error flags • Bit 8 – Poisoned TLP received • Bit 9 – Completion Timeout • Bit 10 – Completer Abort • Bit 11 – Unexpected Completion • Bit 12 – Unsupported Request • Bit 13 – Uncorrectable Internal Error • Bits 21:16 – Advisory Flags • Bit 16 – Poisoned TLP received • Bit 17 – Completion Timeout • Bit 18 – Completer Abort • Bit 19 – Unexpected Completion • Bit 20 – Unsupported Request • Bit 21 – Uncorrectable Internal Error

2.11.4.4. User Error Reporting

The UCFG Interface enables you to log errors that they detect into the core's AER Capability for reporting to software.

The process for reporting an error is as follows:

1. Write the Header of the TLP with the error into the four addresses for Error Report Header. If the error is not generated by a specific TLP, write zeros into Error Report Header.
2. Write the function number and error flags to Error Report to trigger the Core to record the error in the appropriate registers in Configuration Space. One write is needed for each function that must receive the error report. In a single function core, the function number is always 0.

To report an error, the following points must be followed:

- First, write the Error Report Header registers with the associated TLP Header of the TLP with the error, write 0s if the error is not associated with a specific TLP.
- Write Error Report register with the error type. Indicate the function number that should log the error on bits [7:0].
- If the error is not associated with a specific function, then write 0 to assign the error to Function [0].
- Indicate the error type by setting only one bit of bits [13:8] flag.
- If the error is the *advisory* type as defined by the PCIe Specification, set the corresponding bit in bits [21:16] of the error bit set in bits [13:8].

- Advisory errors are downgraded to correctable error status, so the host generally continues the operation unimpeded after an advisory error is reported. If a non-advisory error is reported, the host OS typically faults (blue screen for windows) as these are serious errors that the host OS must either handle through the software or halt operation of the OS.
- The error is only logged when the Error Report register is written with one of bits [13:8] non-zero.
- The TLP Header associated with the error must already have been written and is taken from the Error Report Header registers.

2.12. Soft IP Interface

2.12.1. Data Interface Conversion

2.12.1.1. AXI-Stream Interface

This interface is available if the data interface type selected in the IP generation user interface is *AXI4_STREAM*. The data width of the transaction for x1 lane is 32-bit.

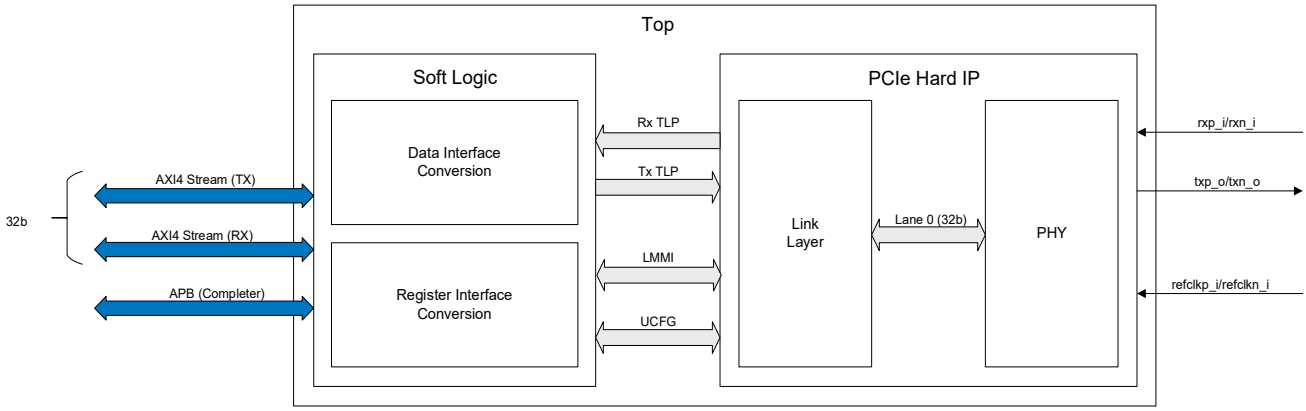

Figure 2.31. AXI-Stream Data Interface, APB Register Interface

PCIe to AXI-Stream Transfer

For the PCIe to AXI-Stream transfer, the PCIe sends the data to the user application. The transaction has the header of size three double-word. After the header transaction, the actual data is transferred as shown in Figure 2.32.

Note: The *DATA* in the transactions below are random data sent by the PCIe IP to compensate for the data width.

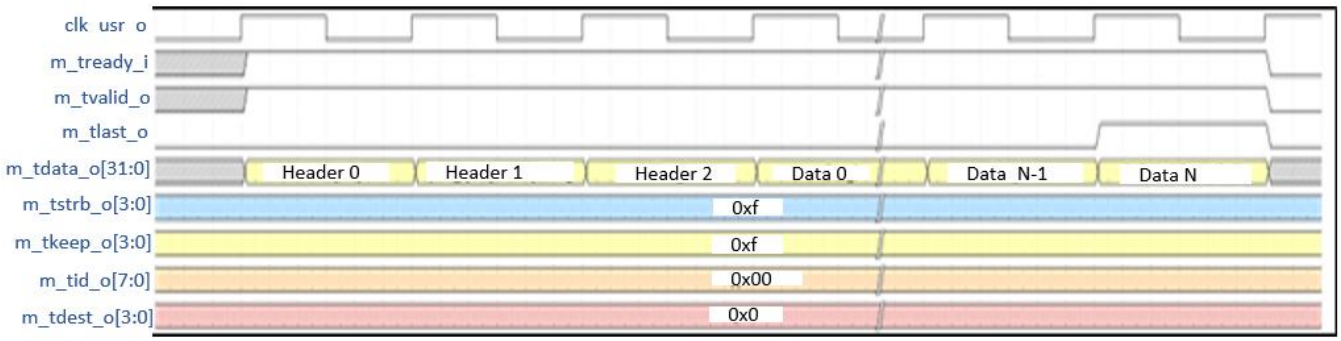

Figure 2.32. PCIe to AXI-Stream Transaction for x1

AXI-Stream to PCIe Transfer

For the AXI-Stream to PCIe transaction, the user's application sends the data to the PCIe Endpoint IP. Similar to the PCIe to AXI-Stream transfer, there is a header data of size three double word, which is transferred first followed by the actual data as shown in Figure 2.33.

Note: The *DATA* in the transactions below are random data sent by the PCIe IP to compensate for the data width.

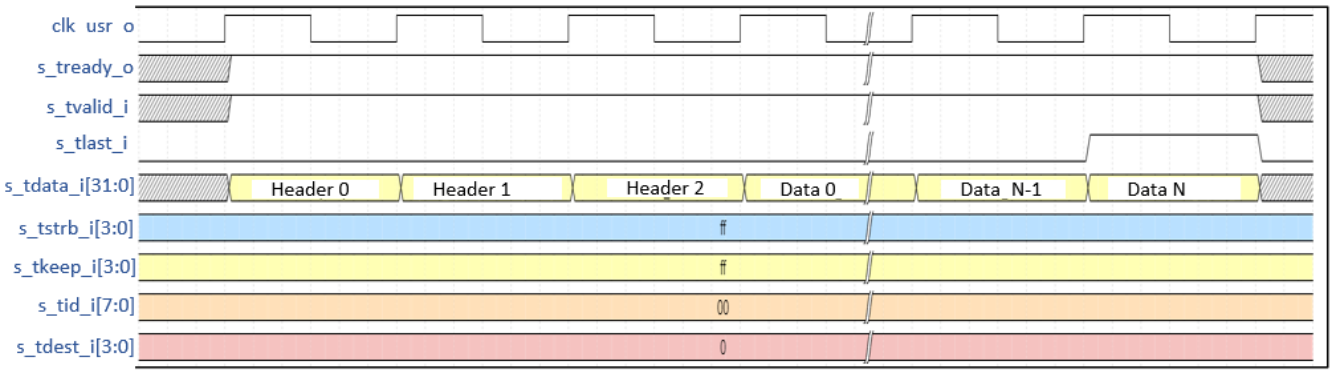

Figure 2.33. AXI-Stream to PCIe Transaction for x1

2.12.1.2. Bridge Mode

Bridge Mode is a non-DMA mode which allows the received MWr and MRd TLP to be converted to AXI-MM or AXI-Lite Manager Interface. Subordinate interface is not supported.

In Radiant user interface, you can configure Bridge Mode interface type (AXI-MM or AXI-L), BAR number that is associated to DMA Bypass, and BAR size.

When a received MWr/MRd TLP targets Bridge Mode BAR, the IP converts the TLP to AXI-MM or AXI-Lite Manager Interface. The BAR value is masked off to 0 when presented at AXI Read/Write Address.

For MRd TLP, the read data at AXI-MM/AXI-Lite Read Data Channel is converted to CplD TLP and be transmitted to PCIe link partner.

The following are the limitations of Bridge Mode:

- Only 1-DW MWr/MRd TLP is supported. If AXI-MM interface is selected, it supports only 32-bit data width without burst mode (AWLEN and ARLEN are always 0).
- Only DW-aligned address is supported. The 4-bit LSB of read/ write address must be 0x0, 0x4, 0x8, or 0xC.
- Only 32-bit addressing BAR is supported.

In addition, when Bridge Mode is selected by the Radiant user interface, user interrupt pins can be enabled. Refer to the User Interrupts section for more details.

Figure 2.34 and Figure 2.35 show the Bridge Mode Radiant user interface settings.

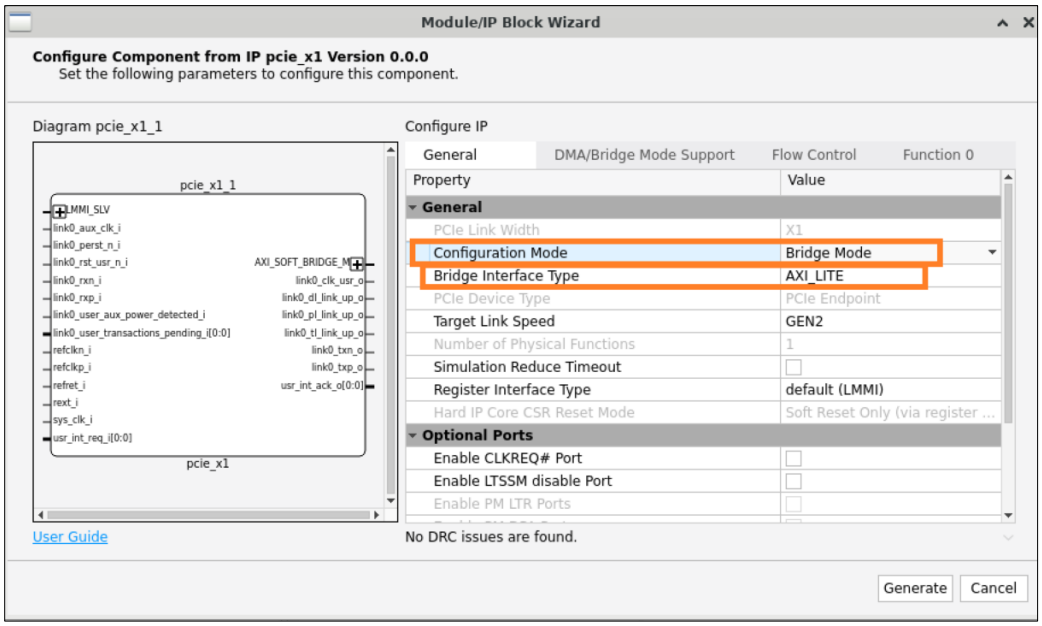

Figure 2.34. Bridge Mode Enablement (General Tab)

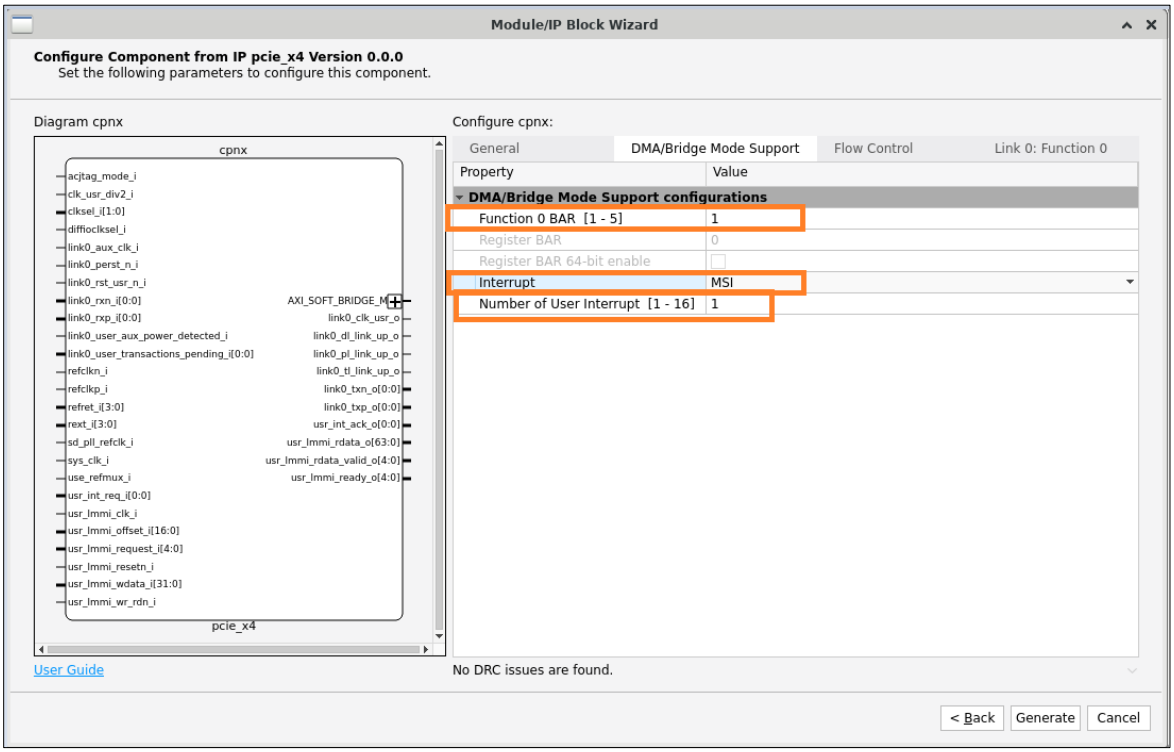

Figure 2.35. Bridge Mode Enablement (DMA/Bridge Mode Tab)

In the Radiant user interface, Bridge Mode is enabled when *Bridge Mode* is selected in *Configuration Mode* drop-down menu in *General* tab (see Figure 2.34).

Bridge Mode interface type (AXI-MM or AXI-Lite) is configured by *Bridge Interface Type* in *General* tab (see Figure 2.34).

The Bridge Mode BAR number is configured by *Function 0 BAR* in the *DMA/Bridge Mode* tab (see Figure 2.35). The available options are BAR 1 to BAR5. BAR0 is reserved for Bridge Mode registers (refer to the Bridge Mode Register section). The

Bridge Mode BAR size is configured in *Link 0: Function 0* tab. DMA interrupt can be MSI or MSI-X, configurable in *DMA/ Bridge Mode Support* tab.

With MSI, the IP supports up to 16 user interrupts. With MSI-X, the IP supports up to 64 user interrupts. The total number of user interrupts is configured by *Number of User Interrupt* in the *DMA/Bridge Mode Support* tab (see Figure 2.35). Refer to the User Interrupts section for more details.

User Interrupts

PCIe IP supports up to 16 user interrupts and 64 user interrupts for MSI and MSI-X, respectively. The number of user interrupts is configured by the Lattice Radiant user interface.

Each user interrupt has a pair of request and acknowledgement pins at the IP interface such as usr_intr_req_i and usr_intr_ack_o, respectively. When user logic asserts any usr_intr_req_i, The PCIe IP transmits MSI/MSI-X TLP to PCIe link partner. If more than one usr_intr_req_i are asserted, an arbiter in the IP arbitrates these requests with round-robin arbitration scheme. The interrupt vector (MSI vector) associated with a user interrupt is configured through the USR_INT_VEC_P* registers. Refer to the Bridge Mode Register section for the details.

The following are the requirements of usr_intr_req_i[NUM-1:0] and usr_intr_ack_o[NUM-1:0]:

1. Each user interrupt has their corresponding usr_intr_req_i[] and usr_intr_ack_o[] pins at the IP's interface. The bit number of these signals refers to the user number.
2. User application logic must assert usr_intr_req_i[] when it requires PCIe IP to send interrupt (MSI or MSI-X) to the host.
3. usr_intr_req_i[] and usr_intr_ack_o[] must comply to full handshake relationship.
   - - usr_intr_req_i[] can only assert when usr_intr_ack_o[] is 0.
   - - usr_intr_req_i[] can only de-assert when usr_intr_ack_o[] is 1.
   - - usr_intr_ack_o[] = 1 means the corresponding user request (via usr_intr_req_i[] assertion) is translated to MSI/MSI-X transmission.

The violation of the rule between usr_intr_req_i[] and usr_intr_ack_o[] may cause undefined behaviour. Figure 2.36 shows the example waveform:

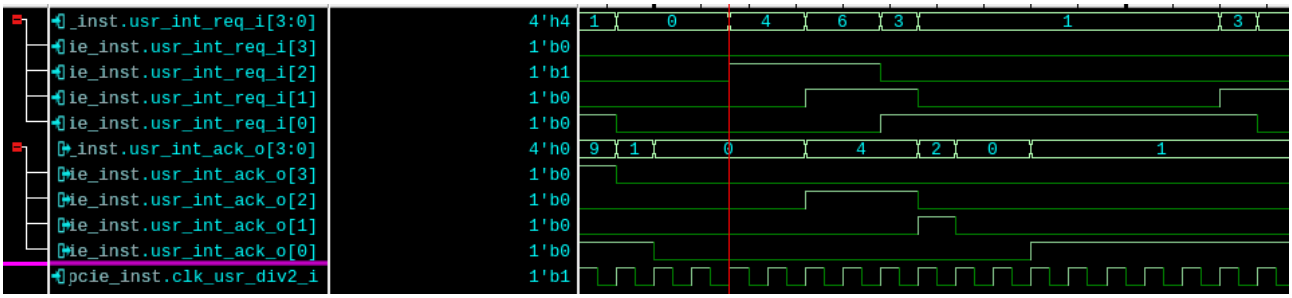

Figure 2.36. User Interrupt Pins Example Waveform

The requests from user logic are translated to MSI or MSI-X TLP. Refer to MSI Bridge Mode MSI Bridge Modeand MSI-X Bridge Mode sections for details.

MSI Bridge Mode

MSI (Message Signaled Interrupts) is supported by PCIe IP as an approach to interrupt Host when user logic asserts specific pins to the IP (see Bridge Mode Register section).

A full 32 vectors are advertised, but only up to 16 interrupt requesters (the number of requesters is configurable) are supported. You can configure user-interrupt-pin-to-MSI-vector mapping through writing to USR_INT_VEC_P* registers. Refer to Bridge Mode Register section for detail.

Per-Vector Masking and Extended Message Data are not supported.

MSI Advertised Capabilities

This section specifies MSI-related capabilities and the advertised values. These values are not configurable.

Table 2.50. MSI Advertised Capabilities

Registers	Advertised value	Remark
Multiple Message Capable	3'b101	32 vectors are advertised despite there are only 16 user interrupt pins.
64-bit address capable	1'b1	64-bit addressing is supported.
Per-Vector Masking Capable	1'b0	Per-Vector Masking is not supported.
Extended Message Data Capable	1'b0	Extended Message Data is not supported.

MSI-X Bridge Mode

Similar to MSI, MSI-X (Message Signaled Interrupts - Extended) is supported by PCIe IP as an approach to interrupt Host when user logic asserts designated pins to the IP (see User Interrupts section).

Up to 16 interrupt requesters (the number of requesters is configurable) are supported. Each requester has a corresponding MSI-X table entry specified by PCIe specification, which means a total of 64 table entries are supported by the IP.

The MSI-X Table entry associated with user interrupts is as below:

Table 2.51. MSI-X Bridge Mode

Offset	MSI-X Table Entries
0x8000	User0 Msg Address
0x8004	User0 Msg Upper Address
0x8008	User0 Msg Data
0x800C	User0 Vector Control
0x8010	User1 Msg Address
0x8014	User1 Msg Upper Address
0x8018	User1 Msg Data
0x801C	User1 Vector Control
…	…
0x83F0	User63 Msg Address
0x83F4	User63 Msg Upper Address
0x83F8	User63 Msg Data
0x83FC	User63 Vector Control

The PBA (Pending Bit Array) table entry associated with user interrupts is as below.

Table 2.52. MSI-X PBA Offsets

Offset	PBA Table Entries
0xC000	user_int_pb[63:0]

MSI-X Table and PBA Table are residing in IP registers at BAR0 (memory space). The offsets in the table above refer to BAR0 offset.

Per PCIe specification requirement, MSI-X must support Function Masking and Per-Vector Masking (PVM).

When Function Mask bit in PCIe Capability register is set to 1, if the IP was to send an MSI-X TLP (due to user interrupt request), this TLP is not sent and instead the IP asserts the corresponding Pending Bit at offset 0xC000.

Once the Function Mask bit is cleared to 0, the IP screens through the Pending Bits, for the bit(s) that is 1, and its corresponding per-vector mask bit is 0, the IP generates and transmits MSI-X TLP to the host. Afterwards, the IP will clear the corresponding Pending Bit(s) to 0. It is possible to have multiple MSI-X TLPs transmitted by the IP after Function Mask bit is cleared by the SW.

When per-vector Mask bit register in Vector Control register is set to 1, if the IP was to send an MSI-X TLP (due to user interrupts) of this vector, this TLP is not sent, and instead the IP asserts the corresponding Pending Bit at offset 0xC000.

Once the Mask bit is cleared to 0 (and Function Mask bit is also 0), the IP will refer to the corresponding vector's Pending Bits. If the Pending Bit is 1, the IP generates and transmits MSI-X TLP to the host. Afterwards, the IP clears the corresponding Pending Bit to 0.

Steering Tag (optional per PCIe specification) is not supported. Therefore, all other bits in the Vector control register (other than bit 0) are reserved.

MSI-X Advertised Capabilities

This section specifies MSI-X-related capabilities and the advertised values. These values are not configurable.

Table 2.53. MSI-X Advertised Capabilities

Registers	Advertised Value	Remark
Table Size	10'h0 – 10'h63	The advertised number of table entries depends on the number of user interrupts.
Table BIR	3'b000	BAR 0
Table Offset	29'b1000_0000_0000_0	MSI-X Table is from BAR 0 Offset 0x8000
PBA BIR	3'b000	BAR 0
PBA Offset	29'b1100_0000_0000_0	PBA Table is from BAR 0 Offset 0xC000

Bridge Mode Register

Bridge Mode registers are accessible by the Host when the received MWr or MRd TLP targets BAR 0. The register access size is limited to maximum 1 DW per TLP.

The access types of each register are defined in Table 2.54.

Table 2.54. Access Types

Access Type	Behavior on Read Access	Behavior on Write Access
RO	Returns register value	Ignores write access
WO	Returns 0	Updates register value
RW	Returns register value	Updates register value
RW1C	Returns register value	Writing 1'b1 on register bit clears the bit to 1'b0. Writing 1'b0 on register bit is ignored.
WHC	Returns register value	Only Write to 1'b1 when the register is 1'b0 takes effect.
RC	Returns register value Clear the register to 0 after read.	Ignores write access
RSVD	Returns 0	Ignores write access

Table 2.55. USR_MSI_VEC_P1 (0x040C)

Field	Name	Access	Width	Default	Description
31:29	RSVD	RO	3	0	Reserved
28:24	USR3_MSI_VEC	RW	5	0	User 3 MSI Vector When MSI is selected: 5'd0: MSI Vector 0 5'd1: MSI Vector 1 And so on. This register is RO if the number of user interrupt is set to less than 4.
23:21	RSVD	RO	3	0	Reserved
20:16	USR2_MSI_VEC	RW	5	0	User 2 MSI Vector

Field	Name	Access	Width	Default	Description
					When MSI is selected: 5'd0: MSI Vector 0 5'd1: MSI Vector 1 And so on. This register is RO if the number of user interrupt is set to less than 3.
15:13	RSVD	RO	3	0	Reserved
12:8	USR1_MSI_VEC	RW	5	0	User 1 MSI Vector When MSI is selected: 5'd0: MSI Vector 0 5'd1: MSI Vector 1 And so on. This register is RO if the number of user interrupt is set to less than 2.
7:5	RSVD	RO	3	0	Reserved
4:0	USR0_MSI_VEC	RW	5	0	User 0 MSI Vector When MSI is selected: 5'd0: MSI Vector 0 5'd1: MSI Vector 1 And so on.

Table 2.56. USR_MSI_VEC_P2 (0x0410)

Field	Name	Access	Width	Default	Description
31:29	RSVD	RO	3	0	Reserved
28:24	USR7_MSI_VEC	RW	5	0	User 7 MSI Vector When MSI is selected: 5'd0: MSI Vector 0 5'd1: MSI Vector 1 And so on. This register is RO if the number of user interrupt is set to less than 8.
23:21	RSVD	RO	3	0	Reserved
20:16	USR6_MSI_VEC	RW	5	0	User 6 MSI Vector When MSI is selected: 5'd0: MSI Vector 0 5'd1: MSI Vector 1 And so on. This register is RO if the number of user interrupt is set to less than 7.
15:13	RSVD	RO	3	0	Reserved
12:8	USR5_MSI_VEC	RW	5	0	User 5 MSI Vector When MSI is selected: 5'd0: MSI Vector 0 5'd1: MSI Vector 1 And so on.

Field	Name	Access	Width	Default	Description
					This register is RO if the number of user interrupt is set to less than 6.
7:5	RSVD	RO	3	0	Reserved
4:0	USR4_MSI_VEC	RW	5	0	User 4 MSI Vector When MSI is selected: 5'd0: MSI Vector 0 5'd1: MSI Vector 1 And so on. This register is RO if the number of user interrupt is set to less than 5.

Table 2.57. USR_MSI_VEC_P3 (0x0414)

Field	Name	Access	Width	Default	Description
31:29	RSVD	RO	3	0	Reserved
28:24	USR11_MSI_VEC	RW	5	0	User 11 MSI Vector When MSI is selected: 5'd0: MSI Vector 0 5'd1: MSI Vector 1 And so on. This register is RO if the number of user interrupt is set to less than 12.
23:21	RSVD	RO	3	0	Reserved
20:16	USR10_MSI_VEC	RW	5	0	User 10 MSI Vector When MSI is selected: 5'd0: MSI Vector 0 5'd1: MSI Vector 1 And so on. This register is RO if the number of user interrupt is set to less than 11.
15:13	RSVD	RO	3	0	Reserved
12:8	USR9_MSI_VEC	RW	5	0	User 9 MSI Vector When MSI is selected: 5'd0: MSI Vector 0 5'd1: MSI Vector 1 And so on. This register is RO if the number of user interrupt is set to less than 10.
7:5	RSVD	RO	3	0	Reserved
4:0	USR8_MSI_VEC	RW	5	0	User 8 MSI Vector When MSI is selected: 5'd0: MSI Vector 0 5'd1: MSI Vector 1 This register is RO if the number of user interrupt is set to less than 9.

Table 2.58. USR_MSI_VEC_P4 (0x0418)

Field	Name	Access	Width	Default	Description
31:29	RSVD	RO	3	0	Reserved
28:24	USR15_MSI_VEC	RW	5	0	User 15 MSI Vector When MSI is selected: 5'd0: MSI Vector 0 5'd1: MSI Vector 1 And so on. This register is RO if the number of user interrupt is set to less than 16.
23:21	RSVD	RO	3	0	Reserved
20:16	USR4_MSI_VEC	RW	5	0	User 14 MSI Vector When MSI is selected: 5'd0: MSI Vector 0 5'd1: MSI Vector 1 And so on. This register is RO if the number of user interrupt is set to less than 15.
15:13	RSVD	RO	3	0	Reserved
12:8	USR13_MSI_VEC	RW	5	0	User 13 MSI Vector When MSI is selected: 5'd0: MSI Vector 0 5'd1: MSI Vector 1 And so on. This register is RO if the number of user interrupt is set to less than 14.
7:5	RSVD	RO	3	0	Reserved
4:0	USR12_MSI_VEC	RW	5	0	User 12 MSI Vector When MSI is selected: 5'd0: MSI Vector 0 5'd1: MSI Vector 1 And so on. This register is RO if the number of user interrupt is set to less than 13.

Table 2.59. USR0_MSIX_TABLE (0x8000)

Field	Name	Access	Width	Default	Description
127:97	RSVD	RW	31	0	Reserved. By default, the value of these bits must be 0. RO when MSI-X is disabled or when User Interrupt is disabled.

Field	Name	Access	Width	Default	Description
96	USR0_MASK_BIT	RW	1	1	User 0 Mask Bit When this bit is Set, the Function is prohibited from sending a message using this MSI-X Table entry for user interrupt 0. However, any other MSI-X Table entries programmed with the same vector is still capable of sending an equivalent message unless they are also masked. Default value of this bit is 1b (entry is masked) RO when MSI-X is disabled or when User Interrupt is disabled.
95:64	USR0_MSG_DATA	RW	32	0	User 0 Message Data Message Data of MSI-X caused by User Interrupt 0. RO when MSI-X is disabled or when User Interrupt is disabled.
63:32	USR0_MSG_UPPER_ADDR	RW	32	0	User 0 Message Upper Address Upper 32-bit address of MSI-X caused by User Interrupt 0. If this field is zero, 32-bit address messages are used. If this field is non-zero, 64-bit address messages are used. RO when MSI-X is disabled or when User Interrupt is disabled.
31:0	USR0_MSG_ADDR	RW	32	0	User 0 Message Address Lower 32-bit address of MSI-X caused by User Interrupt 0. For proper DWORD alignment, software must always write zeroes to LSB two bits; otherwise, the result is undefined. RO when MSI-X is disabled or when User Interrupt is disabled.

Table 2.60. USR1_MSIX_TABLE (0x8010)

Field	Name	Access	Width	Default	Description
127:97	RSVD	RW	31	0	Reserved. By default, the value of these bits must be 0. RO when MSI-X is disabled or when User Interrupt is disabled.
96	USR1_MASK_BIT	RW	1	1	User 1 Mask Bit When this bit is Set, the Function is prohibited from sending a message using this MSI-X Table entry for user interrupt 1. However, any other MSI-X Table entries programmed with the same

Field	Name	Access	Width	Default	Description
					vector will still be capable of sending an equivalent message unless they are also masked. Default value of this bit is 1b (entry is masked) RO when MSI-X is disabled or when User Interrupt is disabled.
95:64	USR1_MSG_DATA	RW	32	0	User 1 Message Data Message Data of MSI-X caused by User Interrupt 1. RO when MSI-X is disabled or when User Interrupt is disabled. Not supported in this release.
63:32	USR1_MSG_UPPER_ADDR	RW	32	0	User 1 Message Upper Address Upper 32-bit address of MSI-X caused by User Interrupt 1. If this field is zero, 32-bit address messages are used. If this field is non-zero, 64-bit address messages are used. RO when MSI-X is disabled or when User Interrupt is disabled.
31:0	USR1_MSG_ADDR	RW	32	0	User 1 Message Address Lower 32-bit address of MSI-X caused by User Interrupt 1. For proper DWORD alignment, software must always write zeroes to LSB two bits; otherwise the result is undefined. RO when MSI-X is disabled or when User Interrupt is disabled.

All Other User Interrupt MSI-X Table (0x8020 to 0x83FF)

For User Interrupt 2 to 62, the register definition is similar to USR0_MSIX_TABLE and USR1_MSIX_TABLE, with only user interrupt number differences. In terms of offset, they are packed in incremental order after USR1_MSIX_TABLE, as each of them are 4DW size, same with USR0_MSIX_TABLE and USR1_MSIX_TABLE.

Table 2.61. PBA_TABLE (0xC000)

Field	Name	Access	Width	Default	Description
63:0	USR_INT_PB	RO	64	0	User Interrupt Pending Bit Each bit is associated to User Interrupt Pending Bits, where LSB refers to User Interrupt 0, in incremental order, up to User Interrupt 63. For each Pending Bit that is Set, the Function has a pending message for the associated MSI-X Table entry, which is suppressed by Function Mask bit and/or the corresponding Mask bit in Vector Control field in MSI-X Table.

2.12.2. Register Interface Conversion

2.12.2.1. APB Interface

This interface is available if the register interface type selected in the IP generation user interface is *APB.* You must provide a 512 kB aligned base address that is used when accessing the Core CSRs and PCIe Configuration Space registers.

Note: Due to a known bug in the APB user interface, it is recommended not using this configuration to prevent unexpected behavior.

The APB interface is used for configuring the PCIe registers when Non-DMA AXI-Stream interface is enabled.

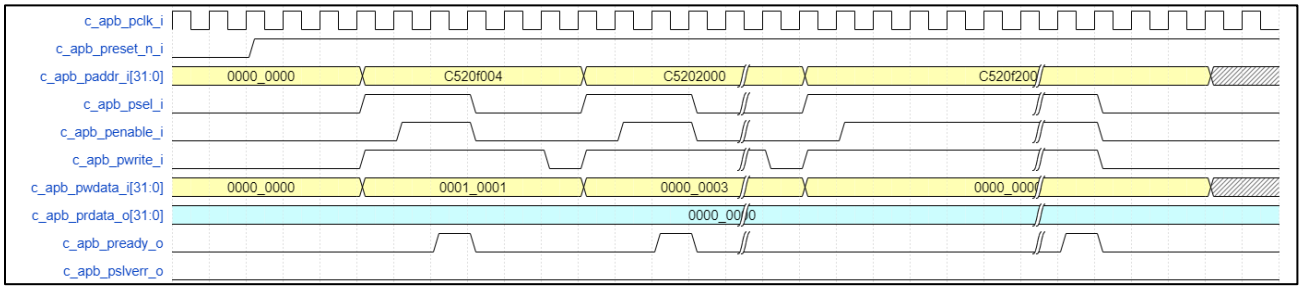

Figure 2.37. PCIe DMA APB Configuration

Figure 2.38 shows an example of the APB address bit signal configuration.

The APB Bus Address bits are mapped as follows:

- Bits[31:19] = CSR Base Address. The initial value is configurable in the PCIe IP user interface: *PCIe CSR Base Address*
- For Link Layer Register access (see registers in Hard IP Core Configuration and Status Registers)
  - Bits[18:16] == 3'b000
  - Bits[15:2] = Link Layer Register offset
  - Bits[1:0] must be tied to 2'b00.
- For PCIe Configuration Space Register access (see registers in the PCI Express Configuration Space Registers section):
  - Bits[18:16] == 3'b010
  - Bits[15:14] = 2'b00 – For Link 0
  - Bits[13:12] = Function number
  - Bits[11:2] = PCIe Configuration Register offset
  - Bits[1:0] must be tied to 2'b00.
- For Soft IP Configuration Register access (see registers in the PCI Express Configuration Space Registers section):
  - Bits[18:0]: base address 0x28000 + offset register
- For PHY register access (see Appendix A in CertusPro-NX SerDes/PCS User Guide (FPGA-TN-02245)):
  - Bits[18:16] == 3'b010
  - Bits[15:9] = 7'h60 – PHY Lane 0
  - Bits[8] = 0 – PMA register, 1 – MPCS register
  - Bits[7:0] = PHY Register offset
  - Bits[1:0] must be tied to 2'b00.

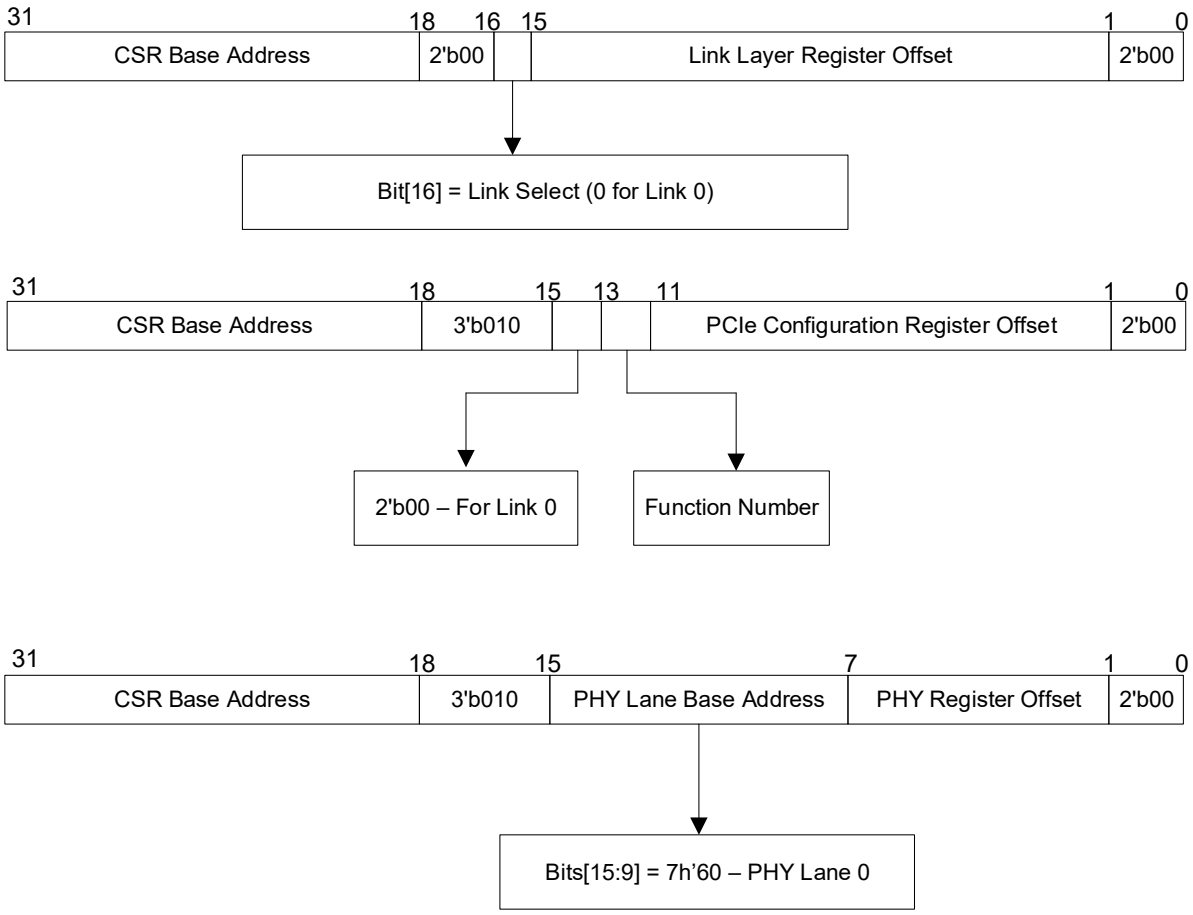

Figure 2.38. PCIe APB Register Set Address Bit Configuration

When you select the DMA Design by default, the configuration is performed through APB. The Core Configuration and Status Registers (CSR) are made accessible to the user design through the APB. The registers that are configured through APB are as follows:

- Register address – 0Xf004 [Base address:-0x0F000, Offset address:-0X4]
  This register is used to assert the PCIe core reset.
- Simulation registers:
  - Register address – 0X2000
    This register is used to reduce the ltssm ts_1 and timeouts to fasten the simulation when asserted as *1*.
  - Register address – 0X3000
    This register is used to reduce the Power Management State Machine timeouts to fasten the simulation when asserted as *1*.
  - Register address – 0X4000
    This register is used to reduce the timeouts to fasten the simulation when asserted as *1*.
- Register address – 0Xf200
  This register is used to read PLL status of each lane.

After configuring all registers, the *0Xf004* register is configured to get the PCIe core out of reset.

2.13. Resizable BAR Capability

The Resizable BAR capability is introduced to improve performance by negotiating the BAR size to optimize system resources. With this capability, the amount of address space consumed by the device can change.

2.13.1. Resizable BAR Registers Configuration

The configuration of the Resizable BAR Capability is done through the PCIe Configuration space register, as shown in Figure 2.39. The extended configuration space register set gives information about the address space size for that function.

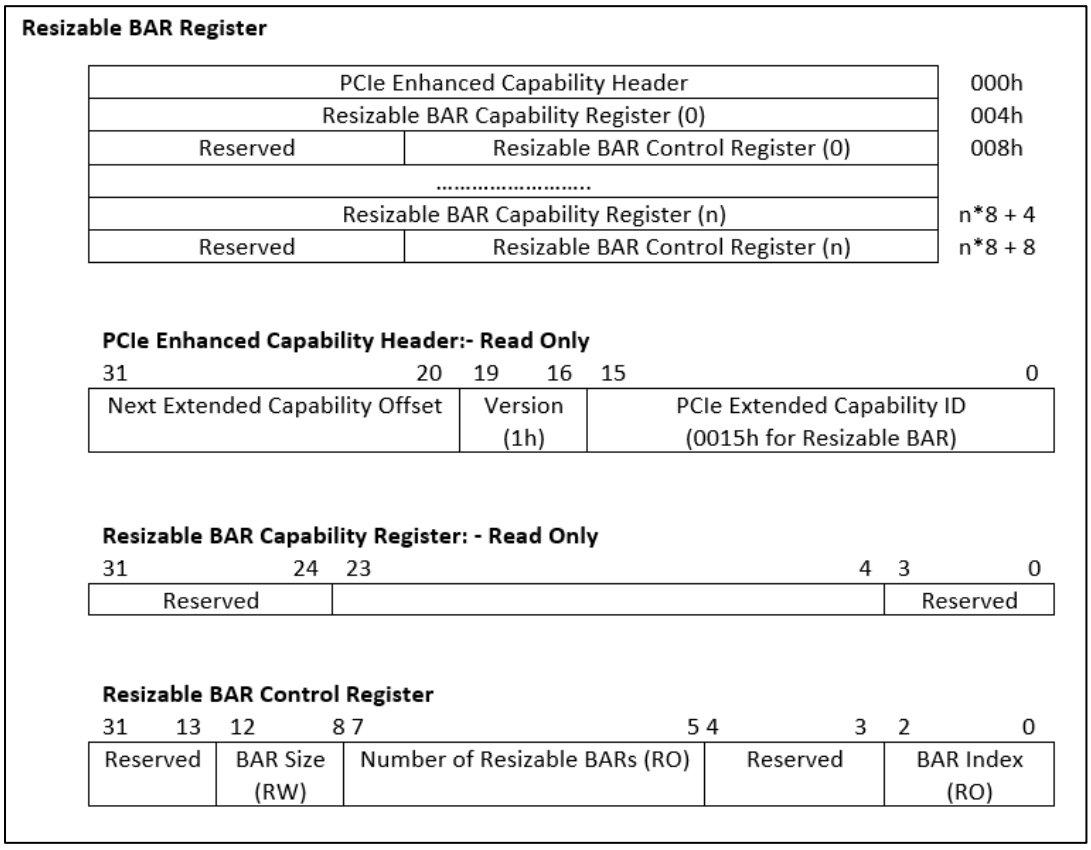

Figure 2.39. Resizable BAR Register Capability Structure

The Resizable BAR Capability and a Control register is implemented for each BAR that is resizable. Since a maximum of six BARs may be implemented by any Function, the Resizable BAR Capability structure can range from 12 bytes long (for a single BAR) to 52 bytes long (for all six BARs).

The Resizable BAR Capability Register gives information about the BAR sizes for the selected function. It is a Read Only register and the bits 4 to 23 is used to determine the BAR size, which is calculated using the following formula:

Consider 'n' indicates the bits between 4 to 23, so for nth bit the BAR size is,

BAR Size = 2^ (n + 16) bytes,

Bit [4] = 1 MB BAR size and Bit [23] = 512 GB BAR size

The Resizable BAR Control register gives information about the selection of the BAR (BAR 0 to BAR 5), the number of the Resizable field, which is only for Control Register zero to know how many of the six possible BAR's have adjustable size and the desired BAR size is programmed by the software for the BAR indicated by the BAR Index field.

Assuming *m* indicates the value of the BAR size field, (example Bits 12 to 8 is set to value m) then,

BAR size = 2^ (m + 20) bytes

If the field BAR Size is set to 3, then BAR size = 2^ (3 + 20) = 8 MB and the maximum value is when m = 19,

BAR size = 2 ^ (19+20) bytes = 512 GB

The enable and disable of the Resizable Register, the default and supported BAR size and the BAR index are programmed through the software by accessing the registers present in the PCIe Hard IP Core CSR.

For different functions, the following are the offset addresses for the resizable BAR capability configuration.

Table 2.62. Offset Address for Resizable Bar Capability Configurations

Offset Address	Description
0x1A0	Enable and Disable of Resizable BAR Capability
0x1A4	Resizable BAR Capability for BAR Configuration 0
0x1A8	Resizable BAR Capability for BAR Configuration 1
0x1AC	Resizable BAR Capability for BAR Configuration 2
0x1B0	Resizable BAR Capability for BAR Configuration 3
0x1B4	Resizable BAR Capability for BAR Configuration 4
0x1B8	Resizable BAR Capability for BAR Configuration 5

For more details on each bit in the register set for the Hard IP CSR and PCIe CSR, refer to the Register Description section.

3. IP Parameter Description

The PCIe Endpoint Core attributes are configurable through the IP Catalog's Module/IP wizard of the Lattice Radiant Software. Refer to Table 3.1 for the description of each attribute.

3.1. General

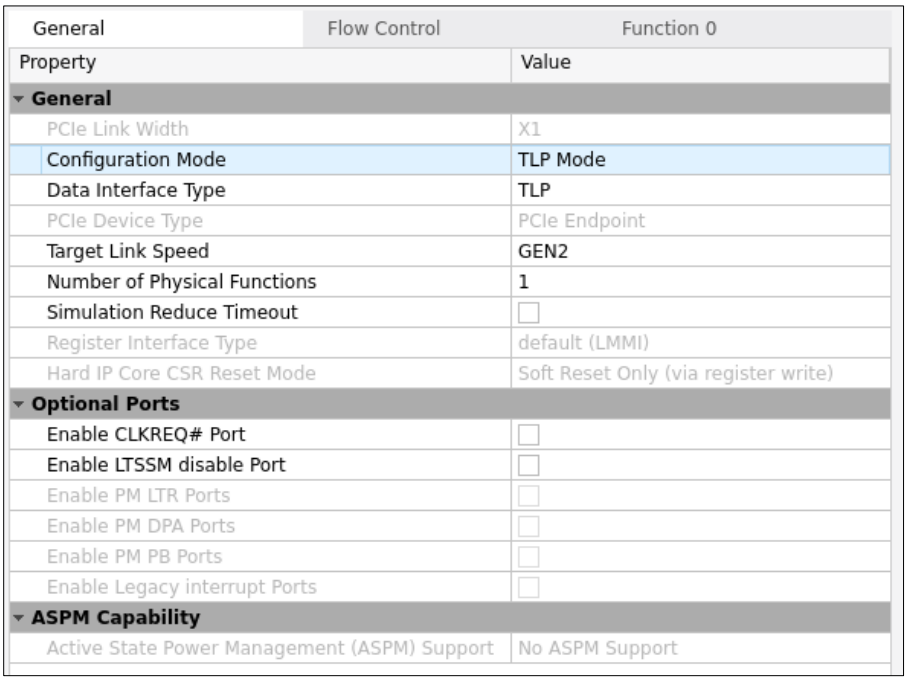

Figure 3.1. Attributes in the General Tab

Table 3.1. General Tab Attributes Descriptions

Attribute	Values	Description
General		
PCIe Link Width	x1	Display only.
Configuration Mode	"TLP Mode" "DMA Only Mode" "Bridge Mode" "DMA with Bridge Mode"	To select configuration mode. Available modes are TLP Mode, DMA Only, Bridge Mode, and DMA with Bridge Mode.
Data Interface Type	"TLP" "AXI_STREAM" "AXI_MM" "AXI_LITE"	Available option per mode: • "TLP Mode": "TLP" and "AXI_STREAM" • "DMA Only Mode": "AXI_MM". • "Bridge Mode": "AXI_MM" and "AXI_LITE". • "DMA with Bridge Mode": "AXI_MM".
PCIe Device Type	PCIe Endpoint	Display Only.
Target Link Speed	Gen1 Gen2	Initial value of Target Link Speed Configuration Register. Determines the maximum initial link speed which can be reached during initial training. Must be set to the lesser of the maximum speed supported by the core and the maximum speed at which the user desires the core to operate. Parameter: MGMT_FTL_INITIAL_TARGET_LINK_SPEEED = {0,1}

Attribute	Values	Description
Number of Physical Functions	1–4	Set the number of enabled functions. Parameter: NUM_FUNCTIONS = {1,2,3,4}
Simulation Reduce Timeout	Checked Unchecked	• Must be checked for simulation run. • Otherwise, leave it unchecked.
Data Interface Type	TLP, AXI4_STREAM, Default (Rx/Tx TLP IF) AXI4_LITE (version 2.0.0) AXI_MM (version 2.0.0)	Available if default interface is not selected. The selected interface replaces the native TLP data interface of the hard IP by adding a soft logic bridge. Parameter: USR_DAT_IF_TYPE = {"TLP", "AXI4_STREAM"} “AXI_MM”, “AXI4_LITE”}
AXI DMA Enabled	Checked Unchecked	Available if selected Data Interface Type is AXI_MM. Parameter: EN_AXI_DMA = {0, 1}
Register Interface Type	APB LMMI	APB is only available in TLP Mode with AXI-Stream data interface type. If APB is selected, APB replaces the native Lattice Memory Mapped Interface (LMMI) of the hard IP by adding a soft logic bridge. Parameter: USR_CFG_IF_TYPE = {"LMMI", "APB"}
Hard IP Core CSR Reset Mode	Soft Reset Only (through register write)	Display only.

3.2. Optional Port

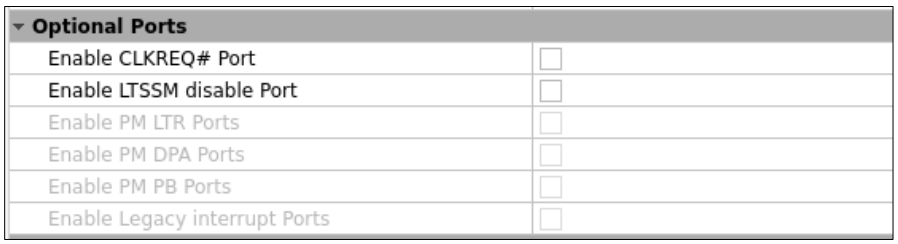

Figure 3.2. Attributes in the Optional Port Tab

Table 3.2. Optional Port Attributes

Attribute	Selectable Values	Description	Parameter
Enable CLKREQ# Port	Checked Unchecked	Set to add the link[LINK]_clkreq_n_io port.	LINK[k]_USE_CLKREQ_SIGNAL = {0,1}
Enable LTSSM disable Port	Checked Unchecked	Set to add the port to stop the LTSSM training. This port can be used to delay start of LTSSM training.	link[LINK]_ltssm_disable_i port.
Enable PM LTR Ports	Checked Unchecked	• Available if LTR capability is enabled. • Set to add the Latency Tolerance Reporting ports.	—

Attribute	Selectable Values	Description	Parameter
Enable PM DPA Ports	Checked Unchecked	• Available if DPA capability is enabled. • Set to add the Dynamic Power Allocation ports.	—
Enable PM PB Ports	Checked Unchecked	• Available if PB capability is enabled. • Set to add the Power Budgeting Ports.	—
Enable Legacy Interrupt Ports	Checked Unchecked	• Available if legacy interrupt is enabled. • Set to add the Legacy interrupt ports.	LINK[k]_MAIN_CTRL_4_EN_PORT_MGMT_INTERRUPT_LEG = {0,1}

3.3. ASPM Capability

Figure 3.3. Attributes in the ASPM Capability Tab

ASPM is not supported in the current version.

3.4. DMA/Bridge Mode Support

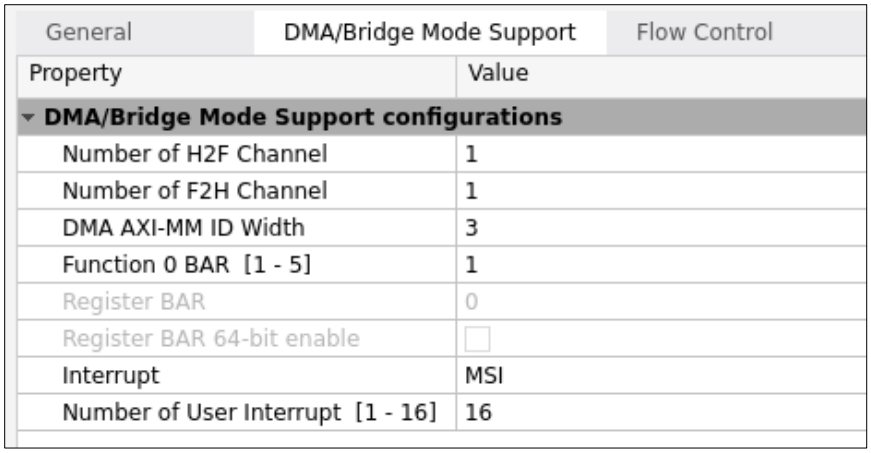

Figure 3.4. DMA/Bridge Mode User Interface

Table 3.3. DMA/ Bridge Mode Support Attributes

Attribute	Selectable Values	Description
Number of H2F Channel	0 - 1	• Number of H2F channel. • Maximum 1 channel is supported in the current release. Parameter: NUM_H2F_CHAN = {0, 1}

Attribute	Selectable Values	Description
Number of F2H Channel	0 - 1	• Number of F2H channel. Maximum 1 channel is supported in the current release. Parameter: NUM_F2H_CHAN = {0, 1}
DMA AXI-MM ID Width	Integer	• Data width for AXI-MM interface's AWID, BID, ARID, and RID. Should use a value not greater than 8. Parameter: DMA_AXI_ID_WIDTH
Function 0 BAR	1 - 5	PCIe Endpoint BAR that is allocated for Bridge Mode.
Register BAR	0	• BAR mapping for DMA/ Bridge register. Only 0 is supported in the current release.
Register BAR 64-bit enable	Checked, Unchecked	• To select if DMA/ Bridge register BAR is 32 bits or 64 bits. Only unchecked (32-bit) is supported in the current release.
Interrupt	MSI, MSI-X	• DMA/ Bridge Interrupt mode. • With DMA: Only MSI is supported in the current release. • Bridge only: MSI or MSI-X.
Number of User Interrupt	MSI: 1 – 16 MSI-X: 1 - 64	Refer to "User Interrupt" chapter in IPUG for more detail

3.5. Flow Control Update

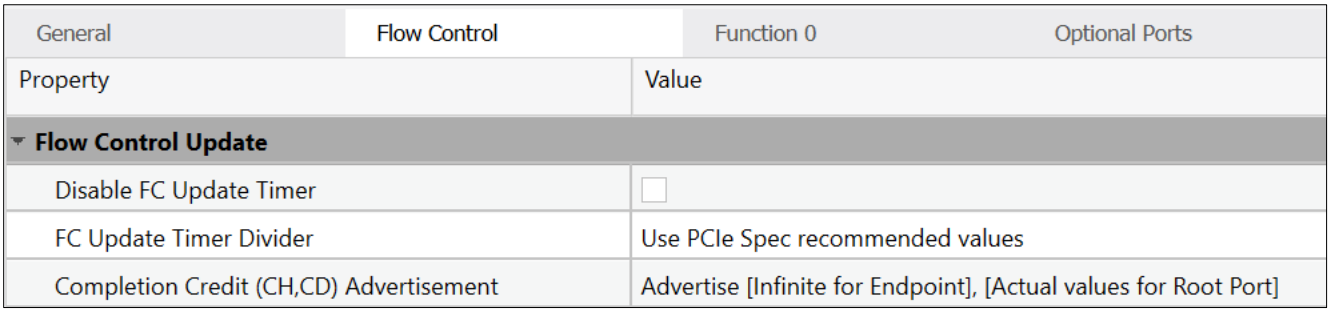

Figure 3.5. Attributes in the Flow Control Update Tab

Table 3.4. Flow Control Attributes

Attribute	Selectable Values	Description	Parameter
Disable FC Update Timer	Checked Unchecked	• Set to disable FC Update Timer (that is, schedule a FC Update on Every Consumed RX TLP • Otherwise, schedule FC Updates in accordance with PCIe Specification recommended values)	MGMT_PTL_RX_CTRL_FC_UPDATE_TIMER_DISABLE = {0,1}
FC Update Timer Divider	Use PCIe Spec recommended values, Divide by 2, Divide by 4, Divide by 8	Select the FC Update frequency of the Receive Buffer when FC update timer is enabled.	MGMT_PTL_RX_CTRL_FC_UPDATE_TIMER_DIV = {0,1,2,3}
Completion Credit (CH, CD) Advertisement	Advertise Infinite for Endpoint and Actual for Root Port, Advertise Actual, Advertise Infinite	Select the completion credit advertisement behavior.	MGMT_PTL_RX_CTRL_ADV_CH_CD_SEL = {0,1,2}

3.6. Receive Buffer Allocation

Receive Buffer Allocation	
Posted Header Credits (20 bytes/credit) [1 - 16]	16
Posted Data Credits (16 bytes/credit) [16 - 108]	108
Non-Posted Header Credits (20 bytes/credit) [1 - 8]	8
Non-Posted Data Credits (16 bytes/credit) [2 - 6]	6
Completion Header Credits (20 bytes/credit) [1 - 32]	32
Completion Data Credits (16 bytes/credit) [16 - 96]	96

Figure 3.6. Attributes in Receive Buffer Allocation Tab

Table 3.5. Receive Buffer Tab Attributes

Attribute	Selectable Values	Description	Parameter
Posted Header Credits	(20 bytes per credit) 1 – 16	• Set the amount of buffer credits for Posted TLP header. • The number of bytes required to allocate the requested PH and PD credits must not exceed the P RAM storage space (2 kB).	MGMT_PTL_RX_ALLOC_P_H = {1 - 16}

Attribute	Selectable Values	Description	Parameter
Posted Data Credits	(16 bytes per credit) 16–108	• Set the amount of buffer credits for Posted TLP data. • The number of bytes required to allocate the requested PH and PD credits must not exceed the P RAM storage space (2 kB).	MGMT_PTL_RX_ALLOC_P_D = {16 - 108}
Non-Posted Header Credits	(20 bytes per credit) 1–8	• Set the amount of buffer credits for Non-Posted TLP header. • The number of bytes required to allocate the requested NH and ND credits must not exceed the N RAM storage space (256 Bytes).	MGMT_PTL_RX_ALLOC_N_H = {1 - 8}
Non-Posted Data Credits	(16 bytes per credit) 2–6	• Set the amount of buffer credits for Non-Posted TLP data. • The number of bytes required to allocate the requested NH and ND credits must not exceed the N RAM storage space (256 Bytes).	MGMT_PTL_RX_ALLOC_N_D = {2 - 6}
Completion Header Credits	(16 bytes per credit) 1 – 32	• Set the amount of buffer credits for Completion TLP header. • The number of bytes required to allocate the requested CH and CD credits must not exceed the C RAM storage space (2 kB).	MGMT_PTL_RX_ALLOC_C_H = {1 - 32}
Completion Data Credits	(16 bytes per credit) 16 – 96	• Set the amount of buffer credits for Completion TLP data. • The number of bytes required to allocate the requested CH and CD credits must not exceed the C RAM storage space (2 kB).	MGMT_PTL_RX_ALLOC_C_D = {16 - 96}

3.7. Transmit Buffer Allocation

Receive Buffer Allocation	
Posted Header Credits (20 bytes/credit) [1 - 16]	16
Posted Data Credits (16 bytes/credit) [16 - 108]	108
Non-Posted Header Credits (20 bytes/credit) [1 - 8]	8
Non-Posted Data Credits (16 bytes/credit) [2 - 6]	6
Completion Header Credits (20 bytes/credit) [1 - 32]	32
Completion Data Credits (16 bytes/credit) [16 - 96]	96

Figure 3.7. Transmit Buffer Allocation Tab Attributes

Table 3.6. Transmit Buffer Tab Attributes

Transmit Buffer Allocation			
Attribute	**Selectable Values**	**Description**	**Parameter**
Posted Header Credits	(20 bytes per credit) 1–16	• Set the amount of buffer credits for Posted TLP header. • The number of bytes required to allocate the requested PH and PD credits must not exceed the P RAM storage space (2 kB).	MGMT_PTL_TX_ALLOC_P_H = {1 - 16}
Posted Data Credits	(16 bytes per credit) 16–108	• Set the amount of buffer credits for Posted TLP data. • The number of bytes required to allocate the requested PH and PD credits must not exceed the P RAM storage space (2 kB).	MGMT_PTL_TX_ALLOC_P_D = {16 - 108}
Non-Posted Header Credits	(20 bytes per credit) 1–8	• Set the amount of buffer credits for Non-Posted TLP header. • The number of bytes required to allocate the requested NH and ND credits must not exceed the N RAM storage space (256 Bytes).	MGMT_PTL_TX_ALLOC_N_H = {1 - 8}

Transmit Buffer Allocation			
Attribute	**Selectable Values**	**Description**	**Parameter**
Non-Posted Data Credits	(16 bytes per credit) 2–6	• Set the amount of buffer credits for Non-Posted TLP data. • The number of bytes required to allocate the requested NH and ND credits must not exceed the N RAM storage space (256 Bytes).	MGMT_PTL_TX_ALLOC_N_D = {2 - 6}
Completion Header Credits	(16 bytes per credit) 1–32	• Set the amount of buffer credits for Completion TLP header. • The number of bytes required to allocate the requested CH and CD credits must not exceed the C RAM storage space (2 kB).	MGMT_PTL_TX_ALLOC_C_H = {1 - 32}
Completion Data Credits	(16 bytes per credit) 16–96	• Set the amount of buffer credits for Completion TLP data. • The number of bytes required to allocate the requested CH and CD credits must not exceed the C RAM storage space (2 kB).	MGMT_PTL_TX_ALLOC_C_D = {16 - 96}

3.8. Function

3.8.1. Configuration

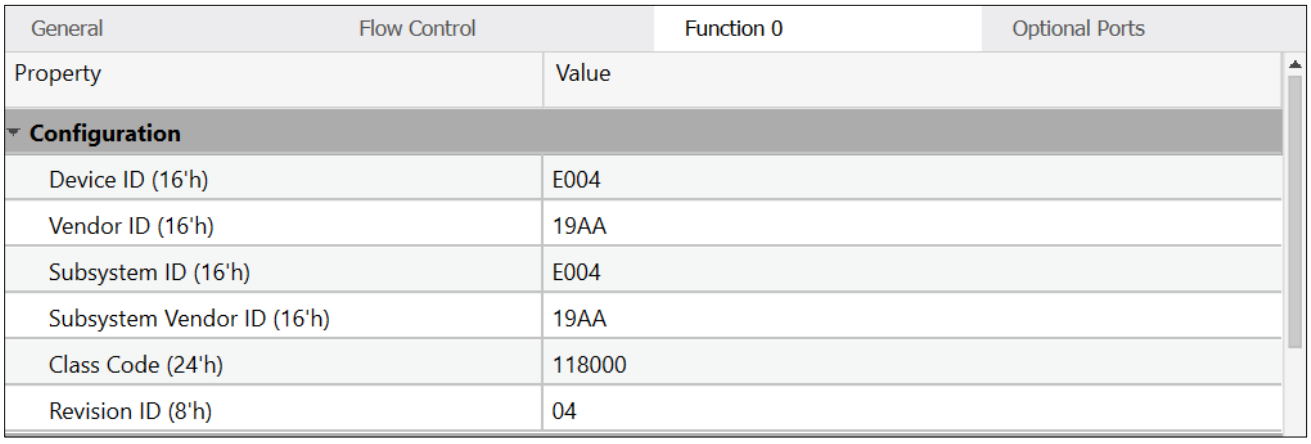

Figure 3.8. Attributes in Function Configuration Tab

Table 3.7. Function Configuration Tab Attributes

Configuration			
Attribute	**Selectable Values**	**Description**	**Parameter**
Disable Function	Unchecked	• Cannot disable function 0. • Display only.	—
Device ID	(Hex) 0000 – FFFF	Value returned when the Device ID Configuration Register is read.	MGMT_FTL_ID1_DEVICE_ID = {16'h0000 – 16'hFFFF}
Vendor ID	(Hex) 0000 – FFFF	Value returned when the Vendor ID Configuration Register is read.	MGMT_FTL_ID1_VENDOR_ID = {16'h0000 – 16'hFFFF}
Subsystem ID	(Hex) 0000 – FFFF	Value returned when the Subsystem ID Configuration Register is read.	MGMT_FTL_ID2_SUBSYSTEM_ID = {16'h0000 – 16'hFFFF}
Subsystem Vendor ID	(Hex) 0000 – FFFF	Value returned when the Subsystem Vendor ID Configuration Register is read.	MGMT_FTL_ID2_SUBSYSTEM_VENDOR_ID = {16'h0000 – 16'hFFFF}
Class Code	(Hex) 00000 – FFFFF	Value returned when the Class Code Configuration Register is read.	MGMT_FTL_ID3_CLASS_CODE = {16'h0000 – 16'hFFFF}
Revision ID	(Hex) 00 – FF	Value returned when the Revision ID Configuration Register is read.	MGMT_FTL_ID3_REVISION_ID = {8'h00 – 8'hFF}

3.8.2. Resizable Bar Capability

Figure 3.9. Attributes in Resizable Bar Capability Tab

Table 3.8. Resizable Bar Capability Attributes

Attributes	Value	Description	Parameters
Enable Resizable BAR Capability	Checked Unchecked	Set to enable the Resizable BAR Capability.	MGMT_FTL_RBAR_CAP_ENABLE = {0,1}

3.8.3. Base Address Register (BAR) [0 to 5]

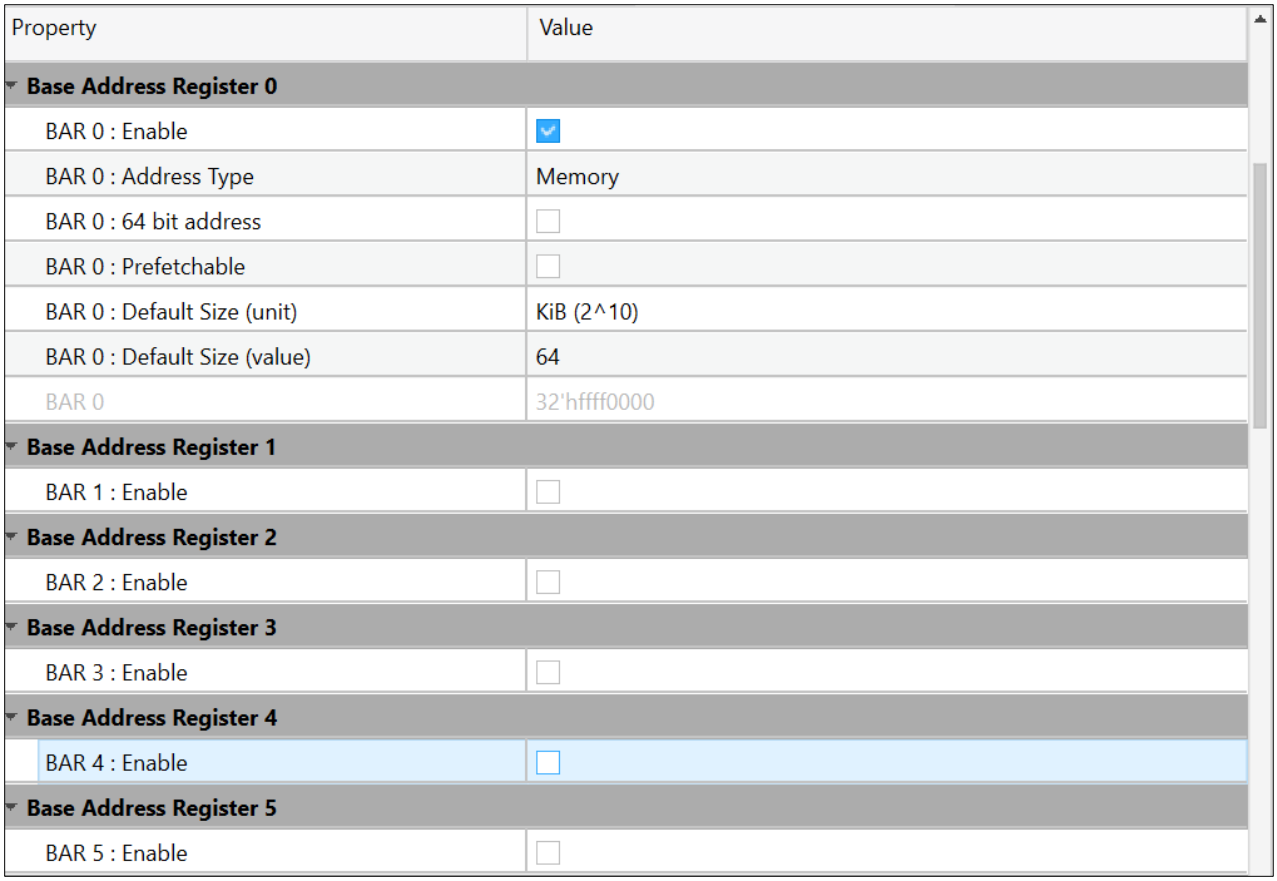

Figure 3.10. Attributes in BAR Tab

Table 3.9. BAR Tab Attributes

Base Address Register n (n == 0 - 5)			
Attribute	**Selectable Values**	**Description**	**Parameter**
BAR n – Enable	Checked Unchecked	Set to enable the BAR.	—
BAR n – Resizable	Checked Unchecked	Set to make this BAR resizable.	—
BAR n – Address Type	Memory, I/O	Select if the BAR is for Memory or I/O space.	—
BAR n – 64 bit Address	Checked Unchecked	• Applicable for memory space only. • Set to use 64-bit address. Note that BAR n and BAR n+1 are used for the 64-bit address.	—
BAR n – Prefetchable	Checked Unchecked	• Applicable for memory space only. • Set to identify the memory address as prefetchable.	—
BAR n – Resizable BAR Supported Sizes [23:4]	(Hex) 00000 – FFFFF	Each bit indicates a supported size which is $2^{(i+16)}$ bytes, where *i* is the index from [23:4]. For example, if bit[4] == 1, then $2^{(4+16)}$ Bytes = 1 MB	—
BAR n – Default Size	Bytes,	Select the size of Memory space.[1]	—

Base Address Register n (n == 0 - 5)			
Attribute	**Selectable Values**	**Description**	**Parameter**
(unit)	KB (2^{10}), MB (2^{20}), GB (2^{30}), TB (2^{40}), PB (2^{50}), EB (2^{60}),		
BAR n – Default Size (value)	(Power of 2) 32 bits Memory Space: 16 bytes – 2 GB 64 bits Memory Space: 64 bits: 4 GB – 8 EB 32 bits I/O Space: 2 Bytes – 256 Bytes	Select the size of Memory or I/O space.[1]	—
BAR n	32 bits: FFFF_FFF0 - 1000_0000 64 bits: FFFF_FFFF_0000_0000 - 1000_0000_0000_0000	Display Only	Function 0: MGMT_FTL_BAR0_CFG … … MGMT_FTL_BAR5_CFG Function m: MGMT_FTL_MF1_BAR0_CFG … … MGMT_FTL_MF[m]_BAR[n]_CFG
Local Memory Base Address n	(Hex, Aligned to BAR size) FFFF_FFF0 – 0000_0000	• Applicable for memory space only. • This is the base address of the local system memory that maps to the configured PCIe BAR. • Must be aligned to the specified BAR size. • Received Memory requests that hits the BAR are forwarded to this address.	Function 0: F0BAR0_TO_LOCADR … … … F0BAR5_TO_LOCADR Function m: F1BAR0_TO_LOCADR … … … F[m]BAR[n]_TO_LOCAD

Note:

1. For Resizable BAR, this is the default size.

3.8.4. Legacy Interrupt

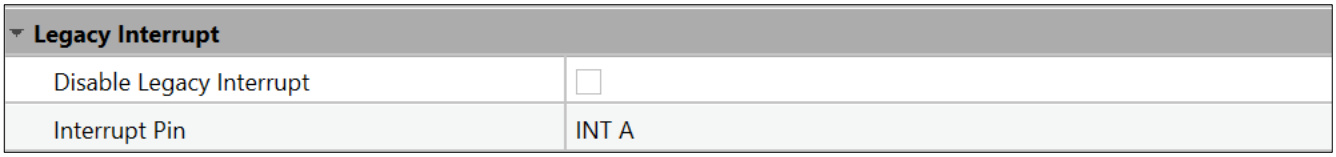

Figure 3.11. Attributes in Legacy Interrupt

Table 3.10. Legacy Interrupt Attribute Descriptions

Attributes	Value	Description	Parameters
Disable Legacy Interrupt	Checked Unchecked	• RTL always supports legacy interrupt. • The current attribute only uses for port activation.	MGMT_FTL_INTERRUPT_DISABLE = {0,1}
Interrupt Pin	INT A, INT B, INT C, INT D	Select which legacy interrupt pin is used.	MGMT_FTL_INTERRUPT_PIN = {0,1,2,3}

3.8.5. MSI Capability

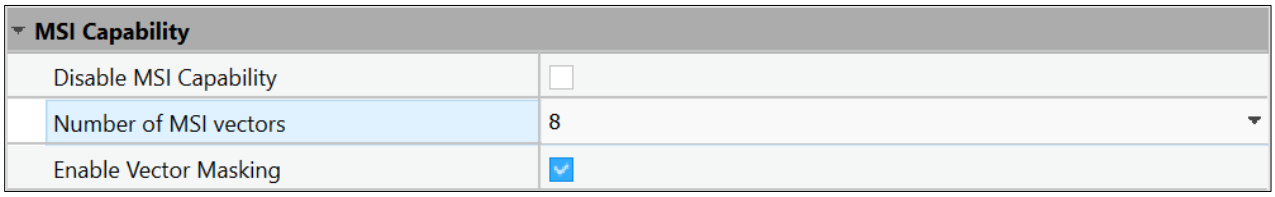

Figure 3.12. Attributes in MSI Capability

Table 3.11. MSI Capability Attributes

Attributes	Value	Description	Parameters
Disable MSI Capability	Checked Unchecked	Set to disable the MSI Capability.	MGMT_FTL_MSI_CAP_DISABLE = {0,1}
Number of MSI vectors	1,2,4,8,16,32	Set the number of requested MSI vectors.	MGMT_FTL_MSI_CAP_MULT_MESSAGE_CAPABLE = {0,1,2,3,4,5}
Enable Vector Masking	Checked Unchecked	Set to enable vector masking capability.	MGMT_FTL_MSI_CAP_VEC_MASK_CAPABLE = {0,1}

3.8.6. MSI-X Capability

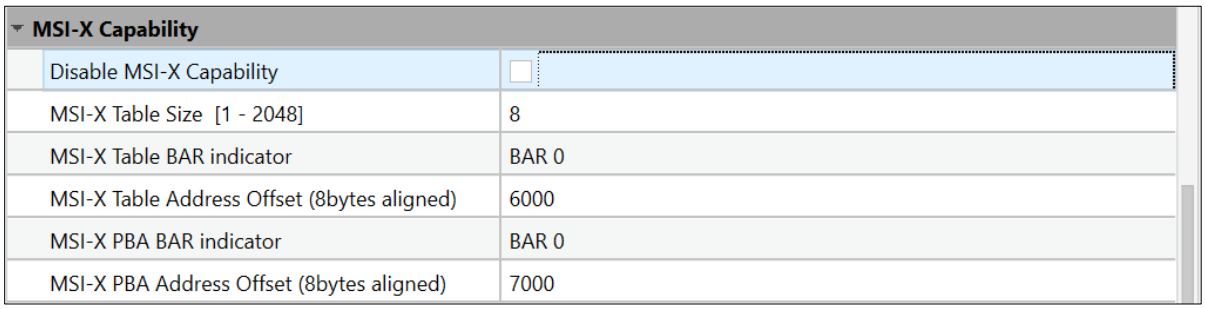

Figure 3.13. Attributes in MSI-X Capability

Table 3.12. MSI-X Capability Attributes

Attributes	Value	Description	Parameters
Disable MSI-X Capability	Checked Unchecked	Set to disable the MSI-X Capability.	MGMT_FTL_MSIX_CAP_DISABLE = {0,1}
MSI-X Table Size	1–2048	Set the number of requested MSI-X vectors.	MGMT_FTL_MSIX_CAP_TABLE_SIZE = {0 – 2047}
MSI-X Table BAR indicator	BAR 0, BAR 1, BAR 2, BAR 3, BAR 4, BAR 5	• Select which Base Address register. • Located beginning at 10h in Configuration Space, is used to map the MSI-X Table into Memory Space.	MGMT_FTL_MSIX_TABLE_BIR = {0,1,2,3,4,5}
MSI-X Table Address Offset	(Hex, 8 bytes aligned) 0000_0000 – FFFF_FFF8	Set the byte address offset (8 bytes aligned), within the BAR selected by MSI-X Table BAR indicator, at which the MSI-X Table begins.	MGMT_FTL_MSIX_TABLE_OFFSET = {29'h00000000 – 29'h1FFFFFF8}
MSI-X PBA BAR indicator	BAR 0, BAR 1, BAR 2, BAR 3, BAR 4, BAR 5	Select which Base Address register, located beginning at 10h in Configuration Space, is used to map the MSI-X PBA into Memory Space.	MGMT_FTL_MSIX_PBA_BIR = {0,1,2,3,4,5}
MSI-X PBA Address Offset	(Hex, 8 bytes aligned) 0000_0000 – FFFF_FFF8	Set the byte address offset (8 bytes aligned), within the BAR selected by MSI-X PBA BAR indicator, at which the MSI-X PBA begins.	MGMT_FTL_MSIX_PBA_OFFSET = {29'h00000000 – 29'h1FFFFFF8}

3.8.7. Device Serial Number Capability

Figure 3.14. Attributes in Device Serial Number Capability

Table 3.13. Device Serial Number Capability Attributes

Link [k] (k == 0 - 1) Device Serial Number Capability			
Attributes	**Value**	**Description**	**Parameters**
Enable DSN Capability	Checked Unchecked	Set to enable the Device Serial Number capability.	MGMT_FTL_DSN_CAP_ENABLE = {0,1}
Serial Number	(Hex) 0000_0000_0000_0000 – FFFF_FFFF_FFFF_FFFF	Set the device serial number.	MGMT_FTL_DSN_SERIAL_NUMBER

3.8.8. PCIe Capability

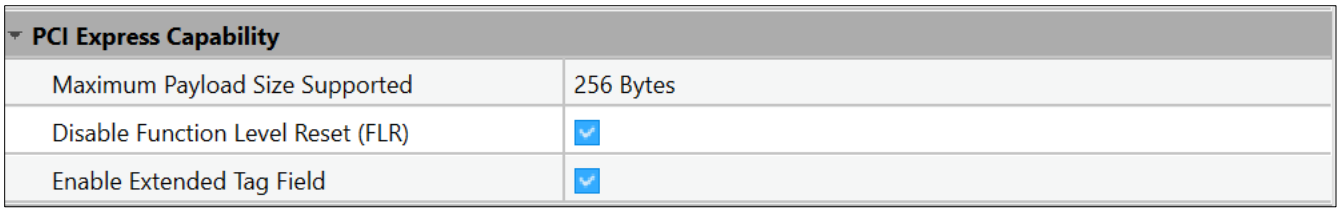

Figure 3.15. Attributes in PCIe Capability

Table 3.14. PCIe Capability Attributes

Link [k] (k == 0 - 1) PCIe Device Capability			
Attributes	**Value**	**Description**	**Parameters**
Maximum Payload Size Supported	128 Bytes, 256 Bytes, 512 Bytes	Select the maximum payload size supported.	MGMT_FTL_PCIE_DEV_CAP_MAX_PAYLOAD_SIZE_SUPPORTED = {0,1,2}
Disable Function Level Reset (FLR)	Checked Unchecked	Set to disable Function Level Reset capability.	MGMT_FTL_PCIE_DEV_CAP_DISABLE_FLR_CAPABILITY = {0,1)
Enable Extended Tag Field	Checked Unchecked	Set to enable Extended Tag Field (8-bit tag field).	MGMT_FTL_PCIE_DEV_CAP_EXTENDED_TAG_FIELD_SUPPORTED = {0,1}

3.8.9. Advance Error Reporting Capability

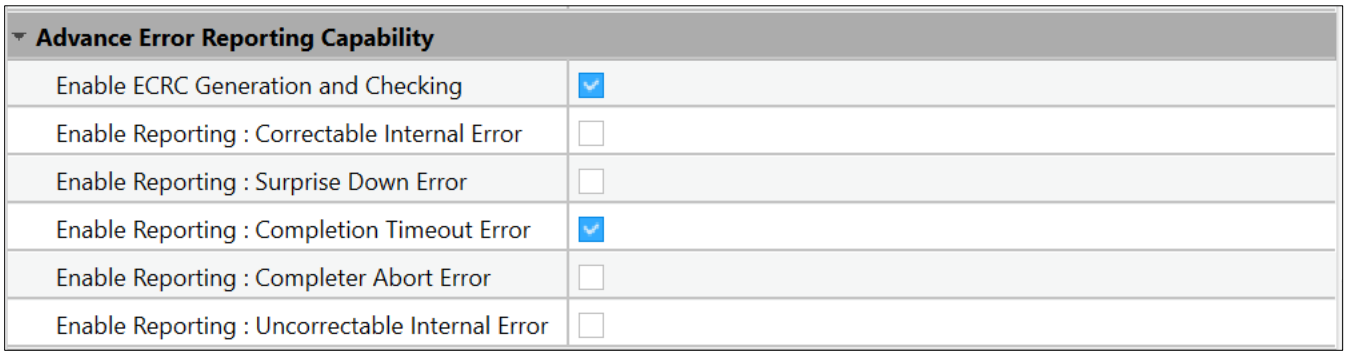

Figure 3.16. Attributes in Advance Error Reporting Capability

Table 3.15. Advance Error Reporting Capability Attributes

Attributes	Value	Description	Parameters
Enable ECRC Generation and Checking	Checked Unchecked	Set to enable ECRC generation and checking.	MGMT_FTL_AER_CAP_ECRC_GEN_CHK_CAPABLE = {0,1}
Enable Reporting: Correctable Internal Error	Checked Unchecked	Set to enable reporting of correctable internal error.	MGMT_FTL_AER_CAP_EN_CORR_INTERNAL_ERROR = {0,1}
Enable Reporting: Surprise Down Error	Checked Unchecked	Set to enable reporting of surprise down error.	MGMT_FTL_AER_CAP_EN_SURPRISE_DOWN_ERROR = {0,1}
Enable Reporting: Completion Timeout Error	Checked Unchecked	Set to enable reporting of completion timeout error.	MGMT_FTL_AER_CAP_EN_COMPLETION_TIMEOUT = {0,1}
Enable Reporting: Completer Abort Error	Checked Unchecked	Set to enable reporting of completer abort error.	MGMT_FTL_AER_CAP_EN_COMPLETER_ABORT = {0,1}
Enable Reporting: Uncorrectable Internal Error	Checked Unchecked	Set to enable reporting of uncorrectable internal error.	MGMT_FTL_AER_CAP_EN_UCORR_INTERNAL_ERROR = {0,1}

3.8.10. ATS Capability

Figure 3.17. Attributes in ATS Capability

Table 3.16. ATS Capability Attribute Description

Link [k] (k == 0 - 1) ATS Capability			
Attributes	**Value**	**Description**	**Parameters**
Enable ATS Capability	Checked Unchecked	Set to enable the ATS Capability.	MGMT_FTL_ATS_CAP_ENABLE = {0,1}

3.8.11. Atomic OP Capability

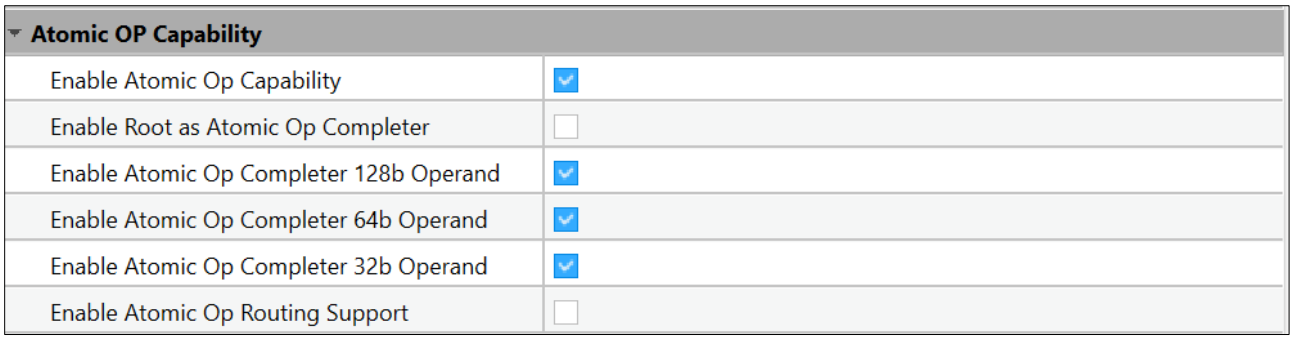

Figure 3.18. Attributes in Atomic OP Capability

Table 3.17. Atomic OP capability Attributes

Attributes	Value	Description	Parameters
Enable Atomic Op Capability	Checked Unchecked	Set to enable Atomic Operations Capability.	MGMT_FTL_ATOMIC_OP_CAP_ENABLE = {0,1}
Enable Root as Atomic Op Completer	Checked Unchecked	Set to enable Root as Atomic OP Completer.	MGMT_ FTL_ATOMIC_OP_CAP_RP_COMPLETER_ENABLE = {0,1}
Enable Atomic Op Completer 128b Operand	Checked Unchecked	Set to support Atomic Op 128b operand.	MGMT_FTL_ATOMIC_OP_CAP_COMPLETER_128_SUPPORTED = {0,1}
Enable Atomic Op Completer 64b Operand	Checked Unchecked	Set to support Atomic Op 64b operand	MGMT_FTL_ATOMIC_OP_CAP_COMPLETER_64_SUPPORTED = {0,1}
Enable Atomic Op Completer 32b Operand	Checked Unchecked	Set to support Atomic Op 32b operand.	MGMT_FTL_ATOMIC_OP_CAP_COMPLETER_32_SUPPORTED = {0,1}
Enable Atomic Op Completer Routing	Checked Unchecked	Set to support Atomic Op routing.	MGMT_ FTL_ATOMIC_OP_CAP_ROUTING_SUPPORTED = {0,1}

3.8.12. Latency Tolerance Reporting Capability

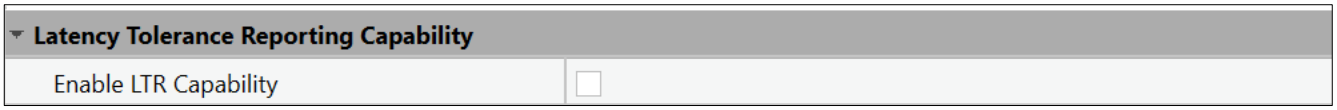

Figure 3.19. Attributes in Latency Tolerance Reporting Capability

Table 3.18. Latency Tolerance Reporting Capability Attributes

Attributes	Value	Description	Parameters
Enable LTR Capability	Checked Unchecked	Set to enable the Latency Tolerance Reporting capability.	MGMT_FTL_LTR_CAP_ENABLE = {0,1}

3.8.13. Power Budgeting Capability

Figure 3.20. Attributes in Power Budgeting Capability

Table 3.19. Power Budgeting Capability Attributes

Attributes	Value	Description	Parameters
Enable PB Capability	Checked Unchecked	Set to enable the Power Budgeting capability.	MGMT_FTL_PWR_BUDGET_CAP_ENABLE = {0,1}

3.8.14. Dynamic Power Allocation Capability

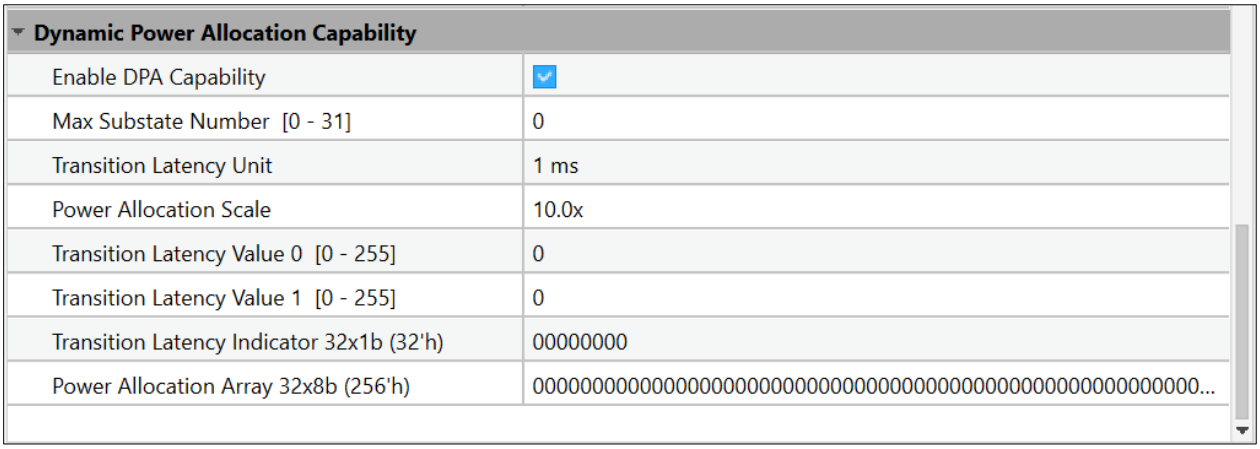

Figure 3.21. Attributes in Dynamic Power Allocation Capability

Table 3.20. Dynamic Allocation capability Attributes

Attributes	Value	Description	Parameters
Enable DPA Capability	Checked Unchecked	Set to enable the Dynamic Power Allocation capability.	MGMT_FTL_DPA_CAP_ENABLE = {0,1}
Max Substate Number	0–31	• Specifies the maximum substate number. • Substates from [substate_max:0] are supported. • For example, substate_max==0 indicates support for 1 substate.	MGMT_FTL_DPA_CAP_SUBSTATE_MAX = {0 - 31}
Transition Latency Unit	1 ms, 10 ms, 100 ms	Specifies Transition Latency Unit.	MGMT_FTL_DPA_CAP_TLUNIT = {0 - 2}
Power Allocation Scale	10.0x, 1.0x, 0.1x, 0.01x	Specifies Power Allocation Scale.	MGMT_FTL_DPA_CAP_PAS = {0 - 3}
Transition Latency Value 0	0–255	Specifies Transition Latency Value 0.	MGMT_FTL_DPA_CAP_XLCY0 = {0 - 3}
Transition Latency Value 1	0–255	Specifies Transition Latency Value 1.	MGMT_FTL_DPA_CAP_XLCY1 = {0 - 3}
Transition Latency Indicator 32x1b	(Hex) 00000000 – FFFFFFFF	• Specifies which Transition Latency Value applies to each substate. • Each bit corresponds to a substate.	MGMT_ FTL_DPA_XLCY_INDICATOR = {32'h00000000 – 32'hFFFFFFFF}
Power Allocation Array 32x8b	(Hex) {32{00}} – {32{FF}}	• Substate Power Allocation Array. • Each entry is 8b value.	MGMT_ FTL_DPA_ALLOC_ARRAY = { {32{8'h00}} – {32{8'hFF}} }

Table 3.21. Function 1-3 Tab

Function n (n == 1 – 3)			
Configuration			
Disable Function	Checked Unchecked	Available if the number of physical functions enabled is set to greater than 1. Set to disable the function.	Parameter: MGMT_FTL_MF1_FUNCTION_DISABLE = {0,1} MGMT_FTL_MF2_FUNCTION_DISABLE = {0,1} MGMT_FTL_MF3_FUNCTION_DISABLE = {0,1}
Device ID	Refer to Function section.		—
Vendor ID			
Subsystem ID			
Subsystem Vendor ID			—
Class Code			—
Revision ID			—

Function n (n == 1 – 3)	
Base Address Register (see the Lattice PCIe x1 Core Configuration user interface in Function section)	—
Legacy Interrupt (see the Lattice PCIe x1 Core Configuration user interface in Function section)	—
MSI Capability (see the Lattice PCIe x1 Core Configuration user interface in Function section)	—
MSI-X Capability (see the Lattice PCIe x1 Core Configuration user interface in Function section)	—
Device Serial Number Capability (see the Lattice PCIe x1 Core Configuration user interface in Function section)	—

4. Signal Description

The Lattice PCIe x1 IP Core Ports are defined in the following sub sections.

4.1. Clock Interface

Table 4.1. Clock Ports

Port	Type	Description

sys_clk_i	Input	• This signal is the User Clock Domain Input Clock
		• It is recommended to use the following minimum clock frequency to achieve the maximum throughput with respect to link data rate:
		  • 5.0G – 125 MHz
		  • 2.5G – 62.5 MHz
		• All ports of the IP, except for the APB and LMMI interfaces, are synchronized to this input clock.
		Note: The u_clk_period_in_ps register (0xF00C) should be updated with the actual value of the clock period used in sys_clk_i.
		You must ensure that sys_clk_i has an active (toggling) clock input when de-asserting the reset ports (link0_perst_n_i/link0_rst_usr_n_i), otherwise the core may get stuck at reset.

link0_clk_usr_o	Output	• This signal is the User Clock Domain Output Clock. • This is the pclk output that comes from the PHY. • By default, the link0_clk_usr_o uses the divide-by-2 version (125 MHz) of the pclk from PHY. • For TLP interface, you have the option to use this clock as input to sys_clk_i. For Non-DMA AXI-Stream interfaces, you must not use this clock as input to sys_clk_i – a separate clock source or PLL is needed for sys_clk_i. **Note:** clk_usr_o is inactive (stays low) when PHY is on reset (perst_n_i is asserted or the register pipe_rst (0x0F004) is asserted).
refclkp_i	Input	Differential Reference Clock, CLK+ (default 100 MHz)
refclkn_i	Input	Differential Reference Clock, CLK- (default 100 MHz)
link0_aux_clk_i	Input	• This is the low-speed auxiliary clock (16 MHz minimum). • This clock is required when L1 Substate is enabled. • During low power mode when the Core enters L1 substate (L1.1 or L1.2), the PHY turns off most of the power consuming blocks including PLLs, thus turning off the Link Layer clock. • The link0_aux_clk_i serves as an always on clock that is used by the Link Layer to wake up and exit from L1 substate. **Note:** The aux_clk_period_in_ps register (0xF010) should be updated with the actual value of the clock period used in link0_aux_clk_i.
link0_clkreq_n_io	InOut	• This signal is the CLKREQ# bidirectional open-drain pin. • The CLKREQ# signal is an open drain, active low signal that is driven low by the add-in card to request that the PCI Express reference clock be available (active clock state) to allow the PCI Express interface to send/receive data. • Operation of the CLKREQ# signal is determined by the state of the Enable Clock Power Management bit in the Link Control Register (offset 010h). • When disabled, the CLKREQ# signal is always asserted (link0_clkreq_n_io = 1'b0) whenever power is applied to the card, with the exception that it may be de-asserted during L1 PM Substates. • When enabled, the CLKREQ# signal may be de-asserted (link0_clkreq_n_io = 1'b1) during an L1 Link state. • The CLKREQ# signal is also used by the L1 PM Substates mechanism. In this case, CLKREQ# can be asserted by either the system or add-in card to initiate an L1 exit.

Port	Type	Description
		• See the PCI Express Base Specification for details on the functional requirements for the CLKREQ# signal when implementing L1 PM Substates. • Whenever dynamic clock management is enabled and when a card stops driving CLKREQ# low, it indicates that the device is ready for the reference clock to transition from the active clock state to a parked (not available) clock state. Reference clocks are not guaranteed to be parked by the host system when CLKREQ# gets de-asserted and module designs shall be tolerant of an active reference clock even when CLKREQ# is de-asserted by the module. **Note:** This signal must be tied to low if CLKREQ# is not used.
rext_i	Input	External Resistance
refret_i	Input	Analog reference return for PMA PLL

4.2. Reset Interface

Table 4.2. Reset Ports

Port	Clock Domain	Type	Description
link0_perst_n_i	Asynchronous	Input	• This signal is the PCI Express Fundamental Reset. • Active-low asynchronous assert, synchronous de-assert (synchronous to sys_clk_i) • Reset the Link Layer, PHY, and Soft Logic blocks. • On link0_perst_n_i and link0_rst_usr_n_i de-assertion the core starts in the Detect.Quiet Link Training and Status State Machine (LTSSM) state with the Physical Layer down and Data Link Layer down. • link0_perst_n_i must remain asserted while the PHY registers are being configured.
link0_rst_usr_n_i	Asynchronous	Input	• User Clock Domain Link Layer Reset (Link Layer Reset). • Asynchronous assert, synchronous de-assert reset to the User clock domain, Link Layer and Soft Logic blocks. • On link0_perst_n_i and link0_rst_usr_n_i de-assertion the core starts in the Detect.Quiet Link Training and Status State Machine (LTSSM) state with the Physical Layer down and Data Link Layer down. • It is recommended that link0_rst_usr_n_i remains asserted while the Link Layer core registers are being configured.
link0_flr_o [NUM_FUNCTIONS-1:0]	sys_clk_i	Output	• Per function Function Level Reset (FLR) indicator • link0_flr_o [i] == 1 indicates FLR is active for function[i] • link0_flr_o [i] == 0 indicates FLR is not active for function[i] • FLR is a function-specific soft reset that occurs when software writes the FLR register in a function's configuration space to 1. When FLR is active, the function's Configuration Space registers are reset to the default values (except Sticky registers as specified by PCIe Specification). • A function's FLR Configuration Space register remains set until link0_flr_ack_i[i] for the associated function[i] is set to 1 for one clock to indicate that you completed resetting the application logic associated with that function.

Port	Clock Domain	Type	Description
link0_flr_ack_i [NUM_FUNCTIONS-1:0]	sys_clk_i	Input	• Per function Function Level Reset (FLR) acknowledge. • Set link0_flr_ack_i [i] == 1 for one clock to indicate that you completed processing an active link0_flr_o[i] for function[i] and is ready to exit FLR for the function. • FLR is only enabled for Endpoints. • FLR support may be disabled through the mgmt_ftl_pcie_dev_cap_disable_flr_capability register, except per PCIe Specification.

4.3. PHY Interface

The Link Layer is used in conjunction with a third-party PCI Express PHY to implement a complete Lattice PCIe x1 Core PCI Express implementation. The PHY implements the high-speed serial and analog functions required to support PCI Express while the Link Layer implements majority of the digital logic as well as the higher levels of the PCI Express protocol.

The PIPE PHY Interface that connects the Link Layer and PHY is not shown here since the interface is only internal and is not visible to you.

Table 4.3. PHY Interface Descriptions

Port	Clock Domain	Type	Description
link0_rxp_i	refclkp_i/refclkn_i	Input	• Differential Receive Serial signal, Rx+
link0_rxn_i	refclkp_i/refclkn_i	Input	• Differential Receive Serial signal, Rx-
link0_txp_o	refclkp_i/refclkn_i	Output	• Differential Transmit Serial signal, Tx+
link0_txn_o	refclkp_i/refclkn_i	Output	• Differential Transmit Serial signal, Tx-
link0_pl_link_up_o	sys_clk_i	Output	• Physical Layer Link Up Status 1 – Link is UP 0 – Link is Down • link0_pl_link_up_ois used as an active-low, synchronous reset for the core's Data Link Layer • You are not expected to use this port except for status since the RTL does not interface directly with the Data Link Layer.
link0_dl_link_up_o	sys_clk_i	Output	• Data Link Layer Link Up Status 1 – Link is UP 0 – Link is Down • link0_dl_link_up_o is used as an active-low, synchronous reset for the Transaction Layer and indicates when TLPs can be successfully transmitted across the link. • For Endpoint-only applications, users must use link0_dl_link_up_o as a synchronous reset for the RTL interfacing to the core's Transaction Layer interfaces.
link0_tl_link_up_o	sys_clk_i	Output	• Transaction Layer Link Up Status. 1 – Link is UP 0 – Link is Down • link0_tl_link_up_o is an active-low, synchronous reset to the core's upper transaction layer.

Port	Clock Domain	Type	Description
link0_ltssm_disable_i	asynchronous	Input	• The LTSSM does not transition from Detect.Quiet to Detect.Active to begin LTSSM training when link0_ltssm_disable_i==1. • link0_ltssm_disable_imay thus be used to delay the start of LTSSM training which otherwise begins as soon as link0_perst_n_iand link0_rst_usr_n_iare deasserted. • link0_ltssm_disable_imust be set to 1 relatively soon (within a few ms) after link0_perst_n_iand link0_rst_usr_n_iare released as the system allocates a finite amount of time for devices to initialize before it begins to scan for devices. • If link0_ltssm_disable_iis held for too long, the software may scan for the device before it becomes operational and assume that no device is present.

Note:

1. NUM_FUNCTIONS – range (1-4)

4.4. Transaction Layer Interface

4.4.1. TLP Transmit Interface

Refer the TLP Transmit Interface section for more information and timing diagrams.

4.4.1.1. TLP Transmit Interface Port Description

Table 4.4. TLP Transmit Interface Ports

Port	Clock Domain	Direction	Description
link0_tx_valid_i	sys_clk_i	Input	Source valid. (1==Valid, 0==Not valid)
link0_tx_ready_o	sys_clk_i	Output	• Destination ready. (1==Ready, 0==Not ready) • A transfer occurs on the transmit interface only when link0_tx_valid_i==link0_tx_ready_o==1.
link0_tx_sop_i	sys_clk_i	Input	Start of packet indicator. Set == 1 coincident with the first link0_tx_data_iword in each TLP.
link0_tx_eop_i	sys_clk_i	Input	End of packet indicator Set == 1 coincident with the last link0_tx_data_iword in each TLP.
link0_tx_eop_n_i	sys_clk_i	Input	• Nullify packet indicator. • Set == 1 coincident with link0_tx_eop_i== 1 to instruct the core to nullify the current TLP (invert LCRC and use EDB framing) instead of transmitting the TLP normally.

Port	Clock Domain	Direction	Description
link0_tx_data_i [31:0]	sys_clk_i	Input	• TLP data to transfer. • link0_tx_data_imust be valid from the assertion of link0_tx_sop_iuntil the TLP is fully consumed with the assertion of link0_tx_eop_i. • TLP data must comprise a complete Transaction Layer Packet (TLP) as defined by the PCI Express Specification including the entire 3 or 4 DWORD TLP header, data payload (if present), and optionally a TLP Digest (ECRC). The core adds the necessary STP/END/EDB framing, Sequence Number, LCRC, and add ECRC (if enabled to do so and ECRC is not already present in the transmission) as part of its Data Link Layer functionality. • Transmitted TLPs are required to be formulated correctly per the PCIe Specification including filling in the Requester/Completer ID, Attributes, and Traffic Class. For Multi-Function, the core uses the Requestor/Completer ID in transmitted TLPs to determine which function's Configuration Registers should be applied to determine the validity of the transmitted TLP.
link0_tx_datap_i [3:0]	sys_clk_i	Input	• Parity of associated vc_tx_data_i and evaluated as: link0_tx_datap_i [i] == ^(link0_tx_data_i [((i+1)*8)-1:(i*8)]). Parity width changes as per data width (one parity bit per 8 bits of data is generated). link0_tx_datap_imust be valid for all bytes in vc_tx_data_i that contain a portion of a TLP (header, payload, and TLP digest (if present)).

4.4.1.2. TLP Transmit Credit Interface Port Description

The Transmit Credit Interface provides the means for flow control of non-posted transmit transactions between the core Transmit Buffer and user. This is important for allowing Posted and Completion TLPs to continue to make progress when non-posted TLPs are blocked (which can be necessary in some cases to avoid potential deadlock conditions). The amount of non-posted TLP storage in the core Transmit Buffer is communicated on the Transmit Credit Interface. You are expected to use this interface to limit simultaneously outstanding TLP transmission of non-posted TLPs to the amount of non-posted TLPs that the core can absorb into the non-posted Transmit Buffer.

Note that core/link partner transmit TLP flow control is not managed via this interface; the core manages to transmit flow control between the core and the PCIe link partner Receive Buffer without user intervention.

Table 4.5. TLP Transmit Credit Interface Ports

Port	Clock Domain	Direction	Description
link0_tx_credit_init_o	sys_clk_i	Output	Transmit layer credit initialization. When the core Transaction Layer for is ready to accept TLP transmissions, the core asserts link0_tx_credit_init_o== 1 for one clock cycle and on the same cycle indicates the non-posted TLP Header storage capacity of the Transmit Buffer on link0_tx_credit_nh_o [11:0]. You are expected to keep and initialize their NH available transmit credit counters on link0_tx_credit_init_o==1. When a non-posted TLP is pending for transmission within user logic, user logic should check the currently available NH credit count for the associated link and hold the transmission until enough NH credits are available to transmit the TLP. Once the TLP has been committed for transmit, the amount of NH credits required by that TLP are decremented from the NH credit count. As the core forwards transmitted TLPs from the Transmit Buffer and thus makes room for new TLPs, the core asserts link0_tx_credit_return_o==1 for one clock cycle and places the number of NH credits being returned on link0_tx_credit_nh_o [11:0]. In this manner you can manage sending only enough non-posted TLPs that the core can hold in its Transmit Buffer. This permits you to know when non-posted TLPs would be blocked and thus send posted and/or completion TLPs instead. This is important for avoiding deadlocks and keeps non-posted TLP blockage from reducing posted and completion throughput. Should the core receive more non-posted TLPs than the core can store in its non-posted TLP transmit storage, the core pauses TLP transmission rather than allow an overflow to occur. Thus, if you do not wish to use the Transmit Credit Interface, you may ignore this interface provided you are willing to permit blocked non-posted TLPs from also blocking following posted and completion TLPs.
link0_tx_credit_return_o	sys_clk_i	Output	As the core forwards transmitted TLPs from the Transmit Buffer and thus makes room for new TLPs, the core asserts link0_tx_credit_return_o==1 for one clock cycle and places the number of NH credits being returned on link0_tx_credit_nh_o [11:0].
link0_tx_credit_nh_o [11:0]	sys_clk_i	Output	Number of NH credits to return through Transmit Interface.

4.4.2. TLP Receive Interface

Refer to the TLP Receive Interface section for more information and timing diagrams.

4.4.2.1. TLP Receive Interface Port Descriptions

Table 4.6. TLP Receive Interface Ports

Port	Clock Domain	Direction	Description
link0_rx_valid_o	sys_clk_i	Output	• The valid signal corresponds to the data sent through link0_rx_data_o • A data transfer occurs when link0_rx_valid_o== 1 and link0_rx_ready_i== 1.
link0_rx_ready_i	sys_clk_i	Input	• Set link0_rx_ready_i== 1 whenever the user logic is ready to accept received TLP data. • A data transfers occur when link0_rx_valid_o== 1 and link0_rx_ready_i== 1.

Port	Clock Domain	Direction	Description

link0_rx_sel_o [1:0]	sys_clk_i	Output	• Receive TLP type indicator:
			  • 0 == Posted Request (write request)
			  • 1 == Non-Posted Request (request requiring a completion)
			  • 2 == Completion (completion to a previous request)
			  • 3 == Reserved
			• link0_rx_sel_ois valid for the entire TLP (from link0_rx_sop_o== 1 to link0_rx_eop_o== 1).
			• link0_rx_sel_ois useful for steering the TLP to the appropriate processing logic. For example, Posted Requests should be directed to receive write logic while Non-Posted Requests should be directed to receive read logic. Completions should be directed back to the original read request source using the TLP Tag information.

link0_rx_cmd_data_o [12:0]	sys_clk_i	Output	• Received TLP Type Indicator.
			• link0_rx_cmd_data_o [12:0] contains following information:
			  • Bits[12:10] – Traffic Class[2:0] of the TLP.
			  • Bit[9] – Completion/Base Address Region indicator.
			  • 1 – Indicates the TLP is a Completion or Message routed by ID.
			  • 0 – Indicates the TLP is a read or write request or a message routed by address that hit an enabled Base Address Region.
			  • Bit[8] – When (1), the packet is a “write” transaction; when (0), the packet is a “read” transaction.
			  • Bit[7] – When (1), the packet requires one or more completion transactions as a response; (0) otherwise.
			  • Bit[6] – (1) the TLP hit the Expansion ROM else (0).
			  • Bit[5] – (1) the TLP hit Base Address Region 5 else (0).
			  • Bit[4] – (1) the TLP hit Base Address Region 4 else (0).
			  • Bit[3] – (1) the TLP hit Base Address Region 3 else (0).
			  • Bit[2] – (1) the TLP hit Base Address Region 2 else (0).
			  • Bit[1] – (1) the TLP hit Base Address Region 1 else (0).
			  • Bit[0] – (1) the TLP hit Base Address Region 0 else (0).
			• link0_rx_cmd_data_ois valid for the entire TLP (from link0_rx_sop_o== 1 to link[LINK]_rx_eop_o == 1).
			• link0_rx_cmd_data_oprovides information about the received TLP to facilitate user TLP processing. This port has a different meaning in Root Port Modes of operation and Endpoint operation. The Lattice PCIe x1 Core decodes received TLPs to determine their destination. The core passes this information to the Transaction Layer Receive Interface by asserting the appropriate bits in this field.

link0_rx_f_o [1:0]	sys_clk_i	Output	• Function hit by the Received TLP
			• link0_rx_f_oindicates which PCIe function received the TLP and response is given as follows:
			  • link0_rx_f_o== 0 indicates Function #0
			  • link0_rx_f_o== 1 indicates Function #1 ...

link0_rx_sop_o	sys_clk_i	Output	Start of TLP indicator link0_rx_sop_o== 1 coincident with the first link0_rx_data_oword in each TLP ; 0-> otherwise.
link0_rx_eop_o	sys_clk_i	Output	End of TLP indicator link0_rx_eop_o== 1 coincident with the last link0_rx_data_oword in each TLP ; otherwise, 0.

Port	Clock Domain	Direction	Description
link0_rx_err_ecrc_o	sys_clk_i	Output	Received TLP ECRC Error Indicator link0_rx_err_ecrc_o == 1 inclusive for received TLPs which contain a detected ECRC error; otherwise, 0. link0_rx_err_ecrc_o only reports ECRC errors when ECRC checking is enabled. ECRC checking is enabled by software through the AER Capability. TLPs with ECRC errors are presented on the Receive Interface in the same format that they are received including the TLP Digest (ECRC). ECRC errors are serious, uncorrectable errors. The user design must decide how to handle/recover from the error including whether to use the TLP with the error. ECRC errors need for higher level software to correct/handle the error. PCIe does not have a standard mechanism for retransmitting TLPs end to end as it does for a given PCIe link (through the LCRC/Sequence Number and Replay mechanisms).
link0_rx_data_o [31:0]	sys_clk_i	Output	• Received TLP Data • Received TLP data comprises a complete Transaction Layer Packet (TLP) as defined by the PCI Express Specification including the entire TLP header, data payload (if present), and TLP Digest (ECRC, if present). The core strips the packet's STP/END/EDB framing, Sequence Number, and Link CRC (LCRC) prior to the TLP appearing on this interface. The core checks TLP ECRC, when present and when checking is enabled, and can be optionally enabled to remove the ECRC from the TLP.
v[3:0]	sys_clk_i	Output	• Received TLP Data Parity • Even parity of associated link0_rx_data_o: link0_rx_datap_o [i] == ^(link0_rx_data_o [((i+1)*8)-1:(i*8)]) • link0_rx_datap_o is valid for all bytes in link0_rx_data_o that contain a portion of a TLP (header, payload (if present), and TLP digest (ECRC, if present). link0_rx_datap_o is not valid for any trailing, unused bytes in the final link0_rx_data_o word in a TLP.

4.4.2.2. TLP Receive Credit Interface Port Description

The Receive Credit Interface provides the means for flow control of non-posted receive transactions between the core Receive Buffer and user receive TLP logic. This is important for allowing Posted and Completion TLPs to continue to make progress when non-posted TLPs are blocked (which can be necessary in some cases to avoid potential deadlock conditions). The amount of non-posted TLP storage in the user design is communicated on the Receive Credit Interface. The core uses this interface to limit the simultaneously outstanding receive non-posted TLPs to the amount of non-posted TLPs that the user design advertises that it can absorb.

Note that link partner/core receive TLP flow control is not managed through this interface; the core manages Receive Buffer flow control between itself and the PCIe link partner transmit gating function without user intervention.

Table 4.7. TLP Receive Credit Interface Ports

Port	Clock Domain	Direction	Description
link0_rx_credit_init_i	sys_clk_i	Input	When the user transaction layer logic is ready to accept non-posted TLP reception, assert the link0_rx_credit_init_i == 1 for one clock cycle and on the same cycle indicates the non-posted TLP header storage capacity of the user design in link0_rx_credit_nh_i [11:0]. You must initialize link0_rx_credit_init_i shortly (within 10s of clocks) after link0_tl_link_up_o for Root Port and shortly after link0_dl_link_up_o for Endpoint. Holding off credit initialization for an extended period can cause received non-posted TLP transactions to timeout in the source component which may be serious errors. The core limits simultaneous outstanding non-posted receive TLPs on the receive interface to ensure no more than the initialized NH credits are simultaneously outstanding to user receive TLP logic. Once the received non-posted TLPs are processed/forwarded such that more room is available to receive new non-posted TLPs, assert link0_rx_credit_return_i ==1 for one clock cycle and places the number of NH credits being returned on link0_rx_credit_nh_i [11:0]. In this manner, you can limit the outstanding core receive TLPs to the user design. This permits the core to know when non-posted TLPs would be blocked and thus send posted and/or completion TLPs to the user design instead. This is important for avoiding deadlocks and keeps non-posted TLP blockage from reducing posted and completion throughput. If you do not wish to implement flow control of NH credits through this interface, link0_rx_credit_init_i ==1 and link0_rx_credit_nh_inf_i is set to 1 to advertise infinite NH credits. The NH credit flow control is not implemented for links that advertised infinite NH credits.
link0_rx_credit_return_i	sys_clk_i	Input	Once the received non-posted TLPs are processed/forwarded such that more room is available to receive new non-posted TLPs, assert link0_rx_credit_return_i 1 for one clock cycle and places the number of NH credits being returned on link0_rx_credit_nh_i [11:0].
link0_rx_credit_nh_i [11:0]	sys_clk_i	Input	Number of NH credits to return through receive interface.
link0_rx_credit_nh_inf_i	sys_clk_i	Input	• Infinite NH Credits • link0_rx_credit_nh_inf_i: 1==Do not limit TLP reception due to NH credits. 0==Limit simultaneously outstanding NH credits to the value of link0_rx_credit_nh_i [11:0] when link0_rx_credit_init_i is 1.

4.5. Lattice Memory Mapped Interface (LMMI)

The Lattice PCIe x1 IP Core implements a bus for configuring core options and obtaining core status. The Core Configuration and Status Registers (CSR) are made accessible to the user design through the Lattice Memory Mapped interface (LMMI).

Table 4.8. Lattice Memory Mapped Interface Ports

Port	Clock Domain	Direction	Description
usr_lmmi_clk_i	usr_lmmi_clk_i	Input	LMMI Clock. You must provide a clock to this port as the PHY relies on this clock during initialization.
usr_lmmi_resetn_i	usr_lmmi_clk_i	Input	Active low, asynchronous assert, synchronous de-assert reset

Port	Clock Domain	Direction	Description
usr_lmmi_offset_i [16:2]	usr_lmmi_clk_i	Input	Register offset Link Layer registers: usr_lmmi_offset_i [16] – 0 (Link Select) usr_lmmi_offset_i [15:2] – word aligned offset usr_lmmi_offset_i [1:0] – reserved (tie to 0) PHY registers: usr_lmmi_offset_i [7:2] – byte aligned offset usr_lmmi_offset_i [16:8] – reserved (tie to 0) Example: Link Layer Register access: mgmt_ftl_bar0 register mgmt_ftl_BASE = 0x4000, bar0 offset = 0x60 reg_byte_offset [15:0] = 0x4060
usr_lmmi_request_i	usr_lmmi_clk_i	Input	The request you sent to PCIe to start the transaction (1==Active; 0==Otherwise). A transaction is started when usr_lmmi_request_i==usr_lmmi_ready_o==1. Consecutive request must be done with at least 1 clock period wait cycle (that is, usr_lmmi_request_i must deassert first after a successful transaction before making another request). When usr_lmmi_request_i==usr_lmmi_ready_o==1, usr_lmmi_wr_rdn_i, and usr_lmmi_offset_i must be valid and describe the transaction to execute; if the transaction is a write as indicated by usr_lmmi_wr_rdn_i==1, usr_lmmi_wdata_i must also be valid.
usr_lmmi_wr_rdn_i	usr_lmmi_clk_i	Input	Direction (1==Write, 0==Read)
usr_lmmi_wdata_i [31:0]	usr_lmmi_clk_i	Input	Write data **Note:** For PHY register access, only bit[7:0] is valid and bit[31:8] should be tied to 0.
usr_lmmi_rdata_o [31:0]	usr_lmmi_clk_i	Output	Read data. Bit[31:0] – Link Layer access read data
usr_lmmi_ready_o	usr_lmmi_clk_i	Output	Indicates the status whether Target is ready to start a new transaction (1==Ready; 0==Not ready)
usr_lmmi_rdata_valid_o	usr_lmmi_clk_i	Output	Indicates usr_lmmi_rdata_o contains valid data (1==Valid; 0==Otherwise)

4.6. Legacy Interrupt Interface

The Legacy Interrupt Interface enables you to generate interrupts. Refer to the Legacy Interrupt section for more details and timing diagrams.

Table 4.9. Legacy Interrupt Interface Ports

Port	Clock Domain	Direction	Description
link0_legacy_interrupt_i [NUM_FUNCTIONS-1:0]	Asynchronous	Input	link0_legacy_interrupt_i is used to generate Legacy interrupts on the PCI Express link. link0_legacy_interrupt_i has one input for each Base (Physical) Function. For each function, system software configures the function to use MSI-X, MSI, or Legacy Interrupt mode as part of the PCI enumeration process. User interrupt logic must behave differently depending upon whether the function is enabled for MSI-X, MSI, or Legacy Interrupts. When Legacy Interrupt Mode is enabled, link0_legacy_interrupt_i implements one level-sensitive interrupt (INTA, INTB, INTC, or INTD) for each Base Function. Each functions' interrupt sources must be logically ORed together and input as link0_legacy_interrupt_i [i] for a given function. Each interrupt source must continue to drive a 1 until it has been serviced and cleared by software at which time it must switch to driving 0. The core ORs together INTA/B/C/D from all functions to create an aggregated INTA/INTB/INTC/INTD. The core monitors high and low transitions on the aggregated INTA/B/C/D and sends an Interrupt Assert message on each 0 to 1 transition and an Interrupt De-Assert Message on each 1 to 0 transition of the aggregated INTA/B/C/D. Transitions which occur too close together to be independently transmitted are merged. When a function has MSI-X or MSI Interrupt Mode enabled, link0_legacy_interrupt_i is not used for that function. MSI-X/MSI interrupts are signaled using MSI-X/MSI Message TLPs which you can generate and transmit on the Transmit Interface.
link0_legacy_interrupt_o	Asynchronous	Output	This signal is to implement the PCI Express Capability and Advanced Error Reporting Capability contain mechanisms to interrupt system software when events occur. The core asserts mgmt_interrupt_o[i] for Link[i] when an event occurs that per the PCI Express Capability or Advance Error Reporting Enhanced Capability must generate an interrupt. You must merge mgmt_interrupt_o with user interrupt sources into mgmt_interrupt_leg, mgmt_interrupt_msix_req, mgmt_interrupt_msix_ack, and mgmt_interrupt_msix_vector just like it would for any user interrupt. The core outputs mgmt_interrupt_o as an active high level-based interrupt when level-based interrupts are in use (MSI-X_Enable == 0 and MSI_Enable == 0) and as an active high single clock pulse when edge based interrupts are in use ~(MSI-X_Enable == 0 and MSI_Enable == 0). The core implements only one interrupt output. In MSI and MSI-X interrupt modes of operation, core interrupts must use MSI/MSI-X interrupt vector mgmt_interrupt_message_num[4:0]. This value is advertised in the configuration registers for software association of interrupts, so it is important that the user route core generated interrupts on mgmt_interrupt_o to the vector advertised.

Note:

1. NUM_FUNCTIONS – range (1-4)

4.7. Power Management Interface

The Lattice PCIe x1 IP Core supports optional capabilities such as Dynamic Power Allocation, Latency Tolerance Reporting, and Power Budgeting.

Table 4.10. Power Management Interface Ports

Port	Clock Domain	Direction	Description
link0_pm_dpa_control_en_o	sys_clk_i	Output	• Dynamic Power Allocation Enable. • If set to 1, the link0_pm_dpa_control_o should be monitored for change requests in the D0 power substate. • If set to 0, link0_pm_dpa_control_o should be ignored.
link0_pm_dpa_control_o	sys_clk_i	Output	• Dynamic Power Allocation Control. • This output specified the desired D0 power substate. • If link0_pm_dpa_control_en_o is set to 1, any change on this output should be used to indicate that the D0 power substate should be changed, and the process should begin to change the power substate. • On completion of the substate transition, this output should be compared again with the current power substate. • If they do not match, and link0_pm_dpa_control_en_o is set to 1, then a new power transition should begin.
link0_pm_dpa_status_i [4:0]	sys_clk_i	Input	• Dynamic Power Allocation Status. This input should be changed to match the link0_pm_dpa_control_o D0 power substate. • If the change on link0_pm_dpa_control_o is to a higher power substate, the link0_pm_dpa_status_i should be updated as soon as the change on link0_pm_dpa_control_o is detected. • If the change on link0_pm_dpa_control_o is to a lower power state, the power state change should complete first, and then link0_pm_dpa_status_i should be updated to the lower power state. • These rules assure that the device never operates at a power level exceeding the power level reported on link0_pm_dpa_status_i.
link0_pm_ltr_msg_send_i	sys_clk_i	Input	• When operating as an upstream port, set to 1 for one clock to cause an LTR message to be transmitted and 0 otherwise. link0_pm_ltr_snoop_i [12:0], link0_pm_ltr_nosnoop_i [12:0], link0_pm_ltr_snoop_req_i, • link0_pm_ltr_nosnoop_req_i Specify the contents of the message. • The LTR capability registers can be access through the UCFG interface. • See Configuration Space Register Interface (UCFG) for details. • Unused for downstream ports.
link0_pm_ltr_snoop_i [12:0]	sys_clk_i	Input	See link0_pm_ltr_msg_send_i.
link0_pm_ltr_nosnoop_i [12:0]	sys_clk_i	Input	See link0_pm_ltr_msg_send_i.
link0_pm_ltr_snoop_req_i	sys_clk_i	Input	See link0_pm_ltr_msg_send_i.
link0_pm_ltr_nosnoop_req_i	sys_clk_i	Input	See link0_pm_ltr_msg_send_i
link0_pm_pb_data_sel_o [7:0]	sys_clk_i	Output	• Power Budgeting Data Select. • Specifies an index into a table of Power Budgeting Status by power rail and operating conditions.

Port	Clock Domain	Direction	Description

link0_pm_pb_data_reg_rd_i [31:0]	sys_clk_i	Input	• Power Budgeting Data Register.
			• This input should be updated with the Power Budgeting Data looked up by the index value on link0_pm_pb_data_sel_o [7:0].
			• A minimum of 2 Status Conditions is needed for each power rail for which the device requires power.
			• The last entry in the table should be identified by having all 32 bits being set to 0.
			• The bit encoding is:
			  • [31:21] reserved (0)
			  • [20:18] pb_rail
			  • [17:15[pb_op_type
			  • [14:13] pb_pm_state
			  • [12:10] pb_pm_substate
			  • [9:8] pb_data_scale
			  • [7:0] pb_base_power

link0_user_aux_power_detected_i	sys_clk_i	Input	• Set to 1 if the user design implements Aux Power and Aux Power is detected as present else set to 0. • The value of this port is reflected in the PCIe Configuration Register: PCIe Status – AUX Power Detected.
link0_user_transactions_pending_i [NUM_FUNCTIONS-1:0]	sys_clk_i	Input	• Set to 1 when you have an outstanding (not yet completed) non-posted requests else set to 0. • The value of this port is reflected in the PCIe Configuration Register: PCIe Status – Transactions Pending.

Note:

1. NUM_FUNCTIONS – range (1-4)

4.8. Configuration Space Register Interface (UCFG)

The UCFG Interface is provided for users to read the current values of the Lattice PCIe x1 Core's PCIe Configuration Registers and to obtain status of the Lattice PCIe x1 Core that may be needed to implement the user design.

The UCFG Interface is a simple SRAM-like interface that accepts write/read transactions. The UCFG Interface supports multiple outstanding transaction requests to enable higher throughput on the interface. Writes and reads are executed out in the same order that they are accepted on the interface.

Table 4.11. Configuration Space Register Interface Ports

Port	Clock Domain	Direction	Description
ucfg_valid_i	sys_clk_i	Input	• Transaction Request Valid • A transaction is started when ucfg_valid_i==ucfg_ready_o==1. • Multiple transactions can be outstanding simultaneously and are executed in the order received. • When ucfg_valid_i==ucfg_ready_o==1, ucfg_f_i, ucfg_wr_rd_n_i, and ucfg_addr_i must be valid and describe the transaction to execute; if the transaction is a write as indicated by ucfg_wr_rd_n_i==1, ucfg_wr_be_i and ucfg_wr_data_i must also be valid.
ucfg_ready_o	sys_clk_i	Output	Transaction Request Ready
ucfg_f_i [2:0]	sys_clk_i	Input	• Transaction Request Function Number • Selects which function in a multi-function core is to be accessed. • This port is only present for cores that are delivered supporting multiple functions.

Port	Clock Domain	Direction	Description

ucfg_wr_rd_n_i	sys_clk_i	Input	• Transaction Request Type
			• Selects the type of transaction:
			  • 1 = Write
			  • 0 = Read

ucfg_addr_i [11:2]	sys_clk_i	Input	• Transaction Request Address • Selects the DWORD (32-bit) address of the register accessed by the transaction.

ucfg_wr_be_i[3:0]	sys_clk_i	Input	• Transaction Request Write Byte Enables
			• Selects which bytes to write during a write transaction.
			• For each ucfg_wr_be_i[i]:
			  • 1 = Write byte
			  • 0 = Do not write byte
			• ucfg_wr_be_i[i] is associated with ucfg_wr_data_i[(i*8)+7:(i*8)].

ucfg_wr_data_i[31:0]	sys_clk_i	Input	• Transaction Request Write Data • Selects data to write during a write transaction. • ucfg_wr_data_i [7:0] is the least significant byte (byte address offset 2'b00) and ucfg_wr_data_i [31:0] is the most significant byte (byte address offset 2'b11).

ucfg_rd_done_o	sys_clk_i	Output	• Read Transaction Done.
			• Indicates that a prior read transaction request has been completed and the resulting data on ucfg_rd_data_o is valid.
			  • 1 = Read done
			  • 0 = Otherwise
			• Read transactions complete in the same order that the transaction requests were accepted.

ucfg_rd_data_o [31:0]	sys_clk_i	Output	• Read Transaction Data. • Provides the read data for a UCFG read transaction. • ucfg_rd_data_o [7:0] is the least significant byte (byte address offset 2'b00) and ucfg_rd_data_o [31:24] is the most significant byte (byte address offset 2'b11).

4.9. APB Configuration Interface

This interface is available if the register interface type selected in the IP generation user interface is *APB*.

You must provide a 512 KB aligned base address that is used when accessing the Core CSRs and PCIe Configuration Space registers.

Table 4.12. APB Configuration Interface Ports

Port	Clock Domain	Direction	Description
c_apb_pclk_i	c_apb_pclk_i	Input	Clock
c_apb_preset_n_i	c_apb_pclk_i	Input	Active-low asynchronous assert, synchronous de-assert reset.
c_apb_paddr_i [31:0]	c_apb_pclk_i	Input	Bus Address (refer to the APB Interface section)
c_apb_psel_i	c_apb_pclk_i	Input	Completer select.
c_apb_penable_i	c_apb_pclk_i	Input	Enable. This signal indicates the second and subsequent cycles of an APB transfer.
c_apb_pwrite_i	c_apb_pclk_i	Input	Indicates write or read access. 0 – Read 1 – Write

Port	Clock Domain	Direction	Description
c_apb_pwdata_i [31:0]	c_apb_pclk_i	Input	Write Data. For PHY register access, only bit[7:0] is valid and bit[31:8] should be tied to 0.
c_apb_prdata_o [31:0]	c_apb_pclk_i	Output	Read Data. For PHY register access, only bit[7:0] is valid and bit[31:8] should be ignored.
c_apb_pready_o	c_apb_pclk_i	Output	Ready. The Completer uses this signal to extend an APB transfer.
c_apb_pslverr_o	c_apb_pclk_i	Output	Completer error. 0 – Otherwise 1 – Error

4.10. AXI-Stream (Non-DMA) Data Interface

This interface is available if the data interface type selected in the IP generation user interface is *AXI_STREAM* when *Configuration Mode* is *TLP_MODE*.

4.10.1. AXI-Stream Transmitter Interface Port Descriptions

Table 4.13. AXI-Stream Transmitter Interface Ports

Port	Clock Domain	Direction	Description
m_tready_i	sys_clk_i	Input	Destination ready. 1==Ready, 0==Not ready. A transfer occurs when m_tvalid_o==m_tready_i==1.
m_tvalid_o	sys_clk_i	Output	Source valid 1==Valid 0==Not valid.
m_tdata_o [31:0]	sys_clk_i	Output	Received TLP Data Received TLP data comprises a complete Transaction Layer Packet (TLP) as defined by the PCI Express Specification including the entire TLP header, data payload (if present), and TLP Digest (ECRC, if present). The core strips the packet's STP/END/EDB framing, Sequence Number, and Link CRC (LCRC) prior to the TLP appearing on this interface. The core checks TLP ECRC, when present and when checking is enabled, and can be optionally enabled to remove the ECRC from the TLP.
m_tstrb_o [3:0]	sys_clk_i	Output	Byte qualifier that indicates whether the content of the associated byte of m_tdata_o is processed as a data byte or a position byte. This is always 4'hF.
m_tkeep_o [3:0]	sys_clk_i	Output	• Byte qualifier that indicates whether the content of the associated byte of m_tdata_o is processed as part of the data stream. • Associated bytes that have the m_tkeep_o byte qualifier deasserted are null bytes and can be removed from the data stream. • This is always 4'hF.
m_tlast_o	sys_clk_i	Output	End of TLP indicator. m_tlast_o == 1 coincident with the last m_tdata_o word in each TLP. Otherwise, 0.

Port	Clock Domain	Direction	Description
m_tid_o [7:0]	sys_clk_i	Output	Data stream identifier that indicates different streams of data. m_tid_o[2:0] has the BAR number information when rx_cmd_data[9] = 0. Otherwise, 0 when rx_cmd_data[9] = 1(completion) m_tid_o[3] = rx_err_par m_tid_o[6:4] = rx_cmd_data[12:10] m_tid_o[7] = rx_err_ecrc
m_tdest_o [3:0]	sys_clk_i	Output	m_tdest_o provides routing information for the data stream. Bits [3:2] – Function Hit by the Received TLP Bits [1:0] – Receive TLP type indicator: • 0 == Posted Request (write request) • 1 == Non-Posted Request (request requiring a completion) • 2 == Completion (completion to a previous request)

4.10.2. AXI-Stream Receiver Interface Port Descriptions

Table 4.14. AXI-Stream Receiver Interface Ports

Port	Clock Domain	Direction	Description
s_tvalid_i	sys_clk_i	Input	Source valid 1==Valid 0==Not valid.
s_tdata_i [31:0]	sys_clk_i	Input	TLP data to transfer
s_tstrb_i [3:0]	sys_clk_i	Input	• Byte qualifier that indicates whether the content of the associated byte of m_tdata_o is processed as a data byte or a position byte. This is always 4'hF for Link 1.
s_tkeep_i [3:0]	sys_clk_i	Input	• Byte qualifier that indicates whether the content of the associated byte of m_tdata_o is processed as part of the data stream. • Associated bytes that have the m_tkeep_o byte qualifier deasserted are null bytes and can be removed from the data stream. This is always 4'hF.
s_tlast_i	sys_clk_i	Input	End of packet indicator. Set == 1 coincident with the last s_tdata_i word in each TLP.
s_tid_i [7:0]	sys_clk_i	Input	Unused. Set to 8'h00.
s_tdest_i [3:0]	sys_clk_i	Input	Unused. Set to 4'h0.
s_tready_o	sys_clk_i	Output	Destination ready. 1==Ready 0==Not ready A transfer occurs when s_tvalid_i==s_tready_o==1.

4.11. DMA Interrupt Interface

This interface is only available if DMA support is enabled.

Table 4.15. DMA Interrupt Interface Ports

Port	Clock Domain	Direction	Description
link0_int_normal_o	—	Output	• Normal Interrupt • When asserted, it means that one or more interrupt status bits classified as “normal” are asserted.
link0_int_critical_o	—	Output	• Critical Interrupt • When asserted, it means that one or more interrupt status bits classified as “critical” are asserted.

4.12. AXI Data Interface (DMA)

Table 4.16. AXI-MM Manager Interface (DMA)

Port	Clock Domain	Direction	Description
m0_dma_axi_awid_o [2:0]	sys_clk_i	Output	This signal is the identification tag for the write address group of signals.
m0_dma_axi_awaddr_o [63:0]	sys_clk_i	Output	The write address in a write transaction.
m0_dma_axi_awlen_o [7:0]	sys_clk_i	Output	The burst length gives the exact number of transfers in a burst.
m0_dma_axi_awsize_o [2:0]	sys_clk_i	Output	This signal indicates the size of each transfer.
m0_dma_axi_awburst_o [1:0]	sys_clk_i	Output	Burst mode. Always 2’b01.
m0_dma_axi_awlock_o	sys_clk_i	Output	This signal is unused and always 0.
m0_dma_axi_awprot_o [2:0]	sys_clk_i	Output	This signal is unused and always 0.
m0_dma_axi_awcache_o [3:0]	sys_clk_i	Output	This signal is unused and always 0.
m0_dma_axi_awvalid_o	sys_clk_i	Output	This signal indicates that the channel is signaling valid write address and control information.
m0_dma_axi_awready_i	sys_clk_i	Input	This signal indicates that the subordinate is ready to accept an address and associated control signals.
m0_dma_axi_wdata_o [31:0]	sys_clk_i	Output	Write data.
m0_dma_axi_wstrb_o [3:0]	sys_clk_i	Output	This signal indicates which byte lanes hold valid data. There is one write strobe bit for each eight bits of the write data bus.
m0_dma_axi_wlast_o	sys_clk_i	Output	This signal indicates the last transfer in a write burst.
m0_dma_axi_wvalid_o	sys_clk_i	Output	This signal indicates that valid write data and strobes are available.
m0_dma_axi_wready_i	sys_clk_i	Input	This signal indicates that the subordinate can accept the write data.
m0_dma_axi_bid_i [3:0]	sys_clk_i	Input	This signal is the ID tag of the write response.
m0_dma_axi_bresp_i [1:0]	sys_clk_i	Input	This signal indicates the status of the write transaction.
m0_dma_axi_bvalid_i	sys_clk_i	Input	This signal indicates that the channel is signaling a valid write response.
m0_dma_axi_bready_o	sys_clk_i	Output	This signal indicates that the manager can accept a write response.
m0_dma_axi_arid_o [2:0]	sys_clk_i	Output	This signal is the identification tag for the read address group of signals.
m0_dma_axi_araddr_o [63:0]	sys_clk_i	Output	The read address gives the address of the first transfer in a read burst transaction.
m0_dma_axi_arlen_o [7:0]	sys_clk_i	Output	This signal indicates the exact number of transfers in a burst.
m0_dma_axi_arsize_o [2:0]	sys_clk_i	Output	This signal indicates the size of each transfer.

Port	Clock Domain	Direction	Description
m0_dma_axi_arburst_o [1:0]	sys_clk_i	Output	Burst mode. Always 2'b01.
m0_dma_axi_arprot_o [2:0]	sys_clk_i	Output	This signal is unused and always 0.
m0_dma_axi_arlock_o	sys_clk_i	Output	This signal is unused and always 0.
m0_dma_axi_arcache_o [3:0]	sys_clk_i	Output	This signal is unused and always 0.
m0_dma_axi_arvalid_o	sys_clk_i	Output	This signal indicates that the channel is signaling valid read address and control information.
m0_dma_axi_arready_i	sys_clk_i	Input	This signal indicates that the subordinate is ready to accept an address and associated control signals.
m0_dma_axi_arqos_o [3:0]	sys_clk_i	Output	This signal is unused and always 0.
m0_dma_axi_aruser_o [7:0]	sys_clk_i	Output	This signal is unused and always 0.
m0_dma_axi_rid_i [2:0]	sys_clk_i	Input	This signal is the identification tag for the read data group of signals generated by the subordinate.
m0_dma_axi_rdata_i [31:0]	sys_clk_i	Input	Read data.
m0_dma_axi_rresp_i [1:0]	sys_clk_i	Input	This signal indicates the status of the read transfer.
m0_dma_axi_rlast_i	sys_clk_i	Input	This signal indicates the last transfer in a read burst.
m0_dma_axi_rvalid_i	sys_clk_i	Input	This signal indicates that the channel is signaling the required read data.
m0_dma_axi_rready_o	sys_clk_i	Output	This signal indicates that the manager can accept the read data and response information.

4.13. AXI Data Interface (Bridge Mode)

Table 4.17. AXI-MM Manager Write Interface (Bridge Mode)

Port	Clock Domain	Direction	Description
m0_aximm_awid_o [7:0]	sys_clk_i	Output	This signal is the identification tag for the write address group of signals.
m0_aximm_awaddr_o [63:0]	sys_clk_i	Output	The write address in a write transaction.
m0_aximm_awlen_o [7:0]	sys_clk_i	Output	Burst mode is not supported. Always 8'h00.
m0_aximm_awsize_o [2:0]	sys_clk_i	Output	This signal indicates the size of each transfer.
m0_aximm_awburst_o [1:0]	sys_clk_i	Output	Burst mode is not supported. Always 2'b00.
m0_aximm_awlock_o	sys_clk_i	Output	This signal is unused and always 0.
m0_aximm_awprot_o [2:0]	sys_clk_i	Output	This signal is unused and always 0.
m0_aximm_awcache_o [3:0]	sys_clk_i	Output	This signal is unused and always 0.
m0_aximm_awvalid_o	sys_clk_i	Output	This signal indicates that the channel is signaling valid write address and control information.
m0_aximm_awready_i	sys_clk_i	Input	This signal indicates that the subordinate is ready to accept an address and associated control signals.
m0_aximm_wdata_o [31:0]	sys_clk_i	Output	Write data.
m0_aximm_wstrb_o [3:0]	sys_clk_i	Output	This signal indicates which byte lanes hold valid data. There is one write strobe bit for each eight bits of the write data bus.
m0_aximm_wlast_o	sys_clk_i	Output	This signal indicates the last transfer in a write burst.
m0_aximm_wvalid_o	sys_clk_i	Output	This signal indicates that valid write data and strobes are available.
m0_aximm_wready_i	sys_clk_i	Input	This signal indicates that the subordinate can accept the write data.
m0_aximm_bid_i [7:0]	sys_clk_i	Input	This signal is the ID tag of the write response.
m0_aximm_bresp_i [1:0]	sys_clk_i	Input	This signal indicates the status of the write transaction.
m0_aximm_bvalid_i	sys_clk_i	Input	This signal indicates that the channel is signaling a valid write response.

Port	Clock Domain	Direction	Description
m0_aximm_bready_o	sys_clk_i	Output	This signal indicates that the manager can accept a write response.
m0_aximm_arid_o [7:0]	sys_clk_i	Output	This signal is the identification tag for the read address group of signals.
m0_aximm_araddr_o [63:0]	sys_clk_i	Output	The read address gives the address of the first transfer in a read burst transaction.
m0_aximm_arlen_o [7:0]	sys_clk_i	Output	This signal indicates the exact number of transfers in a burst.
m0_aximm_arsize_o [2:0]	sys_clk_i	Output	This signal indicates the size of each transfer.
m0_aximm_arburst_o [1:0]	sys_clk_i	Output	Burst mode is not supported. Always 2'b00.
m0_aximm_arprot_o [2:0]	sys_clk_i	Output	This signal is unused and always 0.
m0_aximm_arlock_o	sys_clk_i	Output	This signal is unused and always 0.
m0_aximm_arcache_o [3:0]	sys_clk_i	Output	This signal is unused and always 0.
m0_aximm_arvalid_o	sys_clk_i	Output	This signal indicates that the channel is signaling valid read address and control information.
m0_aximm_arready_i	sys_clk_i	Input	This signal indicates that the subordinate is ready to accept an address and associated control signals.
m0_aximm_arqos_o [3:0]	sys_clk_i	Output	This signal is unused and always 0.
m0_aximm_aruser_o [7:0]	sys_clk_i	Output	This signal is unused and always 0.
m0_aximm_rid_i [7:0]	sys_clk_i	Input	This signal is the identification tag for the read data group of signals generated by the subordinate.
m0_aximm_rdata_i [31:0]	sys_clk_i	Input	Read data.
m0_aximm_rresp_i [1:0]	sys_clk_i	Input	This signal indicates the status of the read transfer.
m0_aximm_rlast_i	sys_clk_i	Input	This signal indicates the last transfer in a read burst.
m0_aximm_rvalid_i	sys_clk_i	Input	This signal indicates that the channel is signaling the required read data.
m0_aximm_rready_o	sys_clk_i	Output	This signal indicates that the manager can accept the read data and response information.

Table 4.18. AXI-Lite Manager Interface (Bridge Mode)

Port	Clock Domain	Direction	Description
m0_axil_awaddr_o [63:0]	sys_clk_i	Output	The write address in a write transaction.
m0_axil_awvalid_o	sys_clk_i	Output	This signal indicates that the channel is signaling valid write address and control information.
m0_axil_awready_i	sys_clk_i	Input	This signal indicates that the subordinate is ready to accept an address and associated control signals.
m0_axi_wdata_o [31:0]	sys_clk_i	Output	Write data.
m0_axil_wstrb_o [3:0]	sys_clk_i	Output	This signal indicates which byte lanes hold valid data. There is one write strobe bit for each eight bits of the write data bus.
m0_axil_wvalid_o	sys_clk_i	Output	This signal indicates that valid write data and strobes are available.
m0_axil_wready_i	sys_clk_i	Input	This signal indicates that the subordinate can accept the write data.
m0_axil_bresp_i [1:0]	sys_clk_i	Input	This signal indicates the status of the write transaction.
m0_axil_bvalid_i	sys_clk_i	Input	This signal indicates that the channel is signaling a valid write response.
m0_axil_bready_o	sys_clk_i	Output	This signal indicates that the manager can accept a write response.
m0_axil_araddr_o [63:0]	sys_clk_i	Output	The read address gives the address of the first transfer in a read burst transaction.

Port	Clock Domain	Direction	Description
m0_axil_arvalid_o	sys_clk_i	Output	This signal indicates that the channel is signaling valid read address and control information.
m0_axil_arready_i	sys_clk_i	Input	This signal indicates that the subordinate is ready to accept an address and associated control signals.
m0_axil_rdata_i [31:0]	sys_clk_i	Input	Read data.
m0_axil_rresp_i [1:0]	sys_clk_i	Input	This signal indicates the status of the read transfer.
m0_axil_rvalid_i	sys_clk_i	Input	This signal indicates that the channel is signaling the required read data.
m0_axil_rready_o	sys_clk_i	Output	This signal indicates that the manager can accept the read data and response information.

5. Register Description

Table 5.1. Register Access Abbreviations

Abbreviation	Meaning
RW	Read and Write access
RO	Read Only
WO	Write Only
RW1C	Read Write 1 to Clear

5.1. Hard IP Core Configuration and Status Registers

The Lattice PCIe x1 IP Core configuration registers have default values that are appropriate for most applications. Customers typically would only want to change a small number of values such as Vendor/Device ID and BAR configuration. Such changes can be made through LMMI writes prior to core reset release or through the IP generation. The registers are configured through LMMI and APB interface.

Table 5.2 lists the base address for the Hard IP Core Registers.

Table 5.2. Base address for Hard IP Core Registers

Registers	Base Address
mgmt_tlb	0x2000
mgmt_ptl	0x3000
mgmt_ftl	0x4000
mgmt_ftl_mf[1]	0x5000
mgmt_ftl_mf[2]	0x6000
mgmt_ftl_mf[3]	0x7000
pcie_ll_BASE	0xF000

5.1.1. EP Configuration Settings

The Lattice PCIe x1 IP Core supports Endpoint (EP) operation. The current mode of operation is determined by the core CSR.

The following table illustrates the CSR values that are recommended for EP and RP applications.

Table 5.3. CSR Values Recommended for EP Applications

Register Field	Base address + Offset	EndPoint
mgmt_tlb_ltssm_port_type_ds_us_n	0x2040	1'b0
mgmt_ftl_cfg_type1_type0_n	0x4030	1'b0
mgmt_ftl_decode_ignore_poison	0x4010	1'b0
mgmt_ftl_decode_t1_rx_bypass_msg_dec	0x4014	1'b0
mgmt_ftl_pcie_cap_slot_implemented	0x4080	1'b0
mgmt_ftl_pcie_cap_device_port_type	0x4080	4'h0
mgmt_ftl_id3_class_code	0x4048	User Application Specific
mgmt_ftl_ari_cap_disable	0x40E0	1'b0
mgmt_ftl_msi_cap_disable	0x40E8	1'b0
mgmt_ftl_msi_cap_mult_message_capable	0x40E8	User Application Specific
mgmt_ftl_msix_cap_table_size	0x40F0	User Application Specific

Register Field	Base address + Offset	EndPoint
mgmt_ftl_msix_cap_disable	0x40F0	1'b0 (Enabled)
mgmt_ftl_aer_cap_en_surprise_down_error	0x4100	1'b0

5.1.2. mgmt_tlb (0x2000)

The following are the register sets with the 0x2000 base address.

5.1.2.1. LTSSM Register Set

ltssm_simulation Register 0x0

This register set is used for LTSSM simulation speed reduction.

Table 5.4. ltssm_simulation Register 0x0

Field	Name	Access	Width	Reset	Description
[31:2]	reserved	RO	30	0x0	—
[1]	reduce_ts1	RW	1	0x0	Reduce the minimum number of TS1 transmitted in Polling.Active from 1024 to 16 to shorten simulation time. 0 – Disable 1 – Enable
[0]	reduce_timeouts	RW	1	0x0	Reduce LTSSM timeouts to shorten simulation time. When enabled, 1 ms-> 20 µs, 2 ms->40 µs, 12 ms->60 µs, 24 ms->80 µs, 32 ms->100 µs, and 48 ms->160 µs. 0 – Disable 1 – Enable

ltssm_cfg_lw_start Register 0x34

This register set is used for LTSSM CFG.LWSTART configuration.

Table 5.5. ltssm_cfg_lw_start Register 0x34

Field	Name	Access	Width	Reset	Description
[31:2]	reserved	RO	30	0x0	—
[1:0]	min_time	RW	2	0x0	Minimum time spent in Cfg.LW.Start before exit is permitted. 0 – 4 µs 1 – 16 µs 2 – 64 µs 3 – 256 µs

ltssm_latch_rx Register 0x38

This register set is used for LTSSM latch RX configuration.

Table 5.6. ltssm_latch_rx Register 0x38

Field	Name	Access	Width	Reset	Description
[31:1]	reserved	RO	30	0x0	—
[0]	link_lane	RW	1	0x1	Enable latching each lane's received link and lane numbers and state exit condition during LTSSM Configuration link width negotiation. 0 – Disable. The lane is included in the link if it is receiving the state exit criteria on the clock cycle that the link width and state exit transition is occurring. A received Physical Layer error occurring close to the clock cycle that the link width is being determined results in a reduction of link width even if the lane had previously recorded valid state exit criteria. 1 – Enable. The lane is included in the link if it meets the state exit criteria at any time during the state. This is the recommended setting since received Physical Layer errors are less likely to result in reduced link width.

ltssm_cfg Register 0x3c

This register set is used for LTSSM configuration.

Table 5.7. ltssm_cfg Register 0x3c

Field	Name	Access	Width	Reset	Description
[31:28]	lw_start_updn_end_delay	RW	4	0x9	LTSSM CFG_[US/DS]_LW_START normal CFG_[US/DS]_LW_START TS1 transmissions and parsing of received TS OS begins (lw_start_updn_end_delay * 64) symbols after the bp_ltssm_cfg_lw_start_updn 1 to 0 transition occurs at the end of PHY adaptation. This delay is intended to flush any corrupted PHY rx data due to the PHY adaptation through the Link Layer Core before the Core begins paying attention to received data again.
[27:24]	lw_start_updn_start_delay	RW	4	0x8	LTSSM CFG_[US/DS]_LW_START bp_ltssm_cfg_lw_start_updn==1 assertion is delayed by (lw_start_updn_start_delay * 64) symbols from CFG_[US/DS]_LW_START state entry. The start delay is intended to avoid the PHY beginning adaptation, and thus corrupting the input data, before the link partner data stream has ended. When the Core reaches CFG_[US/DS]_LW_START before the link partner, the link partner may still be in Recovery.Idle with an active data stream. The start delay must be long enough to delay PHY adaptation until the receive data stream has ended or else SKP Data Parity Errors and Receiver Errors can be detected and recorded by the Core due to the PHY corrupting the receive data stream due to adaptation.
[23:12]	lw_start_updn_count	RW	12	0xfa	LTSSM CFG_[US/DS]_LW_START bp_ltssm_cfg_lw_start_updn==1 duration is set to (lw_start_updn_count * 1024) ns. 0==Disabled.

Field	Name	Access	Width	Reset	Description
[11:8]	lw_start_updn_rate_en	RW	4	0xf	LTSSM CFG_[US/DS]_LW_START bp_ltssm_cfg_lw_start_updn==1 rate enable/disable. Controls for which speeds the bp_ltssm_cfg_lw_start_updn feature is supported. One bit is provided to enable/disable each speed supported {5G, 2.5G}. Bit positions for speeds that are not supported by a given core delivery must be set to 0. 0 – Disable feature when at the associated link speed. 1 – Enable feature when at the associated link speed.
[7:6]	reserved	RO	2	0x0	—
[5]	lw_start_updn_eie_en	RW	1	0x0	*lw_start_updn_eie_en* LTSSM CFG_[US/DS]_LW_START bp_ltssm_cfg_lw_start_updn==1 EIE Tx OS enable. 0 – Disabled 1 – Enabled
[4]	lw_start_updn_en_dir_ds	RW	1	0x0	LTSSM CFG_[US/DS]_LW_START bp_ltssm_cfg_lw_start_updn==1 directed down-configure enable. 0 – Do not assert bp_ltssm_cfg_lw_start_updn==1 when the CFG_[US/DS]_LW_START entry is due to locally directed downconfigure. 1 – Assert bp_ltssm_cfg_lw_start_updn==1 when the CFG_[US/DS]_LW_START entry is due to locally directed down-configure.
[3:2]	reserved	RO	2	0x0	—
[1]	lw_start_updn_timer_en	RW	1	0x0	LTSSM CFG_[US/DS]_LW_START bp_ltssm_cfg_lw_start_updn Timer Enable. Register lw_start_updn_timer_en can be set to stay in adaptation for a fixed time period instead of relying on the PHY to have a port bp_ltssm_cfg_lw_start_updn_ack that is asserted at the end of adaptation. Only one of lw_start_updn_timer_en and lw_start_updn_ack_en can be set to 1. 0 – Disabled. 1 – Deassert bp_ltssm_cfg_lw_start_updn after (lw_start_updn_count * 1024) ns has expired.

Field	Name	Access	Width	Reset	Description
[0]	lw_start_updn_ack_en	RW	1	0x0	LTSSM Configuration Link Width Start bp_ltssm_cfg_lw_start_updn Ack Enable 0 – Disabled. Output port bp_ltssm_cfg_lw_start_updn is held == 0 and input port bp_ltssm_cfg_lw_start_updn_ack is ignored. When CFG_[DS/US]_LW_START is entered from Recovery, the transition from CFG_[DS/US]_LW_START to CFG_[DS/US]_LW_ACCEPT occurs after a minimum of 4 µs. 1 – Enabled. If also enabled, through mgmt_tlb_ltssm_cfg_lw_start_updn_rate_en, at the current link speed, output port bp_ltssm_cfg_lw_start_updn is set upon CFG_[DS/US]_LW_START entry from Recovery and input port bp_ltssm_cfg_lw_start_updn_ack is used. The transition from CFG_[DS/US]_LW_START to CFG_[DS/US]_LW_ACCEPT occurs only after the PHY has asserted bp_ltssm_cfg_lw_start_updn_ack == 1 and additionally a minimum of 4 µs has elapsed. bp_ltssm_cfg_lw_start_updn_ack must not be withheld so long that the state timeout of 24 ms expires or the link exits to detect, and the link goes down, which is a serious error.

ltssm_port_type Register 0x40

This register set is used for the LTSSM port type configuration.

Table 5.8. ltssm_port_type Register 0x40

Field	Name	Access	Width	Reset	Description
[31:1]	reserved	RO	31	0x0	—
[0]	ds_us_n	RW	1	0x0	Determines the PCI Express port type which affects many aspects of LTSSM training. 0 – Upstream Port 1 – Downstream Port

ltssm_ds_link Register 0x44

This register set is used for the LTSSM downstream link configuration.

Table 5.9. ltssm_ds_link Register 0x44

Field	Name	Access	Width	Reset	Description
[31:5]	reserved	RO	27	0x0	—
[4:0]	number	RW	5	0x0	For downstream ports only, unique Link Number assigned to the link and used in TS sets during LTSSM Configuration

ltssm_detect_quiet Register 0x48

This register set is used for the LTSSM Detect.Quiet configuration.

Table 5.10. ltssm_detect_quiet Register 0x48

Field	Name	Access	Width	Reset	Description
[31:2]	reserved	RO	30	0x0	—
[1:0]	min_time	RW	2	0x0	Minimum time spent in Detect.Quiet before an exit is permitted. • 0 – 0 ms • 1 – 1 ms • 2 – 2 ms • 3 – 12 ms

ltssm_rx_det Register 0x4c

This register set is used for the LTSSM receiver detection configuration.

Table 5.11. ltssm_rx_det Register 0x4c

Field	Name	Access	Width	Reset	Description
[31]	override	RW	1	0x0	Lane receiver detection mask enable. 0 – Disable 1 – Enable
[30:16]	reserved	RO	15	0x0	—
[15:0]	mask	RW	16	0x0	Lane receiver detection mask. When override==1, mask determines which lanes attempt receiver detection. For each lane[i]: 0 – Skip receiver detection and exclude the lane from the link. 1 – Perform receiver detection and use result to determine whether to include/exclude the lane from the link.

ltssm_nfts Register 0x50

This register set is used for the LTSSM NFTS configuration.

Table 5.12. ltssm_nfts Register 0x50

Field	Name	Access	Width	Reset	Description
[31:16]	reserved	RO	16	0x0	—
[15:8]	to_extend	RW	8	0x7f	Number of FTS set transfer times to wait in addition to the time required to transmit the requested NFTS sets before timing out to Recovery on Rx_L0s exit.
[7:0]	nfts	RW	8	0xff	Number of FTS sets to request link partner transmit when exiting L0s. NFTS value transmitted in TS1 and TS2 Ordered Sets during training.

ltssm_ds_initial_auto Register 0x54

This register set is used for the LTSSM initial link speed configuration.

Table 5.13. ltssm_ds_initial_auto Register 0x54

Field	Name	Access	Width	Reset	Description
[31]	rate_enable	RW	1	0x0	Determines whether link speed up is requested by the core after the first entry to L0 following state Detect. If neither port directs the link to a higher speed, the link remains at 2.5G unless software initiates a speed change. It is recommended to set rate_enable=1 and rate==maximum supported speed. 0 – Let the link partner or software initiate initial speed changes. 1 – Make 1 attempt to direct the link to the maximum speed specified by rate. The speed achieved is the maximum speed, less than or equal to rate, that both the core and link partner support.
[30:2]	reserved	RO	29	0x0	Number of FTS set transfer times to wait in addition to the time required to transmit the requested NFTS sets before timing out to Recovery on Rx_L0s exit.
[1:0]	rate	RW	2	0x0	When rate_enable==1, indicates the maximum rate that is attempted to negotiate on the initial link training from Detect. Only speeds supported by the core can be indicated. 0 – 2.5G 1 – 5G

ltssm_select_deemphasis Register 0x58

This register set is used for the LTSSM 2.5/5G deemphasis configuration.

Table 5.14. ltssm_select_deemphasis Register 0x58

Field	Name	Access	Width	Reset	Description
[31:1]	reserved	RO	31	0x0	—
[0]	6db_3_5db_n	RW	1	0x1	For 5G capable cores only: For upstream ports only, sets the default deemphasis for 5G operation during LTSSM State Detect. 0 – -3.5dB 1 – -6dB

ltssm_beacon Register 0x5c

This register set is used for the LTSSM Beacon configuration.

Table 5.15. ltssm_beacon Register 0x5c

Field	Name	Access	Width	Reset	Description
[31:1]	reserved	RO	31	0x0	—
[0]	l2_d3hot_enable	RW	1	0x0	L2 wake Beacon transmission control. 0 – Disabled. The customer design must wake the link through WAKE# pin assertion. Set to 0 when using PHY which do not support Beacon transmission. Set to 0 if the core is not clocked (some PHY remove the core's clock in L2 while others supply a keep alive clock) or powered (some applications remove core power in L2 to maximize power savings) in L2, as the core is unable to initiate Beacon generation in these cases. 1 – Transmit beacon when directed to wake the link from L2.

ltssm_mod_cpl Register 0x60

This register set is used for the LTSSM Modified Compliance configuration.

Table 5.16. ltssm_mod_cpl Register 0x60

Field	Name	Access	Width	Reset	Description
[31:2]	reserved	RO	31	0x0	—
[1]	one_eieos	RW	1	0x1	When entering Modified Compliance Pattern determines the number of EIEOS blocks to send. 0 – Send 8 EIEOS blocks to ensure receiver lock 1 – Send 1 EIEOS (as per Spec)
[0]	exit_direct_to_detect	RW	1	0x0	*exit_direct_to_detect* When transmitting Modified Compliance Pattern and cfg_enter_compliance == 0, determines which of the two PCIe Specification optional behaviors is selected. 0 – Do not exit to Detect for this reason. 1 – Exit to Detect.

ltssm_rx_elec_idle Register 0x64

This register set is used for the LTSSM Rx Electrical Idle configuration.

Table 5.17. ltssm_rx_elec_idle Register 0x64

Field	Name	Access	Width	Reset	Description
[31]	rec_spd_infer_rcvr_lock	RW	1	0x0	Recovery Speed successful and unsuccessful inference expand to Recovery.RcvrLock enable. 0 – Do not include time spent in Recovery.RcvrLock when calculating successful and unsuccessful speed change electrical idle inference in Recovery.Speed. 1 – Include time spent in Recovery.RcvrLock when calculating successful and unsuccessful speed change electrical idle inference in Recovery.Speed.

Field	Name	Access	Width	Reset	Description
[30]	rec_pd_infer_rcvr_cfg	RW	1	0x0	Recovery Speed successful and unsuccesful inference expand to Recovery.RcvrCfg enable. 0 – Do not include time spent in Recovery.RcvrCfg when calculating successful and unsuccessful speed change electrical idle inference in Recovery.Speed. 1 – Include time spent in Recovery.RcvrCfg when calculating successful and unsuccessful speed change electrical idle inference in Recovery.Speed.
[29]	rec_spd_infer_eq_ph0123	RW	1	0x0	Recovery Speed successful and unsuccessful inference expand to Recovery.EqPhase0123 enable. 0 – Do not include time spent in Recovery.EqPhase0123 when calculating successful and unsuccessful speed change electrical idle inference in Recovery.Speed. 1 – Include time spent in Recovery.EqPhase0123 when calculating successful and unsuccessful speed change electrical idle inference in Recovery.Speed.
[28:4]	reserved	RO	25	0x0	—
[3:0]	filter	RW	4	0x1	After entering a LTSSM state that monitors, pipe_rx_elec_idle for exit, ignore pipe_rx_elec_idle for 128 * filter) nanoseconds to enable tolerance for pipe_rx_elec_idle not latency matched with the associaetd pipe_rx_data.

ltssm_compliance_toggle Register 0x68

This register set is used for the LTSSM Compliance Toggle configuration.

Table 5.18. ltssm_compliance_toggle Register 0x68

Field	Name	Access	Width	Reset	Description
[31:4]	reserved	RO	28	0x0	—
[3:2]	max_speed	RW	2	0x3	Maximum speed of compliance patterns that should be generated 0 – 2.5G 1 – 5G
[1:0]	min_speed	RW	2	0x0	Minimum speed of compliance patterns that should be generated 0 - 2.5G 1 – 5G

ltssm_prevent_rx_ts_entry_to Register 0x6c

This register set is used for the LTSSM State Rx TS Transition Prevention configuration.

LTSSM State Rx TS Transition Prevention. The fields in this register are provided to disable LTSSM state transitions from occurring due to Rx of TS OS. Certain states such as Disable, Hot Reset, Loopback, and Polling.Compliance are entered after receiving a small number (typically 2) of consecutive TS OS with the appropriate Control Symbol bit set. There is a very low, but non-zero probability that transmission errors could cause two consecutive TS OS to falsely assert the critical control bits and cause a false state transition. Some of these states, such as Polling.Compliance and Loopback cannot be exited if falsely entered in this manner. The fields in this register are set to turn off LTSSM state transitions due to Rx TS OS to prevent the false state transitions from being possible due to bit errors. Directed (requested by software) state transitions are not affected by this register; only state transitions due to Rx TS OS are affected. State transitions must not be disabled through this mechanism if those state transitions are necessary for the customer's application. However, it must be noted that the states, for which disables are provided, are used for test purposes or link disabling and would not typically be entered during normal link operation.

Table 5.19. ltssm_prevent_rx_ts_entry_to Register 0x6c

Field	Name	Access	Width	Reset	Description
[31:4]	reserved	RO	28	0x0	_
[3]	compliance	RW	1	0x0	LTSSM to Polling.Compliance Rx TS state transition disable. 0 – Enabled 1 – Disabled
[2]	loopback	RW	1	0x0	LTSSM to Loopback Slave Rx TS state transition disable. 0 – Enabled 1 – Disabled
[1]	hot_reset	RW	1	0x0	LTSSM to Hot Reset Rx TS state transition disable. 0 – Enabled 1 – Disabled
[0]	disable	RW	1	0x0	LTSSM to Disable Rx TS state transition disable. 0 – Enabled 1 – Disabled

ltssm_link Register 0x80

This register is used for the Current Link Status configuration.

Table 5.20. ltssm_link Register 0x80

Field	Name	Access	Width	Reset	Description
[31]	dl_link_up	RO	1	0x0	Data Link Layer link up status. 0 – Down 1 – Up
[30]	pl_link_up	RO	1	0x0	Physical Layer link up status. 0 – Down 1 – Up
[29:20]	Reserved	RO	10	0x0	—
[19:16]	lane_rev_status	RO	4	0x0	Indicates the current lane reversal status: lane_rev_status[0], 1 == Full Reverse is in effect, else 0 lane_rev_status[1], 1 == x2 Reverse is in effect (≥ 4 lane only) else 0 lane_rev_status[2], 1 == x4 Reverse is in effect (≥ 8 lane only) else 0 lane_rev_status[3], 1 == x8 Reverse is in effect (≥ 16 lane only) else 0

Field	Name	Access	Width	Reset	Description
[15]	idle_infer_rec_rcvr_cfg	RW1C	1	0x0	Electrical Idle inference status in Recovery.RcvrCfg. 0 – Otherwise 1 – Event occurred. Write 1 to clear.
[14]	idle_infer_loopback_slave	RW1C	1	0x0	Electrical Idle inference status in Loopback.Active as a Loopback Slave. 0 – Otherwise 1 – Event occurred. Write 1 to clear
[13]	idle_infer_rec_speed2_success	RW1C	1	0x0	Electrical Idle inference status in Recovery.Speed on a successful speed negotiation. 0 – Otherwise 1 – Event occurred. Write 1 to clear.
[12]	idle_infer_rec_speed2_unsuccess	RW1C	1	0x0	Electrical Idle inference status in Recovery.Speed on an unsuccessful speed negotiation. 0 – Otherwise 1 – Event occurred. Write 1 to clear.
[11]	idle_infer_l0_to_rec_rcvr_lock	RW1C	1	0x0	Electrical Idle inference status in L0 – event causes entry into Recovery.RcvrLock. 0 – Otherwise 1 – Event occurred. Write 1 to clear.
[10:9]	Reserved	RO	2	0x0	—
[8]	speed_change_fail	RW1C	1	0x0	Speed Change Failure error indicator. 0 – Otherwise 1 – Speed change failure occurred. Write 1 to clear.
[7:2]	Reserved	RO	6	0x0	—
[1:0]	speed	RO	2	0x0	Current LTSSM Link Speed. Only link speeds supported by the core is be indicated. 0 – 2.5G 1 – 5G

ltssm_ltssm Register 0x84

This register set is used for LTSSM State Machine State configuration.

Table 5.21. ltssm_ltssm Register 0x84

Field	Name	Access	Width	Reset	Description
[31:20]	Reserved	RO	12	0x0	—
[19:16]	sub_state	RO	4	0x1	Current LTSSM Minor State. Encoding varies depending upon the Current LTSSM Major State. For Major State: 0 – DETECT 0 – DETECT_INACTIVE 1 – DETECT_QUIET 2 – DETECT_SPD_CHG0 3 – DETECT_SPD_CHG1 4 – DETECT_ACTIVE0 5 – DETECT_ACTIVE1 6 – DETECT_ACTIVE2 7 – DETECT_P1_TO_P0 8 – DETECT_P0_TO_P1_0 9 – DETECT_P0_TO_P1_1 10 – DETECT_P0_TO_P1_2

Field	Name	Access	Width	Reset	Description
					For Major State: 1 – POLLING 0 – POLLING_INACTIVE 1 – POLLING_ACTIVE_ENTRY 2 – POLLING_ACTIVE 3 – POLLING_CFG 4 – POLLING_COMP 5 – POLLING_COMP_ENTRY 6 – POLLING_COMP_EIOS 7 – POLLING_COMP_EIOS_ACK 8 – POLLING_COMP_IDLE For Major State: 2 – CONFIGURATION 0 – CONFIGURATION_INACTIVE 1 – CONFIGURATION_US_LW_START 2 – CONFIGURATION_US_LW_ACCEPT 3 – CONFIGURATION_US_LN_WAIT 4 - CONFIGURATION_US_LN_ACCEPT 5 – CONFIGURATION_DS_LW_START 6 – CONFIGURATION_DS_LW_ACCEPT 7 – CONFIGURATION_DS_LN_WAIT 8 – CONFIGURATION_DS_LN_ACCEPT 9 – CONFIGURATION_COMPLETE 10 – CONFIGURATION_IDLE For Major State: 3 – L0 0 – L0_INACTIVE 1 – L0_L0 2 – L0_TX_EL_IDLE 3 – L0_TX_IDLE_MIN For Major State: 4 – RECOVERY 0 – RECOVERY_INACTIVE 1 – RECOVERY_RCVR_LOCK 2 – RECOVERY_RCVR_CFG 3 – RECOVERY_IDLE 4 – RECOVERY_SPEED0 5 – RECOVERY_SPEED1 6 – RECOVERY_SPEED2 7 – RECOVERY_SPEED3 8 – RECOVERY_EQ_PH0 9 – RECOVERY_EQ_PH1 10 – RECOVERY_EQ_PH2 11 – RECOVERY_EQ_PH3 For Major State: 5 – DISABLED 0 – DISABLED_INACTIVE 1 – DISABLED_0 2 – DISABLED_1 3 – DISABLED_2 4 – DISABLED_3 For Major State: 6 – LOOPBACK 0 – LOOPBACK_INACTIVE 1 – LOOPBACK_ENTRY

Field	Name	Access	Width	Reset	Description
					2 – LOOPBACK_ENTRY_EXIT 3 – LOOPBACK_EIOS 4 – LOOPBACK_EIOS_ACK 5 – LOOPBACK_IDLE 6 – LOOPBACK_ACTIVE 7 – LOOPBACK_EXIT0 8 – LOOPBACK_EXIT1 For Major State: 7 – HOT_RESET 1 – HOT_RESET_HOT_RESET 2 – HOT_RESET_ LEADER _UP 3 – HOT_RESET_ LEADER _DOWN For Major State: 8 – TX_L0S 0 – TX_L0S_INACTIVE 1 – TX_L0S_IDLE 2 – TX_L0S_TO_L0 3 – TX_L0S_FTS0 4 – TX_L0S_FTS1 For Major State: 9 – L1 0 – L1_INACTIVE 1 – L1_IDLE 2 – L1_SUBSTATE 3 – L1_TO_L0 For Major State: 10 – L2 0 – L2_INACTIVE 1 – L2_IDLE 2 – L2_TX_WAKE0 3 – L2_TX_WAKE1 4 – L2_EXIT 5 – L2_SPEED
[15:4]	Reserved	RO	12	0x0	—
[3:0]	state	RO	4	0x0	Current LTSSM Major State 0 – DETECT 1 – POLLING 2 – CONFIGURATION 3 – L0 4 – RECOVERY 5 – DISABLED 6 – LOOPBACK 7 – HOT_RESET 8 – TX_L0S 9 – L1 10 – L2

ltssm_rx_l0s Register 0x88

This register set is used for the Rx L0s State Machine State configuration.

Table 5.22. ltssm_rx_l0s Register 0x88

Field	Name	Access	Width	Reset	Description
[31:3]	Reserved	RO	29	0x0	—
[2:0]	state	RO	3	0x0	Current LTSSM RX L0s State. 0 – RX_L0S_L0 1 – RX_L0S_ENTRY 2 – RX_L0S_IDLE 3 – RX_L0S_FTS 4 – RX_L0S_REC

l0_to_rec Register 0x8c

This register set is used to report different events causing L0 state to Recovery state transition.

Table 5.23. l0_to_rec Register 0x8c

Field	Name	Access	Width	Reset	Description
[31:15]	reserved	RO	17	0x0	—
[14]	direct_to_detect_fast	RW1C	1	0x0	Recovery is entered from L0 due to assertion of mgmt_tlb_ltssm_direct_to_detect_fast. 0 – Otherwise 1 – Event occurred. Write 1 to clear.
[13]	direct_to_recovery_ch_bond	RW1C	1	0x0	Recovery is entered from L0 due to more lane skew than the Channel Bond circuit can tolerate or is due to channel bond failing to occur within the expected timeout period. 0 – Otherwise 1 – Event occurred. Write 1 to clear.
[12]	direct_to_loopback_entry	RW1C	1	0x0	Recovery is entered from L0 due to being directed into Leader Loopback. 0 – Otherwise 1 – Event occurred. Write 1 to clear.
[11]	directed_speed_change	RW1C	1	0x0	Recovery is entered from L0 due to being directed to make a speed change. This includes the initial hardware-initiated speed change(s) which are made when first exiting Detect.Quiet to L0. 0 – Otherwise 1 – Event occurred. Write 1 to clear.
[10]	l0_to_rec_rcvr_lock_rx_ts12	RW1C	1	0x0	Recovery is entered from L0 due to receiving TS1 or TS2 ordered sets. The link partner is directing Recovery entry. 0 – Otherwise 1 – Event occurred. Write 1 to clear.
[9]	l0_to_rec_rcvr_lock_rx_8g_eie	RW1C	1	0x0	Recovery is entered from L0 due to receiving EIE ordered sets at ≥ 8G. The link partner is directing Recovery entry. 0 – Otherwise 1 – Event occurred. Write 1 to clear.
[8]	l0_to_rec_rcvr_lock_rx_infer	RW1C	1	0x0	Recovery is entered from L0 due to inferring Electrical Idle due to no SKP ordered set received in 128 µs. 0 – Otherwise 1 – Event occurred. Write 1 to clear.

Field	Name	Access	Width	Reset	Description
[7]	direct_to_recovery_phy	RW1C	1	0x0	Recovery is entered from L0 due to receiving a burst of ~1024 clock cycles of data containing PHY errors at 2.5G or 5G. This normally occurs only when the PHY has lost lock on one or more lanes. 0 – Otherwise 1 – Event occurred. Write 1 to clear.
[6]	direct_to_recovery_frame	RW1C	1	0x0	Recovery is entered from L0 due to receiving one or more framing errors at ≥ 8G. This occurs due to Rx bit errors which are expected every few minutes at PCIe Specified BER of 10^-12. 0 – Otherwise 1 – Event occurred. Write 1 to clear.
[5]	direct_to_recovery_replay	RW1C	1	0x0	Recovery is entered from L0 due to the original and three replay TLP transmissions failing to receive ACK DLLP acknowledgment. 0 – Otherwise 1 – Event occurred. Write 1 to clear.
[4]	direct_to_hot_reset	RW1C	1	0x0	Recovery is entered from L0 due to being directed into Hot Reset (Secondary Bus Reset Register). 0 – Otherwise 1 – Event occurred. Write 1 to clear.
[3]	direct_to_disable	RW1C	1	0x0	Recovery is entered from L0 due to being directed into Disable (Link Disable Register). 0 – Otherwise 1 – Event occurred. Write 1 to clear.
[2]	rx_l0s_direct_to_recovery	RW1C	1	0x0	Recovery is entered from L0 due to failing to receive the complete Rx_L0S FTS exit sequence within the PCIe Specification allowed timeout period. 0 – Otherwise 1 – Event occurred. Write 1 to clear.
[1]	autonomous_width_change	RW1C	1	0x0	Recovery is entered from L0 due to directed autonomous width change. 0 – Otherwise 1 – Event occurred. Write 1 to clear.
[0]	directed_retrain_link	RW1C	1	0x0	Recovery is entered from L0 due to directed retrain link (Retrain Link Register). 0 – Otherwise 1 – Event occurred. Write 1 to clear.

ltssm_rx_detect Register 0x90

This register set is used for the Receiver detection status.

Table 5.24. ltssm_rx_detect Register 0x90

Field	Name	Access	Width	Reset	Description
[31:16]	reserved	RO	16	0x0	—
[15:0]	lanes	RO	16	0x0	Per lane receiver detection status. For each lane: 0 – Unconnected 1 – Present

ltssm_configured Register 0x94

This register set is used for the Configured link status.

Table 5.25. ltssm_configured Register 0x94

Field	Name	Access	Width	Reset	Description
[31:25]	reserved	RO	7	0x0	—
[24:16]	link_num	RO	9	0x1ff	Link Number configured during LTSSM Training. link_num == 0x1FF on fundamental reset, changes to 0x1F7 (KPAD) when entering CFG_US_LW_START or CFG_DS_LW_START (the start of LTSSM Configuration) and then changes to the negotiated Link Number determined during LTSSM Configuration when the LTSSM changes from CFG_COMPLETE to CFG_IDLE. This field is provided for diagnostics.
[15:0]	lanes	RO	16	0x0	Per lane configured link status. Each lane status resets to 0. After Receiver Detection results are available, each lane status is updated to show which lanes detected receivers. After a link has been formed, each lane status is updated to show which lanes are part of the configured link. For each lane: 0 – Lane did not configure into the link. 1 – Lane configured into the link.

ltssm_direct_to_detect Register 0x98

This register set is used for the Rec Rcvr Lock to Detect controls.

Table 5.26. ltssm_direct_to_detect Register 0x98

Field	Name	Access	Width	Reset	Description
[31:16]	reserved	RO	16	0x0	—
[15]	fast	RW	1	0x0	A rising edge on this signal instructs the state machine to proceed from L0 or Recovery to Detect as quickly as possible.
[14:8]	Reserved	RO	7	0x0	—
[7:0]	timer	RW	8	0x0	This value determines the timeout delay for the state machine to proceed from Recovery Rcvr Lock to Detect when no TS sets are received. A value of 0 disables this timeout.

ltssm_equalization Register 0x9c

This register set is used for the for ≥ 8G capable cores only: LTSSM equalization status.

Table 5.27. ltssm_equalization Register 0x9c

Field	Name	Access	Width	Reset	Description
[31:1]	reserved	RO	31	0x0	—
[0]	fail	RO	1	0x0	Equalization Failure error indicator. 0 – Otherwise 1 – Equalization failure.

ltssm_crosslink Register 0xa0

This register set is used for the LTSSM crosslink status.

Table 5.28. ltssm_crosslink Register 0xa0

Field	Name	Access	Width	Reset	Description
[31:2]	reserved	RO	30	0x0	—
[1]	ds_us_n	RO	1	0x0	CrossLink port type. When active==1, indicates which personality the port assumed during crosslink negotiation. 0 – Upstream 1 – Downstream
[0]	active	RO	1	0x0	CrossLink active indicator. 0 – Otherwise 1 – Link is operating in a crosslink configuration.

5.1.2.2. Physical Layer Status Register Set

Physical Layer Tx Underflow Error Status Register – 0xa4

Table 5.29. Physical Layer Tx Underflow Error Status Register – 0xa4

Field	Name	Access	Width	Reset	Description
[31:1]	reserved	RO	31	0x0	—
[0]	err_tx_pipe_underflow	RW1C	1	0x0	0 – Otherwise 1 – Physical Layer Tx data needed to be forwarded to the lanes for transmission and some, but not all lanes, were ready to accept data causing some lanes to under low. This bit stays asserted once set. Write 1 to clear.

Table 5.30 illustrates the physical lane RX Status Register set with its offset and register address.

Table 5.30. Physical Lane Rx Status Registers

Register Name	Offset Address	Description
pl_rx0 Register	0xa8	Lane Rx Status 0 register – TS2 and TS1 OS detection [0 to 15 bits]
pl_rx1 Register	0xac	Lane Rx Status 1 – Inverted TS2 and TS1 OS detection [0 to 15 bits]
pl_rx2 Register	0xb0	Lane Rx Status 2 – FTS and SKP OS detection
pl_rx3 Register	0xb4	Lane Rx Status 3 – EIOS detection and EIE detection
pl_rx4 Register	0xb8	Lane Rx Status 4 – Data Block is received and SDS ordered set detection

pl_rx0 Register 0xa8 – Lane Rx Status 0 Register

Table 5.31. pl_rx0 Register 0xa8 – Lane Rx Status 0 Register

Field	Name	Access	Width	Reset	Description
[31]	ts2_detect15	RW1C	1	0x0	ts2_detect[i] is set to 1 when a TS2 ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear
[30]	ts2_detect14	RW1C	1	0x0	ts2_detect[i] is set to 1 when a TS2 ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear
[29]	ts2_detect13	RW1C	1	0x0	ts2_detect[i] is set to 1 when a TS2 ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear

Field	Name	Access	Width	Reset	Description
[28]	ts2_detect12	RW1C	1	0x0	ts2_detect[i] is set to 1 when a TS2 ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear
[27]	ts2_detect11	RW1C	1	0x0	ts2_detect[i] is set to 1 when a TS2 ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear
[26]	ts2_detect10	RW1C	1	0x0	ts2_detect[i] is set to 1 when a TS2 ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear
[25]	ts2_detect9	RW1C	1	0x0	ts2_detect[i] is set to 1 when a TS2 ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear
[24]	ts2_detect8	RW1C	1	0x0	ts2_detect[i] is set to 1 when a TS2 ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear
[23]	ts2_detect7	RW1C	1	0x0	ts2_detect[i] is set to 1 when a TS2 ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear
[22]	ts2_detect6	RW1C	1	0x0	ts2_detect[i] is set to 1 when a TS2 ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear
[21]	ts2_detect5	RW1C	1	0x0	ts2_detect[i] is set to 1 when a TS2 ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear
[20]	ts2_detect4	RW1C	1	0x0	ts2_detect[i] is set to 1 when a TS2 ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear
[19]	ts2_detect3	RW1C	1	0x0	ts2_detect[i] is set to 1 when a TS2 ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear
[18]	ts2_detect2	RW1C	1	0x0	ts2_detect[i] is set to 1 when a TS2 ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear
[17]	ts2_detect1	RW1C	1	0x0	ts2_detect[i] is set to 1 when a TS2 ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear
[16]	ts2_detect0	RW1C	1	0x0	ts2_detect[i] is set to 1 when a TS2 ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear
[15]	ts1_detect15	RW1C	1	0x0	ts1_detect[i] is set to 1 when a TS1 ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear
[14]	ts1_detect14	RW1C	1	0x0	ts1_detect[i] is set to 1 when a TS1 ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear
[13]	ts1_detect13	RW1C	1	0x0	ts1_detect[i] is set to 1 when a TS1 ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear
[12]	ts1_detect12	RW1C	1	0x0	ts1_detect[i] is set to 1 when a TS1 ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear

Field	Name	Access	Width	Reset	Description
[11]	ts1_detect11	RW1C	1	0x0	ts1_detect[i] is set to 1 when a TS1 ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear
[10]	ts1_detect10	RW1C	1	0x0	ts1_detect[i] is set to 1 when a TS1 ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear
[9]	ts1_detect9	RW1C	1	0x0	ts1_detect[i] is set to 1 when a TS1 ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear
[8]	ts1_detect8	RW1C	1	0x0	ts1_detect[i] is set to 1 when a TS1 ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear
[7]	ts1_detect7	RW1C	1	0x0	ts1_detect[i] is set to 1 when a TS1 ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear
[6]	ts1_detect6	RW1C	1	0x0	ts1_detect[i] is set to 1 when a TS1 ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear
[5]	ts1_detect5	RW1C	1	0x0	ts1_detect[i] is set to 1 when a TS1 ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear
[4]	ts1_detect4	RW1C	1	0x0	ts1_detect[i] is set to 1 when a TS1 ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear
[3]	ts1_detect3	RW1C	1	0x0	ts1_detect[i] is set to 1 when a TS1 ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear
[2]	ts1_detect2	RW1C	1	0x0	ts1_detect[i] is set to 1 when a TS1 ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear
[1]	ts1_detect1	RW1C	1	0x0	ts1_detect[i] is set to 1 when a TS1 ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear
[0]	ts1_detect0	RW1C	1	0x0	ts1_detect[i] is set to 1 when a TS1 ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear

pl_rx1 Register 0xac – Lane Rx Status 1

Table 5.32. pl_rx1 Register 0xac – Lane Rx Status 1

Field	Name	Access	Width	Reset	Description
[31]	ts2i_detect15	RW1C	1	0x0	ts2i_detect[i] is set to: 1 when an inverted TS2 ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[30]	ts2i_detect14	RW1C	1	0x0	ts2i_detect[i] is set to: 1 when an inverted TS2 ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear.

Field	Name	Access	Width	Reset	Description
[29]	ts2i_detect13	RW1C	1	0x0	ts2i_detect[i] is set to: 1 when an inverted TS2 ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[28]	ts2i_detect12	RW1C	1	0x0	ts2i_detect[i] is set to: 1 when an inverted TS2 ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[27]	ts2i_detect11	RW1C	1	0x0	ts2i_detect[i] is set to: 1 when an inverted TS2 ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[26]	ts2i_detect10	RW1C	1	0x0	ts2i_detect[i] is set to: 1 when an inverted TS2 ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[25]	ts2i_detect9	RW1C	1	0x0	ts2i_detect[i] is set to: 1 when an inverted TS2 ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[24]	ts2i_detect8	RW1C	1	0x0	ts2i_detect[i] is set to: 1 when an inverted TS2 ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[23]	ts2i_detect7	RW1C	1	0x0	ts2i_detect[i] is set to: 1 when an inverted TS2 ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[22]	ts2i_detect6	RW1C	1	0x0	ts2i_detect[i] is set to: 1 when an inverted TS2 ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[21]	ts2i_detect5	RW1C	1	0x0	ts2i_detect[i] is set to: 1 when an inverted TS2 ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[20]	ts2i_detect4	RW1C	1	0x0	ts2i_detect[i] is set to: 1 when an inverted TS2 ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[19]	ts2i_detect3	RW1C	1	0x0	ts2i_detect[i] is set to: 1 when an inverted TS2 ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[18]	ts2i_detect2	RW1C	1	0x0	ts2i_detect[i] is set to: 1 when an inverted TS2 ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[17]	ts2i_detect1	RW1C	1	0x0	ts2i_detect[i] is set to: 1 when an inverted TS2 ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear.

Field	Name	Access	Width	Reset	Description
[16]	ts2i_detect0	RW1C	1	0x0	ts2i_detect[i] is set to: 1 when an inverted TS2 ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[15]	ts1i_detect15	RW1C	1	0x0	ts1i_detect[i] is set to: 1 when an inverted TS1 ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[14]	ts1i_detect14	RW1C	1	0x0	ts1i_detect[i] is set to: 1 when an inverted TS1 ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[13]	ts1i_detect13	RW1C	1	0x0	ts1i_detect[i] is set to: 1 when an inverted TS1 ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[12]	ts1i_detect12	RW1C	1	0x0	ts1i_detect[i] is set to: 1 when an inverted TS1 ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[11]	ts1i_detect11	RW1C	1	0x0	ts1i_detect[i] is set to: 1 when an inverted TS1 ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[10]	ts1i_detect10	RW1C	1	0x0	ts1i_detect[i] is set to: 1 when an inverted TS1 ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[9]	ts1i_detect9	RW1C	1	0x0	ts1i_detect[i] is set to: 1 when an inverted TS1 ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[8]	ts1i_detect8	RW1C	1	0x0	ts1i_detect[i] is set to: 1 when an inverted TS1 ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[7]	ts1i_detect7	RW1C	1	0x0	ts1i_detect[i] is set to: 1 when an inverted TS1 ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[6]	ts1i_detect6	RW1C	1	0x0	ts1i_detect[i] is set to: 1 when an inverted TS1 ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[5]	ts1i_detect5	RW1C	1	0x0	ts1i_detect[i] is set to: 1 when an inverted TS1 ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[4]	ts1i_detect4	RW1C	1	0x0	ts1i_detect[i] is set to: 1 when an inverted TS1 ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear.

Field	Name	Access	Width	Reset	Description
[3]	ts1i_detect3	RW1C	1	0x0	ts1i_detect[i] is set to: 1 when an inverted TS1 ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[2]	ts1i_detect2	RW1C	1	0x0	ts1i_detect[i] is set to: 1 when an inverted TS1 ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[1]	ts1i_detect1	RW1C	1	0x0	ts1i_detect[i] is set to: 1 when an inverted TS1 ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[0]	ts1i_detect0	RW1C	1	0x0	ts1i_detect[i] is set to: 1 when an inverted TS1 ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear.

pl_rx2 Register 0xb0 – Lane Rx Status 2

Table 5.33. pl_rx2 Register 0xb0 – Lane Rx Status 2

Field	Name	Access	Width	Reset	Description
[31]	fts_detect15	RW1C	1	0x0	fts_detect[i] is set to: 1 when an FTS ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[30]	fts_detect14	RW1C	1	0x0	fts_detect[i] is set to: 1 when an FTS ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[29]	fts_detect13	RW1C	1	0x0	fts_detect[i] is set to: 1 when an FTS ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[28]	fts_detect12	RW1C	1	0x0	fts_detect[i] is set to: 1 when an FTS ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[27]	fts_detect11	RW1C	1	0x0	fts_detect[i] is set to: 1 when an FTS ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[26]	fts_detect10	RW1C	1	0x0	fts_detect[i] is set to: 1 when an FTS ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[25]	fts_detect9	RW1C	1	0x0	fts_detect[i] is set to: 1 when an FTS ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear.

Field	Name	Access	Width	Reset	Description
[24]	fts_detect8	RW1C	1	0x0	fts_detect[i] is set to: 1 when an FTS ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[23]	fts_detect7	RW1C	1	0x0	fts_detect[i] is set to: 1 when an FTS ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[22]	fts_detect6	RW1C	1	0x0	fts_detect[i] is set to: 1 when an FTS ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[21]	fts_detect5	RW1C	1	0x0	fts_detect[i] is set to: 1 when an FTS ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[20]	fts_detect4	RW1C	1	0x0	fts_detect[i] is set to: 1 when an FTS ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[19]	fts_detect3	RW1C	1	0x0	fts_detect[i] is set to: 1 when an FTS ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[18]	fts_detect2	RW1C	1	0x0	fts_detect[i] is set to: 1 when an FTS ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[17]	fts_detect1	RW1C	1	0x0	fts_detect[i] is set to: 1 when an FTS ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[16]	fts_detect0	RW1C	1	0x0	fts_detect[i] is set to: 1 when an FTS ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[15]	skp_detect15	RW1C	1	0x0	skp_detect[i] is set to: 1 when a SKP ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[14]	skp_detect14	RW1C	1	0x0	skp_detect[i] is set to: 1 when a SKP ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[13]	skp_detect13	RW1C	1	0x0	skp_detect[i] is set to: 1 when a SKP ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear.

Field	Name	Access	Width	Reset	Description
[12]	skp_detect12	RW1C	1	0x0	skp_detect[i] is set to: 1 when a SKP ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[11]	skp_detect11	RW1C	1	0x0	skp_detect[i] is set to: 1 when a SKP ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[10]	skp_detect10	RW1C	1	0x0	skp_detect[i] is set to: 1 when a SKP ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[9]	skp_detect9	RW1C	1	0x0	skp_detect[i] is set to: 1 when a SKP ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[8]	skp_detect8	RW1C	1	0x0	skp_detect[i] is set to: 1 when a SKP ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[7]	skp_detect7	RW1C	1	0x0	skp_detect[i] is set to: 1 when a SKP ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[6]	skp_detect6	RW1C	1	0x0	skp_detect[i] is set to: 1 when a SKP ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[5]	skp_detect5	RW1C	1	0x0	skp_detect[i] is set to: 1 when a SKP ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[4]	skp_detect4	RW1C	1	0x0	skp_detect[i] is set to: 1 when a SKP ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[3]	skp_detect3	RW1C	1	0x0	skp_detect[i] is set to: 1 when a SKP ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[2]	skp_detect2	RW1C	1	0x0	skp_detect[i] is set to: 1 when a SKP ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[1]	skp_detect1	RW1C	1	0x0	skp_detect[i] is set to: 1 when a SKP ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[0]	skp_detect0	RW1C	1	0x0	skp_detect[i] is set to: 1 when a SKP ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear.

pl_rx3 Register 0xb4 – Lane Rx Status 3

Table 5.34. pl_rx3 Register 0xb4 – Lane Rx Status 3

Field	Name	Access	Width	Reset	Description
[31]	eie_detect15	RW1C	1	0x0	eie_detect[i] is set to: 1 when an EIE ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear
[30]	eie_detect14	RW1C	1	0x0	eie_detect[i] is set to: 1 when an EIE ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear
[29]	eie_detect13	RW1C	1	0x0	eie_detect[i] is set to: 1 when an EIE ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear
[28]	eie_detect12	RW1C	1	0x0	eie_detect[i] is set to: 1 when an EIE ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear
[27]	eie_detect11	RW1C	1	0x0	eie_detect[i] is set to: 1 when an EIE ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear
[26]	eie_detect10	RW1C	1	0x0	eie_detect[i] is set to: 1 when an EIE ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear
[25]	eie_detect9	RW1C	1	0x0	eie_detect[i] is set to: 1 when an EIE ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear
[24]	eie_detect8	RW1C	1	0x0	eie_detect[i] is set to: 1 when an EIE ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear
[23]	eie_detect7	RW1C	1	0x0	eie_detect[i] is set to: 1 when an EIE ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear
[22]	eie_detect6	RW1C	1	0x0	eie_detect[i] is set to 1 when an EIE ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear
[21]	eie_detect5	RW1C	1	0x0	eie_detect[i] is set to: 1 when an EIE ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear
[20]	eie_detect4	RW1C	1	0x0	eie_detect[i] is set to: 1 when an EIE ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear

Field	Name	Access	Width	Reset	Description
[19]	eie_detect3	RW1C	1	0x0	eie_detect[i] is set to: 1 when an EIE ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear
[18]	eie_detect2	RW1C	1	0x0	eie_detect[i] is set to: 1 when an EIE ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear
[17]	eie_detect1	RW1C	1	0x0	eie_detect[i] is set to: 1 when an EIE ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear
[16]	eie_detect0	RW1C	1	0x0	eie_detect[i] is set to: 1 when an EIE ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear
[15]	eios_detect15	RW1C	1	0x0	eios_detect[i] is set to: 1 when an EIOS ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[14]	eios_detect14	RW1C	1	0x0	eios_detect[i] is set to: 1 when an EIOS ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[13]	eios_detect13	RW1C	1	0x0	eios_detect[i] is set to: 1 when an EIOS ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[12]	eios_detect12	RW1C	1	0x0	eios_detect[i] is set to: 1 when an EIOS ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[11]	eios_detect11	RW1C	1	0x0	eios_detect[i] is set to: 1 when an EIOS ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[10]	eios_detect10	RW1C	1	0x0	eios_detect[i] is set to: 1 when an EIOS ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[9]	eios_detect9	RW1C	1	0x0	eios_detect[i] is set to: 1 when an EIOS ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[8]	eios_detect8	RW1C	1	0x0	eios_detect[i] is set to: 1 when an EIOS ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[7]	eios_detect7	RW1C	1	0x0	eios_detect[i] is set to: 1 when an EIOS ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear.

Field	Name	Access	Width	Reset	Description
[6]	eios_detect6	RW1C	1	0x0	eios_detect[i] is set to: 1 when an EIOS ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[5]	eios_detect5	RW1C	1	0x0	eios_detect[i] is set to: 1 when an EIOS ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[4]	eios_detect4	RW1C	1	0x0	eios_detect[i] is set to: 1 when an EIOS ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[3]	eios_detect3	RW1C	1	0x0	eios_detect[i] is set to: 1 when an EIOS ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[2]	eios_detect2	RW1C	1	0x0	eios_detect[i] is set to: 1 when a EIOS ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[1]	eios_detect1	RW1C	1	0x0	eios_detect[i] is set to: 1 when a EIOS ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[0]	eios_detect0	RW1C	1	0x0	eios_detect[i] is set to: 1 when a EIOS ordered set is received on Lane[i]. 0 otherwise. This bit stays asserted once set. Write 1 to clear.

pl_rx4 Register 0xb8 – L ane Rx Status 4

Table 5.35. pl_rx4 Register 0xb8 – Lane Rx Status 4

Field	Name	Access	Width	Reset	Description
[31]	data_detect15	RW1C	1	0x0	data_detect[i] is set to: 1 when a Data Block is received on Lane[i] at 8G. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[30]	data_detect14	RW1C	1	0x0	data_detect[i] is set to: 1 when a Data Block is received on Lane[i] at 8G. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[29]	data_detect13	RW1C	1	0x0	data_detect[i] is set to: 1 when a Data Block is received on Lane[i] at 8G. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[28]	data_detect12	RW1C	1	0x0	data_detect[i] is set to: 1 when a Data Block is received on Lane[i] at 8G. 0 otherwise. This bit stays asserted once set. Write 1 to clear.

Field	Name	Access	Width	Reset	Description
[27]	data_detect11	RW1C	1	0x0	data_detect[i] is set to: 1 when a Data Block is received on Lane[i] at 8G. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[26]	data_detect10	RW1C	1	0x0	data_detect[i] is set to: 1 when a Data Block is received on Lane[i] at 8G. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[25]	data_detect9	RW1C	1	0x0	data_detect[i] is set to: 1 when a Data Block is received on Lane[i] at 8G. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[24]	data_detect8	RW1C	1	0x0	data_detect[i] is set to: 1 when a Data Block is received on Lane[i] at 8G. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[23]	data_detect7	RW1C	1	0x0	data_detect[i] is set to: 1 when a Data Block is received on Lane[i] at 8G. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[22]	data_detect6	RW1C	1	0x0	data_detect[i] is set to: 1 when a Data Block is received on Lane[i] at 8G. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[21]	data_detect5	RW1C	1	0x0	data_detect[i] is set to: 1 when a Data Block is received on Lane[i] at 8G. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[20]	data_detect4	RW1C	1	0x0	data_detect[i] is set to: 1 when a Data Block is received on Lane[i] at 8G. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[19]	data_detect3	RW1C	1	0x0	data_detect[i] is set to: 1 when a Data Block is received on Lane[i] at 8G. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[18]	data_detect2	RW1C	1	0x0	data_detect[i] is set to: 1 when a Data Block is received on Lane[i] at 8G. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[17]	data_detect1	RW1C	1	0x0	data_detect[i] is set to: 1 when a Data Block is received on Lane[i] at 8G. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[16]	data_detect0	RW1C	1	0x0	data_detect[i] is set to: 1 when a Data Block is received on Lane[i] at 8G. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[15]	sds_detect15	RW1C	1	0x0	sds_detect[i] is set to: 1 when a SDS ordered set is received on Lane[i] at 8G. 0 otherwise. This bit stays asserted once set. Write 1 to clear.

Field	Name	Access	Width	Reset	Description
[14]	sds_detect14	RW1C	1	0x0	sds_detect[i] is set to: 1 when a SDS ordered set is received on Lane[i] at 8G. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[13]	sds_detect13	RW1C	1	0x0	sds_detect[i] is set to: 1 when a SDS ordered set is received on Lane[i] at 8G. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[12]	sds_detect12	RW1C	1	0x0	sds_detect[i] is set to: 1 when a SDS ordered set is received on Lane[i] at 8G. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[11]	sds_detect11	RW1C	1	0x0	sds_detect[i] is set to: 1 when a SDS ordered set is received on Lane[i] at 8G. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[10]	sds_detect10	RW1C	1	0x0	sds_detect[i] is set to: 1 when a SDS ordered set is received on Lane[i] at 8G. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[9]	sds_detect9	RW1C	1	0x0	sds_detect[i] is set to: 1 when a SDS ordered set is received on Lane[i] at 8G. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[8]	sds_detect8	RW1C	1	0x0	sds_detect[i] is set to: 1 when a SDS ordered set is received on Lane[i] at 8G. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[7]	sds_detect7	RW1C	1	0x0	sds_detect[i] is set to: 1 when a SDS ordered set is received on Lane[i] at 8G. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[6]	sds_detect6	RW1C	1	0x0	sds_detect[i] is set to: 1 when a SDS ordered set is received on Lane[i] at 8G. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[5]	sds_detect5	RW1C	1	0x0	sds_detect[i] is set to: 1 when a SDS ordered set is received on Lane[i] at 8G. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[4]	sds_detect4	RW1C	1	0x0	sds_detect[i] is set to: 1 when a SDS ordered set is received on Lane[i] at 8G. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[3]	sds_detect3	RW1C	1	0x0	sds_detect[i] is set to: 1 when a SDS ordered set is received on Lane[i] at 8G. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[2]	sds_detect2	RW1C	1	0x0	sds_detect[i] is set to: 1 when a SDS ordered set is received on Lane[i] at 8G. 0 otherwise. This bit stays asserted once set. Write 1 to clear.

Field	Name	Access	Width	Reset	Description
[1]	sds_detect1	RW1C	1	0x0	sds_detect[i] is set to: 1 when a SDS ordered set is received on Lane[i] at 8G. 0 otherwise. This bit stays asserted once set. Write 1 to clear.
[0]	sds_detect0	RW1C	1	0x0	sds_detect[i] is set to: 1 when a SDS ordered set is received on Lane[i] at 8G. 0 otherwise. This bit stays asserted once set. Write 1 to clear.

5.1.2.3. Debug Register Set

debugself_crosslink Register 0xc0

This register set is used for debug to allow Rx detection when Tx is externally looped back to Rx.

Table 5.36. debugself_crosslink Register 0xc0

Field	Name	Access	Width	Reset	Description
[31:1]	reserved	RO	31	0x0	—
[0]	enable	RW	1	0x0	0 – Otherwise 1 – For debug use only, configure LTSSM so that it links with itself when core Tx is externally looped back to core Rx

debug_rx_det Register 0xc4

This register set is used for the LTSSM receiver detection bypass configuration.

Table 5.37. debug_rx_det Register 0xc4

Field	Name	Access	Width	Reset	Description
[31:17]	reserved	RO	15	0x0	—
[16]	inhibit	RW	1	0x0	Link receiver detection inhibit. 0 – Perform receiver detection and use result to determine whether to include/exclude lanes from the link. 1 – Skip receiver detection and assume receivers are not present on all lanes.
[15:1]	reserved	RO	15	0x0	—
[0]	bypass	RW	1	0x0	Link receiver detection bypass. If both bypass and inhibit are asserted, bypass takes precedence. 0 – Perform receiver detection and use result to determine whether to include/exclude lanes from the link. 1 – Skip receiver detection and assume receivers are present on all lanes.

debug_force_tx Register 0xc8

This register set is used for debug using TX PIPE signals.

Table 5.38. debug_force_tx Register 0xc8

Field	Name	Access	Width	Reset	Description
[31:10]	reserved	RO	22	0x0	—
[9]	deemph_5g_enable	RW	1	0x0	For 5G capable cores only: Force pipe_tx_deemph at 5G enable. 0 – Disable. 1 – Enable. Force phy_tx_deemph at 5G speed to the value specified by deemph_5g_6db_3_5db_n. The force is applied at 5G speed except during Polling.Compliance, where for compatibility with PCI SIG Workshop Electrical Testing, the force is not applied.
[8]	deemph_5g_3_5db_6db_n	RW	1	0x0	For 5G capable cores only: Force pipe_tx_deemph at 5G value. 0 – -6dB 1 – -3.5dB
[7:4]	reserved	RO	4	0x0	—
[3]	margin_enable	RW	1	0x0	Force pipe_tx_margin enable. 0 – Drive pipe_tx_margin per PCIe Specification. 1 – Drive pipe_tx_margin to value.
[2:0]	margin_value	RW	3	0x0	Force pipe_tx_margin Value.

debug_direct_scramble_off Register 0xcc

This register set is used for scrambling disable control for 2.5G and 5G data rate.

Table 5.39. debug_direct_scramble_off Register 0xcc

Field	Name	Access	Width	Reset	Description
[31:1]	reserved	RO	31	0x0	—
[0]	enable	RW	1	0x0	LTSSM direct scrambling disabled at 2.5G and 5G. 0 – Otherwise 1 – Direct to disable scrambling at 2.5G and 5G during Configuration.Complete.

debug_force_scramble_off_fast Register 0xd0

This register set is used for scrambling disable control for 8G data rate.

Table 5.40. debug_force_scramble_off_fast Register 0xd0

Field	Name	Access	Width	Reset	Description
[31:1]	reserved	RO	31	0x0	—
[0]	enable	RW	1	0x0	LTSSM force scrambling disabled at ≥ 8G. 0 – Otherwise 1 – Disable scrambling at ≥ 8G. Only works for simulation when link partner is also disabling scrambling. Cannot be set for hardware because disabling scrambling at ≥ 8G is not permitted per PCIe Specification.

balign Register 0xd4

This register set is used for pipe_block_align_control generation options for 8G data rate. It is not recommended to change the default values of this register.

Table 5.41. balign Register 0xd4

Field	Name	Access	Width	Reset	Description
[31]	state_data_n	RW	1	0x0	When generating pipe_block_align_control (which may be used by some PHY to aid in acquiring ≥ 8G block alignment), select between the LTSSM State Algorithm and the Rx Data Observation Algorithm. 0 – Use Rx Data Observation Algorithm 1 – Use the LTSSM State Algorithm
[30:6]	reserved	RO	25	0x0	—
[5]	exclude_loopback_master	RW	1	0x0	When generating pipe_block_align_control through the LTSSM State algorithm, exclude/include the Loopback Leader state in driving pipe_block_align_control to 0. 0 – Include 1 – Exclude
[4]	exclude_cfg_complete	RW	1	0x0	When generating pipe_block_align_control through the LTSSM State algorithm, exclude/include the CFG_COMPLETE LTSSM state in driving pipe_block_align_control to 0. pipe_block_align_control is only driven to 0 during CFG_COMPLETE after receiving the required Rx exit criteria to state CFG_IDLE. Due to the v, the core may need to stay in CFG_COMPLETE for a while after receiving the Rx exit criteria to also meet the required Tx exit criteria. 0 – Include 1 – Exclude
[3]	exclude_cfg_idle	RW	1	0x0	When generating pipe_block_align_control through the LTSSM State algorithm, exclude/include the CFG_IDLE LTSSM state in driving pipe_block_align_control to 0. 0 – Include 1 – Exclude
[2]	exclude_rec_rcvr_cfg	RW	1	0x0	When generating pipe_block_align_control through the LTSSM State algorithm, exclude/include the REC_RCVR_CFG LTSSM state in driving pipe_block_align_control to 0. pipe_block_align_control is only driven to 0 during REC_RCVR_CFG after receiving the required Rx exit criteria to state REC_IDLE. Due to the PCIe Specification, the core may need to stay in REC_RCVR_CFG for a while after receiving the Rx exit criteria in order to also meet the required Tx exit criteria. 0 – Include 1 – Exclude
[1]	exclude_rec_idle	RW	1	0x0	When generating pipe_block_align_control through the LTSSM State algorithm, exclude/include the REC_IDLE LTSSM state in driving pipe_block_align_control to 0. 0 – Includee 1 – Exclude
[0]	exclude_l0	RW	1	0x0	When generating pipe_block_align_control through the LTSSM State algorithm, exclude/include the L0 and TX_L0s LTSSM states in driving pipe_block_align_control to 0. 0 – Include 1 – Exclude

debug_pipe_rx Register 0xe0

This register set is used for the PIPE Interface Debug status.

Table 5.42. debug_pipe_rx Register 0xe0

Field	Name	Access	Width	Reset	Description
[31:16]	polarity	RO	16	0x0	PHY PIPE Interface pipe_rx_polarity current value. For each lane: 0 – Otherwise 1 – PHY lane has been instructed to invert its receiver polarity to compensate for serial rx_p and rx_n being swapped.
[15:0]	valid	RO	16	0x0	PHY PIPE Interface pipe_rx_valid current value. For each lane: 0 – Otherwise 1 – PHY lane is locked to data stream

debug_direct_to_loopback Register 0x100

This register set is used to enable LTSSM Leader loopback.

Table 5.43. debug_direct_to_loopback Register 0x100

Field	Name	Access	Width	Reset	Description
[31:1]	reserved	RO	31	0x0	—
[0]	enable	RW	1	0x0	LTSSM leader loopback enable. 0 – Otherwise. 1 – Direct LTSSM to Loopback.Leader. Before this field is set to 1, all relevant registers containing Leader Loopback control options must be set to the desired values. When mgmt_tlb_debug_direct_to_loopback == 1 no Leader Loopback control options may be changed.

debug_loopback_control Register 0x104

This register set is used enable different control features related to loopback for 2.5G and 5G data rates.

Table 5.44. debug_loopback_control Register 0x104

Field	Name	Access	Width	Reset	Description
[31:28]	inject_err_lane_select	RW	4	0x0	Lane selection to inject error in Loopback. Only lanes configured by the core may be programmed. 0 = Lane 0. 15 = Lane 15.
[27]	inject_rx_2bit_data_err	RW	1	0x0	When enabled during loopback, the rising edge of inject_rx_2bit_data_err bit injects a back-to-back error on the received loopback data. This simulates the PHY losing lock and increments the counter by one, and the Loopback Leader restarts the loopback pattern so that the PHY can recover symbol lock.
[26]	inject_rx_1bit_data_err	RW	1	0x0	When enabled during loopback, the rising edge of inject_rx_1bit_data_err bit injects a single clk error on the received loopback data. This causes the error count to increment by 1 for each received data byte.
[25]	inject_rx_valid_err	RW	1	0x0	When enabled during loopback, the rising edge of inject_rx_valid_err bit injects a single clk error on the received PIPE PHY interface phy_rx_valid signal. This simulates the PHY losing lock during Loopback Leader operation which causes the error count to increment by one, and the Loopback Leader restarts the loopback pattern so that the PHY can recover symbol lock.

Field	Name	Access	Width	Reset	Description
[24]	inject_rx_skp_err	RW	1	0x0	When enabled during loopback, the rising edge of inject_rx_skp_err bit injects a single clk error on the next received SKP Ordered Set. When a SKP Ordered Set is corrupted, the lane's RX descrambling LFSR goes out of sync with the transmitter lane's scrambling LFSR causing all the subsequent data checks to fail. This simulates the PHY losing lock and increments the counter by one, and the Loopback Leader restarts the loopback pattern so that the PHY can recover symbol lock.
[23:19]	reserved	RO	5	0x0	—
[18:16]	pattern	RW	3	0x0	Loopback data pattern. 0 – Unscrambled PRBS31 Polynomial Pattern using Galois implementation with non-inverted output. The polynomial representation is G(x) = X31 + X28 +1.
[15:9]	reserved	RO	7	0x0	—
[8]	tx_comp_receive	RW	1	0x0	Loopback compliance receive behavior. 0 – Loopback Leader does not assert Compliance Receive (recommended default) 1 – Loopback Leader asserts Compliance Receive in TS sets transmitted during Loopback Entry
[7:2]	reserved	RO	6	0x0	—
[1:0]	speed	RW	2	0x0	Desired speed in loopback. Only speeds supported by the core may be programmed. A speed change is only implemented if Loopback is entered from Configuration; if entered from Recovery, the speed is not changed. 0 – 2.5G 1 – 5G

debug_loopback_master_5g Register 0x108

This register set is used to select Deemphasis values by loopback Leader for 2.5G and 5G data rates.

Table 5.45. debug_loopback_master_5g Register 0x108

Field	Name	Access	Width	Reset	Description
[31:1]	reserved	RO	31	0x0	—
[0]	deemph	RW	1	0x0	Select Deemphasis value used by Loopback Leader when Loopback.Active occurs at 5G data rate. 0 – -6.0dB 1 – -3.5dB

debug_loopback_slave_5g Register 0x10c

This register set is used to select Deemphasis value transmitted in TS sets during loopback for 2.5G and 5G data rates.

Table 5.46. debug_loopback_slave_5g Register 0x10c

Field	Name	Access	Width	Reset	Description
[31:1]	reserved	RO	31	0x0	—
[0]	deemph	RW	1	0x0	Select Deemphasis value transmitted in TS sets for the Follower to use when Loopback.Active occurs at 5G data rate. 0 – -6.0dB 1 – -3.5dB

debug_loopback_master_8g_deemph Register 0x110

This register set is used to select TX Preset related coefficients for 8G data rate for loopback Leader.

Table 5.47. debug_loopback_master_5g Register 0x108

Field	Name	Access	Width	Reset	Description
[31:1]	reserved	RO	31	0x0	—
[0]	deemph	RW	1	0x0	Select Deemphasis value u**sed** by Loopback Master when Loopback.Active occurs at 5G data rate. 0 – -6.0dB 1 – -3.5dB

debug_loopback_slave_5g Register 0x10c

This register set is used for Loopback slave control.

Table 5.48. debug_loopback_slave_5g Register 0x10c

Field	Name	Access	Width	Reset	Description
[31:1]	reserved	RO	31	0x0	—
[30:26]	deemph	RW	1	0x0	Select Deemphasis value transmitted in TS sets for the Slave to use when Loopback.Active occurs at 5G data rate. 0 – -6.0dB 1 – -3.5dB

debug_direct_to_loopback_status Register 0x118

This register set is used for the Leader loopback status.

Table 5.49. debug_direct_to_loopback_status Register 0x118

Field	Name	Access	Width	Reset	Description
[31:16]	sync	RO	16	0x0	Loopback per lane sync to data pattern indicator. For each lane: 0 – Not locked to loopback pattern. 1 – Locked to loopback pattern.
[15:1]	reserved	RO	15	0x0	—
[0]	cfg_entry	RO	1	0x0	Loopback entered from Configuration or Recovery indicator. 0 – Loopback entry is from Recovery. 1 – Loopback entry is from Configuration.

debug_loopback_err_reset Register 0x11c

This register set is used for the Leader loopback error reset.

Table 5.50. debug_loopback_err_reset Register 0x11c

Field	Name	Access	Width	Reset	Description
[31:1]	reserved	RO	31	0x0	—
[0]	enable	RW	1	0x0	Loopback error counter reset. 0 – Leader Loopback error count increments as errors are detected during Leader Loopback – saturating at maximum value. 1 – Reset the leader loopback error count on all lanes to 0x0. The reset stays in force for as long as mgmt_tlb_debug_loopback_err_reset_enable remains at 1.

debug_loopback_err Register 0x120

This register set is used for the Leader loopback error count.

Table 5.51. debug_loopback_err Register 0x120

Field	Name	Access	Width	Reset	Description
[255:0]	count	RO	256	0x0	Loopback per lane error count – 16 bits per lane. Errors are counted only after the lane is locked to the loopback pattern.

5.1.2.4. Physical control Register Set

phy_control Register 0x140

This register set is used for LTSSM PIPE Interface configuration for 2.5G and 5G data rates.

Table 5.52. phy_control Register 0x140

Field	Name	Access	Width	Reset	Description
[31:1]	reserved	RO	31	0x0	—
[0]	pipe_tx_swing	RW	1	0x0	Directly controls the value of pipe_tx_swing which sets PHY 2.5G/5G Transmitter Amplitude. 0 – Full Swing Full Swing is required for most applications. 1 – Reduced Swing Reduced Swing is useful to support low power form factors which encourage or require reduced transmitter amplitudes.

pl_tx_skp Register 0x344

This register set is used for the Physical Layer Transmit SKP Period Control.

Table 5.53. pl_tx_skp Register 0x344

Field	Name	Access	Width	Reset	Description
[31:30]	reserved	RO	2	0x0	—
[29:24]	period_sris_128b130b	RW	6	0x0	The transmit SKP period used when operating at ≥ 8G with the SRIS capability enabled and configured for SRIS = period_sris_128b130b Blocks. PCIe Specification is < 38 blocks. 0 is a special case that selects 36 Blocks. This register must be configured for a PCIe Specification compliant value.

Field	Name	Access	Width	Reset	Description
[23:16]	period_srns_128b130b	RW	8	0x0	The transmit SKP period used when operating at ≥ 8G with the SRIS capability disabled or with SRIS enabled but configured for SRNS = 256 + period_srns_128b130b Blocks. PCIe Specification is 370-375 blocks. 0 is a special case that selects 116 == 372 Blocks. This register must be configured for a PCIe Specification compliant value.
[15:8]	period_sris_8b10b	RW	8	0x0	The transmit SKP period used when operating at ≤ 5G with the SRIS capability enabled and configured for SRIS = period_sris_8b10b Symbol Times. PCIe Specification is < 154 Symbol Times. 0 is a special case that selects 146 Symbol Times. This register must be configured for a PCIe Specification compliant value. The number of symbol times selected must be a multiple of the PHY per lane symbol data width or the lower bits are truncated. For example: For 16-bit per lane PHY, period_sris_8b10b[0] is always treated as 0. For 32-bit per lane PHY, period_sris_8b10b[1:0] are always treated as 00. For 64-bit per lane PHY, period_sris_8b10b[2:0] are always treated as 000.
[7:0]	period_srns_8b10b	RW	8	0x0	The transmit SKP period used when operating at ≤ 5G with the SRIS capability disabled or with SRIS enabled but configured for SRNS = (256 + period_srns_8b10b) * 4 Symbol Times. PCIe Specification is 1180-1538 Symbol Times. 0 is a special case that selects 44 == 1200 Symbol Times. This register must be configured for a PCIe Specification compliant value. The number of symbol times selected must be a multiple of the PHY per lane symbol data width or the lower bits are truncated. For example: For 64-bit per lane PHY period_srns_8b10b[0] is always treated as 0. For 32, 16, and 8-bit per lane PHY, all period_srns_8b10b is relevant.

pl_tx_debug Register 0x348

This register set is used for the Physical Layer Debug Control.

Table 5.54. pl_tx_debug Register 0x348

Field	Name	Access	Width	Reset	Description
[31:3]	reserved	RO	29	0x0	—
[2]	inject_margin_crc_error	RW	1	0x0	Setting this to 1 injects errors into the margin crc value of the control skp ordered set on all lanes
[1]	inject_margin_parity_error	RW	1	0x0	Setting this to 1 injects errors into the margin parity bit of the control skp ordered set on all lanes
[0]	inject_data_parity_error	RW	1	0x0	Setting this to 1 injects errors into the data parity bit of the control skp ordered set on all lanes

pl_ctrl Register 0x34c

This register set is used for the Physical Layer Control.

Table 5.55. pl_ctrl Register 0x34c

Field	Name	Access	Width	Reset	Description
[31:1]	reserved	RO	31	0x0	—
[0]	8b10b_err_rec_entry_sel	RW	1	0x0	Selects the Physical Layer error threshold required to be received in L0, when operating with 8b10b encoding (2.5G/5G speed), before the link is directed to Recovery. 0 – When in L0 and using 8b10b encoding, direct the link to Recovery after receiving a single Physical Layer Error. This is the more conservative setting but has the disadvantage of causing Recovery entry on all L0 Physical Layer errors – even those errors that the link would be able to recover from on its own without having to go through Recovery. 1 – When in L0 and using 8b10b encoding, direct the link to Recovery only after receiving a massive burst of errors (which typically is an indication that there is a persistent problem, such as PHY loss of lock, for which Recovery entry is required to fix). The core implements an error counter. For each enabled PHY Rx clock cycle, the error counter is incremented when a clock cycle contains a Physical Layer error, and the error counter is decremented when a clock cycle contains no Physical Layer errors. If the counter reaches 1023, the link is directed to Recovery.

pl_ts_matching Register 0x350

This register set is used for the Physical Layer TS Match Control.

Table 5.56. pl_ts_matching Register 0x350

Field	Name	Access	Width	Reset	Description
[31:1]	reserved	RO	31	0x0	—
[0]	legacy_mode	RW	1	0x1	Setting this to 1 compares all symbols when matching TS sets (legacy behavior). Setting this to 0 compares only the symbols required to meet specifications.

5.1.2.5. Data Link Layer Control Register Set

dl_retry_timeout Register 0x380

This register set is used for the Replay Timeout Control.

Table 5.57. dl_retry_timeout Register 0x380

Field	Name	Access	Width	Reset	Description
[31:24]	pcie4_symt_sync	RW	8	0x0	Replay Timeout Timer PCIe 4.0 Simplified REPLAY_TIMER Limits Extended Sync==1 Value. {pcie4_symt_sync, 10'h0} = Symbol times to use for Replay Timer when pcie4_enable==1 and Extended Synch==1. 0 is a special case selecting 8'd80.
[23:16]	pcie4_symt_sync_n	RW	8	0x0	Replay Timeout Timer PCIe 4.0 Simplified REPLAY_TIMER Limits Extended Sync==0 Value. {pcie4_symt_sync_n, 10'h0} = Symbol times to use for Replay Timer when pcie4_enable==1 and Extended Synch==0. 0 is a special case selecting 8'd24.
[15]	pcie4_enable	RW	1	0x1	Replay Timeout Timer PCIe 4.0 Simplified REPLAY_TIMER Limits Enable. 0 – Use PCIe 3.0 Specification and prior REPLAY_TIMER Limits from UNADJUSTED REPLAY_TIMER LIMITS tables in the PCIe Specification. 1 – Use PCIe 4.0 Specification Simplified REPLAY_TIMER Limits.
[14:1]	l0s_adj	RW	14	0x180	Replay Timeout L0s Adjustment. The number of symbol times to add to the recommended PCIe Replay Timer timeout period to compensate for the remote link having to exit L0s before it can send an ACK/NAK DLLP. l0s_adj is added to the Replay Timer timeout period after the optional doubling controlled by mult_enable is applied. Not applicable when pcie4_enable==1.
[0]	mult_enable	RW	1	0x0	Replay Timeout Timer Multiplier Enable. Not applicable when pcie4_enable==1. 0 – The Replay Timer timeout period implemented is the recommended (-0%) value in the PCIe Specification UNADJUSTED REPLAY_TIMER LIMITS FOR 2.5/5.0/8.0 GT/S MODE OPERATION BY LINK WIDTH AND MAX_PAYLOAD_SIZE tables. 1 – The Replay Timer timeout period implemented is 2 times the recommended (-0%) value in the PCIe Specification UNADJUSTED REPLAY_TIMER LIMITS FOR 2.5/5.0/8.0 GT/S MODE OPERATION BY LINK WIDTH AND MAX_PAYLOAD_SIZE tables.

dl_ack_timeout_div Register 0x384

This register set is used for the ACK Timer Control.

Table 5.58. dl_ack_timeout_div Register 0x384

Field	Name	Access	Width	Reset	Descriptions
[31:1]	reserved	RO	31	0x0	—
[0]	enable	RW	1	0x0	ACK Timer Control. 0 – Ack according to specifications. 1 – Ack twice as often as recommended by specifications.

dl_ctrl Register 0x390

This register set is used for the Data Link Layer Control.

Table 5.59. dl_ctrl Register 0x390

Field	Name	Access	Width	Reset	Descriptions
[31:26]	reserved	RO	6	0x0	—
[25]	tx_pfx_par_inject_en	RW	1	0x0	Transmit Data Link Layer Prefix Parity Error Injection Enable. 0 – Do not inject error. 1 – On the rising edge, a single prefix parity error injection, applied prior to assigning the TLP sequence number, is scheduled and injected at the next opportunity (TLP transmit). Tx prefix parity error Handling and Reporting are governed by tx_par2_handle_disable and tx_par2_report_disable.
[24]	rx_early_forward_disable	RW	1	0x0	Receive Data Link Layer down-trained early forwarding disable. 0 – When down-trained, forward Rx Data Link Layer data for processing whenever a TLP/DLLP end occurs without a following TLP/DLLP start the same clock cycle. This setting results in lower Rx TLP/DLLP latency. 1 – When down-trained, always aggregate Rx Data Link Layer data to full width before forwarding the data. For example, a x16 core operating at x1 receives and aggregate 16 clock cycles of 1 lane data before outputting one clock cycle of 16 lane data for further processing.
[23]	reserved	RO	1	0x0	—
[22]	tx_gap_inject_en	RW	1	0x0	Transmit Data Link Layer TX Valid Gap Injection Enable. 0 – Do not inject gap. 1 – On the rising edge, a single clock bp_tx_valid gap is scheduled and is injected at the next opportunity (within a TLP). This gap in the bp_tx_valid can cause a data underflow at the Physical Layer.
[21]	rx_malf_inject_en	RW	1	0x0	Receive Data Link Layer Malformed Length TLP Injection Enable. 0 – Do not inject error. 1 – On the rising edge, a single malformed TLP error injection is scheduled and is injected at the next opportunity (Tx TLP EOP). The TLP is malformed by deleting its end of TLP indicator causing the TLP to end at the incorrect location.
[20]	rx_lcrc_inject_en	RW	1	0x0	Receive Data Link Layer LCRC Injection Enable. 0 – Do not inject error. 1 – On the rising edge, a single LCRC TLP error injection is scheduled and injected at the next opportunity (Tx TLP EOP). The LCRC is corrupted by inverting LCRC bit 0 of the received TLP.
[19]	reserved	RO	1	0x0	—

Field	Name	Access	Width	Reset	Descriptions
[18]	rx_dl_active_disable	RW	1	0x0	Control the use of DL_Active to block reception of TLPs. 0 – Block reception of TLPs when dl_active is low. 1 – Do not block TLP reception based on dl_active.
[17]	rx_inhibit_tlp	RW	1	0x0	Receive Data Link Layer TLP Rx Inhibit Enable. 0 – Process received TLPs per PCIe Specification Required setting for compliant PCIe operation. 1 – For test purposes only, discard and do not accept received TLPs. Received TLPs are processed as if the Sequence Number is one greater than received. This prevents the TLP with the current expected Sequence Number and all following TLPs from being received. This causes the link partner to do TLP replays as received TLPs are incorrect due to perceived Sequence Number errors.
[16]	rx_inhibit_ack_nak	RW	1	0x0	Receive Data Link Layer ACK/NAK Inhibit Enable. 0 – Process received ACK and NAK DLLPs per PCIe Specification. Required setting for compliant PCIe operation. 1 – For test purposes only, discard and do not process received ACK and NAK DLLPs. This causes the core to do TLP replays because TLP acknowledgements do not received.
[15]	reserved	RO	1	0x0	—
[14]	tx_par2_report_disable	RW	1	0x0	Transmit Data Link Layer Parity 2 Error Reporting Disable. 0 – Enable reporting. 1 – Disable reporting of Data Link Layer transmit parity errors.
[13]	tx_par2_handle_disable	RW	1	0x0	Transmit Data Link Layer Parity 2 Error Handling Disable. 0 – Enable handling. TLPs with errors are nullified and not retransmitted. 1 – Disable handling of Data Link Layer transmit parity errors. When error handling is disabled, TLPs with parity errors continue to be transmitted.
[12]	tx_par2_inject_en	RW	1	0x0	Transmit Data Link Layer Parity 2 Error Injection Enable. 0 – Do not inject error. 1 – On the rising edge, a single parity error injection, applied after assigning the TLP sequence number, is scheduled and is injected at the next opportunity (TLP transmit).
[11:9]	reserved	RO	3	0x0	—

Field	Name	Access	Width	Reset	Descriptions
[8]	tx_par1_inject_en	RW	1	0x0	Transmit Data Link Layer Parity 1 Error Injection Enable. 0 – Do not inject error. 1 – On the rising edge, a single parity error injection, applied prior to assigning the TLP sequence number, is scheduled and is injected at the next opportunity (TLP transmit).
[7]	reserved	RO	1	0x0	—
[6]	tx_replay_ecc2_report_disable	RW	1	0x0	Transmit Replay Buffer ECC 2-bit Error Reporting Disable. 0 – Enable reporting. 1 – Disable reporting of ECC 2-bit errors.
[5]	tx_replay_ecc2_handle_disable	RW	1	0x0	Transmit Replay Buffer ECC 2-bit Error Handling Disable. 0 – Enable handling. TLPs with errors are nullified and not retransmitted. 1 – Disable handling of ECC 2-bit errors. When error handling is disabled, TLPs with ECC 2-bit errors continue to be transmitted.
[4]	tx_replay_ecc2_inject_en	RW	1	0x0	Transmit Replay Buffer ECC 2-bit Error Injection Enable. 0 – Do not inject error. 1 – On the rising edge, a single ECC 2-bit error injection is scheduled and is injected at the next opportunity (Replay Buffer RAM write). The error is only seen if a Replay occurs of the TLP receiving the error injection.
[3]	reserved	RO	1	0x0	—
[2]	tx_replay_ecc1_report_disable	RW	1	0x0	Transmit Replay Buffer ECC 1-bit Error Reporting Disable. 0 – Enable reporting. 1 – Disable reporting of ECC 1-bit errors.
[1]	tx_replay_ecc1_handle_disable	RW	1	0x0	Transmit Replay Buffer ECC 1-bit Error Handling Disable. 0 – Enable correction. 1 – Disable correction of ECC 1-bit errors. When error correction is disabled, ECC 1-bit errors are treated the same as uncorrectable ECC 2-bit errors.
[0]	tx_replay_ecc1_inject_en	RW	1	0x0	Transmit Replay Buffer ECC 1-bit Error Injection Enable. 0 – Do not inject error. 1 – On the rising edge, a single ECC 1-bit error injection is scheduled and is injected at the next opportunity (Replay Buffer RAM write). The error is only seen if a Replay occurs of the TLP receiving the error injection.

dl_stat Register 0x394

This register set is used for the Data Link Layer Status.

Table 5.60. dl_stat Register 0x394

Field	Name	Access	Width	Reset	Description
[31]	info_bad_tlp_null_err	RW, W1C	1	0x0	Nullified TLP Received Status. This is not a reported error but is useful information to store for debug. This is a sub-class of err_aer_bad_tlp that provides more information as to why the TLP is bad. 0 – Otherwise 1 – Event occurred. Write 1 to clear.
[30]	info_bad_tlp_phy_err	RW, W1C	1	0x0	TLP PHY Error Status. This is not a reported error but is useful information to store for debug. This is a sub-class of err_aer_bad_tlp that provides more information as to why the TLP is bad. 0 – Otherwise 1 – Event occurred. Write 1 to clear.
[29]	info_bad_tlp_malf_err	RW, W1C	1	0x0	Malformed TLP Status. This is not a reported error but is useful information to store for debug. This is a sub-class of err_aer_bad_tlp that provides more information as to why the TLP is bad. 0 – Otherwise 1 – Event occurred. Write 1 to clear.
[28]	info_bad_tlp_ecrc_err	RW, W1C	1	0x0	TLP ECRC Mismatch Status. This is not a reported error but is useful information to store for debug. This is a sub-class of err_aer_bad_tlp that provides more information as to why the TLP is bad. 0 – Otherwise 1 – Event occurred. Write 1 to clear.
[27]	info_schedule_dupl_ack	RW, W1C	1	0x0	Duplicate TLP Received Status. This is not a reported error but is useful information to store for debug. Duplicate TLPs are received during TLP Replay. 0 – Otherwise 1 – Event occurred. Write 1 to clear.
[26]	info_bad_tlp_seq_err	RW, W1C	1	0x0	TLP Sequence Number Mismatch Status. This is not a reported error but is useful information to store for debug. This is a sub-class of err_aer_bad_tlp that provides more information as to why the TLP is bad. 0 – Otherwise 1 – Event occurred. Write 1 to clear.
[25]	info_bad_tlp_crc_err	RW, W1C	1	0x0	TLP LCRC Mismatch Status. This is not a reported error but is useful information to store for debug. This is a sub-class of err_aer_bad_tlp that provides more information as to why the TLP is bad. 0 – Otherwise 1 – Event occurred. Write 1 to clear.

Field	Name	Access	Width	Reset	Description
[24]	info_nak_received	RW, W1C	1	0x0	NAK Received Status. This is not a reported error but is useful information to store for debug. Receiving a NAK indicates that the link partner requested a Replay. 0 – Otherwise 1 – Event occurred. Write 1 to clear.
[23]	info_deskew_overflow_error	RW, W1C	1	0x0	Rx Deskew FIFO Overflow Error Status. The lane-lane skew of Rx data on one or more lanes are latent from the other lanes that the deskew range of the Rx Deskew FIFO is exceeded. This is a correctable error since the core drives the link to Recovery to fix this issue. 0 – Otherwise 1 – Event occurred. Write 1 to clear.
[22]	info_tx_data_underflow	RW, W1C	1	0x0	Physical Layer TLP Transmit Underflow Error Status. A TLP is transmitted by the physical layer and more data is needed to continue the transmission, but no data is provided. This error is normally caused by the Transaction Layer failing to provide TLP data at ≥ PCIe Line Rate. 0 – Otherwise 1 – Event occurred. Write 1 to clear.
[21]	info_replay_started	RW, W1C	1	0x0	A Replay is started. This is not a reported error but is useful information to store for debug. Indicates a Replay occurred on local TX interface due to either Ack Timeout or Nack Reception. 0 – Otherwise 1 – Event occurred. Write 1 to clear.
[20]	reserved	RO	1	0x0	—
[19]	err_aer_tx_par2	RW, W1C	1	0x0	Transmit Data Link Layer Parity 2 Error Status. Indicates that a Data Link Layer transmit parity error is detected after sequence number application. 0 – Otherwise 1 – Event occurred. Write 1 to clear.
[18]	reserved	RO	1	0x0	—
[17]	err_aer_tx_replay_ecc2	RW, W1C	1	0x0	Transmit Replay Buffer ECC 2-bit Error Status. Indicates that an uncorrectable ECC error occurred during Replay. 0 – Otherwise 1 – Event occurred. Write 1 to clear.
[16]	err_aer_tx_replay_ecc1	RW, W1C	1	0x0	Transmit Replay Buffer ECC 1-bit Error Status. Indicates that a correctable ECC error occurred during Replay. 0 – Otherwise 1 – Event occurred. Write 1 to clear.
[15:8]	reserved	RO	8	0x0	—
[7]	reserved	RO	1	0x0	—

Field	Name	Access	Width	Reset	Description
[6]	err_aer_surprise_down	RW, W1C	1	0x0	Surprise Down Error Status. 0 – Otherwise 1 – Error occurred. Errors of this type must be logged in the Transaction Layer AER Capability. Write 1 to clear.
[5]	err_aer_dl_protocol_error	RW, W1C	1	0x0	DL Protocol Error Status. 0 – Otherwise 1 – Error occurred. Errors of this type must be logged in the Transaction Layer AER Capability. Write 1 to clear.
[4]	err_aer_replay_timer_timeout	RW, W1C	1	0x0	Replay Timer Timeout Error Status. 0 – Otherwise 1 – Error occurred. Errors of this type must be logged in the Transaction Layer AER Capability. Write 1 to clear.
[3]	err_aer_replay_num_rollover	RW, W1C	1	0x0	Replay Num Rollover Error Status. 0 – Otherwise 1 – Error occurred. Errors of this type must be logged in the Transaction Layer AER Capability. Write 1 to clear.
[2]	err_aer_bad_dllp	RW, W1C	1	0x0	Bad DLLP Error Status. 0 – Otherwise 1 – Error occurred. Errors of this type must be logged in the Transaction Layer AER Capability. Write 1 to clear.
[1]	err_aer_bad_tlp	RW, W1C	1	0x0	Bad TLP Error Status. 0 – Otherwise 1 – Error occurred. Errors of this type must be logged in the Transaction Layer AER Capability. Write 1 to clear.
[0]	err_aer_receiver_error	RW, W1C	1	0x0	Receiver Error Status. 0 – Otherwise 1 – Error occurred. Errors of this type must be logged in the Transaction Layer AER Capability. Write 1 to clear.

dl_ack_to_nak Register 0x398

This register set is used for the ACK-to-NAK error injection controls.

Table 5.61. dl_ack_to_nak Register 0x398

Field	Name	Access	Width	Reset	Description
[31]	enable	RW	1	0x0	Enable ACK-to-NAK injection. The write to this field must occur after writes to the other fields in this register have established the desired parameters of the injection. 0 – Do nothing 1 – Enable injection and load parameters into the ACK-to-NAK injector.
[30:24]	reserved	RO	7	0x0	—
[23:16]	count	RW	8	0x0	Number of times to replace the ACK with a NAK.
[15:12]	reserved	RO	4	0x0	—
[11:0]	seq_num	RW	12	0x0	Sequence Number of ACK to be changed to a NAK.

dl_inject Register 0x39c

This register set is used for the DLLP CRC/TLP ECRC error injection controls.

Table 5.62. dl_inject Register 0x39c

Field	Name	Access	Width	Reset	Description
[31]	dllp_crc_err_enable	RW	1	0x0	Enable DLLP CRC error injection. The write to this field must occur after writes to the other fields in this register have established the desired parameters of the injection. 0 – Disable the DLLP CRC Error injector. 1 – Enable DLLP CRC Error injector.
[30:28]	reserved	RO	3	0x0	—
[27:16]	dllp_crc_err_rate	RW	12	0x0	Rate at which DLLP CRC errors are to be injected. A value of 0 injects a single DLLP CRC error. A non-zero value injects errors at intervals of Rate*256*clk_period. This field may not be changed while dllp_crc_err_enable==1.
[15:13]	reserved	RO	3	0x0	—
[12]	dllp_inject_enable	RW	1	0x0	Inject a DLLP (transmit) using the data in the dllp_inject_data register. A single DLLP is injected after each rising edge of this signal.
[11:9]	reserved	RO	3	0x0	—
[8]	tlp_seq_err_enable	RW	1	0x0	Modify the sequence number in the next transmitted TLP to an invalid value (bad sequence number error). A single TLP is altered after each rising edge of this signal.
[7:4]	reserved	RO	4	0x0	—
[3]	tlp_lcrc_err_enable	RW	1	0x0	Enable TLP LCRC error injection. The write to this field must occur after writes to the other fields in this register have established the desired parameters of the injection. 0 – Disable the TLP LCRC Error injector 1 – Enable TLP LCRC Error injector

Field	Name	Access	Width	Reset	Description
[2:0]	tlp_lcrc_err_rate	RW	3	0x0	Rate at which TLP LCRC errors are to be injected. A value of 0 injects LCRC errors into all TLPs. A non-zero value injects an error into a TLP and then pass Rate TLPs without error and then repeat. This field may not be changed while tlp_lcrc_err_enable==1.

dllp_inject Register 0x3a0

This register set is used for the DLLP Injector Data.

Table 5.63. dllp_inject Register 0x3a0

Field	Name	Access	Width	Reset	Description
[31:0]	data	RW	32	0x0	Data to include in injected DLLP. This field may not be changed while dl_inject_dllp_inject_enable==1.

Table 5.64. eq_status_table_info Register 0x3dc

Field	Name	Access	Width	Reset	Description
[31:2]	reserved	RO	30	0x0	—
[1]	16g_speed	RO	1	0x0	Indicates speed at which Equalization is executed. This register is reset only by fundamental reset so that the status survives link down and other soft reset conditions. 0 – Equalization run at 8G speed 1 – Equalization run at 16G speed
[0]	done	RO	1	0x0	Indicates that Equalization has been completed, and the table results are valid. This register is reset only by fundamental reset so that the status survives link down and other soft reset conditions.

5.1.3. mgmt_ptl (0x03000)

The following are the register sets with the 0x3000 base address.

5.1.3.1. Simulation Register

Simulation Register 0x0

This register set is used for the Partial Transaction Layer simulation speed reduction.

Table 5.65. Simulation Register 0x0

Field	Name	Access	Width	Reset	Description
[31:1]	reserved	RO	31	0x0	—
[0]	pm_reduce_timeouts	RW	1	0x0	Reduce Power Management State Machine timeouts from their value in ms to their value in µs to shorten simulation time. 0 – Disable 1 – Enable

5.1.3.2. Power Management State Machine Register Set

pm_aspm_l0s Register 0x40

This register set is used for the Power Management State Machine ASPM L0s entry control.

Table 5.66. pm_aspm_l0s Register 0x40

Field	Name	Access	Width	Reset	Description
[31:16]	entry_time	RW	16	0x0	ASPM L0s TX Entry Time in µs. 0 is a special case == 6.9 µs.
[15:1]	reserved	RO	15	0x0	—
[0]	enable	RW	1	0x1	Enable Power Management State Machine to enter ASPM L0s. 0 – Disable 1 – Enable

pm_aspm_l1 Register 0x50

This register set is used for the Power Management State Machine ASPM L1 entry control.

Table 5.67. pm_aspm_l1 Register 0x50

Field	Name	Access	Width	Reset	Description
[31:16]	entry_time	RW	16	0x0	ASPM L1 TX Entry Time in ms. 0 is a special case == 1000 µs
[15:1]	reserved	RO	15	0x0	—
[0]	enable	RW	1	0x1	Enable Power Management State Machine to enter ASPM L1. 0 – Disable 1 – Enable

pm_aspm_l1_min Register 0x54

This register set is used for the Power Management State Machine ASPM L1 re-entry control.

Table 5.68. pm_aspm_l1_min Register 0x54

Field	Name	Access	Width	Reset	Description
[31:30]	reserved	RO	2	0x0	—
[29:16]	reentry_time	RW	14	0x0	When reentry_disable==0, specifies the minimum time between ASPM L1 requests in ns. 0 is a special case == 9500 ns (PCIe Specification value).
[15:1]	reserved	RO	15	0x0	—
[0]	reentry_disable	RW	1	0x0	Disable enforcing a minimum time between ASPM L1 requests. 0 – Enable 1 – Disable

pm_l1 Register 0x60

This register set is used for the Power Management State Machine L1 entry control.

Table 5.69. pm_l1 Register 0x60

Field	Name	Access	Width	Reset	Description
[31:16]	us_port_ps_entry_time	RW	16	0x0	Upstream Ports only: Number of µs to wait for the transmission of the completion to the PowerState Cfg Write that initiated L1 entry, before beginning to block TLPs and enter L1. 0 is a special case == 4 µs.
[15:1]	reserved	RO	15	0x0	—
[0]	enable	RW	1	0x1	Enable Power Management State Machine to enter L1. 0 – Disable 1 – Enable

pm_l1_min Register 0x64

This register set is used for the Power Management State Machine L1 re-entry control.

Table 5.70. pm_l1_min Register 0x64

Field	Name	Access	Width	Reset	Description
[31:24]	reserved	RO	8	0x0	—
[23:16]	ps_reentry_time	RW	8	0x0	Minimum number of µs to wait following an L1 exit when Power State != D0, before re-entering L1 due to Power State != D0. A wait time is needed to give the transaction layer time to process a Power State Cfg Write to D0 that caused L1 exit. 0 is a special case == 50 µs.
[15:0]	reserved	RO	16	0x0	—

pm_l1pmss Register 0x68

This register set is used for the Power Management State Machine L1PMSS control.

Table 5.71. pm_l1pmss Register 0x68

Field	Name	Access	Width	Reset	Description
[31:1]	reserved	RO	31	0x0	—
[0]	ds_drive_clkreq	RW	1	0x0	Enable driving the clkreq_n signal when operating as a downstream port. 0 – Disable 1 – Enable

pm_l2 Register 0x70

This register set is used for the Power Management State Machine L2 entry control.

Table 5.72. pm_l2 Register 0x70

Field	Name	Access	Width	Reset	Description
[31:1]	reserved	RO	31	0x0	—
[0]	enable	RW	1	0x1	Enable Power Management State Machine to enter L2. 0 – Disable 1 – Enable

pm_pme_to_ack_ep Register 0x80

This register set is used for the Power Management State Machine Endpoint PME_TO_Ack control.

Table 5.73. pm_pme_to_ack_ep Register 0x80

Field	Name	Access	Width	Reset	Description
[31:1]	reserved	RO	31	0x0	—
[0]	user_auto_n	RW	1	0x0	For Endpoints only: PME_TO_Ack message transmission scheduling method. Endpoints are required to respond to a PME_Turn_Off message with a PME_TO_Ack message when they are ready to allow power down. 0 – Schedule PME_TO_Ack message automatically on reception of PME_Turn_Off message. 1 – Schedule PME_TO_Ack message under user control through the pm_l2_enter_ack rising edge.

pm_pme_to_ack_ds Register 0x84

This register set is used for the Power Management State Machine Downstream Port PME_TO_Ack control.

Table 5.74. pm_pme_to_ack_ds Register 0x84

Field	Name	Access	Width	Reset	Description
[31:0]	reserved	RO	32	0x0	—

pm_pme Register 0x88

This register set is used for the Power Management State Machine PM_PME control.

Table 5.75. pm_pme Register 0x88

Field	Name	Access	Width	Reset	Description
[31:12]	reserved	RO	20	0x0	—
[11:0]	timeout_threshold	RW	12	0x0	ms to wait for a transmitted PM_PME message to be acknowledged, by clearing of the PME_Status register, before reissuing the PM_PME message. 0xFFF is a special case that disables the timeout mechanism. 0x000 is a special case == 100 ms.

pm_status Register 0x90

This register set is used for the Power Management State Machine Status.

Table 5.76. pm_status Register 0x90

Field	Name	Access	Width	Reset	Description
[31:5]	reserved	RO	27	0x0	—
[4:0]	state	RO	5	0x0	Power Management State Machine State. 0 – IDLE 1 – L1_WAIT_IDLE 2 – L1_WAIT_REPLAY 3 – L1_READY 4 – L1_STOP_DLLP 5 – L1 6 – L1_1 7 – L1_2_ENTRY 8 – L1_2_IDLE 9 – L1_2_EXIT 10 – L1_EXIT 11 – L2_WAIT_IDLE 12 – L2_WAIT_REPLAY 13 – L23_READY 14 – L2_STOP_DLLP 15 – L2 16 – L0S

5.1.3.3. Receive Buffer

vc_rx_c Register 0x108

This register set is used for the Receive Buffer completion handling configuration.

Table 5.77. vc_rx_c Register 0x108

Field	Name	Access	Width	Reset	Description
[31:2]	reserved	RO	30	0x0	—
[1]	force_ro	RW	1	0x0	Force completion relaxed ordering (RO==1) behavior for all completion TLPs, even those with RO==0. Note that setting this register to 1 is not PCIe Specification compliant but this may be fine for some designs since it is acceptable in many designs for C without RO==1 to pass prior P. 0 – Disable. Received completions are handled using the received RO attribute. 1 – Enable. All received completions are handled as if the RO attribute is 1.
[0]	priority	RW	1	0x0	Completion priority enable. 0 – Disable. Arbitration between Posted, Non-Posted, and Completion TLPs is round robin. 1 – Enable. While arbitrating between putting pending received Posted, Non-Posted, and Completion TLPs on the user received TLP interface, completions are given highest priority. Posted and non-posted requests transact only when a completion is not pending or as needed to prevent starving.

vc_rx_adv Register 0x10c

This register set is used for the Receive Buffer completion credit advertisement configuration.

Table 5.78. vc_rx_adv Register 0x10c

Field	Name	Access	Width	Reset	
[31:2]	reserved	RO	30	0x0	—
[1:0]	ch_cd_sel	RW	2	0x0	PCIe Specification requires CH and CD credit advertisements to be infinite for Endpoints and the finite (actual credit values) for Root Port. ch_cd_sel may be configured to over-ride the default PCIe Specification expected behavior. 0 – Implement CH, CD credit advertisements per port type: Endpoints == infinite, Root Port == actual. 1 – Advertise actual CH, CD credits. 2 – Advertise Infinite CH, CD credits

vc_rx_control Register 0x110

This register set is used for the Receive Buffer Parity/ECC control.

Table 5.79. vc_rx_control Register 0x110

Field	Name	Access	Width	Reset	Description
[31:20]	reserved	RO	12	0x0	—
[19]	reserved	RO	1	0x0	—
[18]	par1_report_disable	RW	1	0x0	Receive Buffer Prefix Parity Error and Parity 1 Error Reporting Disable. 0 – Enable reporting. 1 – Disable reporting of detected parity errors.
[17]	pfx_par_inject_en	RW	1	0x0	Receive Buffer Prefix Parity Error Injection Enable. 0 – Do not inject error. 1 – On the rising edge, a single prefix parity error injection is scheduled and is injected at the next opportunity (TLP receipt).

Field	Name	Access	Width	Reset	Description
[16]	par1_inject_en	RW	1	0x0	Receive Buffer Parity 1 Error Injection Enable. 0 – Do not inject error. 1 – On the rising edge, a single parity error injection is scheduled and is injected at the next opportunity (TLP receipt).
[15:13]	reserved	RO	3	0x0	—
[12]	ecc2_report_disable	RW	1	0x0	Receive Buffer ECC 2-bit Error Reporting Disable. 0 – Enable reporting. 1 – Disable reporting of ECC 2-bit errors.
[11]	ecc2_handle_disable	RW	1	0x0	Receive Buffer ECC 2-bit Error Handling Disable. 0 – Enable handling. 1 – Disable handling of ECC 2-bit errors.
[10:9]	reserved	RO	2	0x0	—
[8]	ecc2_inject_en	RW	1	0x0	Receive Buffer ECC 2-bit Error Injection Enable. 1 – On the rising edge, a single ECC 2-bit error injection is scheduled and is injected at the next opportunity (Receive Buffer RAM read).
[7:5]	reserved	RO	3	0x0	—
[4]	ecc1_report_disable	RW	1	0x0	Receive Buffer ECC 1-bit Error Reporting Disable. 0 – Enable reporting. 1 – Disable reporting of ECC 1-bit errors.
[3]	ecc1_handle_disable	RW	1	0x0	Receive Buffer ECC 1-bit Error Handling Disable. 0 – Enable correction. 1 – Disable correction of ECC 1-bit errors. When error correction is disabled, ECC 1-bit errors are treated the same as uncorrectable ECC 2-bit errors.
[2:1]	reserved	RO	2	0x0	—
[0]	ecc1_inject_en	RW	1	0x0	Receive Buffer ECC 1-bit Error Injection Enable. 1 – On the rising edge, a single ECC 1-bit error injection is scheduled and is injected at the next opportunity (Receive Buffer RAM read).

vc_rx_status Register 0x114

Receive Buffer Parity/ECC Status.

Table 5.80. vc_rx_status Register 0x114

Field	Name	Access	Width	Reset	Description
[31:4]	reserved	RO	28	0x0	—
[3]	err_pfx_par	RW, wr:oneToClear	1	0x0	Receive Buffer Prefix Parity Error Detection Status. 0 – Otherwise 1 – Error occurred.
[2]	err_par1	RW, wr:oneToClear	1	0x0	Receive Buffer Parity 1 Error Detection Status. 0 – Otherwise 1 – Error occurred.
[1]	err_ecc2	RW, wr:oneToClear	1	0x0	Receive Buffer ECC 2-bit Error Detection Status. 0 – Otherwise 1 – Error occurred.
[0]	err_ecc1	RW, wr:oneToClear	1	0x0	Receive Buffer ECC 1-bit Error Detection Status. 0 – Otherwise 1 – Error occurred.

vc_rx_credit_status_cfg Register 0x120

This register set is used for the Receive Buffer credit status configuration.

Table 5.81. vc_rx_credit_status_cfg Register 0x120

Field	Name	Access	Width	Reset	Description
[31:2]	reserved	RO	30	0x0	—
[1:0]	sel	RW	2	0x0	Receive Buffer credit status configuration selection. Configures which credit values are displayed in the remaining vc_rx_credit_status registers. 0 – Display static initial credit advertisements. 1 – Display dynamic current credits available. 2 – Display static number of credits implemented by the receiver buffer. This may be larger than the initial credit advertisement values when number of credits implemented exceeds the PCIe Specification advertised maximum values of 127 H and 2047 D. 3 – Reserved.

vc_rx_credit_status_p Register 0x124

This register set is used for the Receive Buffer Posted credit status.

Table 5.82. vc_rx_credit_status_p Register 0x124

Field	Name	Access	Width	Reset	Description
[31:16]	d	RO	16	0x0	Receive Buffer Posted Data credit status. Content dependent on the value of vc_rx_credit_status_cfg_sel.
[15:0]	h	RO	16	0x0	Receive Buffer Posted Header credit status. Content dependent on the value of vc_rx_credit_status_cfg_sel.

vc_rx_credit_status_n Register 0x128

This register set is used for the Receive Buffer Non-Posted credit status.

Table 5.83. vc_rx_credit_status_n Register 0x128

Field	Name	Access	Width	Reset	Description
[31:16]	d	RO	16	0x0	Receive Buffer Non-Posted Data credit status. Content dependent on the value of vc_rx_credit_status_cfg_sel.
[15:0]	h	RO	16	0x0	Receive Buffer Non-Posted Header credit status. Content dependent on the value of vc_rx_credit_status_cfg_sel.

vc_rx_credit_status_c Register 0x12c

This register set is used for the Receive Buffer Completion credit status.

Table 5.84. vc_rx_credit_status_c Register 0x12c

Field	Name	Access	Width	Reset	Description
[31:16]	d	RO	16	0x0	Receive Buffer Completion Data credit status. Content dependent on the value of vc_rx_credit_status_cfg_sel.
[15:0]	h	RO	16	0x0	Receive Buffer Completion Header credit status. Content dependent on the value of vc_rx_credit_status_cfg_sel.

vc_rx_f_oc_update_timer Register 0x130

This register set is used for the Receive Buffer FC Update Timer Control.

Table 5.85. vc_rx_f_oc_update_timer Register 0x130

Field	Name	Access	Width	Reset	Description
[31:3]	reserved	RO	29	0x0	—
[2:1]	div	RW	2	0x0	Divider Control to Reduce Period of FC Updates for Highly Latent Systems 0 – No adjustment 1 – Divide the PCIe Spec Guideline Value by 2 2 – Divide the PCIe Spec Guideline Value by 4 3 – Divide the PCIe Spec Guideline Value by 8
[0]	disable	RW	1	0x0	Disable Control for the FC Update Timer 0 – Enable the FC Update Timer. 1 – Disable FC Update Timer, sending FC Updates on Every Consumed RX Packet

vc_rx_p_flow_ctrl Register 0x134

Receive Buffer Posted TLP Flow Control.

Table 5.86. vc_rx_p_flow_ctrl Register 0x134

Field	Name	Access	Width	Reset	Description
[31:26]	reserved	RO	6	0x0	—
[25:16]	thresh	RW	10	0x10	Receive Buffer Posted TLP Flow Control Threshold Enable. Threshold to use when thresh_en == 1.
[15:9]	reserved	RO	7	0x0	—
[8]	thresh_en	RW	1	0x0	Receive Buffer Posted TLP Flow Control Threshold. 0 – Use threshold provided on vc_rx_p_thresh/ptl_rx_p_thresh input port. 1 – Use threshold provided by the thresh register
[7:1]	reserved	RO	7	0x0	—
[0]	disable	RW	1	0x0	Receive Buffer Posted TLP Flow Control Disable. 0 – Enable Posted TLP Flow Control. 1 – Disable Posted TLP Flow Control

vc_rx_n_flow_ctrl Register 0x138

This register set is used for the Receive Buffer Non-Posted TLP Flow Control.

Table 5.87. vc_rx_n_flow_ctrl Register 0x138

Field	Name	Access	Width	Reset	Description
[31:26]	reserved	RO	6	0x0	—
[25:16]	thresh	RW	10	0x10	Receive Buffer Non-Posted TLP Flow Control Threshold. 0 – Use threshold provided on vc_rx_np_thresh/ptl_rx_np_thresh input port. 1 – Use threshold provided by the thresh register.
[15:9]	reserved	RO	7	0x0	—
[8]	thresh_en	RW	1	0x0	Receive Buffer Non-Posted TLP Flow Control Disable. 0 – Enable Posted TLP Flow Control. 1 – Disable Posted TLP Flow Control.
[7:1]	reserved	RO	7	0x0	—
[0]	disable	RW	1	0x0	—

vc_rx_alloc_size Register 0x140

This register set is used for the Receive Buffer Size Information.

Table 5.88. vc_rx_alloc_size Register 0x140

Field	Name	Access	Width	Reset	Description
[31:24]	c_hdr	RO	8	0x0	Number of Receive Buffer Data storage bytes required for each unit of alloc_ch.
[23:16]	pn_hdr	RO	8	0x0	Number of Receive Buffer Data storage bytes required for each unit of alloc_ph and alloc_nh.
[15:8]	storage_data	RO	8	0x0	Receive Buffer Data Storage size in bytes == 2^storage_data. The storage space required for alloc_ph, alloc_nh, alloc_ch, alloc_pd, alloc_nd, and alloc_cd must be ≤ (2^storage_data) bytes. The space requried for each unit of alloc_ph, alloc_nh, and alloc_cd is indicated in the fields pn_hdr and c_hdr. 16 bytes of space is required for each unit of alloc_pd, alloc_nd, and alloc_cd. Credit allocations must fit within the Receive Buffer Data Storage, or the Receive Buffer is unable to function correctly, which leads to serious errors.
[7:0]	storage_hdr	RO	8	0x0	Receive Buffer Header Storage size in number of TLPs == 2^storage_hdr. alloc_ph + alloc_nh + alloc_ch must be ≤ 2^storage_hdr. Credit allocations must fit within the Receive Buffer Header Storage, or the Receive Buffer is unable to function correctly, which leads to serious errors.

vc_rx_alloc_p Register 0x144

This register set is used for the Receive Buffer Posted TLP Credit Allocation.

Table 5.89. vc_rx_alloc_p Register 0x144

Field	Name	Access	Width	Reset	Description
[31:16]	d	RW	16	0x440	Number of PCI Express PD credits to allocate in the Receive Buffer. The minimum value is (Max Payload Size Supported in Bytes/16). Number of PCI Express CD credits to allocate in the Receive Buffer. The minimum value is (Max Payload Size Supported in Bytes/16). If more than 2047 credits are allocated, the PCIe Core reserves the requested amount of space in the Receive Buffer, but it advertises the maximum allowed PCIe Specification value of 2047.
[15:12]	reserved	RO	4	0x0	—
[11:0]	h	RW	12	0x20	Number of PCI Express PH credits to allocate in the Receive Buffer. The minimum value is 1. If more than 127 credits are allocated, the PCIe Core reserves the requested amount of space in the Receive Buffer but it advertises the maximum allowed PCIe Specification value of 127.

vc_rx_alloc_n Register 0x148

This register set is used for the Receive Buffer Non-Posted TLP Credit Allocation.

Table 5.90. vc_rx_alloc_n Register 0x148

Field	Name	Access	Width	Reset	Description
[31:16]	d	RW	16	0x40	Number of PCI Express ND credits to allocate in the Receive Buffer. The minimum value is 2.
[15:12]	reserved	RO	4	0x0	—
[11:0]	h	RW	12	0x20	Number of PCI Express NH credits to allocate in the Receive Buffer. The minimum value is 1. If more than 127 credits are allocated, the PCIe Core reserves the requested amount of space in the Receive Buffer but it advertises the maximum allowed PCIe Specification value of 127.

vc_rx_alloc_c Register 0x14c

This register set is used for the Receive Buffer Completion TLP Credit Allocation.

Table 5.91. vc_rx_alloc_c Register 0x14c

Field	Name	Access	Width	Reset	Description
[31:16]	d	RW	16	0x700	Number of PCI Express CD credits to allocate in the Receive Buffer. The minimum value is (Max Payload Size Supported in Bytes/16). If more than 2047 credits are allocated, the PCIe Core reserves the requested amount of space in the Receive Buffer, but it advertises the maximum allowed PCIe Specification value of 2047 (or infinite credits when operating as an Endpoint as required by PCIe Specification.).
[15:12]	reserved	RO	4	0x0	—
[11:0]	h	RW	12	0x1c0	Number of PCI Express CH credits to allocate in the Receive Buffer. The minimum value is 1. If more than 127 credits are allocated, the PCIe Core reserves the requested amount of space in the Receive Buffer but it advertises the maximum allowed PCIe Specification value of 127.

vc_rx_alloc_error Register 0x150

This register set is used for the Receive Buffer Allocation Error Status.

Table 5.92. vc_rx_alloc_error Register 0x150

Field	Name	Access	Width	Reset	Description
[31:4]	reserved	RO	28	0x0	—
[3]	d_sum	RO	1	0x0	Receive Buffer Data Storage Allocation Error. The storage space required for alloc_ph, alloc_nh, alloc_ch, alloc_pd, alloc_nd, and alloc_cd must be ≤ (2^storage_data) bytes. The space requried for each unit of alloc_ph, alloc_nh, and alloc_cd is indicated in the fields pn_hdr and c_hdr. 16 bytes of space is required for each unit of alloc_pd, alloc_nd, and alloc_cd. Credit allocations must fit within the Receive Buffer Data Storage, or the Receive Buffer is unable to function correctly, which leads to serious errors. 0 – No Error 1 – Error
[2]	h_sum	RO	1	0x0	Receive Buffer Header Storage Allocation Error. alloc_ph + alloc_nh + alloc_ch must be ≤ 2^storage_hdr. Credit allocations must fit within the Receive Buffer Header Storage, or the Receive Buffer is unable to function correctly, which leads to serious errors. 0 – No Error 1 – Error
[1]	min_d	RO	1	0x0	Receive Buffer Data Credit Allocation Minimum Error. The alloc_pd and alloc_cd minimum value is Max Payload Size Supported. alloc_nd minimum value is 2. alloc_pd, alloc_nd, and alloc_cd must be a multiple of 2 for cores with DATA_WIDTH==256. 0 – No Error 1 – Error

Field	Name	Access	Width	Reset	Description
[0]	min_h	RO	1	0x0	Receive Buffer Header Credit Allocation Minimum Error. alloc_ph, alloc_nh, and alloc_ch must be > 0. 0 – No Error 1 – Error

vc_tx_np_fifo Register 0x180

This register set is used for the Transmit buffer non-posted TLP FIFO configuration.

Table 5.93. vc_tx_np_fifo Register 0x180

Field	Name	Access	Width	Reset	Description
[31:1]	reserved	RO	31	0x0	—
[0]	disable	RW	1	0x0	Transmit buffer non-posted TLP FIFO disable. The transmit non-posted TLP FIFO is present to allow Posted and Completion TLPs to continue to make progress, to avoid deadlock conditions, when Non-Posted TLP transmission in blocked by available credits in the link partner receive buffer. 0 – Enable non-posted TLP FIFO. PCIe Specification required value. 1 – For debug only: Disable non-posted TLP FIFO in which case non-posted TLPs are carried in the same data path as posted and completion TLPs.

vc_tx_status Register 0x184

This register set is used for the Transmit buffer status.

Table 5.94. vc_tx_status Register 0x184

Field	Name	Access	Width	Reset	Description
[31:1]	reserved	RO	31	0x0	—
[0]	remote_credit_block	RO	1	0x0	Set to one whenever a transmit TLP transmission is blocked by insufficient credits in the remote device's receive buffer.

vc_tx_credit_status_p Register 0x190

This register set is used for the TLP transmit Posted credits currently available in link partner receive buffer.

Table 5.95. vc_tx_credit_status_p Register 0x190

Field	Name	Access	Width	Reset	Description
[31:28]	reserved	RO	4	0x0	—
[27:16]	d	RO	12	0x0	Link partner current Receive Buffer Posted Data credits available.
[15:8]	reserved	RO	8	0x0	—
[7:0]	h	RO	8	0x0	Link partner current Receive Buffer Posted Header credits available.

vc_tx_credit_status_n Register 0x194

This register set is used for the TLP transmit Non-Posted credits currently available in link partner receive buffer.

Table 5.96. vc_tx_credit_status_n Register 0x194

Field	Name	Access	Width	Reset	Description
[31:28]	reserved	RO	4	0x0	—
[27:16]	d	RO	12	0x0	Link partner current Receive Buffer Non-Posted Data credits available.
[15:8]	reserved	RO	8	0x0	—
[7:0]	h	RO	8	0x0	Link partner current Receive Buffer Non-Posted Header credits available.

vc_tx_credit_status_c Register 0x198

This register set is used for the TLP transmit Completion credits currently available in link partner receive buffer.

Table 5.97. vc_tx_credit_status_c Register 0x198

Field	Name	Access	Width	Reset	Description
[31:28]	reserved	RO	4	0x0	—
[27:16]	d	RO	12	0x0	Link partner current Receive Buffer Completion Data credits available.
[15:8]	reserved	RO	8	0x0	—
[7:0]	h	RO	8	0x0	Link partner current Receive Buffer Completion Header credits available.

vc_tx_credit_cleanup Register 0x19c

This register set is used for the TLP transmit error credit cleanup control.

Table 5.98. vc_tx_credit_cleanup Register 0x19c

Field	Name	Access	Width	Reset	Description
[31:1]	reserved	RO	31	0x0	—
[0]	method	RW	1	0x0	TLP Transmit Credit Cleanup Method. 0 – Use the headers of the cleaned-up TLPs to recover the credits. The credits in TLPs with corrupted headers are not recovered. 1 – Use a credit lookup table based on the ID assigned to the TLP. This table is implemented in pcie_user_if.

5.1.3.4. TLP Transmit Control

tlp_tx Register 0x1c4

This register set is used to enable TD bit.

Table 5.99. tlp_tx Register 0x1c4

Field	Name	Access	Width	Reset	Description
[31:1]	reserved	RO	31	0x0	—
[0]	td1_means_add_has_n	RW	1	0x0	TLP Transmit TD==1 Header Field Interpretation. 0 – When a TLP is transmitted with TLP header bit TD==1, this means that the TLP already contains an ECRC. The core transmits the TLP with the TLP's existing ECRC and does not attempt to generate/append a new ECRC. 1 – Not supported for Full Transaction Layer cores like the CrossLink-NX cores. td1_means_add_has_n must be set to 0.

5.1.3.5. FC Credit Init Control.

fc_credit_init Register 0x1c8

This register set is used to force the core to reperform FC credit initialization.

Table 5.100. fc_credit_init Register 0x1c8

Field	Name	Access	Width	Reset	Description
[31:1]	reserved	RO	31	0x0	—
[0]	redo	RW	1	0x0	Force the core to redo FC Credit Initialization without taking the link down. This is only possible if both ends of the link are instructed to redo the initialization.

5.1.4. mgmt_ftl (0x04000)

5.1.4.1. Simulation Register

simulation Register 0x0

This register set is used for simulation only such as Full Transaction Layer simulation speed reduction.

Table 5.101. simulation Register 0x0

Field	Name	Access	Width	Reset	Description
[31:1]	reserved	RO	31	0x0	—
[0]	reduce_timeouts	RW	1	0x0	Reduce timeouts to shorten simulation time. When enabled ms timeouts are shortened to the value in µs. 0 – Disable 1 – Enable

5.1.4.2. Transaction Layer Decode Configuration Register

decode Register 0x10

This register set is used for the Transaction Layer Decode configuration.

Table 5.102. decode Register 0x10

Field	Name	Access	Width	Reset	Description
[31:24]	reserved	RO	8	0x0	—
[23:18]	reserved	RO	6	0x0	—
[17]	tx_bypass_decode_en	RW	1	0x0	Bypass the TLP decode block in the Transmit path. 0 – Decode_in_path module is enabled. 1 – Decode_in_path module is bypassed.
[16]	rx_bypass_decode_en	RW	1	0x0	Bypass the TLP decode block in the Receive path. 0 – Decode_in_path module is enabled. 1 – Decode_in_path module is bypassed.
[15:11]	reserved	RO	5	0x0	—
[10]	tx_convert_ur_to_ca	RW	1	0x0	When decoding TX packets convert Unsupported Request (UR) packets to Completer Abort (CA). 0 – Normal Operation. 1 – Convert UR to CA.
[9]	rx_convert_ur_to_ca	RW	1	0x0	When decoding RX packets convert Unsupported Request (UR) packets to Completer Abort (CA). 0 – Normal Operation. 1 – Convert UR to CA.
[8]	t0_rx_bypass_msg_dec	RW	1	0x0	When implementing Type 0 Configuration Space (Endpoint) – Bypass RX Message TLP Decode Enable. 0 – Normal operation. The core claims and does not forward Message TLPs to the TLP Receive Interface. 1 – All valid Msg TLPs received on PCIe (except Routed by ID and Routed by Address which are routed according to the routing type) are forwarded to the TLP Receive Interface.
[7:3]	reserved	RO	5	0x0	—
[2]	vendor0_ur	RW	1	0x1	Vendor Type 0 Messages received from PCIe are reported as UR. 0 – Do not report received Vendor Type 0 Messages as Unsupported Request (UR). 1 – Report received Vendor Type 0 Messages as Unsupported Request (UR).
[1]	target_only	RW	1	0x0	Target Only. Enable for user designs that implement purely target-only functionality. When enabled all received completions are considered Unexpected Completions and are not forwarded to the TLP Receive Interface. 0 – Disable 1 – Enable

Field	Name	Access	Width	Reset	Description
[0]	ignore_poison	RW	1	0x1	Ignore Poison – Set to 1 to have the core ignore the EP poison indicator for received TLPs with data payload that do not terminate in the core. When set to 1, the core passes all poisoned TLPs to you the same way it would pass the TLP if the TLP is not poisoned. Note that the Ignore Poison control is forced to 1 by the core when the core is configured as a Root-Port. Note that the following TLP types ignore the setting of this bit. Poisoned Configuration Type 0 writes is terminated in the core in all cases, independent of the Ignore Poison bit setting. A completion with UR status is generated and the appropriate error message, ERR COR or ERR FAT, is generated if not masked. Note that Poisoned Configuration Type 0 reads are always treated as if they were not poisoned. The read completes with successful completion status and an optional Advisory Non-Fatal Error status is set provided the severity level is set to NON-FATAL. Poisoned packets without data payload is passed to you in all cases since EP should not be set on packets without data payload and these packets should generally be handled as if they were not poisoned or alternatively handled as Advisory Non-Fatal Errors by user logic. Poisoned Vendor-defined Type 1 messages with data payload are always passed to you and, if ignore poison is 0, additionally an Advisory Non-Fatal Error status is set provided the severity level is set to NON-FATAL. When Ignore Poison is set to 0, the core handles the remaining poisoned TLPs with data payload as follows. Poisoned Write request and poisoned read completions with data TLPs are consumed by the core and handled as TLP Poisoned errors that generate the appropriate poison, ERR NON-FATAL or ERR FATAL, depending upon the error severity register error message. Poisoned Message with data payload (other than vendor-defined type 1) are consumed by the core and handled as TLP Poisoned errors that generate the appropriate poison, ERR NON-FATAL or ERR FATAL depending upon the error severity register, error message. The recommended default for target-only endpoints is to set Ignore Poison == 0 and to have user logic ignore the EP header bit on TLPs that it receives. In this case poisoned TLPs with data payload (other than config 0 writes and vendor-defined type 1 messages) generates a NON-FATAL error message and is discarded by the core. Poisoned TLPs without data payload (for which EP does not apply) is processed as if they were not poisoned. 0 – Disable 1 – Enable

decode_t1 Register 0x14

This register set is used for the Type 1 Configuration Space Transaction Layer Decode configuration.

Table 5.103. decode_t1 Register 0x14

Field	Name	Access	Width	Reset	Description
[31:0]	reserved	RO	32	0x0	—

5.1.4.3. Transaction Layer TLP Processing Configuration Register

tlp_processing Register 0x18

This register set is used for the Transaction Layer TLP Processing configuration.

Table 5.104. tlp_processing Register 0x18

Field	Name	Access	Width	Reset	Description
[31:24]	reserved	RO	8	0x0	—
[23:16]	reserved	RO	8	0x0	—
[15:8]	reserved	RO	8	0x0	—
[7:2]	reserved	RO	6	0x0	—
[1]	ignore_ecrc	RW	1	0x0	Ignore ECRC Error Enable. When enabled ECRC errors are ignored for TLPs passed to you in the TLP Receive Interface. 0 – Disable 1 – Enable
[0]	crs_enable	RW	1	0x0	Configuration Request Retry Status Enable. 0 – Disable. Type 0 Configuration Writes and Reads are performed normally. 1 – Enable. Type 0 Configuration Writes and Reads return Configuration Request Retry Status.

5.1.4.4. Initial Register

Initial Register 0x20

This register set is used for the initial speed and width configuration.

Table 5.105. Initial Register 0x20

Field	Name	Access	Width	Reset	Description
[31:19]	reserved	RO	13	0x0	—
[18:16]	max_link_width	RW	3	0x5	Max Link Width Override. This setting, if different from zero, overrides the value of Maximum Link Width in the PCIe Link Capabilities register. 0 – Maximum core lane width 1 – 1 lane 2 – 2 lanes 3 – 4 lanes 4 – 8 lanes 5 – 16 lanes 6 – Reserved6

Field	Name	Access	Width	Reset	Description
					7 – Reserved7
[15:2]	reserved	RO	14	0x0	—
[1:0]	target_link_speed	RW	2	0x3	Initial value of Target Link Speed Configuration Register. Determines the maximum initial link speed which can be reached during initial training. Must be set to the lesser of the maximum speed supported by the core and the maximum speed at which you desired the core to operate. 0 – 2.5G 1 – 5.0G 2 – 8.0G 3 – 16.0G

5.1.4.5. Configuration Register Type

cfg Register 0x30

This register set is used for the Configuration Register type.

Table 5.106. cfg Register 0x30

Field	Name	Access	Width	Reset	Description
[31:1]	reserved	RO	31	0x0	—
[0]	type1_type0_n	RW	1	0x0	Determines the type of Configuration Registers implemented by the core. 0 – Type 0 – Endpoint 1 – Reserved

5.1.4.6. Downstream Port Configuration

ds_port Register 0x34

This register set is used for Downstream Port configuration.

Table 5.107. ds_port Register 0x34

Field	Name	Access	Width	Reset	Description
[31:17]	reserved	RO	15	0x0	—
[16]	rcb	RW	1	0x0	Read Completion Boundary (RCB). RCB value advertised when the core is operating as a Root Port.
[15:0]	id	RW	16	0x0	Root Port ID. This 16-bit field is used to define the ID used for PCIe Requester ID and Completer ID when the core is operating as a Root Port.

5.1.4.7. Upstream Port Configuration

us_port Register 0x38

This register set is used for the Upstream Port configuration.

Table 5.108. us_port Register 0x38

Field	Name	Access	Width	Reset	Description
[31:1]	reserved	RO	31	0x0	—
[0]	adv_target_link_speed	RW	1	0x0	For an upstream port, advertise the link speeds specified by the target_link_speed field rather than the maximum supported speed.

5.1.4.8. Device ID Configuration

id1 Register 0x40

This register set is used for the ID1 configuration.

Table 5.109. id1 Register 0x40

Field	Name	Access	Width	Reset	Description
[31:16]	device_id	RW	16	0xe004	Value returned when the Device ID Configuration Register is read.
[15:0]	vendor_id	RW	16	0x19aa	Value returned when the Vendor ID Configuration Register is read.

id2 Register 0x44

This register set is used for the ID2 configuration.

Table 5.110. id2 Register 0x44

Field	Name	Access	Width	Reset	Description
[31:16]	subsystem_id	RW	16	0xe004	Value returned when the Subsystem ID Configuration Register is read.
[15:0]	subsystem_vendor_id	RW	16	0x19aa	Value returned when the Subsystem Vendor ID Configuration Register is read.

id3 Register 0x48

This register set is used for the ID3 configuration.

Table 5.111. id3 Register 0x48

Field	Name	Access	Width	Reset	Description
[31:8]	class_code	RW	24	0x118000	Value returned when the Class Code Configuration Register is read. Must be set to the correct value for the type of device being implemented; see PCI Local Bus Specification Revision 2.3 Appendix D for details on setting Class Code.
[7:0]	revision_id	RW	8	0x4	Value returned when the Revision ID Configuration Register is read.

5.1.4.9. Cardbus Configuration

Cardbus Register 0x4c

This register set is used for the Cardbus configuration.

Table 5.112. Cardbus Register 0x4c

Field	Name	Access	Width	Reset	Description
[31:0]	cis_pointer	RW	32	0x0	Value returned when the Cardbus CIS Pointer Configuration Register is read. Set to 0x00000000 unless a Cardbus CIS structure is implemented in memory (which is rare), in which case set to the address of the CIS Structure.

5.1.4.10. Interrupt Configuration

Legacy Interrupt Register 0x50

This register set is used for the Legacy Interrupt configuration.

Table 5.113. Legacy Interrupt Register 0x50

Field	Name	Access	Width	Reset	Description
[31:10]	reserved	RO	22	0x0	—
[9:8]	pin	RW	2	0x0	Selects which legacy interrupt is used. 0 – INTA 1 – INTB 2 – INTC 3 – INTD
[7:1]	reserved	RO	7	0x0	—
[0]	disable	RW	1	0x0	Disable Legacy Interrupt. 0 – Enable 1 – Disable

Note: When using the MSI interrupt, the Legacy interrupt register must also set to Enable, bit[0]=0 (Enable), either through the IP user interface or write through LMMI interface.

5.1.4.11. BAR Configuration

bar0 Register 0x60

This register set is used for the BAR0 configuration.

Table 5.114. bar0 Register 0x60

Field	Name	Access	Width	Reset	Description
[31:0]	cfg	RW	32	0xffff000c	Configuration of BAR0 (Cfg address 0x10). Use to define a 32-bit Memory or I/O region or combine with an adjacent BAR to define a 64-bit Memory region. The following provides an example for requesting address space by setting BAR registers. Bit [0] – 0 for memory space request; 1 for I/O space request. Bits[2:1] – 00 for 32-bit memory address space; 10 for 64-bit memory address space. Bit[3] – 0 for non-prefetchable memory; 1 for prefetchable memory. Bits[31:4] – Indicate the size of required address space by resetting least significant bits. Example 1: 32'hFFFF_F000 requests for memory space(bit[0]=0), 32-bit address space(bit[2:1]=00), non-prefetchable memory(bit[3]=0) and 4KB address space (bits[31:4]=FFFF_F00)

bar1 Register 0x64

This register set is used for the BAR1 configuration.

Table 5.115. bar1 Register 0x64

Field	Name	Access	Width	Reset	Description
[31:0]	cfg	RW	32	0xffffffff	Configuration of bar1 (Cfg address 0x14). Use to define a 32-bit Memory or I/O region or combine with an adjacent BAR to define a 64-bit Memory region.

bar2 Register 0x68

This register set is used for the BAR2 configuration.

Table 5.116. bar2 Register 0x68

Field	Name	Access	Width	Reset	Description
[31:0]	cfg	RW	32	0xffffe00c	Configuration of bar2 (Cfg address 0x18). Use to define a 32-bit Memory or I/O region or combine with an adjacent BAR to define a 64-bit Memory region.

bar3 Register 0x6c

This register set is used for the BAR3 configuration.

Table 5.117. bar3 Register 0x6c

Field	Name	Access	Width	Reset	Description
[31:0]	cfg	RW	32	0xffffffff	Configuration of bar3 (Cfg address 0x1C). Use to define a 32-bit Memory or I/O region or combine with an adjacent BAR to define a 64-bit Memory region.

bar4 Register 0x70

This register set is used for the BAR4 configuration.

Table 5.118. bar4 Register 0x70

Field	Name	Access	Width	Reset	Description
[31:0]	cfg	RW	32	0xffffe00c	Configuration of bar4 (Cfg address 0x20). Use to define a 32-bit Memory or I/O region or combine with an adjacent BAR to define a 64-bit Memory region.

bar5 Register 0x74

This register set is used for the BAR5 configuration.

Table 5.119. bar5 Register 0x74

Field	Name	Access	Width	Reset	Description
[31:0]	cfg	RW	32	0xffffffff	Configuration of bar5 (Cfg address 0x24). Use to define a 32-bit Memory or I/O region or combine with an adjacent BAR to define a 64-bit Memory region.

5.1.4.12. Expansion ROM Configuration

exp_rom Register 0x78

This register set is used for the Expansion ROM configuration.

Table 5.120. exp_rom Register 0x78

Field	Name	Access	Width	Reset	Description
[31:0]	cfg	RW	32	0x0	Configuration of exp_rom. Use to define a 32-bit Memory Expansion ROM region. If an Expansion ROM region is defined, the region must map to PCIe-compliant Expansion ROM code, or the device may fail to boot.

5.1.4.13. PCI Express Configuration

pcie_cap Register 0x80

This register set is used for the PCI Express Capabilities configuration.

Table 5.121. pcie_cap Register 0x80

Field	Name	Access	Width	Reset	Description
[31:14]	reserved	RO	18	0x0	—
[13:9]	interrupt_message_number	RW	5	0x0	MSI/MSI-X vector is used for the interrupt message generated in association with any of the status bits of the PCI Express Capability structure.
[8]	slot_implemented	RW	1	0x0	Indicates that the Link associated with this Port is connected to a slot. This field is valid for Downstream Ports only.

Field	Name	Access	Width	Reset	Description
[7:4]	device_port_type	RW	4	0x0	Indicates the specific type of this PCI Express Function. 0 – PCI Express Endpoint 1 – Legacy PCI Express Endpoint 2 – Reserved 3 – Reserved 4 – Reserved 5 – Reserved 6 – Reserved 7 – Reserved 8 – Reserved 9 – Reserved 10 – Reserved 11 – Reserved 12 – Reserved 13 – Reserved 14 – Reserved 15 – Reserved
[3:0]	capability_version	RW	4	0x2	Indicates PCI-SIG defined PCI Express Capability structure version number. Must be set to 0x2.

5.1.4.14. PCI Express Configuration

pcie_cap Register 0x80

This register set is used for the PCI Express Capabilities configuration.

Table 5.122. pcie_cap Register 0x80

Field	Name	Access	Width	Reset	Description
[31:14]	reserved	RO	18	0x0	—
[13:9]	interrupt_message_number	RW	5	0x0	MSI/MSI-X vector is used for the interrupt message generated in association with any of the status bits of the PCI Express Capability structure.
[8]	slot_implemented	RW	1	0x0	Indicates that the Link associated with this Port is connected to a slot. This field is valid for Downstream Ports only.

Field	Name	Access	Width	Reset	Description
[7:4]	device_port_type	RW	4	0x0	Indicates the specific type of this PCI Express Function. 0 – PCI Express Endpoint 1 – Legacy PCI Express Endpoint 2 – Reserved 3 – Reserved 4 – Reserved 5 – Reserved 6 – Reserved 7 – Reserved 8 – Reserved 9 – Reserved 10 – Reserved 11 – Reserved 12 – Reserved 13 – Reserved 14 – Reserved 15 – Reserved
[3:0]	capability_version	RW	4	0x2	Indicates PCI-SIG defined PCI Express Capability structure version number. Must be set to 0x2.

pcie_dev_cap Register 0x84

This PCI Express Device Capabilities configuration.

Table 5.123. pcie_dev_cap Register 0x84

Field	Name	Access	Width	Reset	Description
[31:29]	reserved	RO	3	0x0	—
[28]	disable_flr_capability	RW	1	0x0	Function Level Reset Capability 0 – Enable 1 – Disable
[27:26]	reserved	RO	2	0x0	—
[25:18]	reserved	RO	8	0x0	—
[17:16]	reserved	RO	2	0x0	—
[15]	reserved	RO	1	0x0	—
[14:13]	reserved	RO	2	0x0	—
[12]	extended_tag_field_en_default	RW	1	0x1	Extended Tag Field Enable Default Value. PCIe Specification allows the Extended Tag Field Enable register to reset to either 1 or 0. This register determines the reset value. 0 – 5-bit Tag field enabled on reset 1 – 8-bit Tag field enabled on reset
[11:9]	endpoint_l1_acceptable_latency	RW	3	0x0	Endpoint L1 Acceptable Latency 0 – Maximum of 1 µs. Must be 0 when not an Endpoint. 1 – Maximum of 2 µs 2 – Maximum of 4 µs 3 – Maximum of 8 µs 4 – Maximum of 16 µs 5 – Maximum of 32 µs 6 – Maximum of 64 µs 7 – No limit

Field	Name	Access	Width	Reset	Description
[8:6]	endpoint_l0s_acceptable_latency	RW	3	0x0	Endpoint L0s Acceptable Latency 0 – Maximum of 64 ns. Must be 0 when not an Endpoint. 1 – Maximum of 128 ns 2 – Maximum of 256 ns 3 – Maximum of 512 ns 4 – Maximum of 1 µs 5 – Maximum of 2 µs 6 – Maximum of 4 µs 7 – No limit
[5]	extended_tag_field_supported	RW	1	0x1	Extended Tag Field Supported 0 – 5-bit Tag field supported 1 – 8-bit Tag field supported
[4:3]	phantom_functions_supported	RW	2	0x0	Phantom Functions Supported 0 – No Function Number bits are used for Phantom Functions 1 – The most significant bit of the Function number in Requester ID is used for Phantom Functions 2 – The two most significant bits of Function Number in Requester ID are used for Phantom Functions 3 – All 3 bits of Function Number in Requester ID used for Phantom Functions.
[2:0]	max_payload_size_supported	RW	3	0x2	Max Payload Size Supported 0 – 128 Bytes 1 – 256 Bytes 2 – 512 Bytes 3 – 1024 Bytes 4 – 2048 Bytes 5 – 4096 Bytes 6 – Reserved 7 – Reserved

pcie_link_cap Register 0x88

This register set is used for the PCI Express Link Capabilities configuration.

Table 5.124. pcie_link_cap Register 0x88

Field	Name	Access	Width	Reset	Description
[31:24]	port_number	RW	8	0x0	Indicates the PCI Express Port number for the PCI Express Link.
[23:18]	reserved	RO	6	0x0	—
[17:15]	l1_exit_latency	RW	3	0x7	L1 Exit Latency. The value reported indicates the length of time this Port requires to complete transition from ASPM L1 to L0. 0 – Less than 1 µs 1 – 1 µs to less than 2 µs 2 – 2 µs to less than 4 µs 3 – 4 µs to less than 8 µs 4 – 8 µs to less than 16 µs 5 – 16 µs to less than 32 µs 6 – 32 µs to 64 µs 7 – More than 64 µs

Field	Name	Access	Width	Reset	Description
[14:12]	l0s_exit_latency	RW	3	0x7	L0s Exit Latency. The value reported indicates the length of time this Port requires to complete transition from ASPM L0s to L0. 0 – Less than 64 ns 1 – 64 ns to less than 128 ns 2 – 128 ns to less than 256 ns 3 – 256 ns to less than 512 ns 4 – 512 ns to less than 1 µs 5 – 1 µs to less than 2 µs 6 – 2 µs to 4 µs 7 – More than 4 µs
[11:10]	aspm_support	RW	2	0x3	Active State Power Management (ASPM) Support 0 – No ASPM Support 1 – L0s Supported 2 – L1 Supported 3 – L0s and L1 Supported
[9:0]	reserved	RO	10	0x0	—

pcie_link_stat Register 0x8c

This register set is used for the PCI Express Link Status configuration.

Table 5.125. pcie_link_stat Register 0x8c

Field	Name	Access	Width	Reset	Description
[31:13]	reserved	RO	19	0x0	—
[12]	slot_clock_configuration	RW	1	0x1	Indicates whether the component uses the physical reference clock that the platform provides on the connector. 0 – Using independent reference clock. 1 – Using reference clock provided by slot.
[11:0]	reserved	RO	12	0x0	—

pcie_slot_cap Register 0x90

This register set is used for the PCI Express Slot Capabilities configuration.

Table 5.126. pcie_slot_cap Register 0x90

Field	Name	Access	Width	Reset	Description
[31:19]	physical_slot_number	RW	13	0x1	Indicates whether the physical slot number attached to this Port. This field must be hardware initialized to a value that assigns a slot number that is unique within the chassis, regardless of the form factor associated with the slot. This field must be initialized to zero for Ports connected to devices that are either integrated on the system board or integrated within the same silicon as the Root Port.

Field	Name	Access	Width	Reset	Description
[18]	no_command_completed_support	RW	1	0x0	Indicates whether the slot generates software notification when an issued command is completed by the Hot-Plug Controller. This bit is only permitted to be 1 if the hot-plug capable Port can accept writes to all fields of the Slot Control register without delay between successive writes. 0 – Software notification provided. 1 – Software notification not provided.
[17]	em_interlock_present	RW	1	0x0	Indicates whether an Electromechanical Interlock is implemented on the chassis for this slot. 0 – Not Supported 1 – Supported
[16:15]	slot_power_limit_scale	RW	2	0x0	Slot Power Limit Scale. In combination with the Slot Power Limit Value, specifies the upper limit on power supplied by the slot or by other means to the adapter. Refer PCIe Specification section for details.
[14:7]	slot_power_limit_value	RW	8	0xa	Slot Power Limit Value. In combination with the Slot Power Limit Scale, specifies the upper limit on power supplied by the slot or by other means to the adapter. Refer PCIe Specification section for details.
[6]	hot_plug_capable	RW	1	0x0	Indicates whether this slot can support hot-plug operations. 0 – Not Supported 1 – Supported
[5]	hot_plug_surprise	RW	1	0x0	Indicates whether an adapter present in this slot might be removed from the system without any prior notification. This is a form factor specific capability. This bit is an indication to the operating system to allow for such removal without impacting continued software operation. 0 – Hot Plug Surprise not possible 1 – Hot Plug Surprise possible
[4]	power_indicator_present	RW	1	0x0	Indicates whether a Power Indicator is electrically controlled by the chassis for this slot. 0 – Not Supported 1 – Supported
[3]	attention_indicator_present	RW	1	0x0	Indicates whether an Attention Indicator is electrically controlled by the chassis. 0 – Not Supported 1 – Supported

Field	Name	Access	Width	Reset	Description
[2]	mrl_sensor_present	RW	1	0x0	Indicates whether a MRL Sensor is implemented on the chassis for this slot. 0 – Not Supported 1 – Supported
[1]	power_controller_present	RW	1	0x0	Indicates whether a software programmable Power Controller is implemented for this slot/adapter. 0 – Not Supported 1 – Supported
[0]	attention_button_present	RW	1	0x0	Indicates whether an Attention Button for this slot is electrically controlled by the chassis. 0 – Not Supported 1 – Supported

pcie_dev_cap2 Register 0x98

This register set is used for the PCI Express Device Capabilities 2 configuration.

Table 5.127. pcie_dev_cap2 Register 0x98

Field	Name	Access	Width	Reset	Description
[31:22]	reserved	RO	10	0x0	—
[21]	end_end_prefixes_supported	RW	1	0x0	End-End TLP Prefix Supported 0 – Not Supported 1 – Supported
[20:19]	reserved	RO	2	0x0	—
[18]	obff_supported	RW	1	0x0	OBFF Supported 0 – OBFF Not Supported 1 – OBFF supported using Message signaling only
[17:16]	reserved	RO	2	0x0	—
[15:8]	reserved	RO	8	0x0	—
[7:5]	reserved	RO	3	0x0	—
[4]	cpl_timeout_disable_supported	RW	1	0x1	Completion Timeout Disable Supported. Completion timeout is not implemented by the core, so the advertised value must match the capabilities of the connected design which is implementing completion timeouts. 0 – Not Supported 1 – Supported

Field	Name	Access	Width	Reset	Description
[3:0]	cpl_timeout_ranges_supported	RW	4	0x0	Completion Timeout Ranges Supported advertised value. Completion timeout is not implemented by the core, so the advertised value must match the capabilities of the connected design which is implementing completion timeouts. 0 – Completion Timeout programming not supported. Timeout value in the range 50 µs to 50 ms is used. 1 – Range A (50 µs to 10 ms) 2 – Range B (10 ms to 250 ms) 3 – Range A (50 µs to 10 ms) and B (10 ms to 250 ms) 4 – Range B (10 ms to 250 ms) and C (250 ms to 4 s) 5 – Range A (50 µs to 10 ms) and B (10 ms to 250 ms) and C (250 ms to 4 s) 6 – Range B (10 ms to 250 ms) and C (250 ms to 4 s) and D (4 s to 64 s) 7 – Range A (50 µs to 10 ms) and B (10 ms to 250 ms) and C (250 ms to 4 s) and D (4 s to 64 s) 8 – Reserved 9 – Reserved 10 – Reserved 11 – Reserved 12 – Reserved 13 – Reserved 14 – Reserved 15 – Reserved

pcie_link_ctl2 Register 0xa0

This register set is used for the PCI Express Link Control 2 configuration.

Table 5.128. pcie_link_ctl2 Register 0xa0

Field	Name	Access	Width	Reset	Description
[31:0]	reserved	RO	32	0x0	—

5.1.4.15. Power Management configuration

pm_cap Register 0xc0

This register set is used for the Power Management Capabilities configuration.

Table 5.129. pm_cap Register 0xc0

Field	Name	Access	Width	Reset	Description
[31:16]	reserved	RO	16	0x0	—
[15:11]	pme_support	RW	5	0x1f	PME Support. Indicates the power states from which the function may generate a PME. For each power state {D3Cold, D3hot, D2, D1, D0}: 0 – PME# not supported 1 – PME# supported
[10]	d2_support	RW	1	0x1	D2 Power Management State support. 0 – Not supported 1 – Supported
[9]	d1_support	RW	1	0x1	D1 Power Management State support. 0 – Not supported 1 – Supported
[8:6]	aux_current	RW	3	0x0	Aux Current. Reports the 3.3Vaux auxiliary current requirements for the PCI function. See PCIe Specification for details. 0 – Self-powered 1 – 55 mA 2 – 100 mA 3 – 160 mA 4 – 220 mA 5 – 270 mA 6 – 320 mA 7 – 375 mA
[5]	dsi	RW	1	0x0	Device Specific Initialization. Indicates whether special initialization of this function is required (beyond the standard PCI configuration header) before the generic class device driver can use it. 0 – No Device Specific Initialization necessary. 1 – Function requires a device specific initialization sequence following transition to the D0 uninitialized state.
[4]	reserved	RO	1	0x0	—
[3]	pme_clock	RW	1	0x0	PME Clock. Does not apply to PCI Express and must be 0.
[2:0]	version	RW	3	0x3	PCI Power Management Interface Specification Version. Must be set to 0x3 to indicate revision 1.2 of the PCI Power Management Interface Specification.

pm Register 0xc4

This register set is used for the Power Management Control/Status configuration.

Table 5.130. pm Register 0xc4

Field	Name	Access	Width	Reset	Description
[31:24]	data	RW	8	0x0	—
[23]	pmcsr_bus_p_c_en	RW	1	0x0	—
[22]	pmcsr_b2_b3_support	RW	1	0x0	—
[21:16]	reserved	RO	6	0x0	—
[15]	reserved	RO	1	0x0	—
[14:13]	cstat_data_scale	RW	2	0x0	0 – Unknown scale 1 – power = data * 0.1 Watts 2 – power = data * 0.01 Watts 3 – power = data * 0.001 Watts
[12:9]	cstat_data_select	RW	4	0x0	0 – D0 Power Consumed 1 – D1 Power Consumed 2 – D2 Power Consumed 3 – D3 Power Consumed 4 – D0 Power Dissipated 5 – D1 Power Dissipated 6 – D2 Power Dissipated 7 – D3 Power Dissipated 8 – Common logic power consumption. For multifunction devices, reported in Function 0 only. 9 – Reserved 10 – Reserved 11 – Reserved 12 – Reserved 13 – Reserved 14 – Reserved 15 – Reserved
[8:0]	reserved	RO	9	0x0	—

pm_aux Register 0xc8

This register set is used for the Power Management Auxiliary Power configuration.

Table 5.131. pm_aux Register 0xc8

Field	Name	Access	Width	Reset	Description
[31:1]	reserved	RO	31	0x0	—

[0]	power_required	RW	1	0x0	• Identifies whether the design requires auxiliary power.
					  • 0 – Aux Power is not required.
					  • 1 – Aux Power is required. If Aux Power is required, PME is advertised supported from D3 Cold, or advertised.
					• aux_current != 0, then the value of Aux Power PM Enable is sticky and preserved through conventional reset when
					• Aux Power is provided.

5.1.4.16. ARI Capability Configuration

ari_cap Register 0xe0

This register set is used for the ARI Capability configuration.

Table 5.132. ari_cap Register 0xe0

Field	Name	Access	Width	Reset	Description
[31:1]	reserved	RO	31	0x0	—
[0]	disable	RW	1	0x0	ARI Capability Disable When disabled, the ARI Capability does not appear in PCIe Configuration Space. This must be enabled when SR-IOV is enabled and must be disabled for downstream ports, Root Complex Integrated Endpoints, and Root Complex Event Collectors.

aer_cap Register 0x100

This register set is used for the AER Capability configuration.

Table 5.133. aer_cap Register 0x100

Field	Name	Access	Width	Reset	Description
[31]	en_tlp_prefix_blocked	RW	1	0x0	Enable TLP Prefix Blocked error reporting. 0 – Disable 1 – Enable
[30]	en_atomicop_egress_blocked	RW	1	0x0	Enable AtomicOp Egress Blocked error reporting. 0 – Disable 1 – Enable
[29]	en_mc_blocked_tlp	RW	1	0x0	Enable MC Blocked TLP error reporting. Not supported by core, so must be 0. 0 – Disable 1 – Enable
[28]	en_ucorr_internal_error	RW	1	0x0	Enable Uncorrectable Internal Error. 0 – Disable 1 – Enable
[27]	en_acs_violation	RW	1	0x0	Enable ACS Violation error reporting. Not supported by core, so must be 0. 0 – Disable 1 – Enable
[26]	en_receiver_overflow	RW	1	0x0	Enable Receiver Overflow error reporting. Not supported by core, so must be 0. 0 – Disable 1 – Enable
[25]	en_completer_abort	RW	1	0x0	Enable Completer Abort error reporting. 0 – Disable 1 – Enable
[24]	en_completion_timeout	RW	1	0x1	Enable Completion Timeout error reporting. 0 – Disable 1 – Enable
[23]	en_surprise_down_error	RW	1	0x0	Enable Surprise Down Error reporting. 0 – Disable 1 – Enable
[22]	en_corr_internal_error	RW	1	0x0	Enable Correctable Internal Error reporting. 0 – Disable 1 – Enable
[21:16]	reserved	RO	6	0x0	—
[15:2]	reserved	RO	14	0x0	—

Field	Name	Access	Width	Reset	Description
[1]	ecrc_gen_chk_capable	RW	1	0x1	ECRC Generation/Checking Capable. 0 – Not supported 1 – Supported
[0]	version	RW	1	0x0	AER Capability Version. 0 – Version 0x1 1 – Version 0x2

5.1.4.17. MSI Capability Configuration

msi_cap Register 0xe8

This register set is used for the MSI Capability configuration.

Table 5.134. msi_cap Register 0xe8

Field	Name	Access	Width	Reset	Description
[31:8]	reserved	RO	24	0x0	—
[7]	reserved	RO	1	0x0	—
[6:4]	mult_message_capable	RW	3	0x5	Number of requested MSI vectors. 0 – 1 1 – 2 2 – 4 3 – 8 4 – 16 5 – 32 6 – Reserved 7 – Reserved
[3:2]	reserved	RO	2	0x0	—
[1]	vec_mask_capable	RW	1	0x1	MSI Capability Per Vector Mask Capable. 0 – Disable 1 – Enable
[0]	disable	RW	1	0x0	MSI Capability Disable. When disabled, the MSI Capability does not appear in PCIe Configuration Space. 0 – Enable 1 – Disable

msix_cap Register 0xf0

This register set is used for the MSI-X Capability configuration.

Table 5.135. msix_cap Register 0xf0

Field	Name	Access	Width	Reset	Description
[31:27]	reserved	RO	5	0x0	—
[26:16]	table_size	RW	11	0x1f	Number of requested MSI-X vectors == (table_size+1).
[15:1]	reserved	RO	15	0x0	—
[0]	disable	RW	1	0x0	MSI-X Capability Disable. When disabled, the MSI-X Capability does not appear in PCIe Configuration Space. 0 – Enable 1 – Disable

msix_table Register 0xf4

This register set is used for the MSI-X Capability – MSI-X Table configuration.

Table 5.136. msix_table Register 0xf4

Field	Name	Access	Width	Reset	Description
[31:3]	offset	RW	29	0xc00	{offset, 3'b000} == byte address offset, within the BAR selected by bir, at which the MSI-X Table begins.
[2:0]	bir	RW	3	0x0	Indicates which Base Address register, located beginning at 10h in Configuration Space, is used to map the MSI-X Table into Memory Space. 0 – 0x10 (BAR0) 1 – 0x14 (BAR1) 2 – 0x18 (BAR2) 3 – 0x1C (BAR3) 4 – 0x20 (BAR4) 5 – 0x24 (BAR5) 6 – Reserved 7 – Reserved

msix_pba Register 0xf8

This register set is used for the MSI-X Capability – MSI-X PBA configuration.

Table 5.137. msix_pba Register 0xf8

Field	Name	Access	Width	Reset	Description
[31:3]	offset	RW	29	0xe00	{offset, 3'b000} == byte address offset, within the BAR selected by bir, at which the MSI-X PBA begins
[2:0]	bir	RW	3	0x0	Indicates which Base Address register, located beginning at 10h in Configuration Space, is used to map the MSI-X PBA into Memory Space. 0 – 0x10 (BAR0) 1 – 0x14 (BAR1) 2 – 0x18 (BAR2) 3 – 0x1C (BAR3) 4 – 0x20 (BAR4) 5 – 0x24 (BAR5) 6 – Reserved 7 – Reserved

5.1.4.18. Vendor-Specific Capability Configuration

vsec_cap Register 0x110

This register set is used for the Vendor-Specific Capability configuration.

Table 5.138. vsec_cap Register 0x110

Field	Name	Access	Width	Reset	Description
[31:16]	id	RW	16	0x1	Vendor-Specific Capability ID.
[15:1]	reserved	RO	15	0x0	—
[0]	enable	RW	1	0x1	Vendor-Specific Capability Enable. When disabled, the VSEC Capability does not appear in PCIe Configuration Space. 0 – Disable 1 – Enable

5.1.4.19. SRIS Capability Configuration

sris_cap Register 0x120

This register set is used for the SRIS Capability configuration.

Table 5.139. sris_cap Register 0x120

Field	Name	Access	Width	Reset	Description
[31:16]	reserved	RO	16	0x0	—
[15:12]	low_skp_generation_speeds	RW	4	0x0	SRIS Lower SKP OS Generation Supported Speeds Vector advertisement
[11:8]	low_skp_reception_speeds	RW	4	0x0	SRIS Lower SKP OS Reception Supported Speeds Vector advertisement
[7:1]	reserved	RO	7	0x0	—
[0]	enable	RW	1	0x0	SRIS Capability Enable. When disabled, the SRIS Capability does not appear in PCIe Configuration Space. 0 – Disable 1 – Enable

5.1.4.20. Device Serial Number

dsn_cap Register 0x130

This register set is used for the DSN capable cores only such as Device Serial Number Capability configuration.

Table 5.140. dsn_cap Register 0x130

Field	Name	Access	Width	Reset	Description
[31:1]	reserved	RO	31	0x0	—
[0]	enable	RW	1	0x0	Device Serial Number Capability Enable. When disabled, the Device Serial Number Capability does not appear in PCIe Configuration Space. 0 – Disable 1 – Enable

dsn_serial Register 0x134

This register set is used for the Device Serial Number Capability – Serial Number.

Table 5.141. dsn_serial Register 0x134

Field	Name	Access	Width	Reset	Description
[63:0]	number	RW	64	0x0	Device Serial Number.

5.1.4.21. Power Budgeting Capability Configuration

pwr_budget_cap Register 0x150

This register set is used for the Power Budgeting Capability configuration.

Table 5.142. pwr_budget_cap Register 0x150

Field	Name	Access	Width	Reset	Description
[31:2]	reserved	RO	30	0x0	—
[1]	sys_alloc	RW	1	0x0	Power Budgeting System Allocated. 0 – Power Budget should use Power Budgeting Capability Values 1 – Power Budget is System Allocated
[0]	enable	RW	1	0x0	Power Budgeting Capability Enable. When disabled, the Power Budgeting Capability does not appear in PCIe. Configuration Space. 0 – Disable 1 – Enable

5.1.4.22. Dynamic Power Allocation Configuration

dpa_cap Register 0x158

This register set is used for the Dynamic Power Allocation Capability configuration.

Table 5.143. dpa_cap Register 0x158

Field	Name	Access	Width	Reset	Description
[31:24]	xlcy1	RW	8	0x0	Transition Latency Value 1. When the Transition Latency Indicator for a substate is 1, this value is multiplied by the Transition Latency Unit to determine the maximum Transition Latency for the substate.
[23:16]	xlcy0	RW	8	0x0	Transition Latency Value 0. When the Transition Latency Indicator for a substate is 0, this value is multiplied by the Transition Latency Unit to determine the maximum Transition Latency for the substate.
[15:14]	reserved	RO	2	0x0	—
[13:12]	pas	RW	2	0x0	Power Allocation Scale. The value of the substate Power Allocation Register is multiplied by the decoded value of this field to determine the power allocation of the substate. 0 – 10x 1 – 1x 2 – 0.1x 3 – 0.01x
[11:10]	reserved	RO	2	0x0	—

Field	Name	Access	Width	Reset	Description
[9:8]	tlunit	RW	2	0x0	Transition Latency Unit. The substate Transition Latency Value is multiplied by the decoded Transition Latency Unit to Determine the maximum Transition Latency for the substate. 0 – 1 ms 1 – 10 ms 2 – 100 ms 3 – Reserved
[7:3]	substate_max	RW	5	0x0	Substate_Max. Specifies the maximum substate number. Substates from [substate_max:0] are supported. For example, substate_max==0 indicates support for 1 substate.
[2:1]	reserved	RO	2	0x0	—
[0]	enable	RW	1	0x0	Dynamic Power Allocation (DPA) Capability Enable. When disabled, the Dynamic Power Allocation Capability does not appear in PCIe Configuration Space. 0 – Disable 1 – Enable

dpa_xlcy Register 0x15c

This register set is used for the Dynamic Power Allocation – Transition Latency.

Table 5.144. dpa_xlcy Register 0x15c

Field	Name	Access	Width	Reset	Description
[31:0]	indicator	RW	32	0x0	Transition Latency Indicator. Indicates which Transition Latency Value applies to each substate. For each substate[i], indicator[i] indicates which Transition Latency Value applies: 0 – Use Transition Latency Value 0 1 – Use Transition Latency Value 1

dpa_alloc Register 0x160

This register set is used for the Dynamic Power Allocation Capability – Dynamic Power Allocation Array.

Table 5.145. dpa_alloc Register 0x160

Field	Name	Access	Width	Reset	Description
[255:0]	array	RW	256	0x0	Substate Power Allocation Array. For each substate[i], multiply array[(i*8)+7i*8)] times the Power Allocation Scale to determine the power allocation in Watts for the associated substate.

5.1.4.23. Latency Tolerance Reporting Capability Configuration

ltr_cap Register 0x180

This register set is used for the Latency Tolerance Reporting Capability configuration.

Table 5.146. ltr_cap Register 0x180

Field	Name	Access	Width	Reset	Description
[31:1]	reserved	RO	31	0x0	—
[0]	enable	RW	1	0x0	Latency Tolerance Reporting Capability Enable. When disabled, the Latency Tolerance Reporting Capability does not appear in PCIe Configuration Space. 0 – Disable 1 – Enable

5.1.4.24. L1 PM Substates Capability Configuration

l1pmss_cap Register 0x188

This register set is used for the L1 PM Substates Capability configuration.

Table 5.147. l1pmss_cap Register 0x188

Field	Name	Access	Width	Reset	Description
[31:24]	cm_restore_time	RW	8	0x0	Default Common Mode Restore Time. Default time, in microseconds, used by the Downstream Port for timing the re establishment of common mode. See the L1 PM Substates ECN for further details.
[23:16]	port_cm_restore_time	RW	8	0x0	Port Common Mode Restore Time. Time, in microseconds, required for this port to re-establish common mode. See the L1 PM Substates ECN for further details
[15:11]	port_tpower_on_value	RW	5	0x0	Port TPOWER_ON Value. Required for ports supporting PCI-PM L1.2 or ASPM L1.2. The value of TPOWER_ON is calculated by multiplying the value in this field by the decoded TPOWER_ON Scale field.
[10]	reserved	RO	1	0x0	—
[9:8]	port_tpower_on_scale	RW	2	0x0	Port TPOWER_ON Scale. Required for ports supporting PCI-PM L1.2 or ASPM L1.2. 0 – 2 µs 1 – 10 µs 2 – 100 µs 3 – Reserved
[7]	pcipm_l1_1_supported	RW	1	0x1	PCI-PM L1.1 Substate Supported. Must be set to 1 for all ports supporting L1 PM Substates. 0 – Not supported 1 – Supported
[6]	pcipm_l1_2_supported	RW	1	0x1	PCI-PM L1.2 Substate Supported. 0 – Not supported 1 – Supported
[5]	aspm_l1_1_supported	RW	1	0x1	ASPM L1.1 Substate Supported. 0 – Not supported 1 – Supported
[4]	aspm_l1_2_supported	RW	1	0x1	ASPM L1.2 Substate Supported. 0 – Not supported 1 – Supported

Field	Name	Access	Width	Reset	Description
[3]	l1pm_supported	RW	1	0x1	L1 PM Substates Supported. 0 – Not supported 1 – Supported
[2:1]	reserved	RO	2	0x0	—
[0]	enable	RW	1	0x0	L1 PM Substates Capability Enable. When disabled, the L1 PM Substates Capability does not appear in PCIe. Configuration Space. 0 – Disable 1 – Enable

5.1.4.25. Resizable BAR Capability Configuration

rbar_cap Register 0x1a0

This register set is used for the Resizable BAR Capability configuration.

Table 5.148. rbar_cap Register 0x1a0

Field	Name	Access	Width	Reset	Description
[31:1]	reserved	RO	31	0x0	—
[0]	enable	RW	1	0x0	Resizable BAR Capability Enable. When disabled, the Resizable BAR Capability does not appear in PCIe Configuration. Space. 0 – Disable 1 – Enable

rbar_cfg0 Register 0x1a4

This register set is used for the Resizable BAR Capability – BAR Configuration 0.

Table 5.149. rbar_cfg0 Register 0x1a4

Field	Name	Access	Width	Reset	Description
[31:29]	reserved	RO	3	0x0	—
[28:24]	size	RW	5	0x0	Default BAR Size. Indicates the default size after reset for this BAR. BAR Size == 2 size+20 bytes. For example, if this field is set to a value of 3, that indicates this BAR has a default size of 2^23=8 MB. The max value is 19 (2^39==512 GB). The default value must be one of the supported BAR sizes indicated by supported_sizes
[23:4]	supported_sizes	RW	20	0xf	Supported BAR Sizes. supported_sizes[i] indicates a BAR Size of 2i+20 is supported for this BAR. For example, if supported_sizes[0] is set, then a BAR size of 2^20=2 MB is supported.
[3]	reserved	RO	1	0x0	—

Field	Name	Access	Width	Reset	Description
[2:0]	bar_index	RW	3	0x0	BAR Index. BAR offset for which this configuration is valid. For a 64-bit BAR, this index must indicate the lower DWORD used for the BAR. 0 – BAR located at Configuration Register address offset 0x10. 1 – BAR located at Configuration Register address offset 0x14. 2 – BAR located at Configuration Register address offset 0x18. 3 – BAR located at Configuration Register address offset 0x1C. 4 – BAR located at Configuration Register address offset 0x20. 5 – BAR located at Configuration Register address offset 0x24. 6 – Reserved 7 – Reserved

rbar_cfg1 Register 0x1a8

This register set is used for the Resizable BAR Capability – BAR Configuration 1.

Table 5.150. rbar_cfg1 Register 0x1a8

Field	Name	Access	Width	Reset	Description
[31:29]	reserved	RO	3	0x0	—
[28:24]	size	RW	5	0x0	Default BAR Size. Indicates the default size after reset for this BAR. BAR Size == 2 size+20 bytes. For example, if this field is set to a value of 3, that indicates this BAR has a default size of 2^23=8 MB. The max value is 19 (2^39==512 GB). The default value must be one of the supported BAR sizes indicated by supported_sizes.
[23:4]	supported_sizes	RW	20	0x0	Supported BAR Sizes. supported_sizes[i] indicates a BAR Size of 2i+20 is supported for this BAR. For example. If supported_sizes[0] is set, a BAR size of 2^20= 2 MB is supported.
[3]	reserved	RO	1	0x0	—

Field	Name	Access	Width	Reset	Description
[2:0]	bar_index	RW	3	0x1	BAR Index. BAR offset for which this configuration is valid. For a 64-bit BAR, this index must indicate the lower DWORD used for the BAR. 0 – BAR located at Configuration Register address offset 0x10. 1 – BAR located at Configuration Register address offset 0x14. 2 – BAR located at Configuration Register address offset 0x18. 3 – BAR located at Configuration Register address offset 0x1C. 4 – BAR located at Configuration Register address offset 0x20. 5 – BAR located at Configuration Register address offset 0x24. 6 – Reserved 7 – Reserved

rbar_cfg2 Register 0x1ac

This register set is used for the Resizable BAR Capability – BAR Configuration 2.

Table 5.151. rbar_cfg2 Register 0x1ac

Field	Name	Access	Width	Reset	Description
[31:29]	*reserved*	RO	3	0x0	—
[28:24]	size	RW	5	0x0	Default BAR Size. Indicates the default size after reset for this BAR. BAR Size == 2 size+20 bytes. For example, if this field is set to a value of 3, that indicates this BAR has a default size of 2^23=8 MB. The max value is 19 (2^39==512 GB). The default value must be one of the supported BAR sizes indicated by supported_sizes.
[23:4]	supported_sizes	RW	20	0x0	Supported BAR Sizes. supported_sizes[i] indicates a BAR Size of 2i+20 is supported for this BAR. For example, if supported_sizes[0] is set, then a BAR size of 2^20=2 MB is supported.
[3]	*reserved*	RO	1	0x0	—
[2:0]	bar_index	RW	3	0x2	BAR Index. BAR offset for which this configuration is valid. For a 64-bit BAR, this index must indicate the lower DWORD used for the BAR. 0 – BAR located at Configuration Register address offset 0x10. 1 – BAR located at Configuration Register address offset 0x14. 2 – BAR located at Configuration Register address offset 0x18. 3 – BAR located at Configuration Register address offset 0x1C. 4 – BAR located at Configuration Register address offset 0x20. 5 – BAR located at Configuration Register address offset 0x24. 6 – Reserved 7 – Reserved

rbar_cfg3 Register 0x1b0

This register set is used for the Resizable BAR Capability – BAR Configuration 3.

Table 5.152. rbar_cfg3 Register 0x1b0

Field	Name	Access	Width	Reset	Description
[31:29]	reserved	RO	3	0x0	—
[28:24]	size	RW	5	0x0	Default BAR Size. Indicates the default size after reset for this BAR. BAR Size == 2 size+20 bytes. For example, if this field is set to a value of 3, that indicates this BAR has a default size of 2^23= 8MB. The max value is 19 (2^39==512 GB). The default value must be one of the supported BAR sizes indicated by supported_sizes.
[23:4]	supported_sizes	RW	20	0x0	Supported BAR Sizes. supported_sizes[i] indicates a BAR Size of 2i+20 is supported for this BAR. For example. If supported_sizes[0] is set, then a BAR size of 2^20=2 MB is supported.
[3]	reserved	RO	1	0x0	—
[2:0]	bar_index	RW	3	0x3	BAR Index. BAR offset for which this configuration is valid. For a 64-bit BAR, this index must indicate the lower DWORD used for the BAR. 0 – BAR located at Configuration Register address offset 0x10. 1 – BAR located at Configuration Register address offset 0x14. 2 – BAR located at Configuration Register address offset 0x18. 3 – BAR located at Configuration Register address offset 0x1C. 4 – BAR located at Configuration Register address offset 0x20. 5 – BAR located at Configuration Register address offset 0x24. 6 – Reserved 7 – Reserved

rbar_cfg4 Register 0x1b4

This register set is used for the Resizable BAR Capability – BAR Configuration 4.

Table 5.153. rbar_cfg4 Register 0x1b4

Field	Name	Access	Width	Reset	Description
[31:29]	reserved	RO	3	0x0	—
[28:24]	size	RW	5	0x0	Default BAR Size. Indicates the default size after reset for this BAR. BAR Size == 2 size+20 bytes. For example, if this field is set to a value of 3, that indicates this BAR has a default size of 2^23=8 MB. The max value is 19 (2^39==512 GB). The default value must be one of the supported BAR sizes indicated by supported_sizes.
[23:4]	supported_sizes	RW	20	0x0	Supported BAR Sizes. supported_sizes[i] indicates a BAR Size of 2i+20 is supported for this BAR. For example. If supported_sizes[0] is set, then a BAR size of 2^20=2 MB is supported.
[3]	reserved	RO	1	0x0	—
[2:0]	bar_index	RW	3	0x4	BAR Index BAR offset for which this configuration is valid. For a 64-bit BAR, this index must indicate the lower DWORD used for the BAR. 0 – BAR located at Configuration Register address offset 0x10. 1 – BAR located at Configuration Register address offset 0x14. 2 – BAR located at Configuration Register address offset 0x18. 3 – BAR located at Configuration Register address offset 0x1C. 4 – BAR located at Configuration Register address offset 0x20. 5 – BAR located at Configuration Register address offset 0x24. 6 – Reserved 7 – Reserved

rbar_cfg5 Register 0x1b8

This register set is used for the Resizable BAR Capability – BAR Configuration 5.

Table 5.154. rbar_cfg5 Register 0x1b8

Field	Name	Access	Width	Reset	Description
[31:29]	reserved	RO	3	0x0	—
[28:24]	size	RW	5	0x0	Default BAR Size. Indicates the default size after reset for this BAR. BAR Size == 2 size+20 bytes. For example, if this field is set to a value of 3, that indicates this BAR has a default size of 2^23=8 MB. The max value is 19 (2^39==512 GB). The default value must be one of the supported BAR sizes indicated by supported_sizes.
[23:4]	supported_sizes	RW	20	0x0	Supported BAR Sizes. supported_sizes[i] indicates a BAR Size of 2i+20 is supported for this BAR. For example. If supported_sizes[0] is set, then a BAR size of 2^20=2 MB is supported.
[3]	reserved	RO	1	0x0	—

Field	Name	Access	Width	Reset	Description
[2:0]	bar_index	RW	3	0x5	BAR Index. BAR offset for which this configuration is valid. For a 64-bit BAR, this index must indicate the lower DWORD used for the BAR. 0 – BAR located at Configuration Register address offset 0x10. 1 – BAR located at Configuration Register address offset 0x14. 2 – BAR located at Configuration Register address offset 0x18. 3 – BAR located at Configuration Register address offset 0x1C. 4 – BAR located at Configuration Register address offset 0x20. 5 – BAR located at Configuration Register address offset 0x24. 6 – Reserved 7 – Reserved

5.1.4.26. ATS Capability Configuration

ats_cap Register 0x1c0

This register set is used for the ATS capable cores only such as ATS Capability configuration.

Table 5.155. ats_cap Register 0x1c0

Field	Name	Access	Width	Reset	Description
[31:17]	reserved	RO	15	0x0	—
[16]	global_inval_support	RW	1	0x1	Cores with both ATS and PASID support only: ATS/PASID Global Invalidate Support. If set to 1, the function supports Invalidate Requests with the Global Invalidate bit set.
[15:13]	reserved	RO	3	0x0	—
[12:8]	inval_q_depth	RW	5	0x0	ATS Invalidate Queue Depth. Number of invalidate requests that can be queued. 0 is a special case that indicates a queue depth of 32.
[7:1]	reserved	RO	7	0x0	—
[0]	enable	RW	1	0x0	ATS Capability Enable. When disabled, the ATS Capability does not appear in PCIe Configuration Space. 0 – Disable 1 – Enable

5.1.4.27. Atomic Op Capability Configuration

atomic_op_cap Register 0x1cc

This register set is used for the Atomic Op Capability configuration.

Table 5.156. atomic_op_cap Register 0x1cc

Field	Name	Access	Width	Reset	Description
[31:6]	reserved	RO	26	0x0	—
[5]	rp_completer_enable	RW	1	0x0	Enable Root Port to be an Atomic Op Completer which means that the Root Port completes rather than forwards Atomic Op TLPs. 0 – Disable 1 – Enable
[4]	completer_128_supported	RW	1	0x0	Atomic Op Completer 128-bit Operand Support. 0 – Not Supported 1 – Supported

Field	Name	Access	Width	Reset	Description
[3]	completer_64_supported	RW	1	0x0	Atomic Op Completer 64-bit Operand Support. 0 – Not Supported 1 – Supported
[2]	completer_32_supported	RW	1	0x0	Atomic Op Completer 32-bit Operand Support. 0 – Not Supported 1 – Supported
[1]	routing_supported	RW	1	0x0	Atomic Op Routing Supported. 0 – Not Supported 1 – Supported
[0]	enable	RW	1	0x0	Atomic Op Capability Enable. When disabled, the Atomic Op Capability does not appear in PCIe Configuration Space. 0 – Disable 1 – Enable

5.1.5. mgmt_ftl_mf[3:1] (0x05000,0x06000,0x07000)

The base address for multifunction mgmt_ftl_mf is shown in Table 5.157.

Table 5.157. Base Address for mgmt_ftl_mf

Port	Base Address
mgmt_ftl_mf1_BASE	0x5000 (Function 1)
mgmt_ftl_mf2_BASE	0x6000 (Function 2)
mgmt_ftl_mf3_BASE	0x7000 (Function 3)

5.1.5.1. Function Register 0x08

This register set is used for the Function disable for Functions[3:1]. Function[0] may not be disabled.

Table 5.158. Function Register 0x08

Field	Name	Access	Width	Reset	Description
[31:1]	reserved	RO	31	0x0	—
[0]	disable	RW	1	0x0	Function disable for Functions[3:1]. Function[0] may not be disabled. 0 – Enable 1 – Disable

5.1.5.2. us_port Register 0x38

This register set is used for the Upstream Port Configuration.

Table 5.159. us_port Register 0x38

Field	Name	Access	Width	Reset	Description
[31:1]	reserved	RO	31	0x0	—
[0]	adv_target_link_speed	RW	1	0x0	For an upstream port, advertise the link speeds specified by the target_link_speed field rather than the maximum supported speed.

5.1.6. pcie_ll(0x0F000)

5.1.6.1. Main Control Register

main_ctrl_0 Register 0x0

This register set is used for the Main Control 0 register.

Table 5.160. main_ctrl_0 Register 0x0

Field	Name	Access	Width	Reset	Description
[31]	en_user_write	RW	1	0x1	This option allows you to modify the values of this register (excluding this field). By default, you have write and read access to this register. 0 – Read only access 1 – Read/Write access
[30:17]	reserved	RO	14	0x0	—
[16]	disable_csr_reset_port	RW	1	0x0	Disables the reset of configuration and status registers (CSR) through reset port. 0 – Asserting the usr_lmmi_resetn_i resets the CSRs 1 – Disable reset port. (You can still use soft reset by writing to the reset register pcie_ll_main_ctrl_2[0]).
[15:6]	reserved	RO	10	0x1	—
[5]	sel_pclk_div2	RW	1	0x1	This field selects the clock output on port link[LINK]_clk_usr_o. 0 – pclk (250 MHz) 1 – pclk_div2 (125 MHz)
[4:2]	num_lanes	RW	3	0x1	This field indicates the maximum number of lanes that is used when PCIe LL core is enabled. 1 – 1 Lane
[1]	reserved	RO	1	0x0	—
[0]	core_enable	RW	1	0x1	Enable or disable the PCIe Link Layer Core. 0 – Disable 1 – Enable

main_ctrl_1 Register 0x4

This register set is used for the Main Control 1 register.

Table 5.161. main_ctrl_1 Register 0x4

Field	Name	Access	Width	Reset	Description
[31:17]	reserved	RO	15	0x0	—
[16]	hold_reset	RW	1	0x0	Controls the core_reset and pipe_reset if it remains asserted or automatically deasserts. 0 – (not supported on this version) – writing 1 to core_reset/pipe_reset field toggles the PCIe Link Layer core reset or PIPE reset for 1 clock cycle 1 – Hold the core_reset/pipe_reset (core_reset/pip_reset does not automatically deasserts unless 0 is written to the corresponding field).
[15:9]	reserved	RO	7	0x0	—
[8]	pipe_reset	RW	1	0x0	This field controls the PIPE reset (PCS reset). The behaviour of pipe_reset depends on the hold_reset field. 0 – Deassert PIPE Reset (Normal operation) 1 – Assert PIPE Reset
[7:1]	reserved	RO	7	0x0	—

Field	Name	Access	Width	Reset	Description
[0]	core_reset	RW	1	0x0	This field controls the PCIe Link Layer core reset. The behaviour of core_reset depends on the hold_reset field. 0 – Deassert Core Reset (Normal operation) 1 – Assert Core Reset

main_ctrl_2 Register 0x8

This register set is used for the Main Control 2 register.

Table 5.162. main_ctrl_2 Register 0x8

Field	Name	Access	Width	Reset	Description
[31:2]	reserved	RO	30	0x0	—
[1]	ll_csr_reset	RW	1	0x0	This field controls the reset of PCIe Link Layer mgmt_* configuration and status registers. Automatically returns to 0 after a write of 1. 0 – reserved 1 – Assert Link Layer CSR Reset, writing 1 to ll_csr_reset field toggles the Link Layer CSR reset for 1 clock cycle.
[0]	phy_csr_reset	RW	1	0x0	This field controls the reset of PHY configuration and status registers. Automatically returns to 0 after a write of 1. 0 – reserved 1 – Assert PHY CSR Reset, writing 1 to phy_csr_reset field toggles the PHY CSR reset for 1 clock cycle.

main_ctrl_3 Register 0xC

This register set is used for the Main Control 3 register.

Table 5.163. main_ctrl_3 Register 0xC

Field	Name	Access	Width	Reset	Description
[31:16]	u_clk_period_in_ps	RW	16	0x1F40	The current period of clk_usr in picoseconds. This is used for time events with fixed time duration such as LTSSM state machine timeouts. Default is 8000 ps (125 MHz).
[15:0]	p_clk_period_in_ps	RW	16	0xFA0	The current period of pclk in picoseconds. This is used for time events with fixed time duration such as LTSSM state machine timeouts. Default is 4000 ps (250 MHz).

main_ctrl_4 Register 0x10

This register set is used for the Main Control 4 register.

Table 5.164. main_ctrl_4 Register 0x10

Field	Name	Access	Width	Reset	Description
[31:16]	aux_clk_period_in_ps	RW	16	0xF424	The current period of phy aux_clk in picoseconds. This is used for time events with fixed time duration such as LTSSM state machine timeouts. Default is 62500 ps (16 MHz).
[15:3]	reserved	RO	13	0x0	—

Field	Name	Access	Width	Reset	Description
[2]	merge_cfgreg_lmmi_rdata	RW	1	0x0	This option is provided to allow the reduction of ports and merge the PCIe Configuration Register read data port (ucfg_rd_data_o) with the LMMI read data (usr_lmmi_rdata_o) port. When enabled, it is expected that you do not issue a simultaneous read access on CSR and PCIe Configuration Registers 0 – Disable 1 – Enable
[1]	en_port_mgmt_interrupt_leg	RW	1	0x1	Enables the input port mgmt_interrupt_leg, otherwise use register access. 0 – Disable 1 – Enable
[0]	en_port_mgmt_ltssm_disable	RW	1	0x0	Enables the input port link[LINK]_ltssm_disable_i, otherwise use register access (see register pcie_ll_conv_port_0). 0 – Disable 1 – Enable

main_ctrl_5 Register 0x14

This register set is used for the Main Control 5 register.

Table 5.165. main_ctrl_4 Register 0x10

Field	Name	Access	Width	Reset	Description
[31:1]	reserved	RO	31	0x0	—
[0]	en_pipe_if_ctrl	RW	1	0x0	When enabled, allows you to control the following pipe interface signals: pipe_pclkreq_n, pipe_rx_ei_disable, pipe_tx_cm_disable, pipe_power_down. This should not be enabled during normal operation. 0 – Disable 1 – Enable

5.1.6.2. Converted Port Register Set

conv_port_0 Register 0x100

This register set is used for the Converted Port 0 register.

Table 5.166. conv_port_0 Register 0x100

Field	Name	Access	Width	Reset	Description
[31:1]	reserved	RO	31	0x0	—
[0]	mgmt_ltssm_disable	RW	1	0x0	(refer to register pcie_ll_main_ctrl_4) The LTSSM does not transition from Detect.Quiet to Detect. Active to begin LTSSM training while mgmt_ltssm_disable ==1. mgmt_ltssm_disable may be used to delay the start of LTSSM training which otherwise begins as soon as rst_usr_n is deasserted. mgmt_ltssm_disable must be set to 1 relatively soon (within a few ms) after rst_usr_n is released as the system allocates a finite amount of time for devices to initialize before it begins to scan for devices. If mgmt_ltssm_disable is held for too long, software may scan for the device before it becomes operational and assume that no device is present.

conv_port_1 Register 0x104

This register set is used for the Converted Port 1 register.

Table 5.167. conv_port_1 Register 0x104

Field	Name	Access	Width	Reset	Description
[31:4]	reserved	RO	28	0x0	—
[3:0]	mgmt_interrupt_leg	RW	4	0x0	(refer to register pcie_ll_main_ctrl_4) When Legacy Interrupt Mode is enabled, mgmt_interrupt_leg implements one level-sensitive interrupt (INTA, INTB, INTC, or INTD) for each Base Function. Each functions' interrupt sources must be logically ORed together and input as mgmt_interrupt_leg[i] for a given function. Each interrupt source must continue to drive a 1 until it has been serviced and cleared by software at which time it must switch to driving 0. The core ORs together INTA/B/C/D from all functions to create an aggregated INTA/INTB/INTC/INTD. The core monitors high and low transitions on the aggregated INTA/B/C/D and sends an Interrupt Assert message on each 0 to 1 transition and an Interrupt De-Assert Message on each 1 to 0 transition of the aggregated INTA/B/C/D. Transitions which occur too close together to be independently transmitted are merged.

conv_port_2 Register 0x108

This register set is used for the Converted Port 2 register.

Table 5.168. conv_port_2 Register 0x108

Field	Name	Access	Width	Reset	Description
[31:5]	reserved	RO	27	0x0	—
[4:3]	pipe_power_down	RW	2	0x0	Applicable if en_pipe_if_ctrl == 1 (pcie_ll_main_ctrl_5[0]). Set this register to force drive the pipe interface signal. Power up or down the transceiver. 00 – P0, normal operation 01 – P0s, low recovery time latency power saving state 10 – P1, longer recovery time latency power saving state 11 – P2, lowest power state
[2]	pipe_tx_cm_disable	RW	1	0x0	Applicable if en_pipe_if_ctrl == 1 (pcie_ll_main_ctrl_5[0]). Set this register to force drive the pipe interface signal. L1 substate disable Tx common mode voltage. Through this signal the Link Layer effectively configure the Tx driver into Hi-Z (power down) and move the PHY to L1.2 (Tx common mode voltage is disabled).
[1]	pipe_rx_ei_disable	RW	1	0x0	Applicable if en_pipe_if_ctrl == 1 (pcie_ll_main_ctrl_5[0]). Set this register to force drive the pipe interface signal. L1 substate disable activity detector. Through this signal the Link Layer effectively disable the activity detector circuit (power down) and move the PHY to either L1.1 (Tx common mode voltage is still valid) or L1.2 (Tx common mode voltage is disabled).

Field	Name	Access	Width	Reset	Description
[0]	pipe_pclkreq_n	RW	1	0x0	Applicable if en_pipe_if_ctrl == 1 (pcie_ll_main_ctrl_5[0]). Set this register to force drive the pipe interface signal. L1 substate request. Active low request to enter L1 substate. The Link Layer should wait for pipe_pclkack_n assertion (low) before to effectively gate the reference clock on the board through CLKREQ# out of band signal.

5.1.6.3. Status Port Register

stat_port_0 Register 0x200

This register set is used for the Status Port 0 register.

Table 5.169. stat_port_0 Register 0x200

Field	Name	Access	Width	Reset	Description
[31:16]	reserved	RO	16	0x0	—
[15:14]	phy_sts_pipe_power_down	RO	2	0x0	Power up or down the transceiver. 00 – P0, normal operation 01 – P0s, low recovery time latency power saving state 10 – P1, longer recovery time latency power saving state 11 – P2, lowest power state
[13]	phy_sts_pipe_tx_cm_disable	RO	1	0x0	Signal from PIPE interface. This register may not reflect the current value due to synchronization. L1 substate disable Tx common mode voltage. Through this signal the LL effectively configure the Tx driver into Hi-Z (power down) and move the PHY to L1.2 (Tx common mode voltage is disabled).
[12]	phy_sts_pipe_rx_ei_disable	RO	1	0x0	Signal from PIPE interface. This register may not reflect the current value due to synchronization. L1 substate disable Tx common mode voltage. Through this signal the LL effectively configure the Tx driver into Hi-Z (power down) and move the PHY to L1.2 (Tx common mode voltage is disabled).
[11]	phy_sts_pipe_pclkack_n	RO	1	0x0	Signal from PIPE interface. This register may not reflect the current value due to synchronization. L1 substate acknowledge. Active low acknowledge signal to enter L1 substate. The Link Layer should wait for pipe_pclkack_n assertion (low) before to effectively gate the reference clock on the board through CLKREQ# out of band signal.
[10]	phy_sts_pipe_pclkreq_n	RO	1	0x0	Signal from PIPE interface. This register may not reflect the current value due to synchronization. L1 substate request. Active low request signal to enter L1 substate. The Link Layer should wait for pipe_pclkack_n assertion (low) before to effectively gate the reference clock on the board through CLKREQ# out of band signal

Field	Name	Access	Width	Reset	Description
[9]	phy_sts_pipe_phy_status	RW, W1C	1	0x0	Signal from PIPE interface. 0 – Otherwise 1 – pipe_phy_status is asserted. Write 1 to clear.
[8]	phy_sts_pipe_rstn	RW, W1C	1	0x0	Signal from PIPE interface. 0 – Otherwise 1 – pipe_rstn wis asserted. Write 1 to clear.
[7:6]	reserved	RO	2	0x0	—
[5]	phy_sts_arxpllstable	RO	1	0x0	Signal from PMA interface. Rx PLL locked.
[4]	phy_sts_atxpllstable	RO	1	0x0	Signal from PMA interface. Tx PLL locked
[3]	phy_sts_acdrdiagout	RO	1	0x0	Signal from PMA interface. CDR PLL locked on data.
[2]	phy_sts_atrandet	RO	1	0x0	Signal from PMA interface. Activity detected
[1]	phy_sts_acdrpllrstb	RO	1	0x0	Signal from PMA interface. Rx PLL reset.
[0]	phy_sts_txpllrstb	RO	1	0x0	Signal from PMA interface. Tx PLL reset.

5.2. PCI Express Configuration Space Registers

The Lattice PCIe x1 IP Core implements Header Type 00 and Header Type 01 Configuration Registers, including Capability and Extended Capability Items, as detailed in the PCI Express Base Specification, Rev 3.0, PCI Local Interface Specification Revision 3.0, and PCI Bus Power Management Interface Specification Revision 1.2.

Type 00 and Type 01 Configuration Registers implement the first 64 bytes of Configuration Space differently:

- Type 00 – Implemented by Endpoints; refer Table 5.170.
- Type 01 – Implemented by Root Ports; refer Table 5.171.

Capability and Extended Capability Items are located at the same addresses regardless of which the header type is implemented, see Table 5.172 for details.

Table 5.170 , Table 5.171, and Table 5.172 illustrate the core's PCIe Configuration Register map.

The Configuration Registers provide the ability for standard PCI/PCIe BIOS/OS software to discover the device, determine its capabilities, and configure the core's features. Since there are a tremendous variety of applications, the core's Configuration Registers are highly configurable.

5.2.1. Type 00 Configuration Header

Table 5.170. Type 00 Configuration Header

Addr	Byte3	Byte2	Byte1	Byte0
00	Device ID		Vendor ID	
04	Status		Command	
08	Class Code			Revision ID
0C	BIST	Header Type	Latency Timer	Cache Line Size
10	Base Address Register 0			
14	Base Address Register 1			
18	Base Address Register 2			
1C	Base Address Register 3			
20	Base Address Register 4			
24	Base Address Register 5			
28	Cardbus CIS Pointer			
2C	Subsystem ID		Subsystem Vendor ID	
30	Expansion ROM Base Address			
34	Reserved			Capabilities Pointer
38	Reserved			

Addr	Byte3	Byte2	Byte1	Byte0
3C	Max Latency	Min Grant	Interrupt Pin	Interrupt Line

5.2.2. Type 01 Configuration Header

Table 5.171. Type 01 Configuration Header

Addr	Byte3	Byte2	Byte1	Byte0
00	Device ID		Vendor ID	
04	Status		Command	
08	Class Code			Revision ID
0C	BIST	Header Type	Primary Latency Timer	Cache Line Size
10	Base Address Register 0			
14	Base Address Register 1			
18	Secondary Latency Timer	Subordinate Bus Number	Secondary Bus Number	Primary Bus Number
1C	Secondary Status		I/O Limit	I/O Base
20	Memory Limit		Memory Base	
24	Prefetchable Memory Limit		Prefetchable Memory Base	
28	Prefetchable Base Upper 32 Bits			
2C	Prefetchable Limit Upper 32 Bits			
30	I/O Limit Upper 16 Bits		I/O Base Upper 16 Bits	
34	Reserved			Capability Pointer
38	Expansion ROM Base Address			
3C	Bridge Control		Interrupt Pin	Interrupt Line

5.2.3. Capability and Extended Capability Address Locations

Table 5.172. Capability and Extended Capability Items

Addr	Byte3	Byte2	Byte1	Byte0
7B-40	PCI Express Capability			
7F-7C	Reserved			
87-80	Power Management Capability			
8F-88	Reserved			
9B-90	MSI-X Capability			
9F-9C	Reserved			
B7-A0	MSI Capability			
FF-B8	Reserved			
147-100	Advanced Error Reporting Capability			
14F-148	ARI Capability			
17F-150	Vendor-Specific Extended Capability			
1AB-180	Secondary PCI Express Extended Capability			
1FF-1AC	Reserved			
207-200	ATS Capability			
20F-208	Reserved			
21B-210	DSN Capability			
26B-240	Reserved			
2BF-280	Resizable BAR Capability			
38F-2C0	Reserved			
39F-390	Power Budgeting Capability			

Addr	Byte3	Byte2	Byte1	Byte0
3CF-3A0	Dynamic Power Allocation (DPA) Capability			
3DF-3D0	L1 PM Substates Extended Capability			
3E7-3E0	Latency Tolerance Reporting (LTR) Capability			
FFF-3E8	Reserved			

5.2.4. Type 00 Configuration Registers

Table 5.173. Type 00 Configuration Registers

Addr	Config Register	Register Description
01–00	Vendor ID	Read Only: This field identifies the manufacturer of the device.
03–02	Device ID	Read Only: This field identifies the device.
05–04	Command Register	Command Register Bits: Bits 10, 8, 6, and 2..0 are Read/Write. Bits[15:11] = 00000. Not implemented. Bit[10] – Interrupt Disable – If set, interrupts are disabled and cannot be generated; if clear interrupts are enabled Bit[9] = 0. Not implemented. Bit[8] – SERR Enable – When set enables the reporting of fatal and non-fatal errors detected by the device to the root complex (not supported). Bit[7] = 0. Not implemented. Bit[6] – Parity Error Enable – Affects the mapping of PCI Express errors to legacy PCI errors. See *PCI Express Base Specification Rev1.1*, Section 6.2 for details. Bit[5] = 0. Not implemented. Bit[4] = 0. Not implemented. Bit[3] = 0. Not implemented. Bit[2] – Bus Master Enable – Memory and I/O Requests can only be generated on the Transaction Layer Interface if this bit is set. Bit[1] – Memory Space Enable – If set, the core decodes the packets to determine memory BAR hits; if clear, memory BARs are disabled. Bit[0] – I/O Space Enable – If set, the core decodes the packets to determine I/O BAR hits; if clear, I/O BARs are disabled.
07–06	Status Register	Status Register Bits: Bits 15..11 and 8 are Read/Write. Writing a 1 to a bit location clears that bit. Writing a 0 to a bit location has no affect. Bit[15] – Set by a device whenever it receives a Poisoned TLP. Bit[14] – Set when a device sends an ERR_FATAL or ERR_NONFATAL Message and the SERR Enable bit in the Command Register is set. Bit[13] – Set when a requestor receives a completion with Unrecognized Request Completion Status Bit[12] – Set when a requestor receives a completion with Completer Abort Completion Status Bit[11] – Set when a device completes a request using Completer Abort Completion Status Bits[10:9] = 00. Not implemented. Bit[8] – Leader Data Parity Error – This bit is set by a Requestor if its Parity Error Enable bit is set and either a Completion is received that is marked poisoned or the requestor poisons a write request. Bits[7:5] = 000. Not implemented. Bit[4] = 1 to indicate the presence of a Capabilities List. Bit[3] – Interrupt Status – Reflects the value of mgmt_interrupt. Bits[2:0] = 000. Reserved.
08	Revision ID	Read Only: This register specifies the device specific revision identifier.
0B–09	Class Code	Read Only: The Class Code identifies the generic function of the device.

Addr	Config Register	Register Description
0C	0x0C: Cache Line Size	Read/Write: Cache Line Size is not used with PCI Express but is still implemented as read/write register for legacy compatibility purposes.
0D	0x0D: Latency Timer	Read Only returning 0x00.
0E	0x0E: Header Type	Read Only: This register reads 0x00 to indicate that the core complies to the standard PCI configuration register mapping and that it is a single function device.
0F	0x0F: BIST	Not implemented. Reads return 0x00.
13–10	Base Address Register 0	Read/Write: Base Address Register0, Base Address Register1, Base Address Register2, Base Address Register3, Base Address Register4, and Base Address Register5 inform system software of the device's resource requirements and are subsequently programmed to allocate memory and I/O resources to the device.
17–14	Base Address Register 1	See Base Address Register 0 description
1B–18	Base Address Register 2	See Base Address Register 0 description
1F–1C	Base Address Register 3	See Base Address Register 0 description
23–20	Base Address Register 4	See Base Address Register 0 description
27–24	Base Address Register 5	See Base Address Register 0 description
2B–28	Card Bus CIS Pointer	Read Only: Reads return the value of the Cardbus CIS Pointer.
2D–2C	Subsystem Vendor ID	Read Only: Additional vendor information. Reads return the value of the Subsystem Vendor ID.
2F–2E	Subsystem ID	Read Only: Additional device information. Reads return the value of the Subsystem ID.
33–30	Expansion ROM Base Addr. Reg.	Informs system software of the device's Expansion ROM resource requirements and is subsequently programmed to allocate memory resources to the device. Read/Write: Expansion ROM Base Address Register Bits[31:11] – Written to specify where to locate this region in memory space Bits[10:1] = 0..0 Reserved Bit[0] = Set by S/W to enable decoding the Expansion ROM and clear to disable
34	Capabilities Pointer	Read Only: Reads return 0x40 which is the beginning address of the PCI Express Capabilities Item.
37–35	Reserved	Not implemented. Reads return 0x000000.
3B–38	Reserved	Not implemented. Reads return 0x00000000.
3C	Interrupt Line	Legacy interrupt is always ENABLED.
3D	Interrupt Pin	Interrupt support is enabled/disabled by CSR register. When interrupts are enabled, Interrupt Pin returns 0x01 indicating the core implements INTA# and when interrupts are disabled, Interrupt Pin returns 0x00 indicating no interrupts are used.
3E	Minimum Grant	Read Only: Returns 0x00.
3F	Maximum Latency	Read Only: Returns 0x00.

5.2.5. PCI Express Capability

Table 5.174. PCI Express Capability

Addr	Config Register	Register Description
40	PCI Express Capability ID	Read Only = 0x10 (Beginning of PCI Express Capability Item)
41	Next Capability Pointer	Read Only = 0x80 (Pointer to beginning of Power Management Capability)

43-42	PCI Express Capabilities	Read Only
		• Bits[15:14] – Reserved = 00
		• Bits[13:9] – Interrupt Message Number[4:0]; MSI/MSI-X interrupt vector associated with interrupts generated by Configuration Register events (change in link bandwidth and root port error)
		• Bit[8] – Slot Implemented; Downstream Switch/Root Port only
		• Bits[7:4] – Device/Port Type – Must match the core application since the value programmed enables/hides Configuration Registers and functionality that is only applicable to some Device/Port types:
		  • 0000 – PCI Express Endpoint
		  • Required for Endpoint applications
		  • 0001 – Legacy PCI Express Endpoint
		  • 0100 – Reserved
		  • 0101 – Reserved
		  • 0110 – Reserved
		  • 0111 – Reserved
		  • 1000 – Reserved
		  • 1001 – Reserved
		  • 1010 – Reserved
		• Bits[3:0] – Capability Version – Must be 0x2 for PCIe 3.0

47-44	Device Capabilities Register	Read Only
		• Bits[31:29] – Reserved.
		• Bit[28] – Function Level Reset Capability
		  • 1 – Capability Present
		  • 0 – Capability Not Present
		• Bits[27:26] – Captured Slot Power Limit Scale
		• Bits[25:18] – Captured Slot Power Limit Value
		• Bits[17:16] = 00. Reserved.
		• Bit[15] = 1. Role-based Error Reporting
		• Bit[14] = 0 – Reserved
		• Bit[13] = 0 – Reserved
		• Bit[12] = 0 – Reserved
		• Bits[11:9] – Endpoint L1 Acceptable Latency
		• Bit[8:6] – Endpoint L0s Acceptable Latency
		• Bit[5] – Extended Tag Field Supported
		• Bits[4:3] – Phantom Functions Supported
		• Bits[2:0] – Max Payload Size Supported
		  • 000 – 128 bytes max payload size
		  • 001 – 256 bytes max payload size
		  • 010 – 512 bytes max payload size
		  • 011 – 1024 bytes max payload size
		  • 100 – 2048 bytes max payload size
		  • 101 – 4096 bytes max payload size
		  • 110 – Reserved
		  • 111 – Reserved

Addr	Config Register	Register Description

49-48	Device Control Register	Read/Write
		• Bit[15] – Bridge Configuration Retry Enable/Initiate Function Level Reset
		• Bits[14:12] – Max Read Request Size; the Transmit Interface may not transmit a read request TLP with a length larger than the size indicated by Max Read Request Size:
		  • 000 == 128 bytes
		  • 001 == 256 bytes
		  • 010 == 512 bytes
		  • 011 == 1024 bytes
		  • 100 == 2048 bytes
		  • 101 == 4096 bytes
		  • 110 == Reserved
		  • 111 == Reserved
		• Bit[11] – Enable No Snoop
		• Bit[10] – Aux Power PM Enable
		• Bit[9] – Phantom Functions Enable
		• Bit[8] – Extended Tag Field Enable
		• Bits[7:5] – Max Payload Size; the Transmit Interface may not transmit a TLP with a payload larger than the size indicated by Max Payload Size:
		  • 000 == 128 bytes
		  • 001 == 256 bytes
		  • 010 == 512 bytes
		  • 011 == Reserved
		  • 100 == Reserved
		  • 101 ==Reserved
		  • 110 == Reserved
		  • 111 == Reserved
		• Bit[4] – Enable Relaxed Ordering
		• Bit[3] – Unsupported Request Reporting Enable
		• Bit[2] – Fatal Error Reporting Enable
		• Bit[1] – Non-Fatal Error Reporting Enable
		• Bit[0] – Correctable Error Reporting Enable

4B-4A	Device Status Register	Bits[15:4] are Read Only. Bits[3:0] are cleared by writing a 1 to the corresponding bit location. • Bits[15:6] = 0000000000. Reserved • Bit[5] – Transactions Pending • Bit[4] – AUX Power Detected • Bit[3] – Unsupported Request Detected • Bit[2] – Fatal Error Detected • Bit[1] – Non-Fatal Error Detected • Bit[0] – Correctable Error Detected

4F-4C	Link Capabilities Register	Read Only.
		• Bits[31:24] – Port Number
		• Bits[22] = 1. ASPM Optional Compliance
		• Bit[21] – Link Bandwidth Notification Capability
		  • == 1 when operating as a Downstream Port; else 0
		• Bit[20] – Data Link Layer Active Reporting Capable
		  • == 1 when operating as a Downstream Port; else 0
		• Bit[19] – Surprise Down Error Reporting Capable
		  • == 1 when operating as a Downstream Port; else 0
		• Bit[18] = 0. Clock Power Management
		• Bits[17:15] – L1 Exit Latency

Addr	Config Register	Register Description

		• Bits[14:12] – L0s Exit Latency
		• Bits[11:10] – Active State Power Management (ASPM) Support
		  • 00 – No ASMP Support
		  • 01 – L0s Supported
		  • 10 – L1 Supported
		  • 11 – L0s and L1 Supported
		• Bits[9:4] – Maximum Link Width
		  • 000001 – x1
		  • 000010 – x2
		  • 000100 – x4
		  • 001000 – x8
		  • 010000 – x16
		• Bits[3:0] – Maximum Link Speed
		  • 0001 (2.5GT/s)
		  • 0010 (5GT/s)
		  • 0011 (8GT/s)

51-50	Link Control Register	Read/Write
		• Bits[15:12] = 0000. Reserved.
		• Bit[11] – Link Autonomous Bandwidth Interrupt Enable
		• Bit[10] – Link Bandwidth Management Interrupt Enable
		• Bit[9] – Hardware Autonomous Width Disable
		• Bit[8] = 0. Enable Clock Power Management
		• Bit[7] – Extended Sync
		• Bit[6] – Common Clock Configuration
		• Bit[5] – Retrain Link
		• Bit[4] – Link Disable
		• Bit[3] – Read Completion Boundary (RCB)
		  • 0 == 64 bytes
		  • 1 == 128 bytes
		• Bit[2] = 0. Reserved.
		• Bits[1:0] – Active State Power Management (ASPM) Control
		  • 00 – Disabled
		  • 01 – L0s Enabled
		  • 10 – L1 Enabled
		  • 11 – L0s and L1 Enabled

53-52	Link Status Register	Read Only
		• Bit[15] – Link Autonomous Bandwidth Status
		• Bit[14] – Link Bandwidth Management Status
		• Bit[13] – Data Link Layer Active
		• Bit[12] – Slot Clock Configuration
		• Bit[11] – Link Training
		• Bit[10] = 0. Reserved.
		• Bits[9:4] Negotiated Link Width – indicates the number of lanes currently in use
		  • 010000 = x16
		  • 001000 = x8
		  • 000100 = x4
		  • 000010 = x2
		  • 000001 = x1
		• Bits[3:0] Link Speed
		  • 0001 (2.5 GT/s)

Addr	Config Register	Register Description
		• 0010 (5.0 GT/s) • 0011 (8.0 GT/s)
57-54	Slot Capabilities Root Port/Switch Only	Normally Read Only; Writable when HW.Init Write Enable == 1 (see Table 5.180) • Bits[31:19] – Physical Slot Number • Bit[18] – No Command Completed Support • Bit[17] – Electromechanical Interlock Present • Bits[16:15] – Slot Power Limit Scale[1:0] • Bits[14:7] – Slot Power Limit Value[7:0] • Bit[6] – Hot-Plug Capable • Bit[5] – Hot-Plug Surprise • Bit[4] – Power Indicator Present • Bit[3] – Attention Indicator Present • Bit[2] – MRL Sensor Present • Bit[1] – Power Controller Present • Bit[0] – Attention Button Present
59-58	Slot Control Root Port/Switch Only	Read Only • Bits[15:13] = 0. Reserved. • Bit[12] – Data Link Layer State Changed Enable • Bit[11] = 0. Electromechanical Interlock Control • Bit[10] – Power Controller Control • Bit[9:8] – Power Indicator Control • Bit[7:6] – Attention Indicator Control • Bit[5] – Hot-Plug Interrupt Enable • Bit[4] – Command Completed Interrupt Enable • Bit[3] – Presence Detect Changed Enable • Bit[2] – MRL Sensor Changed Enable • Bit[1] – Power Fault Detected Enable • Bit[0] – Attention Button Pressed Enable
5b-5a	Slot Status Root Port/Switch Only	Read Only • Bits[15:9] = 0. Reserved. • Bit[8] – Data Link Layer State Changed • Bit[7] – Electromechanical Interlock Status • Bit[6] – Presence Detect State • Bit[5] – MRL Sensor State • Bit[4] – Command Completed • Bit[3] – Presence Detect Changed • Bit[2] – MRL Sensor Changed • Bit[1] – Power Fault Detected • Bit[0] – Attention Button Pressed
5d-5c	Root Control Root Port Only	Read/Write • Bits[15:5] = 0. Reserved. • Bit[4] = CRS Software Visibility Enable • Bit[3] – PME Interrupt Enable • Bit[2] – System Error on Fatal Error Enable • Bit[1] – System Error on Non-Fatal Error Enable • Bit[0] – System Error on Correctable Error Enable

Addr	Config Register	Register Description
5f-5e	Root Capabilities Root Port Only	Read Only; Bit[16] – Write 1 to clear. • Bits[15:1] = 0. Reserved • Bit[0] = 1. CRS Software Visibility supported.
63-60	Root Status Root Port Only	Read Only; Bit[16] – Write 1 to clear. • Bits[31:18] = 0. Reserved • Bit[17] – PME Pending • Bit[16] – PME Status • Bits[15:0] – PME Requester ID
67-64	Device Capabilities 2	Read Only • Bits[31:24] = 0. Reserved • Bits[23:22] = 00. Max End-End TLP Prefixes • Bit[21] = 0. End-End TLP Prefix Supported • Bit[20] = 0. Extended Fmt Field Supported • Bit[19:18] = 00. OBFF Supported • Bits[17:14] = 0000. Reserved • Bits[13:12] = 00. TPH Completer Supported • Bit[11] = LTR Mechanism Supported • Bit[10] = 0. No RO-enabled PR-PR Passing • Bit[9] = 0. 128-bit CAS Completer Supported • Bit[8] = 0. 64-bit AtomicOp Completer Supported • Bit[7] = 0. 32-bit AtomicOp Completer Supported • Bit[6] = 0. AtomicOp Routing Supported • Bit[5] = 0. ARI Forwarding Supported • Bit[4] – Completion Timeout Disable Supported • Bits[3:0] – Completion Timeout Ranges Supported
69-68	Device Control 2	Read/Write • Bit[15] – End-End TLP Prefix Blocking • Bits[14:13] – OBFF Enable; not supported • Bits[12:11] = 00. Reserved. • Bit[10] – LTR Mechanism Enable • Bit[9] – IDO Completion Enable • Bit[8] – IDO Request Enable • Bit[7] – AtomicOp Egress Blocking • Bit[6] – AtomicOp Request Enable • Bit[5] – ARI Forwarding Enable • Bit[4] – Completion Timeout Disable – Set by system software to disable this device from generating completion timeouts. You must disable completion timeout error generation when this bit is set. • Bits[3:0] – Completion Timeout Value – Set by system software to select the completion timeout range which must be used by users which are implementing completion timeouts. See PCI Express Specification Table 7.24 for details.
6B-6A	Device Status 2	Reserved by PCI SIG for future use. Reads return 0x00000000.
6F-6C	Link Capabilities 2	Read Only • Bits[31:23] – Reserved • Bits[22:16] – Lower SKP OS Reception Supported Speeds Vector • Bits[15:9] – Lower SKP OS Generation Supported Speeds Vector • Bit[8] – CrossLink Supported • Bits[7:1] – Supported Link Speeds Vector • Bit[0] = 0. Reserved
71-70	LinkControl 2	Read/Write • Bit[15:12] – Compliance Preset/De-emphasis[3:0] • Bit[11] – Compliance SOS

Addr	Config Register	Register Description

		• Bit[10] – Enter Modified Compliance
		• Bits[9:7] – Transmit Margin
		• Bit[6] – Selectable De-emphasis
		• Bit[5] – Hardware Autonomous Speed Disable
		• Bit[4] – Enter Compliance
		• Bits[3:0] – Target Link Speed[3:0]
		  • 0001 (2.5 GT/s)
		  • 0010 (5.0 GT/s)
		  • 0011 (8.0 GT/s)

73-72	Link Status 2	Read Only; Bit[5] – write 1 to clear:
		• Bits[15:6] = 0000000000. Reserved.
		• Bit[5] – Link Equalization Reset
		• Bit[4] – Equalization Phase 3 Successful
		• Bit[3] – Equalization Phase 2 Successful
		• Bit[2] – Equalization Phase 1 Successful
		• Bit[1] – Equalization Complete
		• Bit[0] – Current De-emphasis Level
		  • 1==-3.5 dB
		  • 0==-6 dB

77-74	Slot Capabilities 2 Root Port/Switch Only	Reserved by PCI SIG for future use. Reads return 0x00000000.
79-78	Slot Control 2 Root Port/Switch Only	Reserved by PCI SIG for future use. Reads return 0x00000000.
7b-7a	Slot Status 2 Root Port/Switch Only	Reserved by PCI SIG for future use. Reads return 0x00000000.
7F-7C	Reserved	Reads return 0x00000000.

5.2.6. Power Management Capability

Table 5.175. Power Management Capability

Addr	Config Register	Register Description
80	Power Management Capability ID	Read Only = 0x01 (Beginning of Power Management Capability Item)
81	Next Capability Pointer	Read Only. Pointer to next Capability Item in the list.
83-82	Power Management Capabilities	Read Only. • Bits[15:11] – PME Support; recommended default == 0. • Bits[10] – D2 Support (1) Yes (0) No; this bit must be set for the core to allow Power State to be written to D2; recommended default == 0. • Bit[9] – D1 Support (1) Yes (0) No; this bit must be set for the core to allow Power State to be written to D1; recommended default == 0. • Bit[8:6] – Aux Current; recommended default = 0. • Bit[5] – Device Specific Initialization(DSI); recommended default = 0. • Bit[4] – Reserved; set to 0. • Bit[3] – PME Clock; recommended default = 0. • Bits[2:0] – Version; set to 011 (complies with revision 1.2 of the PCI Power Management Interface Specification). Refer Error Handling for additional detail.

Addr	Config Register	Register Description

85-84	Power Management Control/Status	Read/Write.
		• Bit[15] – PME Status; if Power Management Capabilities[15] == 1 indicating that PME is generated from D3cold, then PME_Status is implemented by the core; otherwise PME_Status == 0.
		• Bits[14:13] – Data Scale; recommend == 0 (Data not implemented)
		• Bits[12:9] – Data Select; recommend == 0 (Data not implemented)
		• Bit[8] – PME En – ; if Power Management Capabilities[15:11] == 0 indicating that PME is not generated from any power state then PME_En == 0; is implemented by the core and written by system software to enable PME generation from D3cold; otherwise PME_En == 0.
		• Bits[7:4] – Reserved – set to 0
		• Bit[3] – No Soft Reset – Core sets to 1 since the core is not reset when transitioning from D3hot to D0 purely due to power state changes. This bit is used by system software to know whether the device needs to be reinitialized when transitioning between D3hot and D0.
		• Bit[2] – Reserved; set to 0
		• Bits[1:0] – Power State; software writes this field to transition a device into a different power state; increasing Dx numbers represent increasingly lower power states
		  • 00 – D0; normal operation
		  • 01 – D1; not allowed to be written unless D1 Support == 1
		  • 10 – D2; not allowed to be written unless D2 Support == 1
		  • 11 – D3hot; “off”
		Refer Error Handling for additional detail.

86	PMCSR PCI to PCI Bridge Support	Read Only. • Bit[7] – Bus Power/Clock Control Enable; set to 0 • Bit[6] – B2/B3 Support for D3bat; set to 0 • Bits[5:0] – Reserved; set to 0
87	Data	Read Only. • Bits[7:0] – Data; recommended default = 0; not implemented
8F-88	Reserved	Reads return 0x00000000.

5.2.7. MSI-X Capability

Table 5.176. MSI-X Capability

Addr	Config Register	Register Description
90	MSI-X Capability ID	Read Only = 0x11 (Beginning of MSI-X Capability Item) MSI-X Support may be enabled/disabled through the CSR registers. If present, its capability is defined as follows otherwise all the following registers reads 0x0.
91	Next Capability Pointer	Read Only. Pointer to next Capability Item in the list.
93-92	Message Control	Only bits[15:14] are Read/Write. • Bit[15] – MSI-X Enable (Read/Write) • Bit[14] – Function Mask (Read/Write) • Bits[13:11] – Reserved – 000 (Read Only) • Bit[10:0] – Table Size[10:0] (Read Only) • The number of MSI-X vectors requested/supported by the user’s design is Table Size + 1.
97-94	Table_Offset, Table_BIR	Read Only. Bits[31:3] – Table_Offset[31:3] • {Table_Offset[31:3], 000} is the offset into the BAR indicated by Table_BIR where the MSI-X Table begins. • Bits[2:0] – Table BIR[2:0]

Addr	Config Register	Register Description

		• Indicates which BAR location contains the MSI-X Table. In the case of a 64-bit BAR Table BIR indicates the BAR that contains the lower 32-bit address:
		  • 000 – BAR0
		  • 001 – BAR1
		  • 010 – BAR2
		  • 011 – BAR3
		  • 100 – BAR4
		  • 101 – BAR5
		  • 110, 111 – Reserved

9B-98	PBA_Offset, PBA_BIR	Read Only. Bits[31:3] – PBA_Offset[31:3] • Same as Table Offset above but indicates the location of the PBA (Pending Bit Array). • Bits[2:0] – PBA BIR[2:0] • Same as Table BIR above, but indicates the location of the PBA

5.2.8. MSI Capability

Table 5.177. MSI Capability

Addr	Config Register	Register Description
9F-9C	Reserved	Reads return 0x00000000.
A0	Message Capability ID	Read Only = 0x05 (Beginning of Message Capability Item); MSI Support is enabled/disabled by CSR registers. If present, its capability is defined as follows otherwise all the following registers read 0x0.
A1	Next Capability Pointer	Read Only. Pointer to next Capability Item in the list.
A3-A2	Message Control	• Bits[6:4] and Bit[0] are Read/Write; remainder are Read Only • Bits[15:9] = 0x00. Reserved • Bit[8] = 0. Note per vector masking capable. • Bit[7] – 64-bit Address Capable = 1 (Capable of generating 64-bit messages) • Bits[6:4] – Multiple Message Enable – system software writes the number of allocated messages; 000==1, 001==2, 010==4, 011==8, 100==16, 101==32, 110 Reserved, 111 Reserved • Bits[3:1] – Multiple Message Capable – Number of messages requested by the device == 000 (1 Message) • Bit[0] – MSI Enable – System software sets this bit to enable MSI. When set, the core uses the MSI mechanism instead of the legacy interrupt mechanism to forward user interrupts on mgmt_interrupt to PCI Express.
A7-A4	Message Address	Bits[31:2] are Read/Write; Bits[1:0] are Read Only • Bits[31:2] Message Address[31:2] • Bits[1:0] – Reserved – Message Address[1:0] is always 00
AB-A8	Message Upper Address	Read/Write • Bits[31:0] Message Address[63:32]; if Message Address[63:32] == 0, then the core uses only Message Address[31:0] and does 32-bit address MSI writes. If Message Address[63:32] != 0 then the core uses Message Address[63:0] and does 64-bit address MSI writes.
AD-AC	Message Data	Read/Write • Bits[15:0] Message Data[15:0] – An MSI Message is sent by writing Message Data to Message Address.

5.2.9. Advanced Error Reporting Extended Capability

Table 5.178. Advanced Error Reporting Extended Capability

Addr	Config Register	Register Description
103-100	Advanced Error Reporting Enhanced Capability Header	Beginning of Advanced Error Reporting (AER) Capability; the AER capability is only present if AER support is enabled in the design, however, AER support is a standard core feature that is present unless AER removal has been specifically requested to be excluded at core deliver time (which is unusual). • Bits[15:0] – Read Only = 0x0001 == AER Capability ID • Bits[19:16] – Read Only = 0x01 == AER Capability Version (PCIe 2.0/1.1) • Bits[31:20] – Read Only. Pointer to next Enhanced/Extended Capability Item in the list.
107-104	Uncorrectable Error Status	• Read/Write: Bit set when corresponding error event occurs, and the error is not masked by the Uncorrectable Error Mask register; clear set bits by writing a 1: • Bits[3:0] – Reserved == 0 • Bit[4] – DataLink_Protocol_Error_Status • Bit[5] – Surprise_Down_Error_Status • Bits[11:6] – Reserved == 0 • Bit[12] – Poisoned_TLP_Status • Bit[13] – Flow_Control_Protocol_Error_Status • Bit[14] – Completion_Timeout_Status • Bit[15] – Completer_Abort_Status • Bit[16] – Unexpected_Completion_Status • Bit[17] – Receiver_Overflow_Status • Bit[18] – Malformed_TLP_Status • Bit[19] – ECRC_Error_Status • Bit[20] – Unsupported_Request_Error_Status • Bit[21] – Reserved = 0 • Bit[22] – Uncorrectable Internal Error Status • Bits[31:23] – Reserved == 0
10B-108	Uncorrectable Error Mask	Read/Write: Set corresponding bit to mask (not report) selected error events; clear to unmask (report): • Bits[3:0] – Reserved == 0 • Bit[4] – DataLink_Protocol_Error_Mask • Bit[5] – Surprise_Down_Error_ Mask • Bits[11:6] – Reserved == 0 • Bit[12] – Poisoned_TLP_ Mask • Bit[13] – Flow_Control_Protocol_Error_ Mask • Bit[14] – Completion_Timeout_ Mask • Bit[15] – Completer_Abort_ Mask • Bit[16] – Unexpected_Completion_ Mask • Bit[17] – Receiver_Overflow_ Mask • Bit[18] – Malformed_TLP_ Mask • Bit[19] – ECRC_Error_ Mask • Bit[20] – Unsupported_Request_Error_ Mask • Bit[21] – Reserved = 0 • Bit[22] – Uncorrectable Internal Error Mask • Bits[31:23] – Reserved == 0
10F-10C	Uncorrectable Error Severity	Read/Write: Set corresponding bit to mark selected error events as FATAL errors; clear to mark selected error events as NON-FATAL errors: • Bits[3:0] – Reserved == 0 • Bit[4] – DataLink_Protocol_Error_Severity • Bit[5] – Surprise_Down_Error_Severity • Bits[11:6] – Reserved == 0

Addr	Config Register	Register Description
		• Bit[12] – Poisoned_TLP_Severity • Bit[13] – Flow_Control_Protocol_Error_Severity • Bit[14] – Completion_Timeout_Severity • Bit[15] – Completer_Abort_Severity • Bit[16] – Unexpected_Completion_Severity • Bit[17] – Receiver_Overflow_Severity • Bit[18] – Malformed_TLP_Severity • Bit[19] – ECRC_Error_Severity • Bit[20] – Unsupported_Request_Error_Severity • Bit[21] – Reserved = 0 • Bit[22] – Uncorrectable Internal Error Severity • Bits[31:23] – Reserved == 0
113-110	Correctable Error Status	Read/Write: Bit set when corresponding error event occurs, and the error is not masked by the Correctable Error Mask register; clear set bits by writing a 1: • Bit[0] – Receiver_Error_Status • Bits[5:1] – Reserved == 0 • Bit[6] – Bad_TLP_Status • Bit[7] – Bad_DLLP_Status • Bit[8] – Replay_Num_Rollover_Status • Bits[11:9] – Reserved == 000 • Bit[12] – Replay_Timer_Timeout_Status • Bit[13] – Advisory_Non_Fatal_Error_Status • Bit[14] – Corrected Internal Error Status • Bit[15] – Header Log Overflow Status • Bits[31:16] – Reserved == 0
117-114	Correctable Error Mask	Read/Write: Set corresponding bit to mask (not report) selected error events; clear to unmask (report): • Bit[0] – Receiver_Error_Mask • Bits[5:1] – Reserved == 0 • Bit[6] – Bad_TLP_Mask • Bit[7] – Bad_DLLP_Mask • Bit[8] – Replay_Num_Rollover_Mask • Bits[11:9] – Reserved == 000 • Bit[12] = Replay_Timer_Timeout_Mask • Bit[13] = Advisory_Non_Fatal_Error_Mask • Bit[14] – Corrected Internal Error Mask • Bit[15] – Header Log Overflow Mask • Bits[31:16] – Reserved == 0

11B-118	Advanced Error Capabilities and Control	Read/Write: Misc Capabilities and Control
		• Bits[4:0] = Read Only – First_Error_Pointer[4:0]
		• Bit[5] = Read Only – ECRC_Generation_Capable
		  • 1==Device can generate ECRC; set if the core includes ECRC generation logic (non-standard core option).
		  • 0 == Device is not capable of generating ECRC.
		• Bit[6] = Read/Write – ECRC_Generation_Enable
		  • Software sets to control whether ECRCs are generated and inserted for TLPs transmitted by the core; if ECRC support is not implemented in the core, this bit is Read Only == 0.
		  • 1== Generate and insert ECRC for TLPs transmitted by the core.
		  • 0 == Do not generate and insert ECRC.

Addr	Config Register	Register Description

		• Bit[7] = Read Only – ECRC_Check_Capable
		  • 1==Device is capable of checking ECRC; set if the core includes ECRC generation logic (non-standard core option).
		  • 0 == Device is not capable of checking ECRC.
		• Bit[8] = Read/Write – ECRC_Check_Enable
		  • Software sets to control whether ECRCs are checked for TLPs received by the core; if ECRC support is not implemented in the core, this bit is Read Only == 0.
		  • 1== Check ECRC for all TLPs with ECRC received by the core.
		  • 0 == Do not check ECRC.
		• Bits[31:9] – Reserved = 0

12B-11C	Header Log	Header[127:0] of the TLP associated with the error. TLP format is in same order as illustrated in PCIe Specification: • 0x11F-11C – {Byte0, Byte1, Byte2, Byte3} • 0x123-120 – {Byte4, Byte5, Byte6, Byte7} • 0x127-124 – {Byte8, Byte9, Byte10, Byte11} • 0x12B-0x128 – {Byte12, Byte13, Byte14, Byte15}
137-12C	Reserved	Only implemented by AER Root Ports. Reads return 0x00000000.
147-138	Reserved	TLP Prefix Log Register

5.2.10. ARI Extended Capability

ARI Is located at offset 0x148 unless AER is not present in which case it is moved to 0x100.

Table 5.179. ARI Extended Capability

Addr	Config Register	Register Description
14B-148 or 103-100	ARI Capability Extended Capability Header	Beginning of ARI Extended Capability – Read Only • Bits[31:20] – Pointer to next Enhanced/Extended Capability Item in the list. • Bits[19:16] = 0x1 == Capability Version • Bits[15:0] = 0x000E == Capability ID
14D-14C or 105-104	ARI Capability Register	Read Only • Bits[15:8] – Next Function Number = 0 (not implemented) • Bits[7:2] – Reserved = 0 • Bit[1] – ACS Function Groups Capability = 0 (not implemented) • Bit[0] – MFVC Function Groups Capability = 0 (not implemented)
14F-14E or 107-106	ARI Control Register	Read Only • Bits[15:7] – Reserved • Bit[6:4] – Function Group = 0 (not implemented) • Bits[3:2] – Reserved = 0 • Bit[1] – ACS Function Groups Enable = 0 (not implemented) • Bit[0] – MFVC Function Groups Enable = 0 (not implemented)

5.2.11. Vendor-Specific Extended Capability

Table 5.180. Vendor-Specific Extended Capability

Addr	Config Register	Register Description
153-150	Vendor-Specifc PCI Express Extended Capability Header	Beginning of Vendor-Specific Extended Capability (VSEC) • Bits[31:20] – Read Only. Pointer to next Enhanced/Extended Capability Item in the list. • Bits[19:16] – Read Only = 0x1 == Capability Version • Bits[15:0] – Read Only = 0x000B == Capability ID
157-154	Vendor-Specific Header	Read Only • Bits[31:20] – VSEC Length = 0x24 (36 bytes) • Bits[19:16] – VSEC Rev = 0x1 • Bits[15:0] – VSEC ID

15B-158	HW.Init	Read/Write
		• Bits[31:1] = 0. Reserved
		• Bit[0] – HW.Init Write Enable – Used to allow software to write some Configuration Registers which are type *HW.Init* when they would otherwise not be writable; default value == 0
		  • 1 – HW.Init Write Enabled – Allow specific HW.Init fields to be written by software; only relevant for Configuration Registers in this document which specifically state they are writable when HW.Init Write Enable == 1 (for example, PCI Express Capability: Slot Capabilities).
		  • 0 – HW.Init Write Disabled

15F-15C	Link Power Down Root Port / Downstream Switch Port Only	Read/Write – Used by system software in a Root Port or Downstream Switch Port application to cause a PME_Turn_Off Message to be transmitted on PCI Express to request that the downstream PCI Express heirarchy prepare for Power Down. • Bits[31:3] = 0. Reserved. • Bit[2] – L2 Request Timeout; indicates when an L2 Request completed due to a timeout; L2 Request Timeout is cleared (0) when L2_Request is written to 1 or when 1 is written to this register; L2 Request Timeout is set (1) when a PME_TO_ACK message is not received in response to a transmitted PME_Turn_Off within the expected 100 ms (100 µs for simulation when mgmt_short_sim == 1) timeout window. • Bit[1] – L2 Request Status; indicates when an L2 Request has completed either due to receiving the expected PME_TO_Ack message response or due to timeout; L2 Request Status is cleared (0) when L2_Request is written to 1 or when 1 is written to this register; L2 Request Status is set (1) when a PME_TO_ACK message is received or a timeout occurs • Bit[0] – L2 Request; write to 1 to cause a PME_Turn_Off Message to be transmitted downstream to the PCI Express hierarchy; after all downstream devices have prepared for power-down the core should receive a PME_TO_Ack message in response indicating the downstream PCIe hierarchy is ready for removal of power; L2 Request stays set until a PME_TO_ACK message is received or a timeout occurs.

Addr	Config Register	Register Description
163-160	Autonomous Recovery, Speed, and Width	Read/Write – Used by system software in an US Port to perform autonomous speed change, width change, or entry to recovery. Bits[31:16] – Lane Width Mask. A 1 indicates that the lane can be used. [16] = Lane 0 .. [31] = Lane 15. Bits[15:12] – Reserved Bits[11:8] – Target Speed. (1=2.5G, 2=5G,3=8G,4=16G). Bits[7:3] – Reserved Bit[2] – Autonomous Entry to Recovery command. When Set to 1, Bits[1] and [0] must both be set to 0. Setting this bit to 1 causes the Link to immediately transition to recovery. Bit[1] – Autonomous Width Change command. When set to 1, the Link transitions to recovery to perform a link width change, using the Lane Width Mask field. This bit is ignored if HW Autonomous Width Disable has been set in the Link Control register. Bit[0] – Autonomous Speed Change command. When set to 1, the Link transitions to Recovery to perform a speed change, using the Target Speed field. This bit is ignored if HW Autonomous Speed Disable has been set in the Link Control 2 register. Speed and width changes can be signaled together. However, entry to recovery must be signaled independently from speed or width changes.
173-164	Reserved	Reserved

5.2.12. Secondary PCI Express Extended Capability

Table 5.181. Secondary PCI Express Extended Capability

Addr	Config Register	Register Description
183-180	Secondary PCI Express Extended Capability Header	Beginning of Secondary PCI Express Extended Capability; this capability is only present if the PCIe 3.0 support is enabled in the design. If the AER capability is not present, this capability is located at offset 0x100 instead. Bits[31:20] – Read Only. Pointer to next Enhanced/Extended Capability Item in the list. • Bits[19:16] – Read Only = 0x1 == Capability Version • Bits[15:0] – Read Only = 0x0019 == Capability ID
187-184	Link Control 3	Read/Write • Bits[31:16] – Reserved = 0 • Bits[15:9] – Enable Lower SKP OS Generation Vector • Bits[8:2] – Reserved = 0 • Bit[1] – Link Equalization Request Interrupt Enable • Bit[0] – Perform Equalization
18B-188	Lane Error Status	Read Only: Indicates lane-specific error status. • Bits[31:NUM_LANES] – Reserved = 0 • Bit[Lane#] – 1 == Error detected on lane[[Lane#]; 0 == no error
1AB-18C	Lane Equalization Control Register	Read Only: Control and status fields for link equalization; 16-bits per lane starting with Lane[0] with higher lane #s at higher addresses. Per lane format is as follows: • Bit[15] – Reserved = 0 • Bits[14:12] – Upstream Port Receiver Preset Hint • Bits[11:8] –Upstream Port Transmitter Preset • Bit[7] – Reserved = 0 • Bits[6:4] – Downstream Port Receiver Preset Hint • Bits[3:0] –Downstream Port Transmitter Preset

5.2.13. ATS Extended Capability

Table 5.182. ATS Extended Capability

Addr	Config Register	Register Description
203-200	ATS Capability Extended Capability Header	Beginning of ATS Extended Capability – Read Only • Bits[31:20] – Pointer to next Enhanced/Extended Capability Item in the list. • Bits[19:16] = 0x1 == Capability Version • Bits[15:0] = 0x000F == Capability ID
205-204	ATS Capability Register	Read Only • Bits[4:0] – Invalidate Queue Depth • Bit[5]Page Aligned Request • Bits[15:6] – Reserved
207-206	ATS Control Register	Read/Write • Bits[4:0] – Smallest Translation Unit • Bits[14:5] – Reserved • Bit[15] – Enable

5.2.14. DSN Extended Capability

Table 5.183. DSN Extended Capability

Addr	Config Register	Register Description
213-210	DSN Capability Extended Capability Header	Beginning of DSN Extended Capability – Read Only • Bits[31:20] – Pointer to next Enhanced/Extended Capability Item in the list. • Bits[19:16] = 0x1 == Capability Version • Bits[15:0] = 0x0003 == Capability ID
21B-214	DSN Serial NUmber	Read Only • Bits[63:0] – DSN Serial Number

5.2.15. Resizable BAR Capability

Table 5.184. Resizable BAR Capability

Addr	Config Register	Register Description
283-280	Resizable BAR Extended Capability Header	Resizable BAR Capability Header – Read Only • Bits[31:20] – Pointer to next Enhanced/Extended Capability Item in the list. • Bits[19:16] = 0x1 == Capability Version • Bits[15:0] = 0x0015 == Capability ID
287-284	Resizable BAR Capability(0)	Read Only • Bits[31:24] – Reserved • Bits[23:4] – When Bit n is Set, The BAR Indicated by the BAR Index in the Control Register operates with BAR sized to 2^(n+16) Bytes. For example, bit[4] = 2^20 Bytes = 1 MB. • Bits[3:0] – Reserved

Addr	Config Register	Register Description
28B-288	Resizable Bar Control Register(0)	Read Only • Bits[31:13] – Reserved R/W • Bits[12:8] – BAR Size. Encoded Value for the Size this BAR should use. Read Only • Bits[7:5] – Number of Resizable BARs. Value must be between 1 and 6. These bits are only valid in the Resizable BAR Control Register (0). In Control Registers (1) or higher, these bits are Reserved. Bits[2:0] – BAR Index for this BAR: 0 = BAR located at offset 0x10 1 = BAR located at offset 0x14 2 = BAR located at offset 0x18 3 = BAR located at offset 0x1C 4 = BAR located at offset 0x20 5 = BAR located at offset 0x24 Other values reserved. For a 64-bit BAR, this index should point to the lower DWORD.
2BF-28C	Resizable BAR Capability and Control Registers (1..6)	See Resizable BAR Capability (0). See Resizable Bar Control Register(0). The number of Implemented BAR Capability and Control Registers depends on the setting of *Number of Resizable BARs* Control Register (0).

5.2.16. Power Budgeting Capability

Table 5.185. Power Budgeting Capability

Addr	Config Register	Register Description
393-390	Power Budgeting Capability Extended Capability Header	Beginning of Power Budgeting Extended Capability – Read Only • Bits[31:20] – Pointer to next Enhanced/Extended Capability Item in the list. • Bits[19:16] = 0x1 == Capability Version • Bits[15:0] = 0x0004 == Capability ID
394	Data Select Register	Read/Write • Bits[7:0] – Data Select Register
397-395	Reserved	Reserved
39B-398	Data Register	Read Only • Bits[31:21] – Reserved • Bits[20:18] – Power Rail (0:12V,1:3.3V,2:1.5/1.8V,7:Thermal) • Bits[17:15] – Type (0:PME Aux,1:Aux,2:Idle,3:Sustained,7:Max) • Bits[14:13] – PM State (0:D0,1:D1,2:D2,3:D3) • Bits[12:10] – PM Sub State (0:Default,others: Device Specific) • Bits[9:8] – Data Scale (0:1x,1:0.1x,2:0.01x,3:0.001x) • Bits[7:0] – Base Power
39C	Capabilities Register	Read Only • Bits[7:1] – Reserved • Bit[0] – System Allocated – Set to 1 to indicate that the Power Budget Should be System Allocated, and the values from the Data Register should NOT be used for System Power Budgeting. Set to 0 to indicate that the values provided in the Data Register should be used for System Power Budgeting.

5.2.17. Dynamic Power Allocation Capability

Table 5.186. Dynamic Power Allocation (DPA) Capability

Addr	Config Register	Register Description
3A3-3A0	DPA Capability Extended Capability Header	Beginning of DPA Extended Capability – Read Only • Bits[31:20] – Pointer to next Enhanced/Extended Capability Item in the list. • Bits[19:16] = 0x1 == Capability Version • Bits[15:0] = 0x0016 == Capability ID
3A7-3A4	DPA Capability Register	Read Only • Bits[31:24] – Transition Latency Value1 (xlcy1) • Bits[23:16] – Transition Latency Value0 (xlcy0) • Bits[15:14] – Reserved • Bits[13:12] – Power Allocation Scale (PAS) • Bits[11:10] – Reserved • Bits[9:8] – Transition Latency Unit (tlunit) • Bits[7:5] –Reserved • Bits[4:0] – Substate_Max
3AB-3A8	DPA Latency Indicator Register	Read Only • Bits[31:Substate_Max+1] – Reserved • Bits[Substate_Max:0] – Transition Latency Indicator Bits
3AD-3AC	DPA Status Register	Read Only • Bits[15:9] – Reserved Read, Write 1 to Clear • Bits[8] – Substate Control Enabled Read Only • Bits[7:0] – Substate Status
3EF-3AE	DPA Control Register	Read Only • Bits[15:5] – Reserved Read/Write • Bits[4:0] – Substate Control
3CF-3B0	DPA Power Allocation Array	Read Only • Bits[7:0] – Substate Power Allocation Register Address 3B0 is for Substate 0 Address 3B1 is for Substate 1, up to Substate Substate_Max

5.2.18. L1 PM Substates Extended Capability

Table 5.187. L1 PM Substates Extended Capability

Addr	Config Register	Register Description
3D3-3D0	L1 PM Substates Capability Extended Capability Header	Beginning of L1 PM Substates Extended Capability – Read Only • Bits[31:20] – Pointer to next Enhanced/Extended Capability Item in the list. • Bits[19:16] = 0x1 == Capability Version • Bits[15:0] = 0x001E == Capability ID

Addr	Config Register	Register Description
3D7-3D4	L1 PM Substates Capabilities Register	HwInit • Bits[31:24] – Reserved • Bits[23:19] – Port TPOWER_ON Value • Bits [18] – Reserved • Bits[17:16] – Port TPOWER_ON Scale • Bits[15:8] – Port Common_Mode_Restore_Time (in µs) • Bits[7:5] – Reserved • Bit[4] – L1 PM Substates Supported • Bit[3] – ASPM L1.1 Supported • Bit[2] – ASPM L1.2 Supported • Bit[1] – PCI-PM L1.1 Supported • Bit[0] – PCI-PM L1.2 Supported
3DB-3D8	L1 PM Substates Control 1 Register	Read/Write • Bits[31:29] – LTR_L1.2_THRESHOLD_Scale • Bits[28:26] – Reserved • Bits[25:16] – LTR_L1.2_THRESHOLD_Value • Bits[15:8] – Common_Mode_Restore_Time • Bits[7:4] – Reserved • Bit[3] – ASPM L1.1 Enable • Bit[2] – ASPM L1.2 Enable • Bit[1] – PCI-PM L1.1 Enable • Bit[0] – PCI-PM L1.2 Enable
3DF-3DC	L1 PM Substates Control 2 Register	Read/Write • Bits[31:8] – Reserved • Bits[7:3] – TPOWER_ON Value • Bit[2] – Reserved • Bits[1:0] – T_{POWER_ON} Scale

5.2.19. Latency Tolerance Reporting Capability

Table 5.188. Latency Tolerance Reporting (LTR) Capability

Addr	Config Register	Register Description
3E3-3E0	LTR Capability Extended Capability Header	Beginning of LTR Extended Capability – Read Only • Bits[31:20] – Pointer to next Enhanced/Extended Capability Item in the list. • Bits[19:16] = 0x1 == Capability Version • Bits[15:0] = 0x0018 == Capability ID
3E5-3E4	Max Snoop Latency Register	R/W • Bits[15:13] – Reserved • Bits[12:10] – Max Snoop LatencyScale • Bits[9:0] – Max Snoop LatencyValue
3E7-3E6	Max No-Snoop Latency Register	R/W • Bits[15:13] – Reserved • Bits[12:10] – Max No-Snoop LatencyScale • Bits[9:0] – Max No-Snoop LatencyValue

6. Example Design

The PCIe x1 IP can be generated for two types of designs:

- DMA Design
- Non-DMA Design

The PCIe x1 IP example design for Non-DMA allows you to compile, simulate, and test the PCIe x1 IP on the Lattice evaluation boards as it is both simulation and hardware capable. The example design simulation is not supported in Modelsim OEM and Modelsim Pro. Use the QuestaSim OEM and QuestaSim Pro for simulation.

6.1. Example Design Supported Configuration

The Example Design Supported Configuration is shown in Table 6.1.

Table 6.1. PCIe x1 IP Configuration Supported by the Example Design

PCIe x1 IP User Interface Parameter	PCIe x1 IP Configuration	
	DMA Design	Non-DMA Design
PCIe Link Width	x1	x1
Target Link Speed	Gen1, Gen2	Gen1, Gen2
Use TLP Interface	X	TLP Interface is supported
Data Interface Type	AXI-MM	AXI-Stream AXI-MM Manager
Number of Physical Function	1 is supported (Function 0)	1 is supported (Function 0)
Enable Descriptor ID	X	X
PCIe CSR Base Address (512 kB aligned)	X	0xC5200000
Optional Ports: Enable Clkreq port Enable LTSSM disable port	X	X
Flow Control Tab	Refer to Flow Control Update, Receive Buffer Allocation, and Transmit Buffer Allocation section for the configuration performed in this tab.	Refer to Flow Control Update, Receive Buffer Allocation, and Transmit Buffer Allocation section for the configuration performed in this tab.
Configuration Device ID and Vendor ID Subsystem ID Subsystem Vendor ID Class Code and Revision ID	Device ID and Vendor ID is Configured from APB and rest are default	Device ID and Vendor ID is Configured from APB or LMMI and rest are default
Enable Resizable Bar Capabilities	X	X
BAR 0 Enable	✓	✓
BAR 1 Enable	Depends on DMA Bypass Mode and the BAR assigned to it.	Yes except for when TLP mode is being used.
BAR 3, BAR 4, BAR 5	Depends on DMA Bypass Mode and the BAR assigned to it.	X
Disable Legacy Interrupt	✓	✓

PCIe x1 IP User Interface Parameter	PCIe x1 IP Configuration	
	DMA Design	Non-DMA Design
Disable MSI Capability	X	✓
Disable MSI-X capability	✓	✓
Enable DSN Capability	X	X
Maximum Payload Size Supported	128 bytes, 256 bytes, or 512 bytes	128 bytes, 256 bytes, or 512 bytes
Disable Function Level Reset	✓	✓
Enable Extended Tag Filed	✓	✓
Advance Error Reporting Capability	Refer to the Advance Error Reporting Capability section for the configuration done in this tab.	Refer to the Advance Error Reporting Capability section for the configuration done in this tab.
ATS Capability	Disabled through APB configuration	Disabled through APB/LMMI Configuration

Note: ✓ refers to a checked option in the PCIe X4 IP example design and X refers to an unchecked option or a non-applicable option in the PCIe x1 IP example design.

6.2. Overview of the Example Design and Features

The Example Design contains the PCIe DMA design and PCIe non-DMA design. Using the graphical user interface, you can test the PCIe in any configuration. The testbench adapts and generates the testcases based on the configuration.

You can configure the parameters like the PCIe generation (Gen1 or Gen2), PCIe lane width x1, PCIe DMA enabled or disable, and data interface to be used.

To perform a DMA write or read process, you must select the following parameters:

- PCIe Gen speed
- DMA support enable/disable.

Based on these instructions, the BFM selects the type of testcase that needs to be implemented. If a Gen2x1 PCIe with non-DMA with TLP interface is selected, the BFM sends a testcase that is compatible with the DUT (PCIe Endpoint).

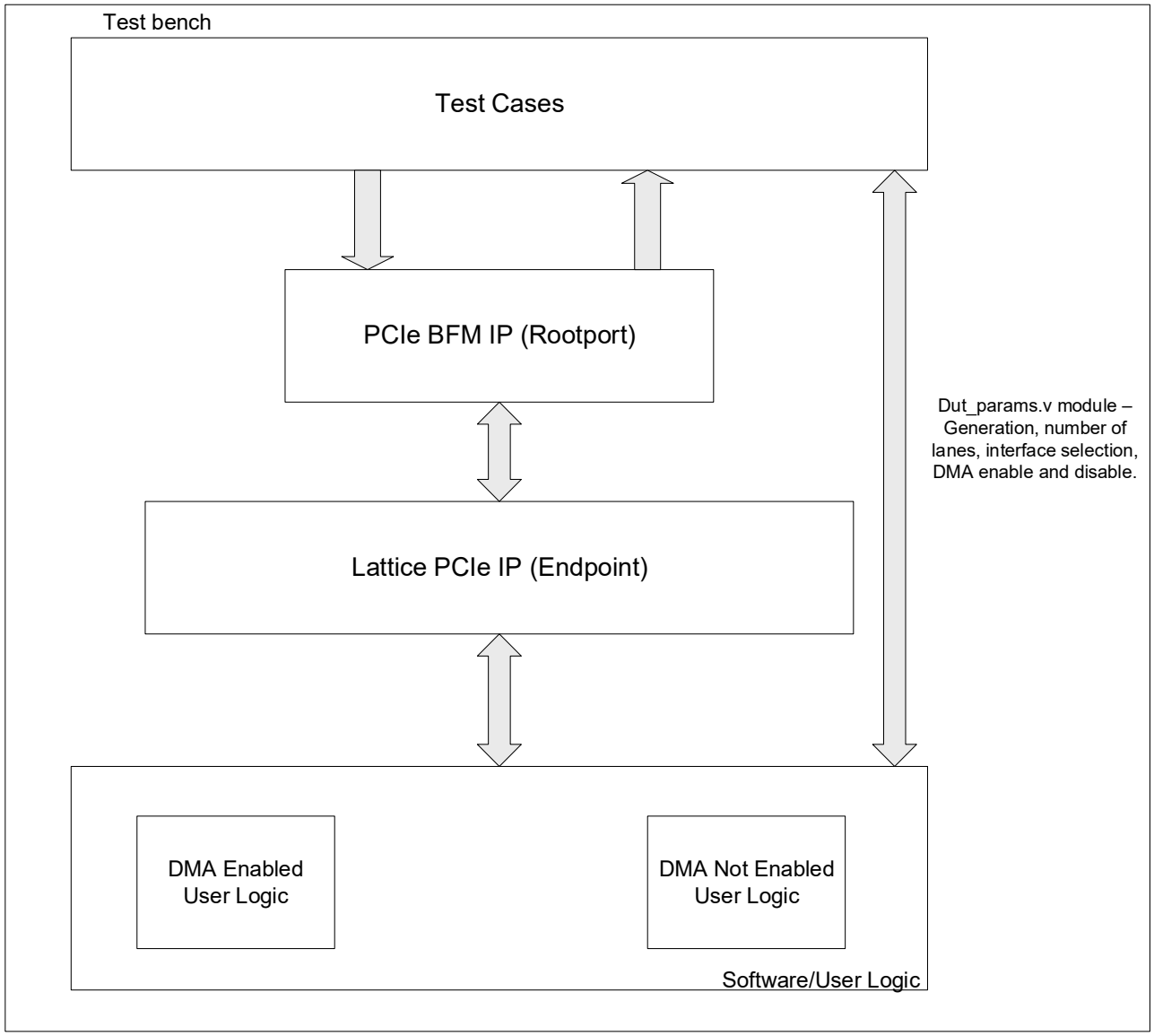

Figure 6.1. PCIe x1 IP Example Design Block Diagram

6.3. Example Design Components

6.3.1. DMA Design (AXI-MM)

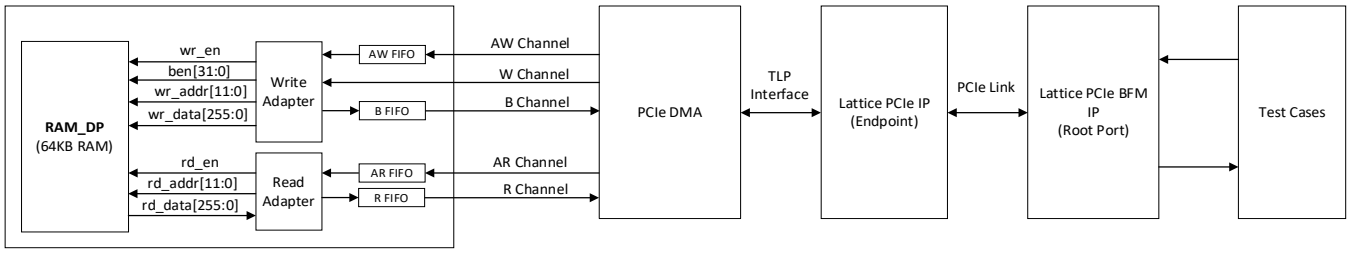

Figure 6.2. Components within AXI-MM DMA Example Design

The PCIe Example Design implements the DMA Design with the following components:

- AR FIFO, B FIFO, AR FIFO, R FIFO – FIFOs that stores information from AXI-MM interface.
- RAM_DP – A True Dual Port RAM that is configured to 128 kB size. It is a storage on FPGA for F2H and H2F data transfer.
- Write Adapter – A component that converts AXI-MM write to RAM_DP write interface.
- Read Adapter – A component that converts AXI-MM read to RAM_DP read interface.
- PCIe DMA – The PCIe IP DMA is used to implement the DMA Operations.

The following shows the DMA Design's data flow:

- Read the configuration of the Lattice PCIe IP DMA
- The BFM waits for the Linkup to occur.
- The BFM setups a single entry of H2F descriptor table with transfer size of 16 kB and with INTR and EOP bits set to 1.
- The BFM programs the PCIe DMA H2F registers to start DMA transfer.
- The BFM waits for MSI interrupt from DUT.
- The BFM setups a single entry of F2H descriptor table with transfer size of 16 kB and with INTR and EOP bits set to 1.
- The BFM programs the PCIe DMA F2H registers to start DMA transfer.
- The BFM waits for MSI interrupt from DUT.

The BFM does data comparison for F2H and H2H to make sure they are intact.

6.3.1.1. Generating the AXI-MM DMA Example Design

To generate the AXI-MM DMA example design:

1. Create a Lattice Radiant software project. Double-click the **PCIE_X1** in the **IP Catalog** and generate the IP with selecting **DMA only Mode** at **Configuration Mode** and select **AXI_MM** at **Data Interface Type** drop-down menu.
2. Configure **Target Link Speed**.
3. Right-click on Input Files and select **Add > Existing Files**.
4. Add **<Component Name>/testbench/example_design_top.sv**.

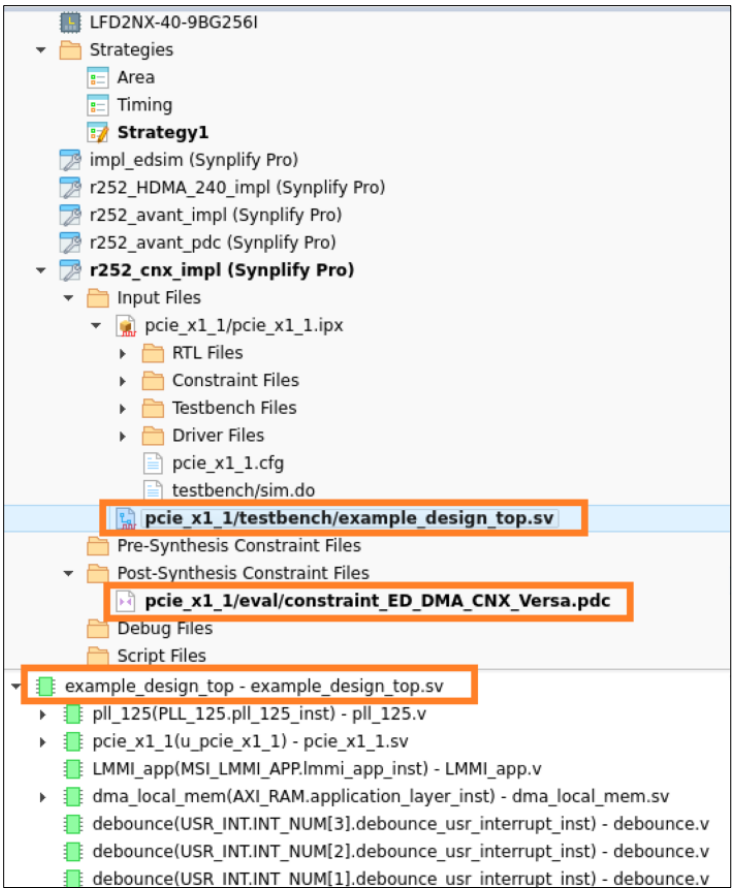

Figure 6.3. File List View of the Created AXI-MM DMA Example Design

5. Right-click on **Post-Synthesis Constraint Files**.
6. Add **<Component Name>/eval/constraint_ED_DMA_CNX_Versa.pdc** or **constraint_ED_DMA_MachXO5_Versa.pdc**
7. Proceed to the Radiant flow if the hierarchical view shows **example_design_top** as the top module.

6.3.2. Non-DMA Design (Bridge Mode)

The PCIe x1 Example Design implements the Non-DMA Design (Bridge Mode) with the following components:

- Write Adapter – Convert AXI Write to the pmi_fifo write interface.
- Read Adapter – Convert AXI Read to the pmi_fifo read interface.
- RAM_DP – pmi_fifo that contains EBR-based RAM.

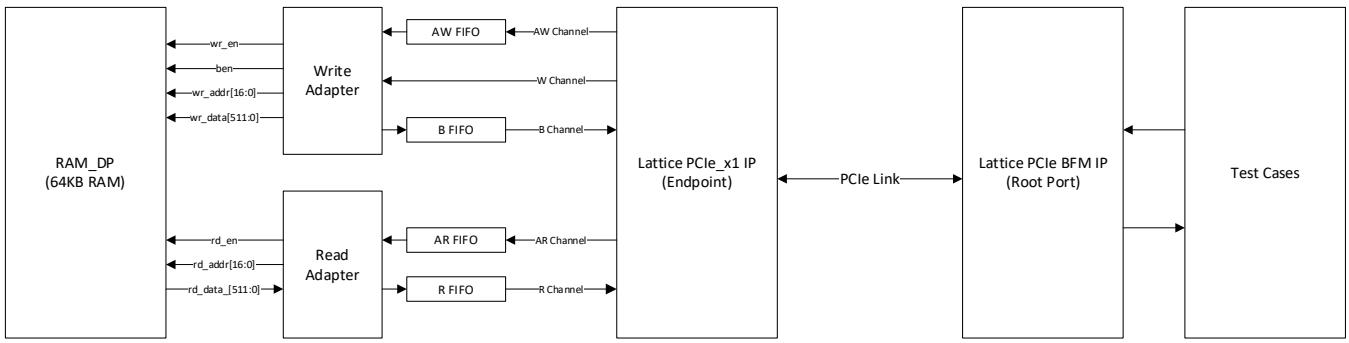

Figure 6.4. Components within NON-DMA Design (Bridge Mode)

The following shows the Non-DMA (Bridge Mode) data flow:

- Reading the configuration of Lattice PCIe x1 IP
- BFM performs the linkup with PCIe IP.
- Once linkup is done, the BFM sends the header info of the data packet to the PCIe whether to write/read into/from the BAR1 location of the PCIe.
- PCIe sends the packet information to application layer via AXI Write or AXI Read Channels, which the *Write Adapter*/Read Adapter decodes AXI Write/Read Channel and performs write/read operation accordingly.
- For write operations, the data is written into RAM which stores the data received.
- For read operations, the data is read from RAM and sent to the IP.

6.3.2.1. Generating the Bridge Mode Example Design

To generate the Bridge Mode example design:

1. Create a Lattice Radiant software project. Double-click the **PCIE_X1** in the **IP Catalog** and generate the IP by selecting **Bridge Mode** in **Configuration Mode** drop-down menu (see Figure 2.34 and Figure 2.35 for the settings).
2. Right-click on Input Files and select **Add > Existing Files**.
3. Add **<Component Name>/testbench/example_design_top.sv**.

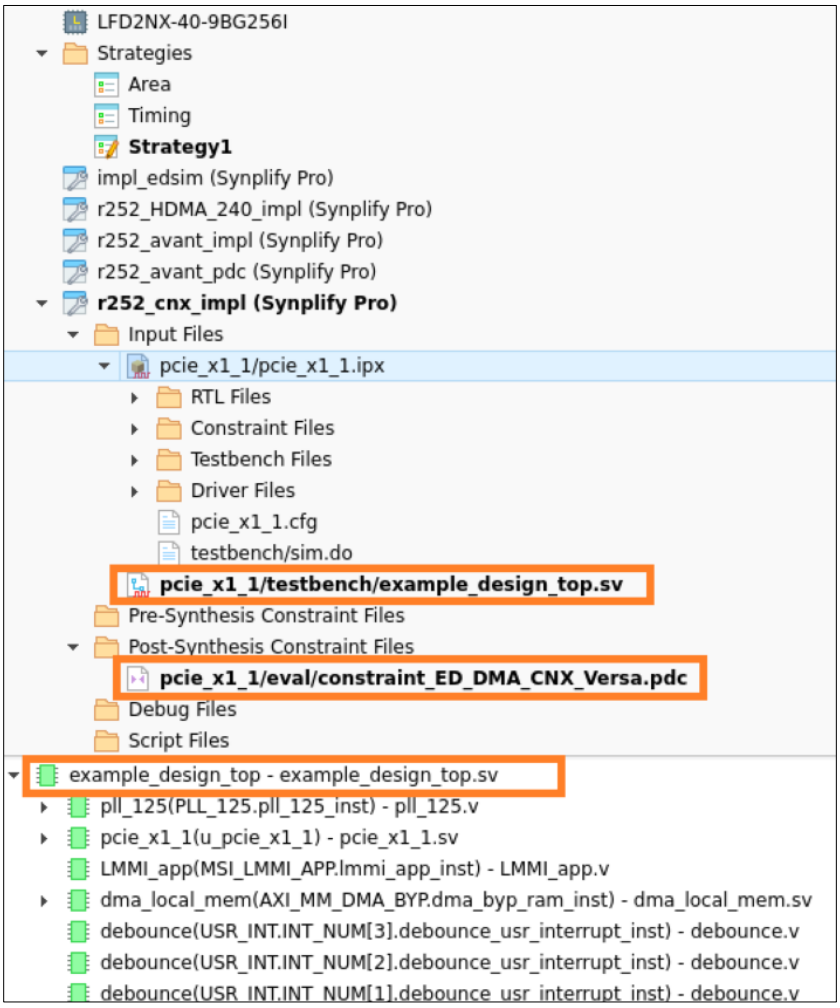

Figure 6.5. File List View of the Created Bridge Example Design

4. Right-click on **Post-Synthesis Constraint Files**.
5. Add **<Component Name>/eval/constraint_ED_NDMA_CNX_Versa.pdc** or **constraint_ED_DMA_MachXO5_Versa.pdc**.
6. Proceed to the Radiant flow if the hierarchical view shows **example_design_top** as the top module.

Most of the above steps are expected to be same as the steps to generate AXI-MM DMA example design.

6.3.3. Non-DMA Design (TLP Interface)

The PCIe x1 Example Design implements the Non-DMA Design with the following components:

- PCIe_rx_engine – The received TLPs on the Rx TLP Interface is decoded in this block. For write(posted) operations, the received TLP data is sent to the *pcie_ep_mem* block to store this data into a RAM. For read(non-posted) operations, the received TLP header information is sent to the *pcie_tx_engine* block for the completion TLP.
- PCIe_tx_engine – The transmitted TLPs on Tx TLP Interface managed by this block. This block sends out the completion packets in response to the received non-posted TLP packets to meet the PCIe specification requirements. For example, in case of memory read TLP type packet. The required header information for the completion packet is received from the *pcie_rx_engine*. The data payload is read from the *pcie_ep_mem* block for transmitting along with the completion header.
- PCIe_ep_mem – The *PCIe_ep_mem* module receives the instructions from *rx_engine*, whether the data is written or read.This module consist of a RAM ,which is used to store the received data. The design consists of two bars (BAR 0 and BAR 1) enabled in the PCIe endpoint. Based on the address of the BAR, the RAM writes/reads the data from/to the bar specified.

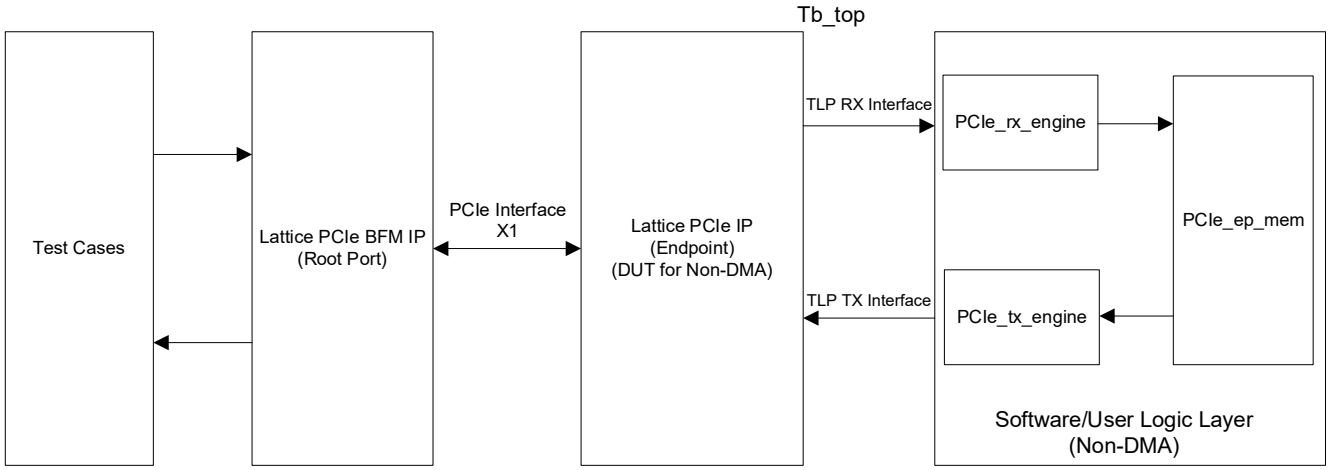

Figure 6.6. Components within Non-DMA Design (TLP Interface)

The following shows the Non-DMA data flow:

- Reading the configuration of Lattice PCIe x1 IP
- BFM performs the linkup with PCIe IP.
- Once linkup is done, the BFM sends the header info of the data packet to the PCIe whether to write/read into/from the preferred BAR location of the PCIe.
- PCIe sends the packet information to application layer, which the *pcie_rx_engine* decodes the header data and performs read/write operation accordingly.
- For write operations, the data is written into RAM which stores the data received.
- For read operations, the data is read from RAM and sent to the PCIe along with the header information of the packet.

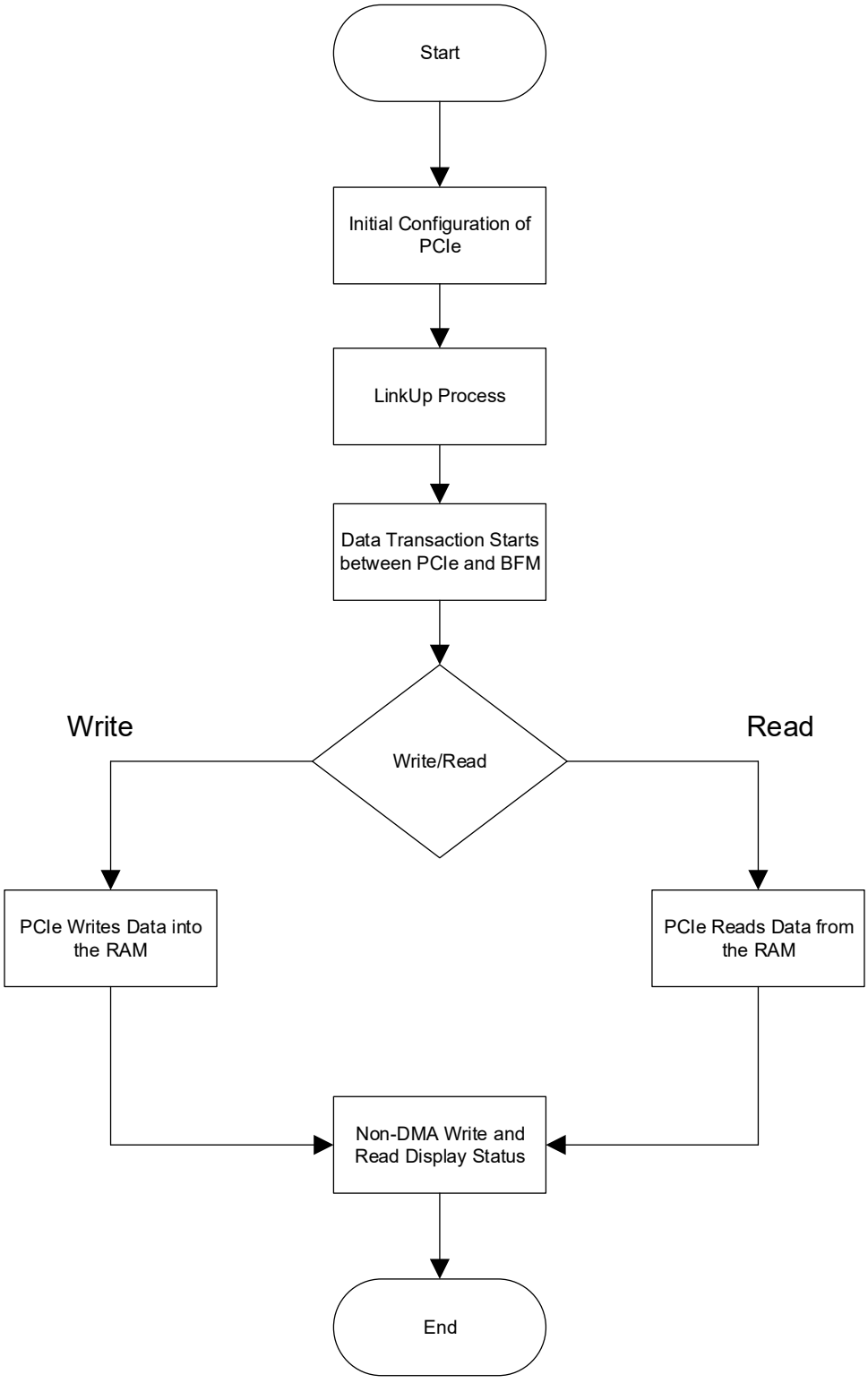

Figure 6.7. Non-DMA Design Data Flow

6.3.3.1. Generating the TLP Interface Example Design

To generate the TLP Interface example design:

1. Create a Lattice Radiant software project. Double-click the **PCIE_X1** in the **IP Catalog** and generate the IP with your desired choice of PCIe generation support, bifurcation and ensure that the **Configuration Mode** being chosen is **TLP Mode**.
2. Right-click on Input Files and select **Add > Existing Files**.
3. Add Add **<Component Name>/testbench/NON_DMA/example_design_tlp_top.sv**.

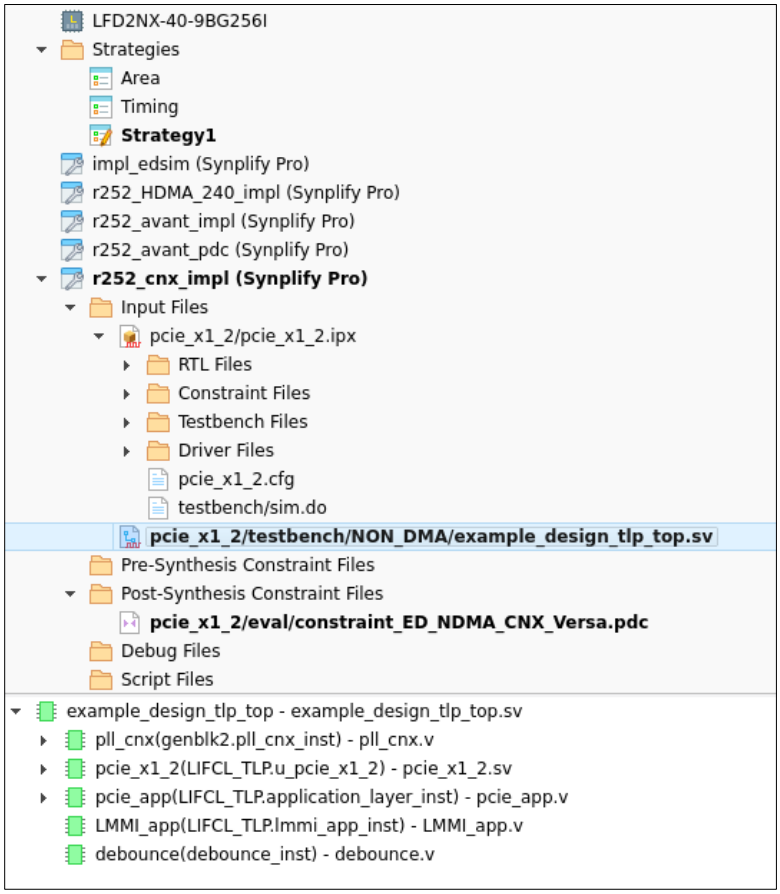

Figure 6.8. File List View of the Created TLP Interface Example Design

4. Right-click on **Post-Synthesis Constraint Files**.
5. Add **<Component Name>/eval/constraint_ED_NDMA_CNX_Versa.pdc** or **constraint_ED_NDMA_MachXO5_Versa.pdc**.
6. Proceed to the Radiant flow if the hierarchical view shows **example_design_tlp_top** as the top module.

6.3.4. PDC Settings for Hardware Example Design

When using the SPI Flash programming on the Certus-NX and MachXO5-NX board, the following system configuration constraints must be applied :-

- For Certus-NX Versa board:
```
ldc_set_sysconfig {JTAG_PORT=ENABLE PROGRAMN_PORT=ENABLE BOOTMODE=SINGLE
MASTER_SPI_PORT=SERIAL CONFIG_SECURE=OFF CONFIG_IOSLEW=FAST MCCLK_FREQ=28.1}
```
- For MachXO5-NX board:
```
ldc_set_sysconfig {CONFIG_IOSLEW=FAST FLASH_CLK_FREQ=56.2 JTAG_PORT=ENABLE
PROGRAMN_PORT=ENABLE BOOTMODE=SINGLE COMPRESS_CONFIG=ON}
```

6.4. Simulating the Example Design

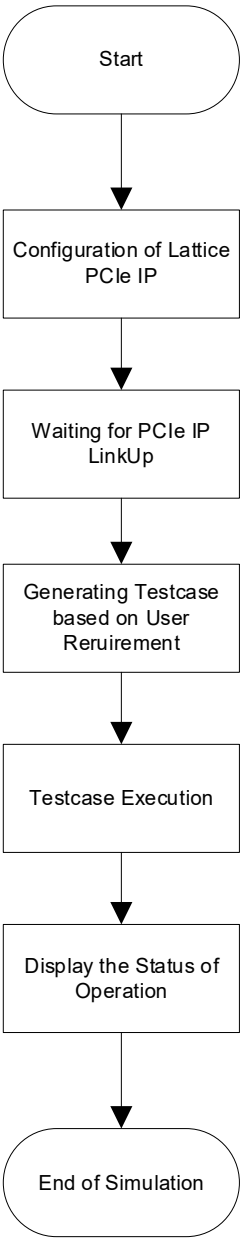

Figure 6.9. PCIe x1 IP Example Design Flowchart

The Example Design can run in simulation as follows:

1. Generate the PCIe x1 IP with the required configuration. Some of the configurations of PCIe can be done through the APB interface for DMA Design or through LMMI for the Non-DMA Design. The Testbench then waits for the linkup to occur.
2. Enumeration is started and wait for completion.
3. The BFM waits for the PCIe to link up.
4. The BFM starts sending the testcase based on the user requirement.
5. The status of the testcase is displayed as PASS or FAIL.

Functional Simulation can be performed after the IP is generated through the Example Design testbench. For more details on the Example Design configuration and test cases, refer to the Example Design Supported Configuration section.

6.4.1. QuestaSim Lattice-Edition

To run the functional simulation in QuestaSim Lattice-Edition:

1. Create a new Radiant project, select the target device that supported PCIE_X1 IP.
2. Select **IP on Server** and install the latest version of PCIE_X1 IP if not already installed.
3. Switch to **IP on Local**, double-click **PCIE_X1** and enter your desired **Component name.**

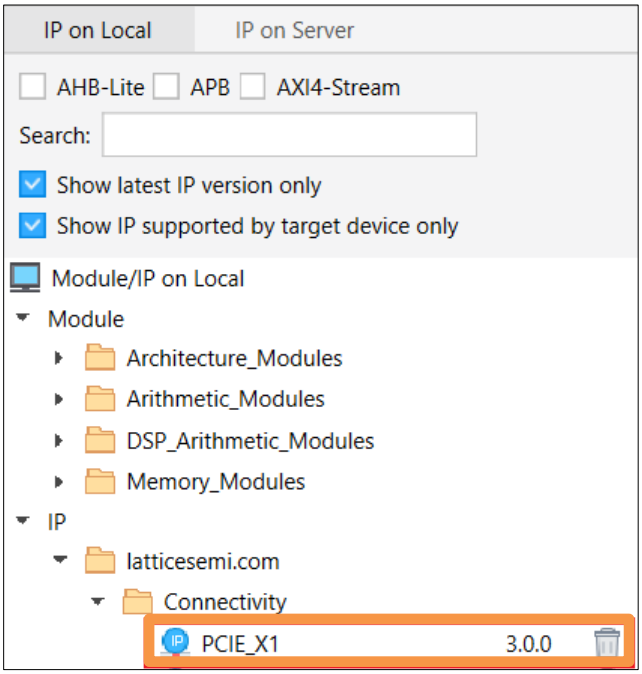

Figure 6.10. IP on Local

4. Parameterize the PCIE_X1, for simulation purpose, it is important to tick the **Simulation Reduce Timeout** option. Other parameters can be left as default (or changed). Click **Generate** and **Finish**.

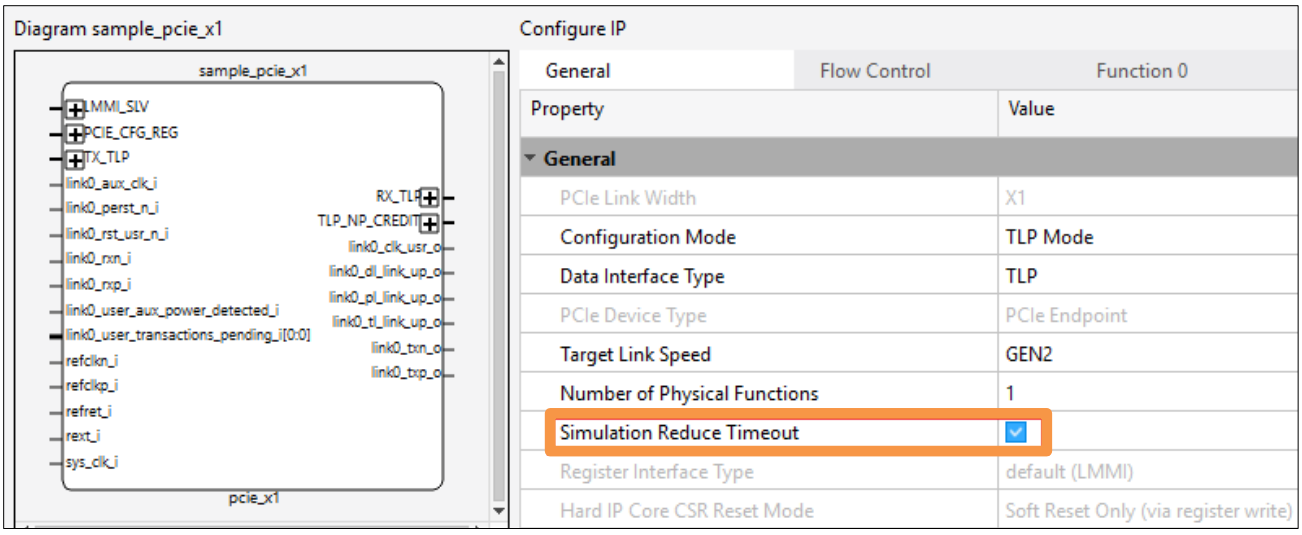

Figure 6.11. Parameterize the PCIE_X1

5. Make sure that the testbench files are generated during PCIE x1 IP generation.

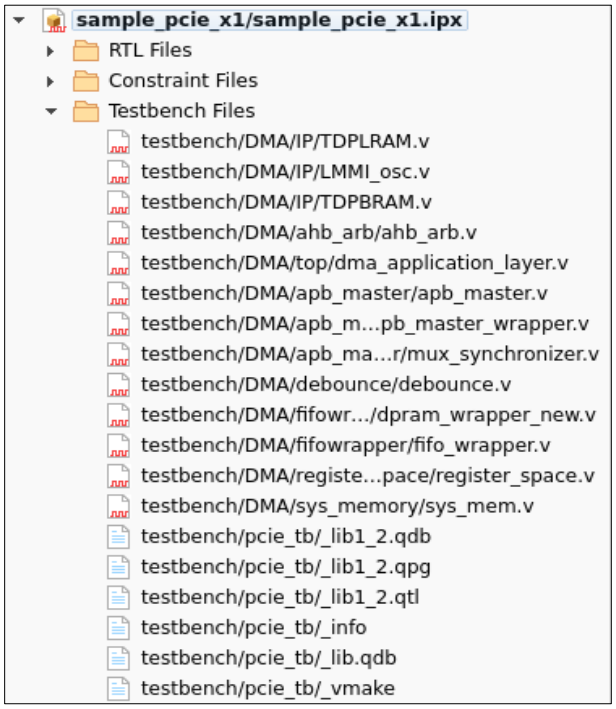

Figure 6.12.Testbench Files

6. Click the  icon to initiate the Simulation Wizard and create a new simulation project.
7. Name the project.

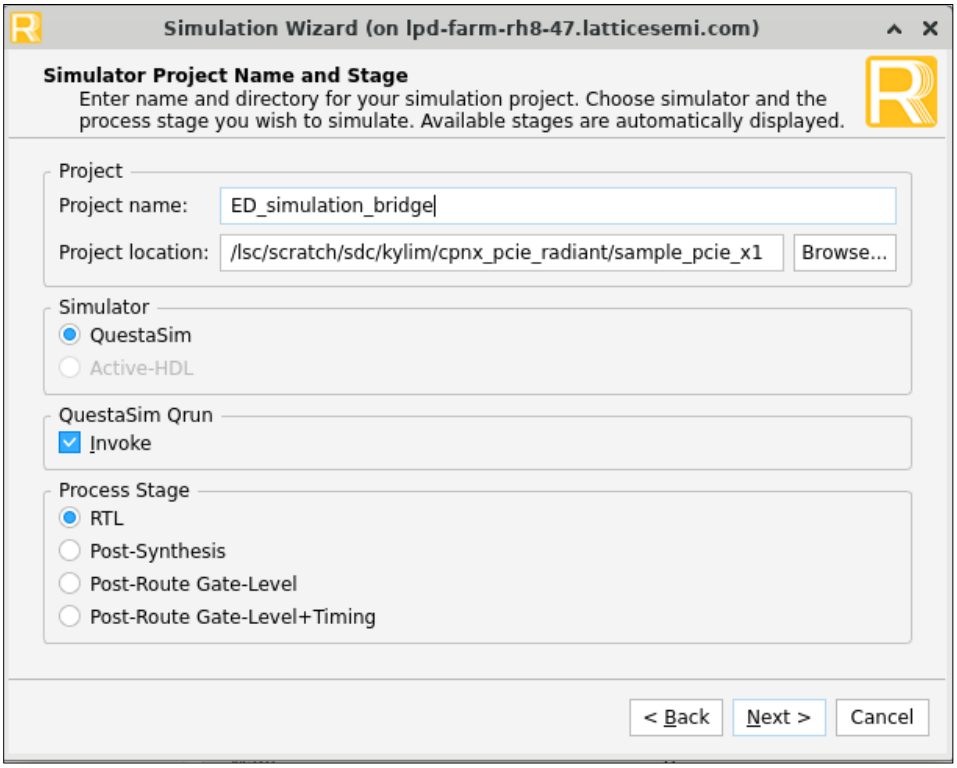

Figure 6.13. Project Naming

8. Select **tb_top** as Simulation Top Module.

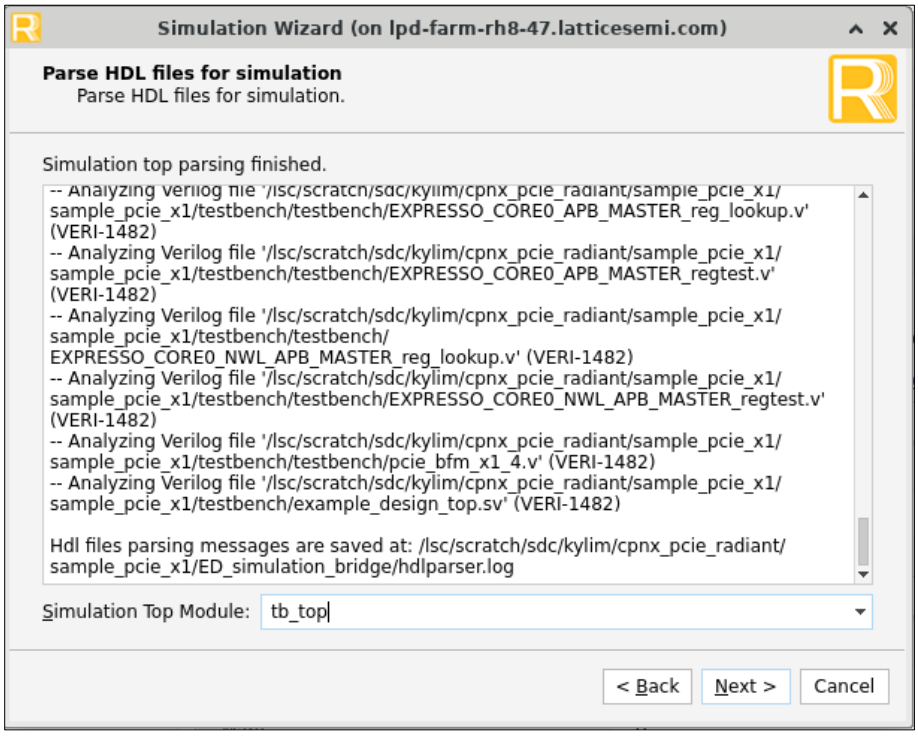

Figure 6.14. Simulation Top Module

9. Use the following simulation settings. Default Run set to **0** is required.

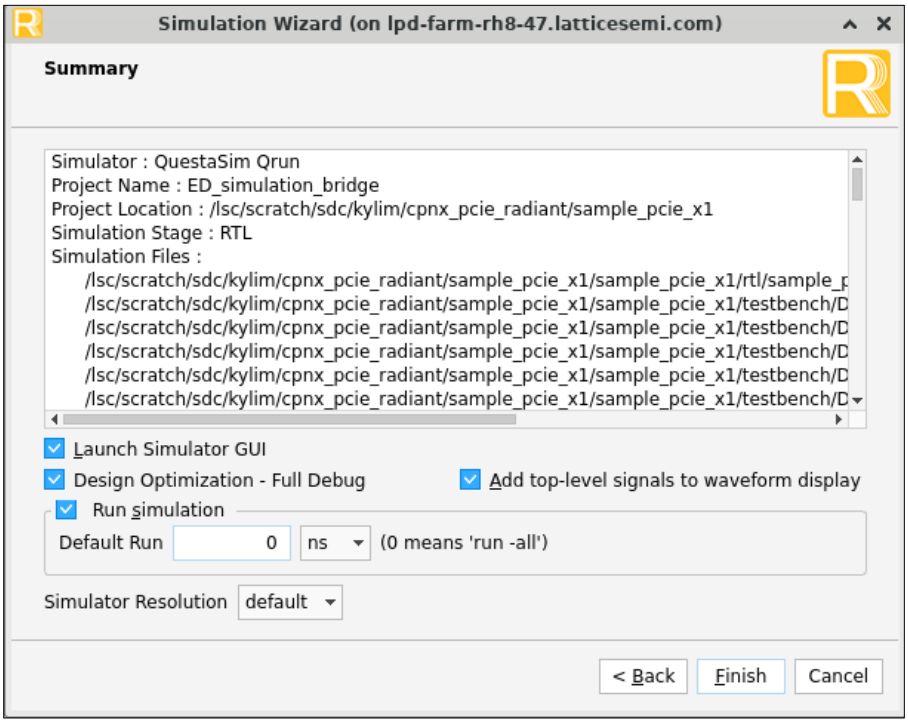

Figure 6.15. Simulation Setting

10. Once simulation is completed, below log printing should be observed.

```
#
# 169564550 INFO: tb_top.ref_design_ts: DUT used the following tags[255:0]: 0x0000000000000000000000000000000000000000000000000000000000000000
# 169564550 INFO: tb_top.ref_design_ts: ######## SIMULATION COMPLETE ########
# 169564550 INFO: tb_top.report_status: SIMULATION STATUS: ALL TESTS PASS
# ** Note: $stop    : /lsc/scratch/sdc/kylim/cpnx_pcie_radiant/sample_pcie_x1/sample_pcie_x1/testbench/testbench/tb_top.v(1791)
#    Time: 169564550 ps  Iteration: 5  Instance: /tb_top
# Break in Module tb_top at /lsc/scratch/sdc/kylim/cpnx_pcie_radiant/sample_pcie_x1/sample_pcie_x1/testbench/testbench/tb_top.v line 1791

VSIM 7>
```

Figure 6.16. Expected Log Printing

11. The simulation run completion is indicated by *SIMULATION STATUS of ALL TESTS PASS* or *ERRORS DETECTED* in the QuestaSim transcript window. Figure 6.17 shows the example of a simulation waveform.

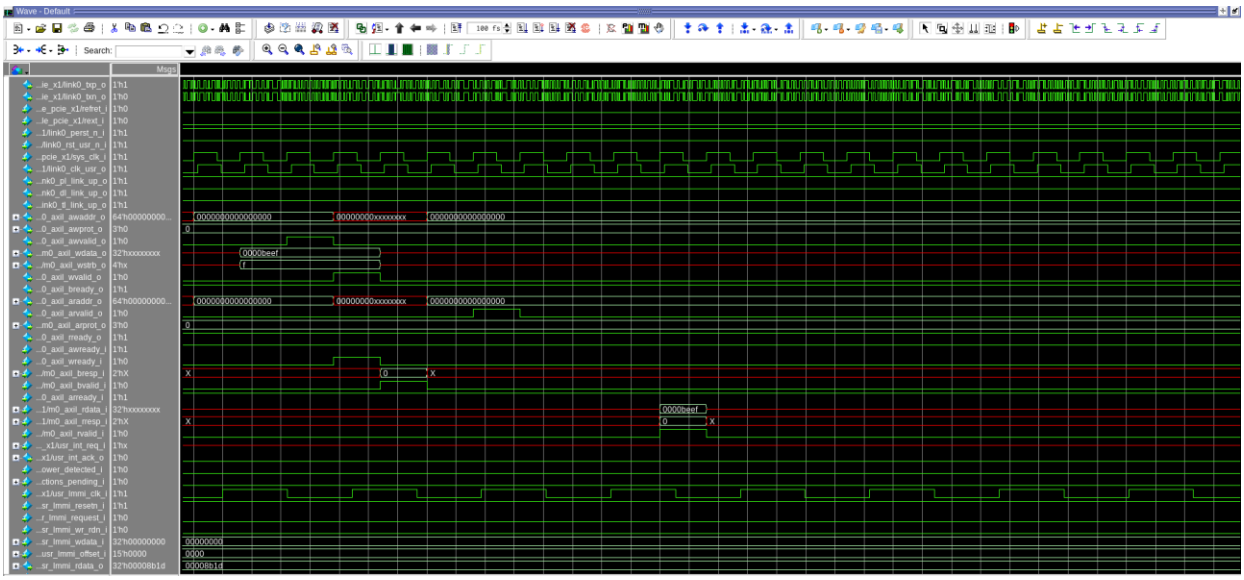

Figure 6.17. Simulation Waveform

6.4.2. QuestaSim Pro

To run the functional simulation in QuestaSim Pro:

1. Make sure that the testbench files are generated during PCIe x1 IP generation.

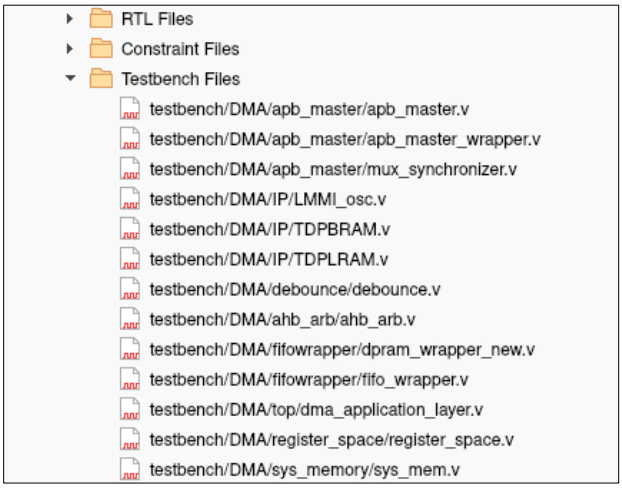

Figure 6.18. Testbench Files

2. Click the  icon to initiate the Simulation Wizard and create a new simulation project.
3. Name the project.

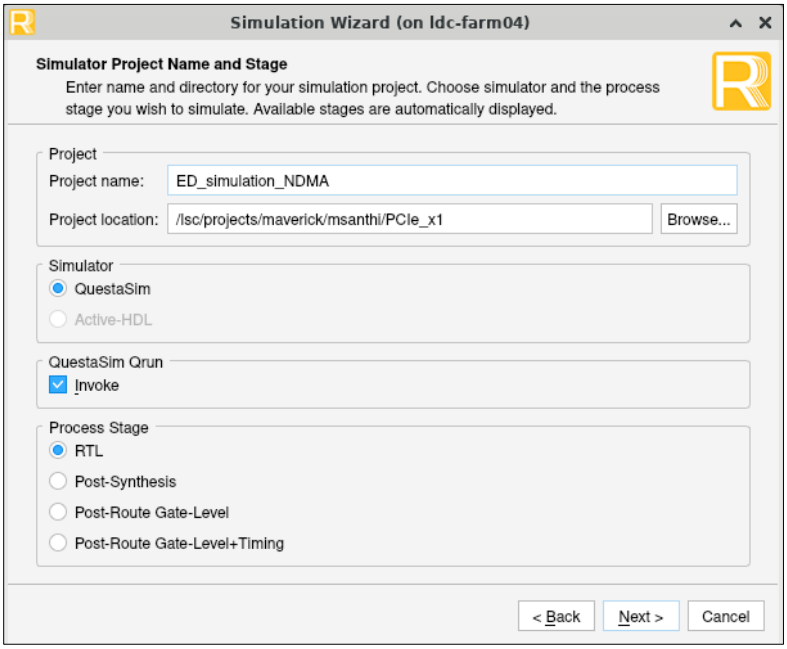

Figure 6.19. Project Naming

4. Select **tb_top** as *Simulation Top Module*.

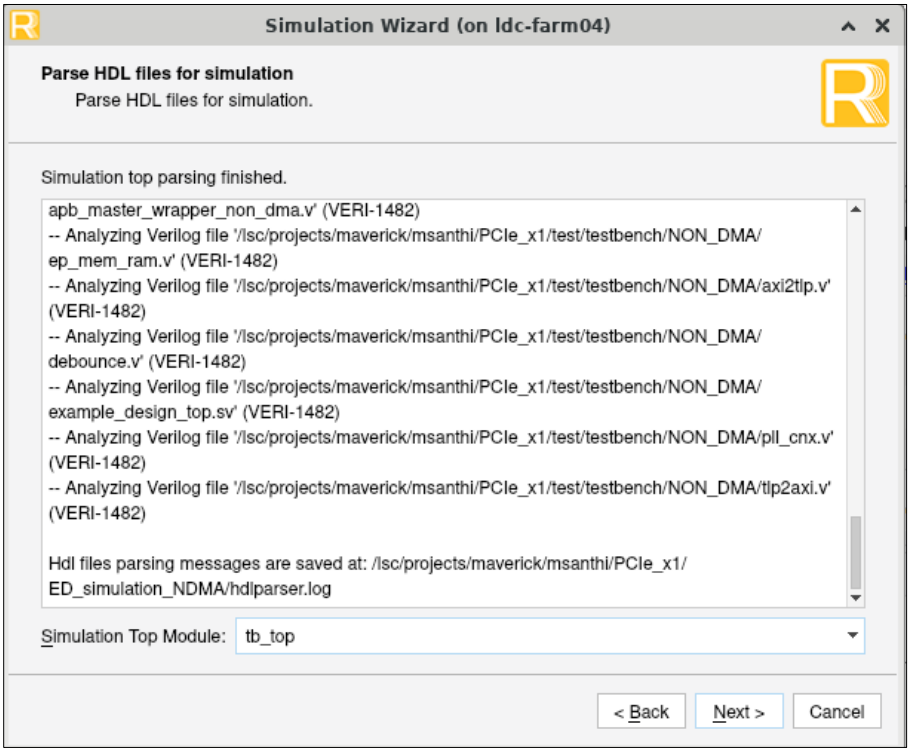

Figure 6.20. Simulation Top Module

5. Use the following simulation settings. Untick *Run simulation* option.

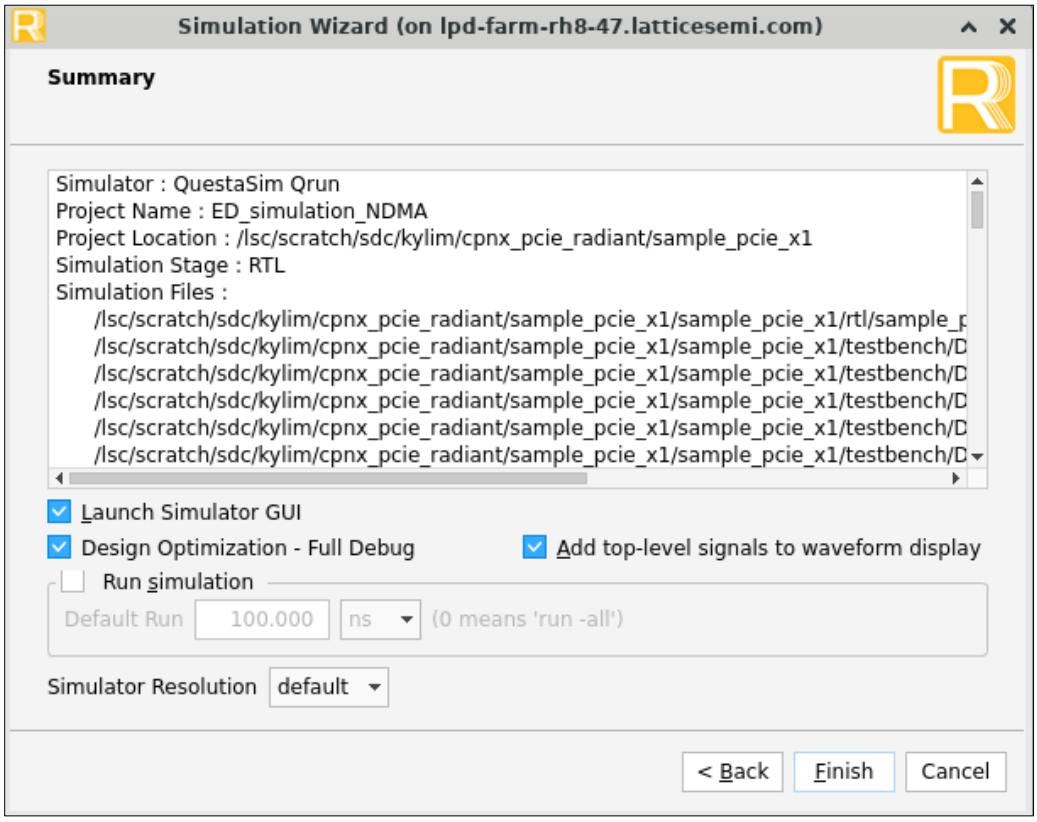

Figure 6.21. Simulation Setting

6. QuestaSim Lattice-Edition is being launched to perform design compilation. Proceed to close QuestaSim Lattice-Edition window once design compilation is completed.

Figure 6.22. Transcript Log Printing

7. Open the *<project>.f* and then update the *-reflib* to following:

   For Linux:

   ```
   -reflib <radiant_installation_directory>/cae_library/simulation/libs/pmi_work
   -reflib <radiant_installation_directory>/cae_library/simulation/libs/lifcl
   ```

   Example for Radiant Installation directory: /home/rel/ng2025_2

   For Windows:

   ```
   -reflib C:/lscc/radiant/2025.2/cae_library/simulation/libs/pmi_work
   -reflib C:/lscc/radiant/2025.2/cae_library/simulation/libs/lifcl
   ```

8. Update the *<project>.f* file to include the BFM files.

   ```
   "<project_path>/testbench/testbench/pcie_model_x1_4.v"
   "<project_path>/testbench/testbench/pcie_bfm_x1_4.v"
   ```

9. Add the following simulation run command at the end of the line in *<project>.f* .

   ```
   -do "run -all"
   ```

10. In the *<project>.f* file, remove the following lines:

    ```
    +noacc+/lsc/scratch/sdc/kylim/cpnx_pcie_radiant/sample_pcie_x1/sample_pcie_x1/testbench/pcie_tb
    -reflib pcie_tb
    ```

11. In the *<project>.vdo* file, remove the following line:

    ```
    -do "<project_path>/testbench/sim.do"
    ```

12. Set up the environment variable for FOUNDRY before launching the simulator.

    For Linux:

    a. In the terminal used to launch the simulator, set the environment variable as follows:

    ```
    setenv FOUNDRY <radiant_installation_directory>/cae_library
    Example: setenv FOUNDRY /home/rel/ng2025_2/cae_library
    ```

    For Windows:

    a. Run the Command Prompt as Admin, then set the FOUNDRY as shown below.

    ```
    set FOUNDRY=<radiant_installation_directory>/ispfpga
    ```

    Example: set FOUNDRY=C:/lscc/radiant/2025.2/ispfpga

    a. After that, run the QuestaSim Pro full version.

13. Launch full license QuestaSim Pro. Make sure QuestaSim is in the correct project directory and run the following qrun command.

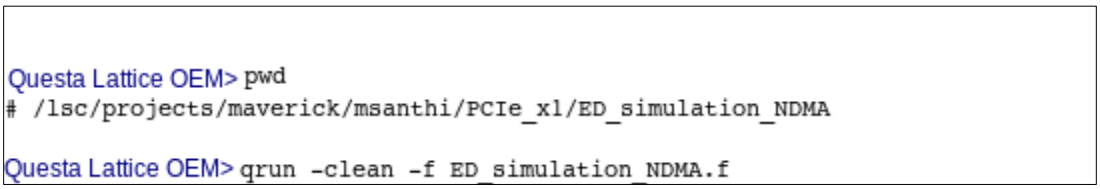

Figure 6.23. Command of Full License QuestaSim Pro

14. Once simulation is completed, the log printing below must be observed.

```
# 169045850 INFO: tb_top.gen_pcie_bfm[0].pcie_bfm.receive_initial_block: U :   Data (Tag 0x75) == 0x0000aa10
# 169045851 INFO: tb_top.ref_design_ts.configure_pci.legi_cap: Configuring Legacy Interrupts of all discovered functions
# 169045851 INFO: tb_top.ref_design_ts.configure_pci.legi_cap: Bus[  0], Dev[ 0], Func[0] : Legacy Int : MSI Interrupts have already been allocated to this function; skipping
# 169045851 INFO: tb_top.ref_design_ts.configure_pci.legi_cap: Bus[  1], Dev[ 0], Func[0] : Legacy Int : MSI Interrupts have already been allocated to this function; skipping
# 169045851 INFO: tb_top.ref_design_ts.configure_pci: Enumeration Completed
# 169425850 INFO: tb_top.ref_design_ts.single_read_write_test: **** Begin test SINGLE_READ_WRITE_TEST ****
# 169425850 INFO: tb_top.ref_design_ts.single_read_write_test: do_test : 1
# 169425850 INFO: tb_top.ref_design_ts.single_read_write_test: ** SINGLE Write Read Tests(testing BAR          0) **
# 169425850 INFO: tb_top.ref_design_ts.single_read_write_test: ** Alternate DWORD write read Test **
# 169425850 INFO: tb_top.ref_design_ts.single_read_write_test: Single DWORD Write
# 169425851 INFO: tb_top.gen_pcie_bfm[0].pcie_bfm.xfer_action: D : MWr32 -- 0000 ---- 001 -------- 81f80000 0 f - -- --- -- -- - --- --     169425
# 169431550 INFO: tb_top.gen_pcie_bfm[0].pcie_bfm.xfer_action: D :   Data == 0x00010000
# 169433450 INFO: tb_top.ref_design_ts.single_read_write_test: Single DWORD Read
# 169623451 INFO: tb_top.gen_pcie_bfm[0].pcie_bfm.xfer_action: D : MRd32 76 0000 ---- 001 -------- 81f80000 0 f - -- --- -- -- - --- --     169623
# 170448050 INFO: tb_top.gen_pcie_bfm[0].pcie_bfm.receive_initial_block: U : CplD  76 0000 0064 001 -------- -------- - - 0 00 004 -- -- - --- --     170448
# 170449950 INFO: tb_top.gen_pcie_bfm[0].pcie_bfm.receive_initial_block: U :   Data (Tag 0x76) == 0x00010000
# 170829950 INFO: tb_top.ref_design_ts.single_read_write_test: **** End test SINGLE_READ_WRITE_TEST ****
# 171779950 INFO: tb_top.ref_design_ts: DUT used the following tags[255:0]: 0x00000000000000000000000000000000000000000000000000000000000000000000000000000000
# 171779950 INFO: tb_top.ref_design_ts: ######## SIMULATION COMPLETE ########
# 171779950 INFO: tb_top.report_status: SIMULATION STATUS: ALL TESTS PASS
# ** Note: $stop    : /lsc/projects/maverick/msanthi/PCIe_x1/test/testbench/testbench/tb_top.v(1589)
#    Time: 171779950 ps  Iteration: 5  Instance: /tb_top
# Break in Module tb_top at /lsc/projects/maverick/msanthi/PCIe_x1/test/testbench/testbench/tb_top.v line 1589
```

Figure 6.24. Expected Log Printing

15. The simulation run completion is indicated by *SIMULATION STATUS of ALL TESTS PASS* or *ERRORS DETECTED* in the QuestaSim transcript window. Figure 6.25 shows the example of a simulation waveform.

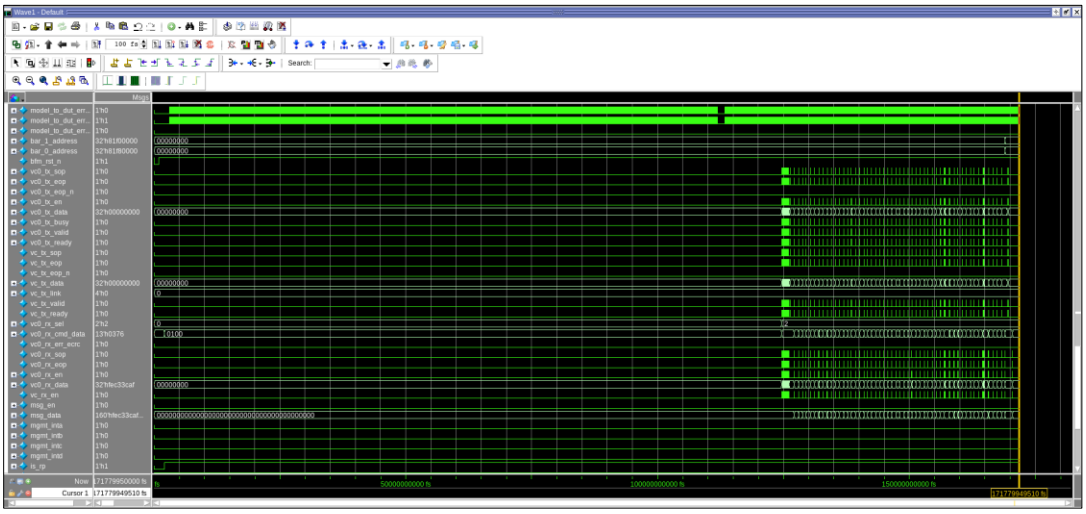

Figure 6.25. Simulation Waveform

6.5. Debugging Example Design Issues

6.5.1. Signals to Debug

6.5.1.1. Simulation Debug for DMA (AXI-MM) Design

Table 6.2. AXI-MM DMA Signals to Debug Description

Module Name	Signal Name	Description
tb_top	link0_pl_link_up_o	PCIe IP Physical Layer linkup
tb_top	link0_dl_link_up_o	PCIe IP Data Link Layer linkup
tb_top	link0_tl_link_up_o	PCIe IP Transaction Layer linkup
Host-to-FPGA		
tb_top	m0_dma_axi_awaddr_o	Write address. The write address gives the address of the first transfer in a write burst transaction.

Module Name	Signal Name	Description
tb_top	m0_dma_axi_awlen_o	Burst length. The burst length gives the exact number of transfers (beat) in a burst. This information determines the number of data transfers associated with the address. 0x00 – 1 beat 0x01 – 2 beats … 0xFF – 256 beats
tb_top	m0_dma_axi_awvalid_o	Write address valid. This signal indicates that the channel is signaling valid write address and control information.
tb_top	m0_dma_axi_awready_i	Write address ready. This signal indicates that the subordinate is ready to accept an Address and associated control signals.
tb_top	m0_dma_axi_wdata_o	Write data.
tb_top	m0_dma_axi_wstrb_o	Write strobes. This signal indicates which byte lanes hold valid data. There is one write strobe bit for each eight bits of the write data bus.
tb_top	m0_dma_axi_wlast_o	Write last. This signal indicates the last transfer in a write burst.
tb_top	m0_dma_axi_wready_i	Write ready. This signal indicates that the subordinate can accept the write data.
tb_top	m0_dma_axi_bresp_i	Write response. This signal indicates the status of the write transaction.
tb_top	m0_dma_axi_bvalid_i	Write response valid. This signal indicates that the channel is signaling a valid write response.
tb_top	m0_dma_axi_bready_o	Response ready. This signal indicates that the manager can accept a write response.
FPGA-to-Host		
tb_top	m0_dma_axi_araddr_o	Read address. The read address gives the address of the first transfer in a read burst transaction.
tb_top	m0_dma_axi_arlen_o	Burst length. The burst length gives the exact number of transfers (beat) in a burst. This information determines the number of data transfers associated with the address. 0x00 – 1 beat 0x01 – 2 beats … 0xFF – 256 beats
tb_top	m0_dma_axi_arvalid_o	Read address valid. This signal indicates that the channel is signaling valid read address and control information.
tb_top	m0_dma_axi_arready_i	Read address ready. This signal indicates that the subordinate is ready to accept an Address and associated control signals.
tb_top	m0_dma_axi_rdata_i	Read data.
tb_top	m0_dma_axi_rresp_i	Read response. This signal indicates the status of the read transfer.
tb_top	m0_dma_axi_rlast_i	Read last. This signal indicates the last transfer in a read burst.
tb_top	m0_dma_axi_rvalid_i	Read valid. This signal indicates that the channel is signaling the required read data.
tb_top	m0_dma_axi_rready_o	Read ready. This signal indicates that the manager can accept the read data and response information.

6.5.1.2. Simulation Debug for Non-DMA (Bridge Mode) Design

Table 6.3. AXI-Lite Bridge Mode to Debug Description

Module Name	Signal Name	Description
tb_top	link0_pl_link_up_o	PCIe IP Physical Layer linkup
tb_top	link0_dl_link_up_o	PCIe IP Data Link Layer linkup
tb_top	link0_tl_link_up_o	PCIe IP Transaction Layer linkup
AXI-Lite		
tb_top	m0_axil_awaddr_o	Write address. The write address gives the address in a write transaction.
tb_top	m0_axil_awvalid_o	Write address valid. This signal indicates that the channel is signaling valid write address.
tb_top	m0_axil_awready_i	Write address ready. This signal indicates that the subordinate is ready to accept an Address.
tb_top	m0_axil_wdata_o	Write data.
tb_top	m0_axil_wstrb_o	Write strobes. This signal indicates which byte lanes hold valid data. There is one write strobe bit for each eight bits of the write data bus.
tb_top	m0_axil_wready_i	Write ready. This signal indicates that the subordinate can accept the write data.
tb_top	m0_axil_bresp_i	Write response. This signal indicates the status of the write transaction.
tb_top	m0_axil_bvalid_i	Write response valid. This signal indicates that the channel is signaling a valid write response.
tb_top	m0_axil_bready_o	Response ready. This signal indicates that the manager can accept a write response.
tb_top	m0_axil_araddr_o	Read address. The read address gives the address in a read transaction.
tb_top	m0_axil_arvalid_o	Read address valid. This signal indicates that the channel is signaling valid read address.
tb_top	m0_axil_arready_i	Read address ready. This signal indicates that the subordinate is ready to accept an Address.
tb_top	m0_axil_rdata_i	Read data.
tb_top	m0_axil_rresp_i	Read response. This signal indicates the status of the read transfer.
tb_top	m0_axil_rvalid_i	Read valid. This signal indicates that the channel is signaling the required read data.
tb_top	m0_axil_rready_o	Read ready. This signal indicates that the manager can accept the read data and response information.
User Interrupt		
tb_top	usr_int_req_i	Request by application logic to trigger interrupt to the Host via the IP.
tb_top	usr_int_ack_o	Acknowledgement by the IP with respect to the request from signal usr_int_req_i.

6.5.1.3. Simulation Debug for Non-DMA (TLP Interface) Design

Table 6.4. Non-DMA Signals to Debug Description

Module Name	Signal Name	Description
tb_top	lmmi_offset	Lower 17-bit address of LMMI interface registers
tb_top	lmmi_wdata	Data written into PCIe IP through LMMI interface
tb_top	lmmi_rdata	Data read from PCIe IP through LMMI interface
tb_top	pl_link_up	PCIe IP physical layer linkup
tb_top	dl_link_up	PCIe IP data layer linkup
tb_top	tl_link_up	PCIe IP transaction layer linkup

The following are the steps to debug the non-DMA design:

- Check whether the initial configuration is performed properly. You can check the *lmmi_offset*, *lmmi_wdata*, and *lmmi_rdata* signals to verify. Note that in the actual application, the register configuration may not be necessary if the corresponding register is configured through the IP Catalog's Module/IP wizard.
- The linkup signals such as *pl_link_up*, *dl_link_up*, and *tl_link_up* must be asserted.

7. Designing with the IP

This section provides information on how to generate and customize the Lattice PCIe x1 Core using the IP generation wizard of the Lattice Radiant Software. For more details on the Lattice Radiant Software, refer to the Lattice Radiant Software User Guide.

Note: The screenshots provided are for reference only. Details may vary depending on the version of the IP or software being used. If there have been no significant changes to the user interface, a screenshot may reflect an earlier version of the IP.

7.1. Instantiating the IP Core

The Lattice PCIe x1 Core is available for download from the Lattice IP server through the IP Catalog of Lattice Radiant Software.

1. Open Lattice Radiant Software Program and create a new project. (Refer to Lattice Radiant Software User Guide for details).
2. Select the **IP Catalog** tab then select **IP on Server**.
3. Select **PCIE x1** and install. After installation, the IP should be on the **IP on Local** tab.

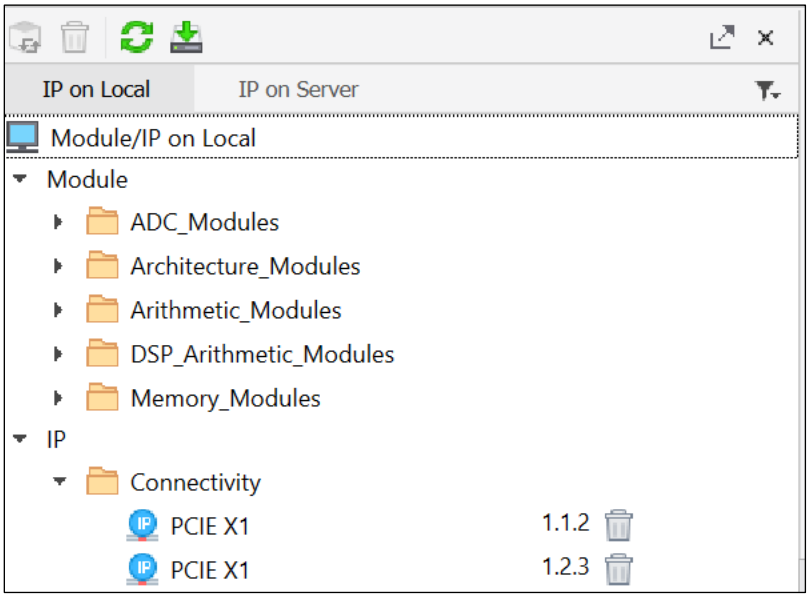

Figure 7.1. Select PCIE x1 IP

4. Double-click on *PCIe_Endpoint* to open the **Module/IP Block Wizard**.
5. Fill out the required information on the dialog box (such as component name and directory) and click **Next**.

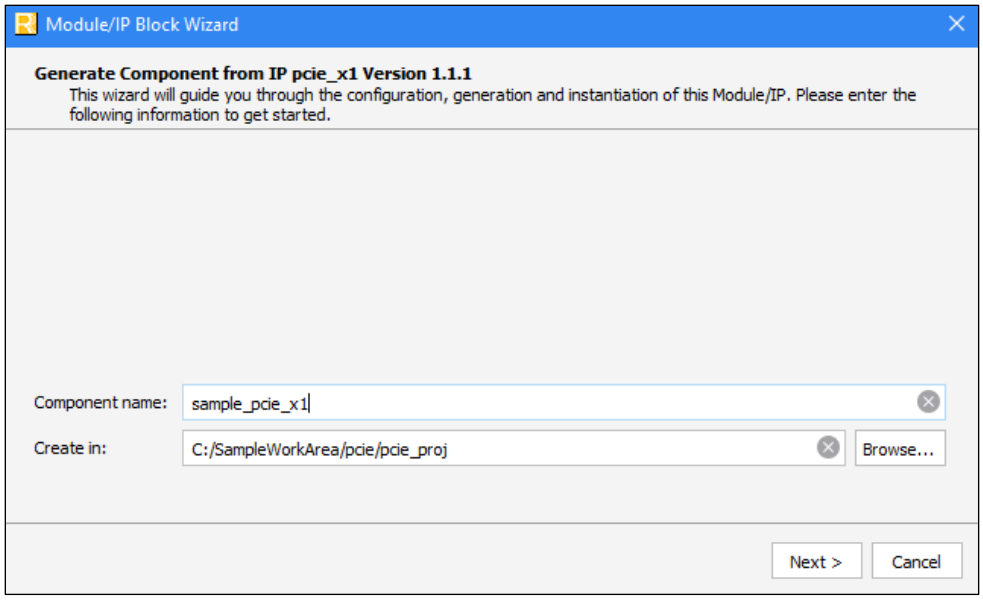

Figure 7.2. Configure Module/IP Block Wizard

7.2. Configuring the IP Core

Figure 7.3 shows the IP Configuration interface where you can select and set the IP parameters.

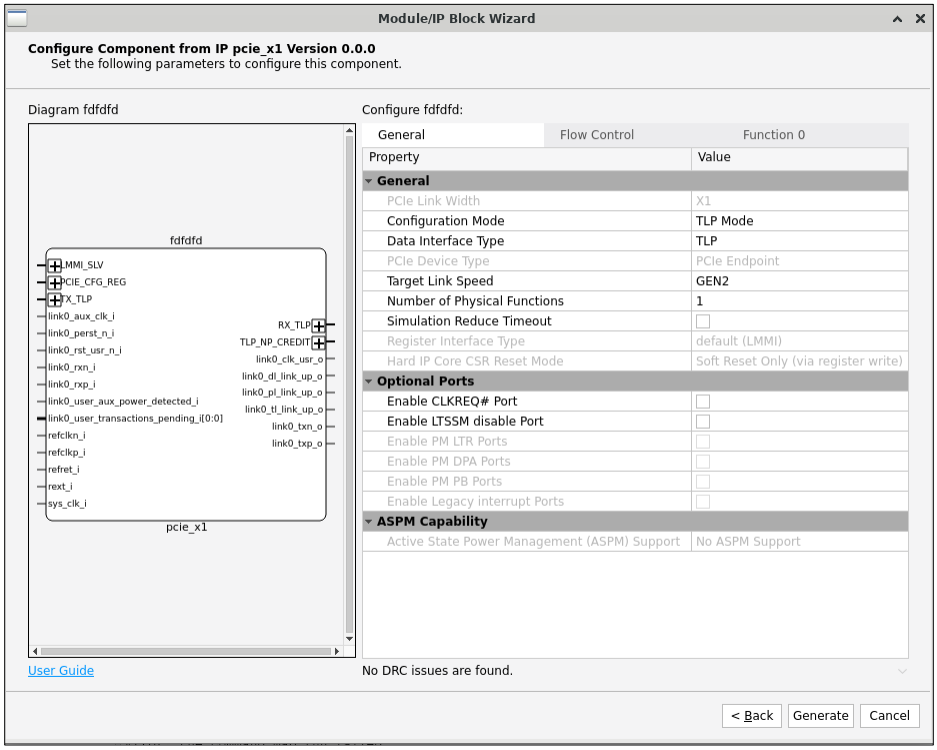

Figure 7.3. Lattice PCIe x1 Core Configuration User Interface (General Tab)

You can configure the IP by setting the user interface attributes applicable to their application. You may also configure the CSR through the register interface. It is recommended to use the Configuration interface to ensure that only valid parameter values are set. The details of attributes and settings are described in IP Parameter Description section.

Figure 7.4 shows the configuration options in *Flow Control* tab.

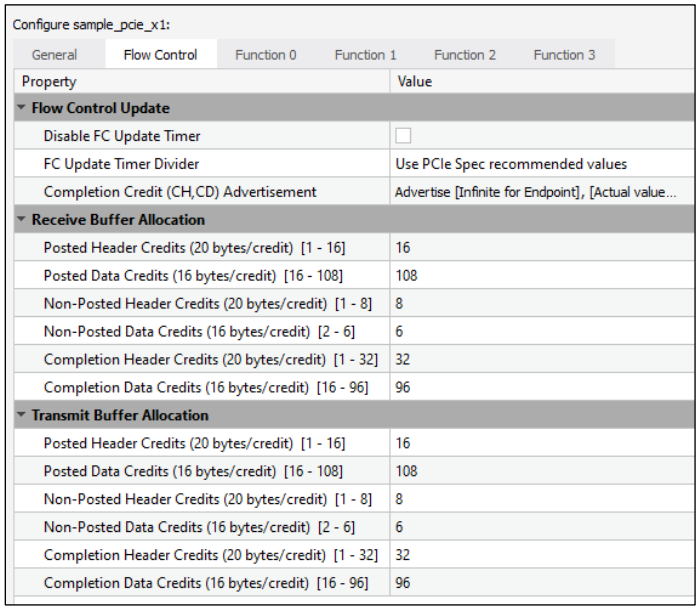

Figure 7.4. Lattice PCIe x1 Core Configuration User Interface (Flow Control Tab)

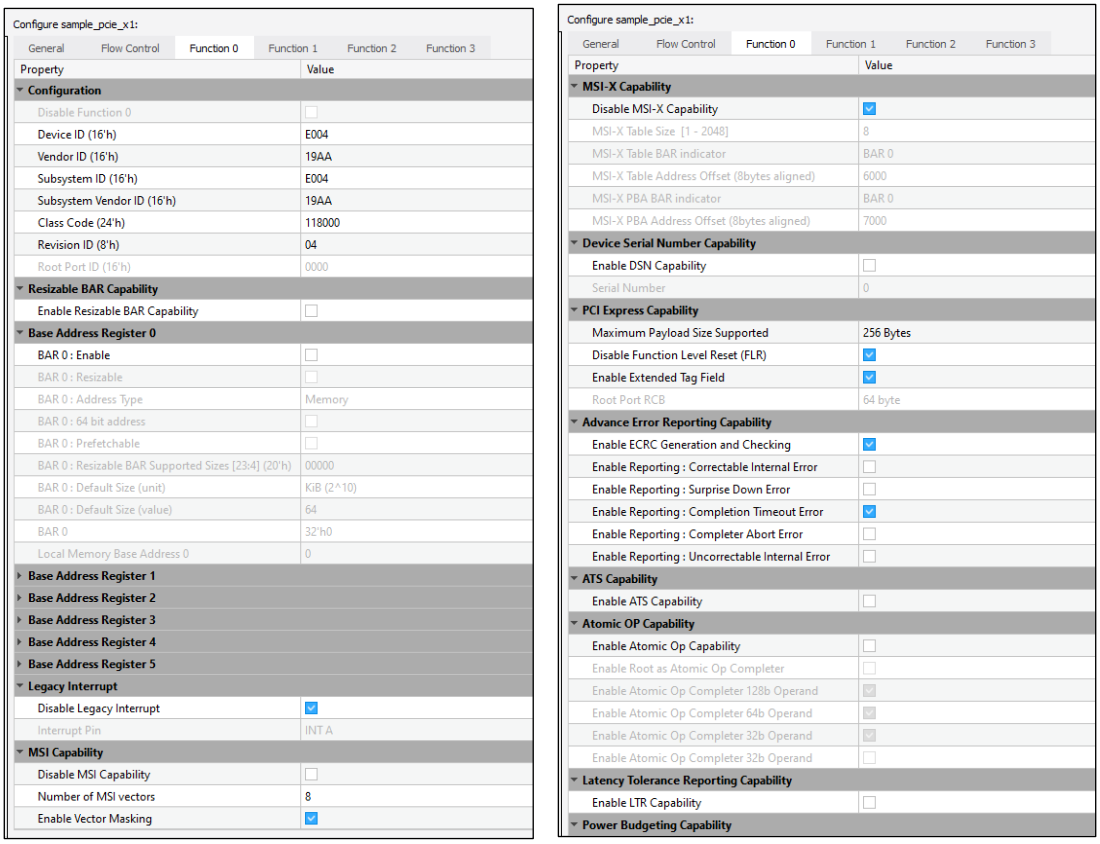

Figure 7.5. Lattice PCIe x1 Core Configuration User Interface (Function 0 Tab)

7.3. Generating the IP Core

To generate the IP Core:

1. After configuring the IP Core, click the **Generate** button.
2. Checked the **Insert to project** option. Click **Finish**.

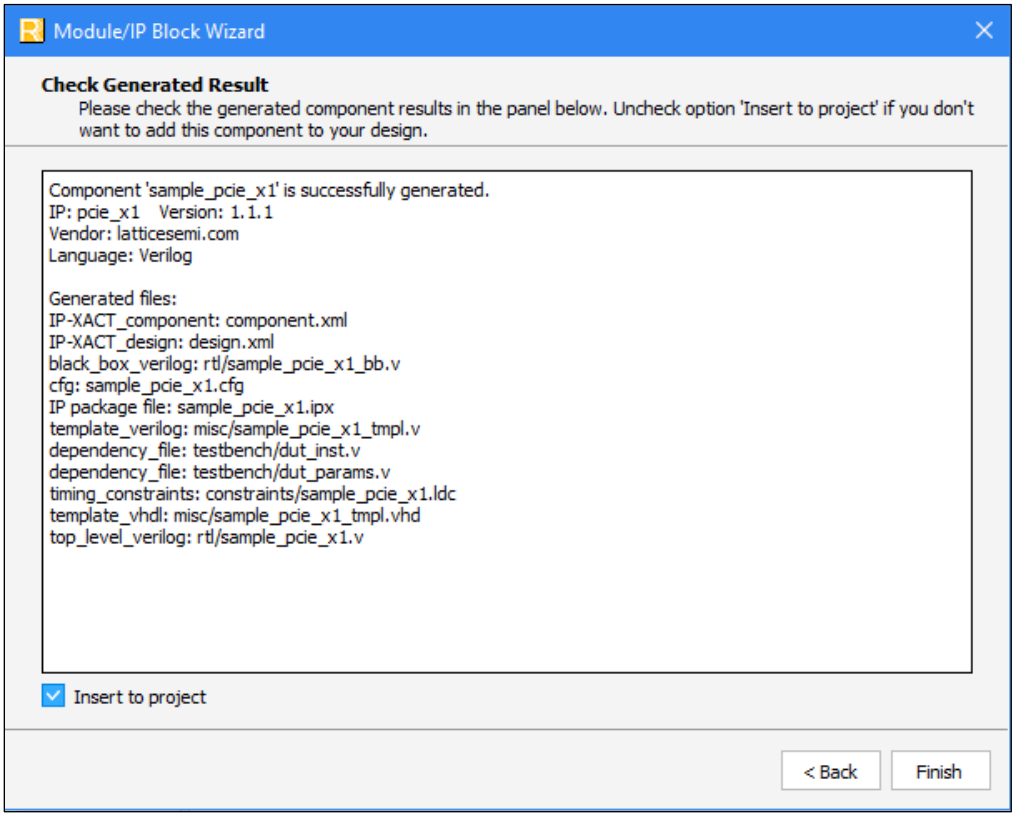

Figure 7.6. Check Generated IP

7.3.1. Generated Files and File Structure

The generated PCIe x1 module package includes the black box (<Component name>_bb.v) and instance templates (<Component name>_tmpl.v/vhd) that can be used to instantiate the core in a top-level design. An example RTL top-level reference source file (<Component name>.v) that can be used as an instantiation template for the module is also provided. You may also use this top-level reference as the starting template for the top-level for the complete design. The generated files are listed in below.

Table 7.1. Generated File List

Attribute	Description
<Component name>.ipx	This file contains the information on the files associated to the generated IP.
<Component name>.cfg	This file contains the parameter values used in IP configuration.
component.xml	Contains the ipxact: component information of the IP.
design.xml	Documents the configuration parameters of the IP in IP-XACT 2014 format.
rtl/<Component name>.v	This file provides an example RTL top file that instantiates the module.
rtl/<Component name>_bb.v	This file provides the synthesis black box.
misc/<Component name>_tmpl.v misc /<Component name>_tmpl.vhd	These files provide instance templates for the module.

The IP Core and other supporting files ARE generated in the specified directory. Figure 7.7 shows the directory structure of the generated IP Core.

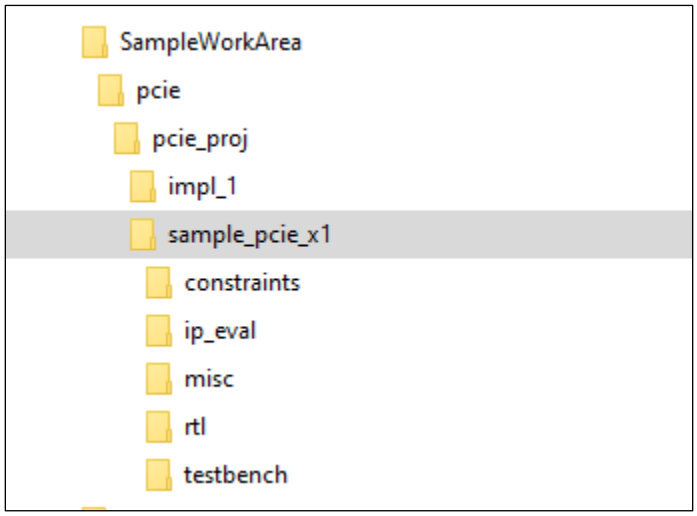

Figure 7.7. Generated IP Core Directory Structure

The testbench directory contains bus functional models for simulation. It also includes a generated file *dut_inst.v* that can be used to instantiate the IP Core either by including the file or by copying the module instance. A simple reference design for IP evaluation is provided in the *ip_eval* directory that is used when running synthesis flow and simulation.

7.4. Timing Constraints the IP Core

To run the synthesis flow using the IP evaluation reference design:

1. Check the **File List** tab. An ipx file must be in the **Input Files** after the IP Core is generated.

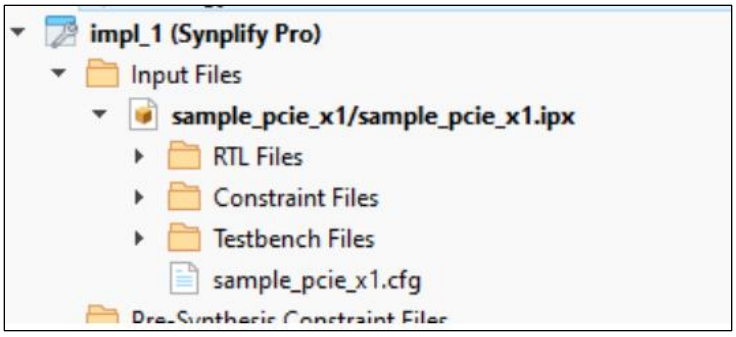

Figure 7.8. Generated IP Core Directory Structure

2. Right-click on the **Post-Synthesis Constraint Files**, select **Add** then select **Existing File**.
3. Browse through the **eval** folder and add *constraint.pdc* file.

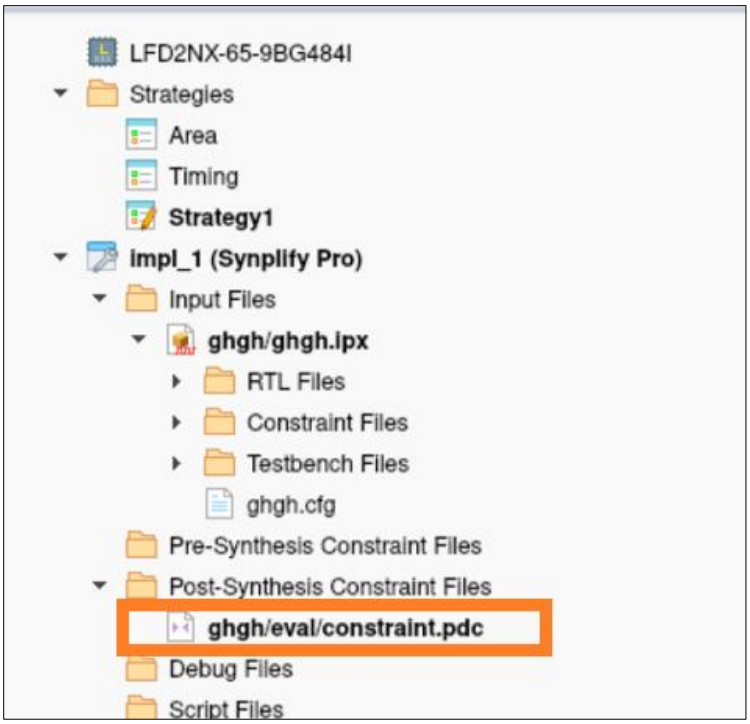

Figure 7.9. Include Timing Constraint pdc File

4. Run the synthesis flow, map, place, and route by clicking the **Toolbar** button.

Figure 7.10. Run Synthesis Flow

The corresponding button turns green with a check mark once the flow is done.

Figure 7.11. Synthesis Flow Status

7.5. Production Driver

7.5.1. DMA

For more information, refer to the Lattice Avant and Nexus PCIe Host DMA Driver User Guide (FPGA-TN-02386) document.

7.5.2. Non-DMA

For more information, refer to the Lattice Avant and Nexus PCIe Basic Memory-Mapped Host Driver (Non-DMA) User Guide (FPGA-TN-02387) document.

7.6. Known Issue

It is a known issue that Radiant user interface setting fails to translate to PCIe EP setting.

As a walkaround, you can refer to the LMMI_app module in Example Design to configure PCIe EP settings such as multi-function support, and BAR setting.

8. Debugging

The PCIe protocol involves the interface between a root port and endpoint with both sides being linked up. Hence, PCIe issues can range from device recognition issues, link training issue, flow control errors, enumeration issues, link down due to fatal errors, and others. This section provides the debug flow diagrams for some of the most common issues when using the PCIe x1 IP. Several debug flow charts are introduced with additional information on critical debug registers to refer to and loopback diagnostic features. This section also provides a short description on signals to be used for debugging simulation.

8.1. Debug Methods

8.1.1. Debug Flow Charts

One debugging method is to identify the type of PCIe issue. The following sections show the steps to debug various issues.

8.1.1.1. Hardware Detection Failure

Follow the steps shown in the flow diagram below if the system is not detecting the hardware.

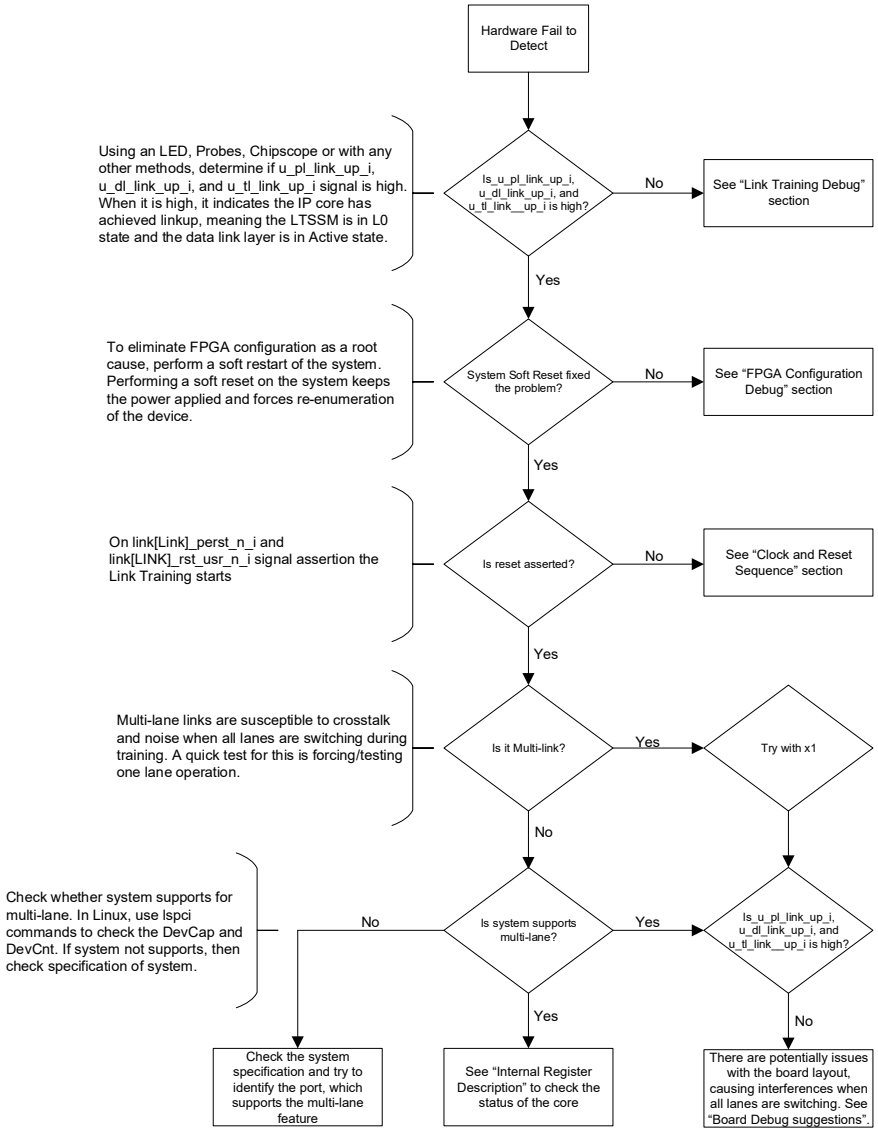

Figure 8.1. Hardware Detection Failure Debugging Flow

8.1.1.2. Link Training Debug

For link training debug refer the flow chart as shown in Figure 8.1:

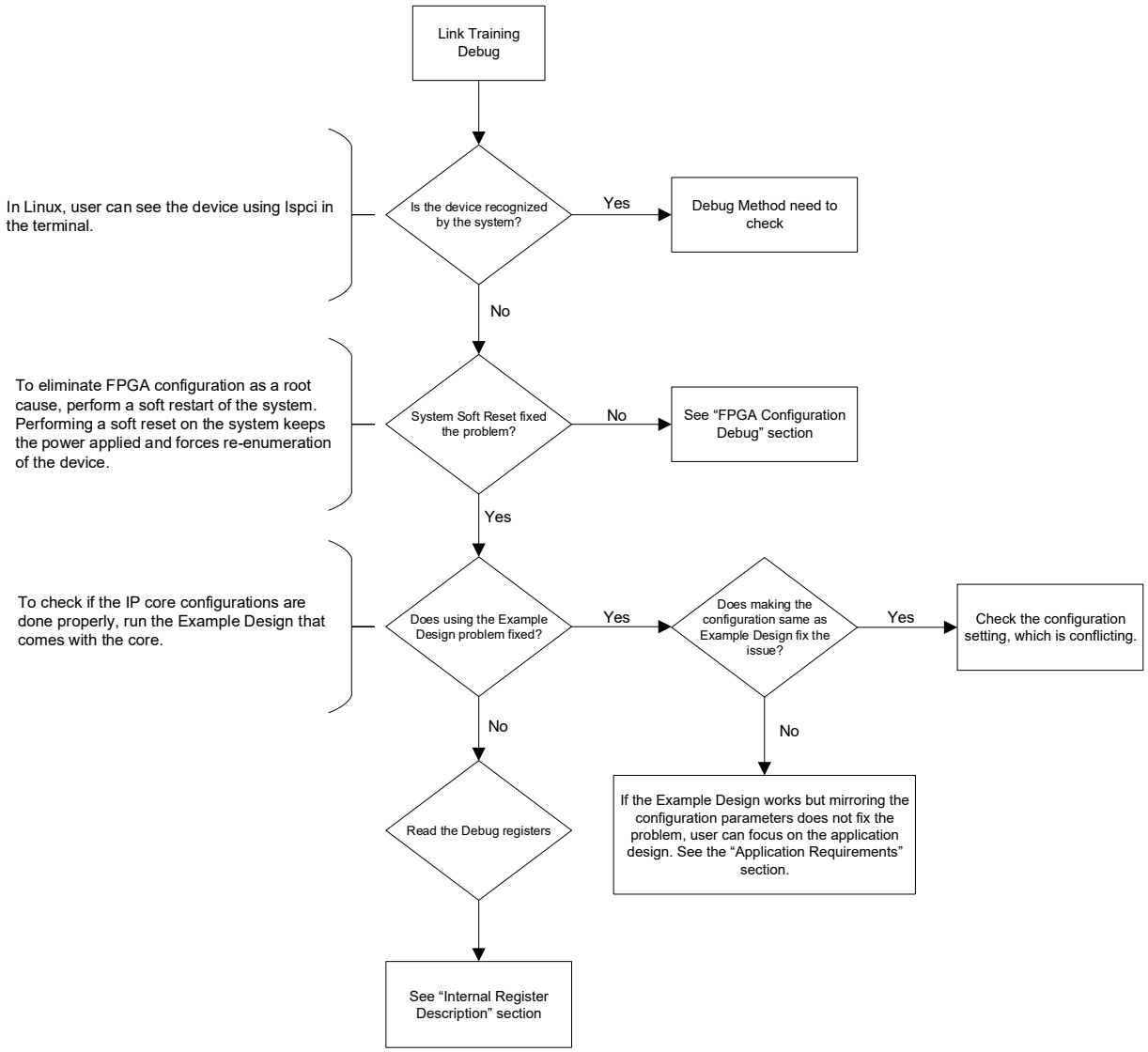

Figure 8.2. Link Training Issue Debugging Flow

8.1.1.3. Data Transfer Debug

If data transfer fails, refer to the steps shown in Figure 8.3.

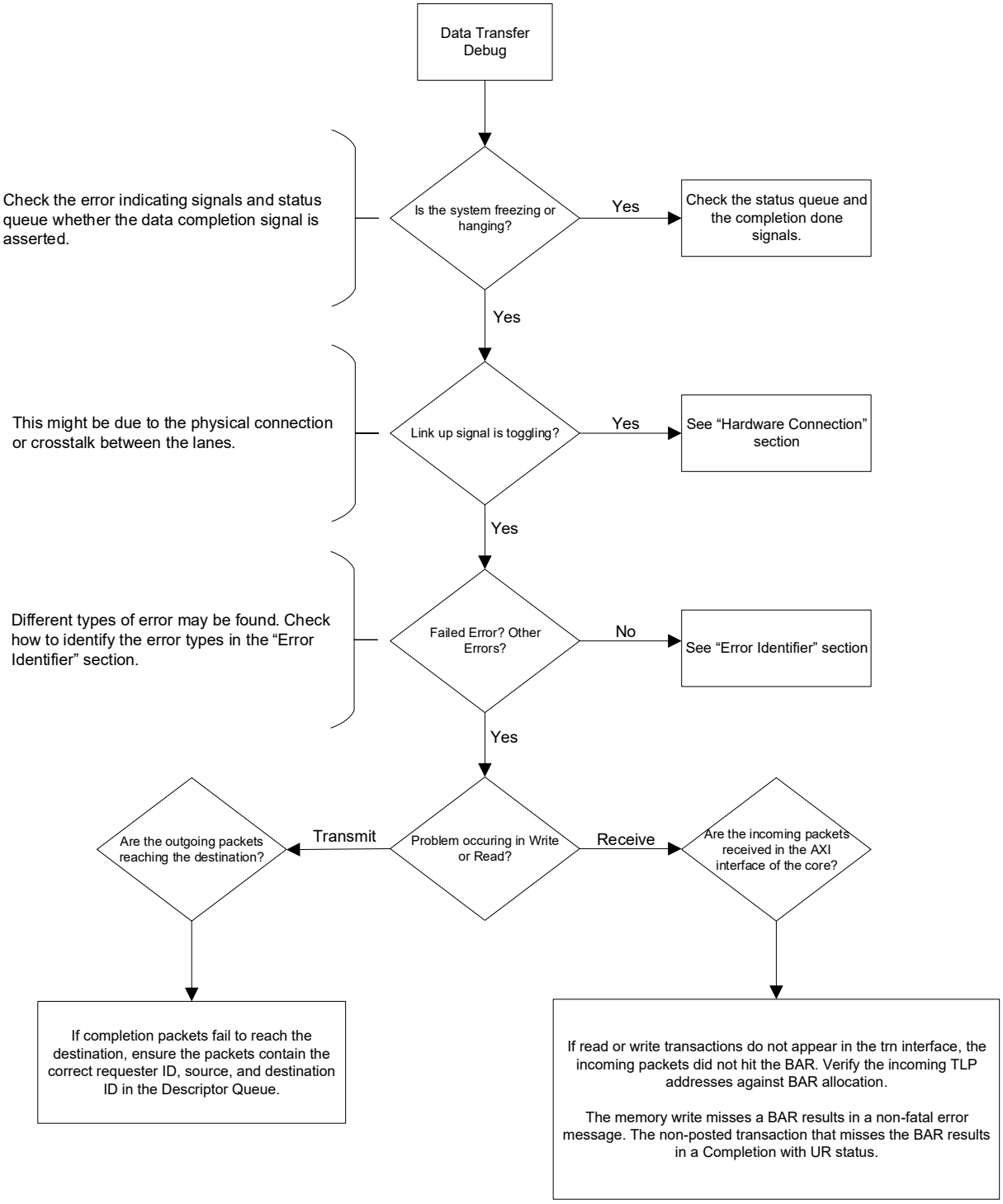

Figure 8.3. Data Transfer Issue Debugging Flow

8.1.1.4. FPGA Configuration Debug

Device initialization and configuration issues can be caused by not having the FPGA configured fast enough to enter link training and be recognized by the system. Prior to becoming operational, the FPGA goes through a sequence of states, including initialization, configuration, and wake-up. After programming the FPGA, a soft reset is required to configure the FPGA from flash. Performing the soft reset operation restarts the enumeration process.

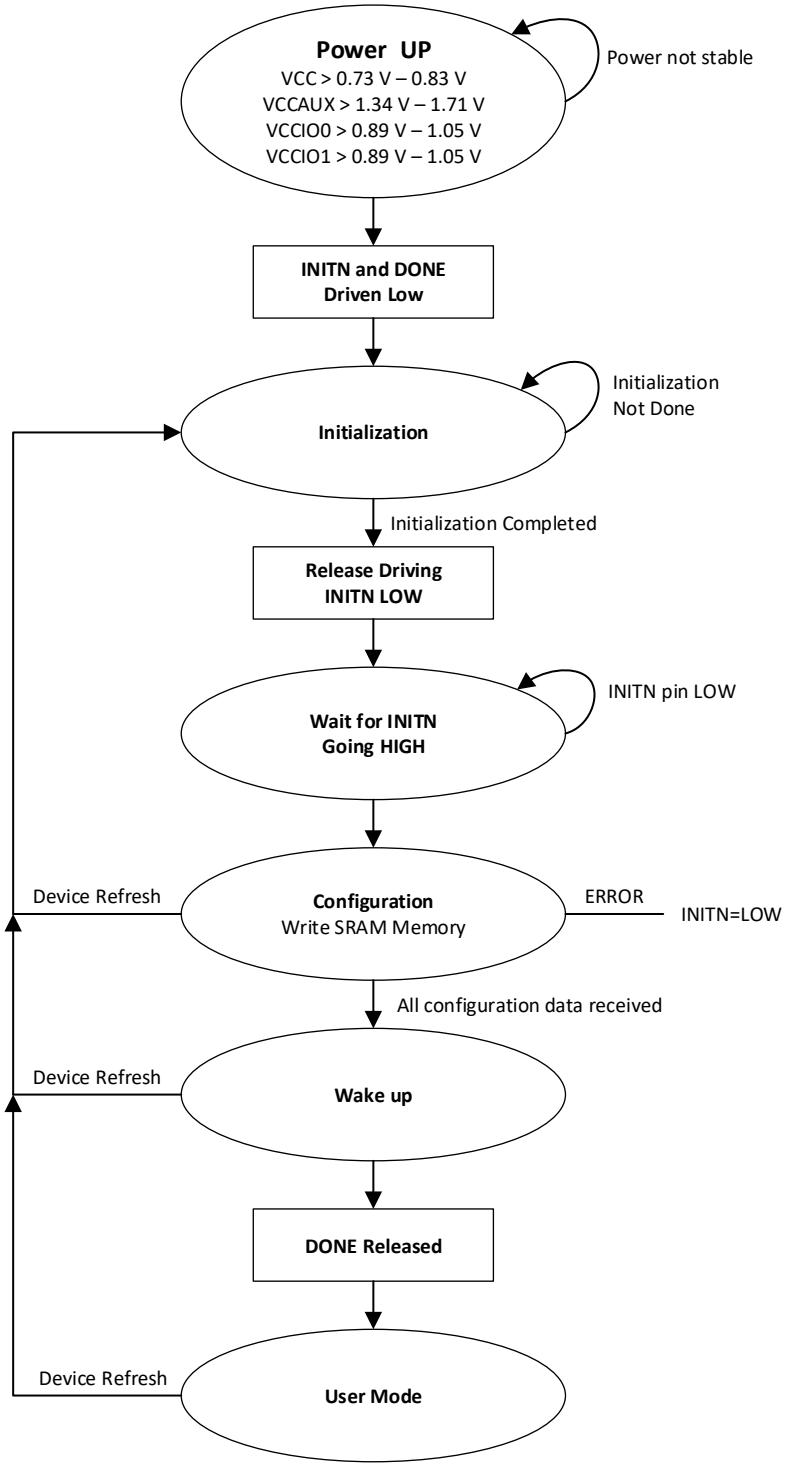

Figure 8.4. Debugging the FPGA Configuration Issues Flow

You can refer to the sysCONFIG User Guide for Nexus Platform (FPGA-TN-02099) document for more information on FPGA configuration.

8.1.2. Internal Register Read for Debug

If the above flowcharts do not capture the issues mentioned, you can read the 0x03114 (*vc_rx_status*) register, which indicates the Receive Buffer Parity/ECC Status where the error detection status can be seen.

The PCIe capability register addresses listed below also provide relevant information for debugging the PCIe issues:

- 0x47-0x44 (Device capability register) address for checking the different supported capabilities of the connected system.
- 0x49-0x48 (Device Control Register) address for checking the supported capabilities of the device.
- 0x4B-0x4A (Device Status Register) address for the device status. You can obtain the error status through this register address.

8.1.3. PCIe Loopback Test

The PCIe loopback test is a diagnostic feature specified by the PCIe Specification that can aid in debugging. The LTSSM Loopback is a state of the Link Training and Status State Machine (LTSSM), which is a mechanism for managing the link state of a serial bus such as PCI Express. In this state, the link partner can test its own transmitter and receiver by sending and receiving data packets without involving the link partner.

The LTSSM Loopback state can be entered from two different states: Configuration or Recovery. The entry into Loopback state is initiated by a Leader Loopback. Before register 0x2100 field is set to 1, all relevant registers containing Leader Loopback control options must be set to the desired values.

When mgmt_tlb_debug_direct_to_loopback = 1, no Leader Loopback control options may be changed. The LTSSM Loopback state has three substates: Entry, Active, and Exit. In Entry substate, both link partners wait for eight EIOS (End of Initialization Ordered Sets) before transitioning to Active substate. In Active substate, both link partners exchange data packets for testing purposes. In Exit substate, both link partners wait for eight EIOS before transitioning to Recovery or Configuration state depending on whether they received an Electrical Idle signal or not.

The LTSSM Loopback is useful for debugging and characterizing the performance of PCI Express links during Link Equalization Training which is a process of optimizing the signal quality between two link partners by adjusting various parameters such as amplitude, de-emphasis, preshoot, and jitter. For more information on the Loopback state, see Table 2.6.

9. Design Considerations

9.1. DMA Based Design

To create a DMA based design:

1. According to the PCIe IP configuration, select the appropriate clocking architecture. Refer to the Clocking section.
2. Select *DMA only Mode* in *Configuration Mode* drop-down menu in the IP wizard General section. Configure AXI DMA in the IP wizard in the DMA Support section.
3. Set up Descriptors in Host Memory. Program DMA registers to initiate DMA transfer.
4. If AXI-MM DMA is selected, verify F2H and H2F data transfer through AXI-MM interface.

9.2. Non-DMA Based Design

To create a Non-DMA based design:

1. According to the PCIe IP configuration, select the appropriate clocking architecture. Refer to the Clocking section.
2. Select the proper Configuration Mode in IP according to the design requirement. Refer to the General section.
3. Initialize the register using LMMI interface/APB interface which is configured from the IP wizard. Refer to the LMMI Interface section.
4. Verify the TLP write and read Transactions. Refer to the Transaction Layer Interface section.
5. Verify the AXI-Stream write and read Transactions. See AXI-Stream Interface section.
6. Verify the AXI-MM transactions. See AXI Data Interface (Bridge Mode) section.
7. Verify the AXI-Lite transactions. See AXI Data Interface (Bridge Mode) section.
8. Select the BAR's with BAR size according to the requirements. See Base Address Register (BAR) [0 to 5] section.

Appendix A. Resource Utilization

The Lattice PCIe IP core utilization report is provided in this section. You can check the resource utilized by the IP core and design top logic based on the available resource in the Certus-NX FPGA device.

Table A.1 shows a sample resource utilization of the Lattice PCIe x1 IP Core on LFD2NX-40.

Table A.1. Lattice PCIe IP Core Resource Utilization

PCIe Core Config	Device Family	Map Resource Utilization				
		LUT4	PFU Register	I/O Buffer	EBR	Data Interface Type
1×1 EP	LFD2NX-40	717	470	101	3	AXI-STREAM/ APB
1x1 EP	LFD2NX-40	303	0	6	0	TLP/ LMMI
1x1 EP	LFD2NX-40	2574	1831	76	0	AXI-MM/ LMMI
1x1 EP	LFD2NX-40	2541	1831	38	0	AXI-Lite/ LMMI
1x1 EP DMA	LFD2NX-40	7907	6358	44	32	AXI-MM/ LMMI

Note: Resource utilization differs with different configurations of the PCIe IP. The above resource utilization is provided for reference only. You can view the resource utilization under *Report > Map > Map Resource Usage*. To view the resource usage, you must run the Synthesize and Map Design.The LIFCL-40, LFD2NX-28, LFD2NX-35, LFD2NX-65, LFMXO5-35T, and LFMXO5-65T devices have similar run results.

References

- PCI Express x1 and x4 IP Core for Nexus-based FPGAs
- PCI Express Base Specification, Rev 3.0 and Rev 3.1
- PCIe x1 IP Release Notes (FPGA-RN-02060)
- Lattice Avant and Nexus PCIe Host DMA Driver User Guide (FPGA-TN-02386)
- Lattice Avant and Nexus PCIe Basic Memory-Mapped Host Driver (Non-DMA) User Guide (FPGA-TN-02387)
- CrossLink-NX web page
- Certus-NX web page
- MachXO5-NX web page
- Lattice Radiant FPGA design software
- Lattice Insights for Lattice Semiconductor training courses and learning plan.

Technical Support Assistance

Submit a technical support case through www.latticesemi.com/techsupport.

For frequently asked questions, refer to the Lattice Answer Database at www.latticesemi.com/Support/AnswerDatabase.

Revision History

Note: In some instances, the IP may be updated without changes to the user guide. The user guide may reflect an earlier IP version but remains fully compatible with the later IP version. Refer to the IP Release Notes for the latest updates.

Revision 2.2, IP v3.0.0, December 2025

Section	Change Summary
All	• Made editorial fixes across the document. • Updated IP version on the cover page.
Introduction	• Updated the IP core and Radiant version; removed AHB-Lite; updated Simulation row to update QuestaSim and ModelSim information; updated table note references; removed and added table notes in Table 1.1. Summary of the PCIe x1 IP. • Updated Soft IP and Licensing and Ordering Information section content.
Functional Description	• Updated PCIe IP Architecture Overview section content, including Figure 2.1. Lattice PCIe x1 IP Core Block Diagram to remove AHB-Lite. • Changed Physical Layer Packets to *Ordered Sets* in Protocol Layers. • Updated DMA Support section content, including updating and adding new sub-sections and content: DMA Registers, DMA Performance (AXI-MM), DMA With Bridge Mode, and DMA User Interrupts. • Updated Table 2.10. DMA_LEN (0x04). • Updated Soft IP Interface section content, including: removing AHB-L Interface sub-section, updating x1 lane bit to 32 in AXI-Stream Interface, updating Bridge Mode and Bridge Mode Register sub-section name and content (including figures and tables), added All Other User Interrupt MSI-X Table (0x8020 to 0x83FF) sub-section, and updating APB Interface sub-section content (including adding a note regarding a known bug).
IP Parameter Description	• Updated Figure 3.1. Attributes in the General Tab and Table 3.1. General Tab Attributes Descriptions content. • Updated Figure 3.15. Attributes in PCIe Capability. • Updated Optional Port section content. • Added ASPM Capability, and DMA/Bridge Mode Support. • Removed DMA Support and RX TLP Destination Base Address sections.
Signal Description	• Updated Table 4.1. Clock Ports to change AHB-Lite reference to *Non-DMA*. • Removed AHB-Lite Data Interface section. • Updated section content, including section name to AXI-Stream (Non-DMA) Data Interface. • Updated Table 4.15. DMA Interrupt Interface Ports to remove reference to a deleted section. • Added AXI Data Interface (DMA) section. • Updated Table 4.17. AXI-MM Manager Write Interface (Bridge Mode) and Table 4.18. AXI-Lite Manager Interface (Bridge Mode) to update ports to *m0_axi_mm* and *m0_axil*.
Register Description	Removed Soft IP Configuration, Control, and Status Register section.
Example Design	• Updated section content, including adding information that the example design is note supported in Modelsim OEM and Pro. • Updated Table 6.1. PCIe x1 IP Configuration Supported by the Example Design. • Updated Overview of the Example Design and Features section to remove AXI or AHB-Lite bullet point. • Updated Example Design Components to update DMA Design (AXI-MM) and Non-DMA Design (TLP Interface) section content including section name, tables, and figures; added Non-DMA Design (Bridge Mode) and PDC Settings for Hardware Example Design section. • Updated Simulating the Example Design section content, including updating section content and name from Steps to Simulate Example Design to QuestaSim Pro and adding QuestaSim Lattice-Edition. • Removed Design Test Case Examples section.

Section	Change Summary

	• Updated Debugging Example Design Issues to update the following in Signals to Debug:
	  • Added Simulation Debug for DMA (AXI-MM) Design and Simulation Debug for Non-DMA (Bridge Mode) Design sub-sections.
	  • Updated sub-section name to Simulation Debug for Non-DMA (TLP Interface) Design.
	  • Removed Simulation Debug for DMA Design sub-section.

Designing with the IP	• Added note for screenshots at the beginning of the section. • Updated Instantiating the IP Core to remove IP release information. • Updated Figure 7.3. Lattice PCIe x1 Core Configuration User Interface (General Tab). • Updated Production Driver section to add DMA sub-section. • Added Known Issue section.
Design Considerations	Updated DMA Based Design and Non-DMA Based Design section content.
Appendix A. Resource Utilization	Updated content of Table A.1. Lattice PCIe IP Core Resource Utilization.
References	Added reference document *Lattice Avant and Nexus PCIe Host DMA Driver User Guide (FPGA-TN-02386)*.
Revision History	Added note regarding IP version.

Revision 2.1, IP v2.1.0, October 2025

Section	Change Summary
All	Updated IP version on the cover page.
Introduction	• Updated the IP core and Radiant version; corrected typo to LFMXO5-35T in the Supported Devices row in Table 1.1 Summary of the PCIe x1 IP. • Updated Table 1.2 PCIe x1 IP Support Readiness to add MachXO5-NX support. • Updated Table 1.4. Lattice PCIe IP Core Supported Speed Grade to add CrossLink-NX and MachXO5-NX support and table note.
Designing with the IP	Added Production Driver section.
References	Added reference document *Lattice Avant and Nexus PCIe Basic Memory-Mapped Host Driver (Non-DMA) User Guide (FPGA-TN-02387)*.

Revision 2.0, IP v2.0.0, June 2025

Section	Change Summary
All	• Updated IP version on the cover page. • Updated AXI4_Stream to *AXI-Stream* and AHB_Lite to *AHB-Lite* across the document.
Acronyms in This Document	Added AXI-MM definition.
Introduction	• Added MachXO5-NX support in Overview of the IP. • Updated section name from Supported FPGA Family to *Supported Devices*, added MachXO5-NX supported devices and added reference to table note 2 in Certus-NX; updated the IP core and Radiant version and updated Supported User Interface row including adding AXI-Lite support, added reference to table note 3 for AHB-Lite, and removing Targeted Devices row; added table notes for CABGA484 and IP versions in Table 1.1 Summary of the PCIe x1 IP. • Added AXI-MM and AXI-Lite bullet points and IP version footnotes in the Soft IP section.
Functional Description	• Updated Figure 2.1. Lattice PCIe x1 IP Core Block Diagram and added AXI-MM and AXI-Lite bullet points in PCIe IP Architecture Overview. • Updated clk_usr_o to *link0_ clk_usr_o* in Clocking Overview. • Added AHB-Lite DMA support information in the DMA Support section. • Updated figure caption to Figure 2.15. Non-DMA Application Data Flow – TLP Interface, added Figure 2.16. Non-DMA Application Data Flow – AXI-Stream Interface to Figure 2.18. Non-DMA Application Data Flow – AXI-Lite Interface (Bridge Mode), and updated Table 2.10. Register Access for Different Data Interfaces to add AXI-MM and AXI-Lite

Section	Change Summary
	rows. • Added AHB-L Interface support information in the AHB-L Interface section. • Updated section name to AXI-Stream to PCIe Transfer. • Added AXI Bridge Mode section.
IP Parameter Description	Added *This is only supported in IP version 1.2.5 or older* text in the DMA Support section.
Signal Description	• Updated all tables in the section (except Table 4.8 Lattice Memory Mapped Interface Ports and Table 4.15 APB Configuration Interface Ports) to updated Port, Clock Domain, and Description values across the section, specifically adding *link0* at the beginning of port names and updating port name from clk_usr_i to *sys_clk_i*: • Added *This is only supported in IP version 1.2.5 or older* text in the AHB-Lite Data Interface section. • Added AXI Data Interface (Bridge Mode) section.
Example Design	• Updated section content including adding LMMI in Non-DMA Design for Configuration in Table 6.1. PCIe x1 IP Configuration Supported by the Example Design. • Added AHB-L DMA support text in the DMA Design and Simulation Debug for DMA Design section. • Updated Non-DMA Design section content. • Added Steps to Simulate Example Design section.
Designing with the IP	Updated Timing Constraints the IP Core section content including Figure 7.9. Include Timing Constraint pdc File.
Design Considerations	• Added AHB-L DMA support text in the DMA Based Design section. • Updated AHB-L item and added AXI-MM and AXI-Lite items in the Non-DMA Based Design section.
Appendix A. Resource Utilization	Updated section content including removing text mentioning the IP and Radiant version, updated Table A.1. Lattice PCIe IP Core Resource Utilization to update values and add rows for 1x1 EP AXI-MM and AXI-Lite.
References	Added MachXO5-NX web page and removed Avant-G/X web page references.

Revision 1.9, IP v1.2.5, December 2024

Section	Change Summary
All	• Removed Root Port description, as this is not supported, across the document. • Added IP version to the cover page and revision history.
Introduction	• Updated Table 1.1 Summary of the PCIe x1 IP to add LFD2NX-28 in Targeted Devices, add *IP Changes* row, add TLP in Supported User Interface, update IP Core and Radiant software version, and updated Simulation to *QuestaSim Lattice FPGA Edition* and added table note for Modelsim OEM. • Updated Table 1.3. Ordering Part Number to put Certus-NX in a separate row and update column name to *Single Seat Annual*. • Renamed IP Validation Summary section to Hardware Support and updated section content. • Added IP Support Summary and Speed Grade Supported section.
Functional Description	• Updated Clocking Overview to add reference clock support info. • Updated Table 2.15. Offset Address for Resizable Bar Capability Configurations to update Bit 17 1 value to *Reserved* in Vendor Specific Capability.
Register Description	• Updated Table 5.74. pm_pme_to_ack_ds Register 0x84 to update Field and Width values to *31:0* and *32* respectively and remove 7:0 row. • Updated • Table 5.103. decode_t1 Register 0x14 to remove rows and update Field and Width values to *31:0* and *32* respectively. • Updated Table 5.106. cfg Register 0x30 to change [0] Description value to *1 – Reserved*. • Updated Table 5.121. pcie_cap Register 0x80 and Table 5.122. pcie_cap Register 0x80 to change [7:4] Description, from 4 to 10, values to *Reserved*.

Section	Change Summary
	• Updated Table 5.128. pcie_link_ctl2 Register 0xa0 to update Field and Width values to *31:0* and *32* respectively and remove rows. • Removed Root Port information in PCI Express Configuration Space Registers section. • Updated Table 5.174. PCI Express Capability to change Register Description values (from 0100 to 1010) for 43-42 Addr to *Reserved*.
Example Design	Updated section description to add evaluation boards text.
Designing with the IP	Updated Running Functional Simulation section content including steps 1 to 9.
Appendix A. Resource Utilization	Updated section content including redoing the Table A.1. Lattice PCIe IP Core Resource Utilization content.
References	Added document reference for PCIe x1 release notes.

Revision 1.8, April 2024

Section	Change Summary
Designing with the IP	Updated Instantiating the IP Core section to add information on using 2023.1 and 2023.2 versions.

Revision 1.7, January 2024

Section	Change Summary
All	Revamped document structure for clarity by re-arranging sections and sub-sections.
Disclaimers	Updated this section.
Introduction	• Revamped this section. • Updated Features to add sub-sections. • Moved Ordering Part Number to Licensing and Ordering Information as sub-section and changed to table format. • Added Licensing and Ordering Information and IP Validation Summary.
Functional Description	Revamped this section, including updating diagrams and tables.
IP Parameter Description	Added this section.
Signal Description	Converted subsection (previously under Functional Description) to a main section.
Register Description	Converted subsection (previously under Functional Description) to a main section.
Example Design	Added this section.
Designing with the IP	Changed name from IP Generation to *Designing with the IP* and revamped this section.
Debugging	Added this section.
Design Considerations	Added this section.
Appendix A. Resource Utilization	Updated this section to add more information.
References	Added Rev 3.0 and 3.1 for PCI Express Base Specification and modified webpages for Avant devices.

Revision 1.6, September 2023

Section	Change Summary
IP Generation	Updated the Synthesizing and Implementing the IP Core and Running Functional Simulation Deleted Running Example Design (ED) Simulation section.
Ordering Part Numbers	Updated the OPN information in section Ordering Part Numbers.
References	Added web links for CrossLink-NX, Certus-NX, Lattice Radiant, and Lattice Insights.

Revision 1.5, June 2023

Section	Change Summary
All	Fixed minor formatting issues in the document.
Functional Description	Updated the descriptions of Configuration Headers and PCIe Express Capability 0x10 in Table 2.19. UCFG Address Space. Updated inputs rext_i and refret_i in Figure 2.2. Lattice PCIe x1 Core Hard IP.
IP Generation	Added Running Functional Simulation section.
Reference	Added web page link for PCI Express x1 & x4 IP Core for Nexus-based FPGAs.
Technical Support Assistance	Added Lattice Frequently Asked Questions website link.

Revision 1.4, August 2021

Section	Change Summary
Functional Description	Added information to the sys_clk_idescription in Table 2.8. Clock and Reset Port Descriptions.

Revision 1.3, December 2020

Section	Change Summary
Introduction	Changed PCI Express Base Specification Revision to 3.0 in the Features section.
Functional Description	• Revised perst_n_i description in Table 2.8. Clock and Reset Port Descriptions. • Added information in the AHB-Lite Data Interface and the AHB-Lite Configuration Interface sections. • Adjusted level of pcie_ll (0x0F000) heading.
All	Updated reference to Lattice Radiant Software User Guide.

Revision 1.2, June 2020

Section	Change Summary
All	• Changed IP name to PCIE x1. • Changed document title to PCIe x1 IP Core - Lattice Radiant Software.
Disclaimers	Added this section.
Introduction	Updated Table 1.1: • Added Certus-NX support. • Added LFD2NX-40 as targeted device. • Updated Synopsis Synplify Pro version. • Updated Lattice Implementation to Lattice Radiant 2.1.
Functional Description	• Added Root Port mode support. • Updated diagrams to match the current IP. • Corrected IP Register names and offset mapping.
Designing with the IP	Removed reference to Lattice Radiant 2.0 Tutorial.
Ordering Part Number	Added this section.
Appendix A. Resource Utilization	Updated device to LIFCL-40-9BG400I.

Revision 1.1, March 2020

Section	Change Summary
All	• Removed details for unsupported features. • Minor adjustments in formatting/styles.

Revision 1.0, December 2019

Section	Change Summary
All	Initial release.

www.latticesemi.com