POWER9 PCIe Controller

Functional Specification

OpenPOWER

Version 1.1
27 July 2018

Printed in the United States of America July 2018

IBM Systems
294 Route 100, Building SOM4
Somers, NY 10589-3216

The IBM home page can be found at **ibm.com®**.

Version 1.1
27 July 2018

Contents

List of Tables

List of Figures

Revision Log

Each release of this document supersedes all previously released versions. The revision log lists all significant changes made to the document since its initial release. In the rest of the document, change bars in the margin indicate that the adjacent text was modified fom the previous release of this document.

Revision Date	Description
27 July 2018	Version 1.1. • Revised a paragraph in *Section 5 Processor Bus Common Queue Registers* on page 45. • Added *Table 6-2 PHB Lane Configuration and Swap Bit Register Mapping* on page 88.
31 May 2018	Version 1.0 (Initial release).

About this Document

This document describes the design and operation of the IBM® POWER9™ PCI Express Controller (PEC). It provides details on its basic requirements, function, operation, and usage. It serves as a reference document for designers, simulators, testers, and programmers.

Related Documents

The documents available in the IBM Portal for OpenPOWER, an online IBM technical library, are helpful in understanding the contents of this document.

Conventions Used in this Document

The following typographical conventions are used to define special terms and command syntax:

Convention	Description
`Monospaced` typeface	Used for code examples.
Number conventions	Numbers are generally shown in decimal format, unless designated as follows: • Hexadecimal values are preceded by an “0x”. For example: 0x0A00. • Binary values ar preceded by an “0b” (for example: 0b0A00) or enclosed in single quotes (‘0’).
Underline	Indicates that the definition of an acronym is displayed when the user hovers the cursor over the term.
Hyperlink	Web-based URLs are displayed in blue text to denote a virtual link to an external document. For example: http://www.ibm.com
Note: This is note text.	The note block denotes information that emphasizes a concept or provides critical information.
This is an inline footnote reference.[1] 1. Descriptive footnote text.	A footnote is an explanatory note or reference inserted at the foot of the page or under a table that explains or expands upon a point within the text or indicates the source of a citation or peripheral information.

1. Overview

The POWER9 PCIe controller (PEC) provides PCIe Gen4 root-complex ports to connect to an adapter slot or as a link to a PCIe switch. It acts as a PCIe host bridge (PHB) from the internal, coherent SMP interconnect to the PCIe I/O.

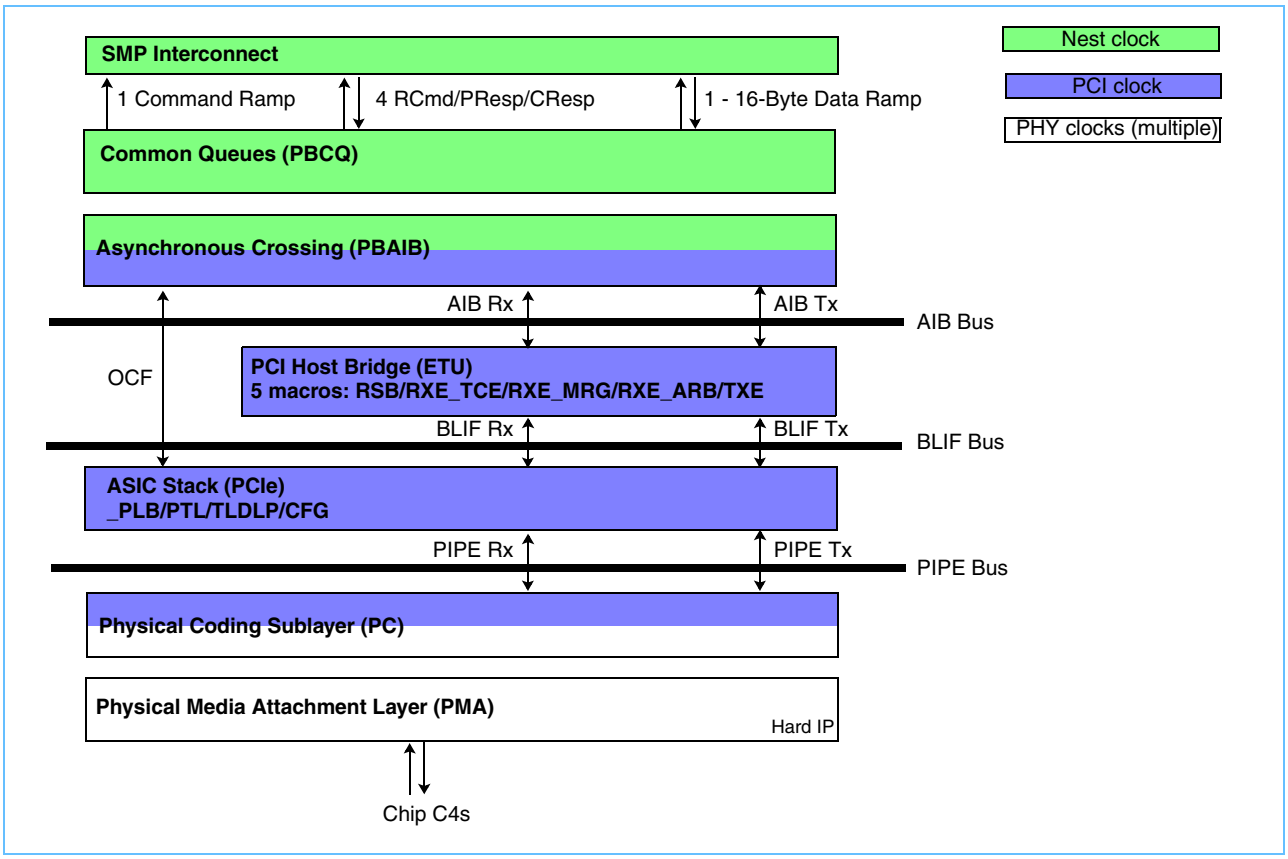
1.1 Description

The PEC is made up of six major building blocks.

- Processor bus common queue (PBCQ) logic
- Processor bus to AIB interface (PBAIB), which is an asynchronous boundary crossing
- Express transaction unit (ETU)
- PCIe ASIC building blocks (PCIASIC)
- Physical coding sublayer (PCS)
- Physical media access (PMA)

Figure 1-1 shows an overview of the major blocks and defined interfaces. The following sections describe the building blocks.

Figure 1-1. High-Level Block Diagram



1.1.1 Processors Bus Common Queue (PBCQ)

The processor bus common queue (PBCQ) logic is responsible for managing the transactions on the coherent processor/cache fabric, also known as the SMP interconnect. Key features of the PBCQ are as follows:

- CAPI support
- Tunnelled operations
  - Atomics
  - AS_Notify support
- Inbound DMA capability
  - Supports 64 (128 on PEC0) DMA read transactions on the SMP interconnect. DMA read transactions are sourced from nonposted read transactions from the PCIe.
  - Supports 32 DMA write transactions on the SMP interconnect. DMA write transactions are sourced from posted write transactions from the PCIe.
  - Peer-to-peer write capability.
- Tunnelled operations
  - Atomics. Atomic transactions are sourced from posted write transactions on the PCIe and, if necessary, return data back to the PCIe using an MMIO store.
  - AS_Notify is a quick method to communicate with the core.
  - CAPI support.
- Outbound MMIO capability
  - Two Base Address Registers (BARs) for external MMIO address ranges.
  - Sixteen MMIO stores.
  - Sixteen MMIO loads.
- The ability to share resources with more than one PHB stack.

1.1.2 Processor Bus AIB interface (PBAIB)

The PBAIB logic provides an asynchronous crossing between the PBCQ and the AIB 2.0 interface.

1.1.3 Express Transaction Unit

The ETU is responsible for address translation, interrupt management, and error isolation. Key features of the ETU (×8 or ×4 lane versions) are as follows:

- 512 KB (256 KB) partitionable endpoints
- 1 KB (512 KB) 4-way set-associative translation cache
- 4K (2K) MSI interrupts supported
- Eight LSI interrupts supported

1.1.4 PCIe ASIC Intellectual Property

The PCIe ASIC building block is composed of the packet buffer layer (PBL), the packet transaction layer (PTL), the transaction and data link layer (TLDLP), and the PCIe Configuration Register core (CFG). These blocks implement the PCIe transaction and data link layers.

1.1.5 Physical Coding Sublayer

The physical coding sublayer (PCS) manages the low-level networking protocol and signaling between the physical media and the higher-level link protocol layer across the PIPE interface. The 16 lanes of the PCS can be bifurcated into two ×8 lanes or trifurcated into one ×8 and two ×4 lanes.

1.1.6 Physical Media Access

The physical media access (PMA) provides the SERDES and analog protocols necessary to connect to the chip C4s. It also provides the PLLs used to drive the PCI clock grid.

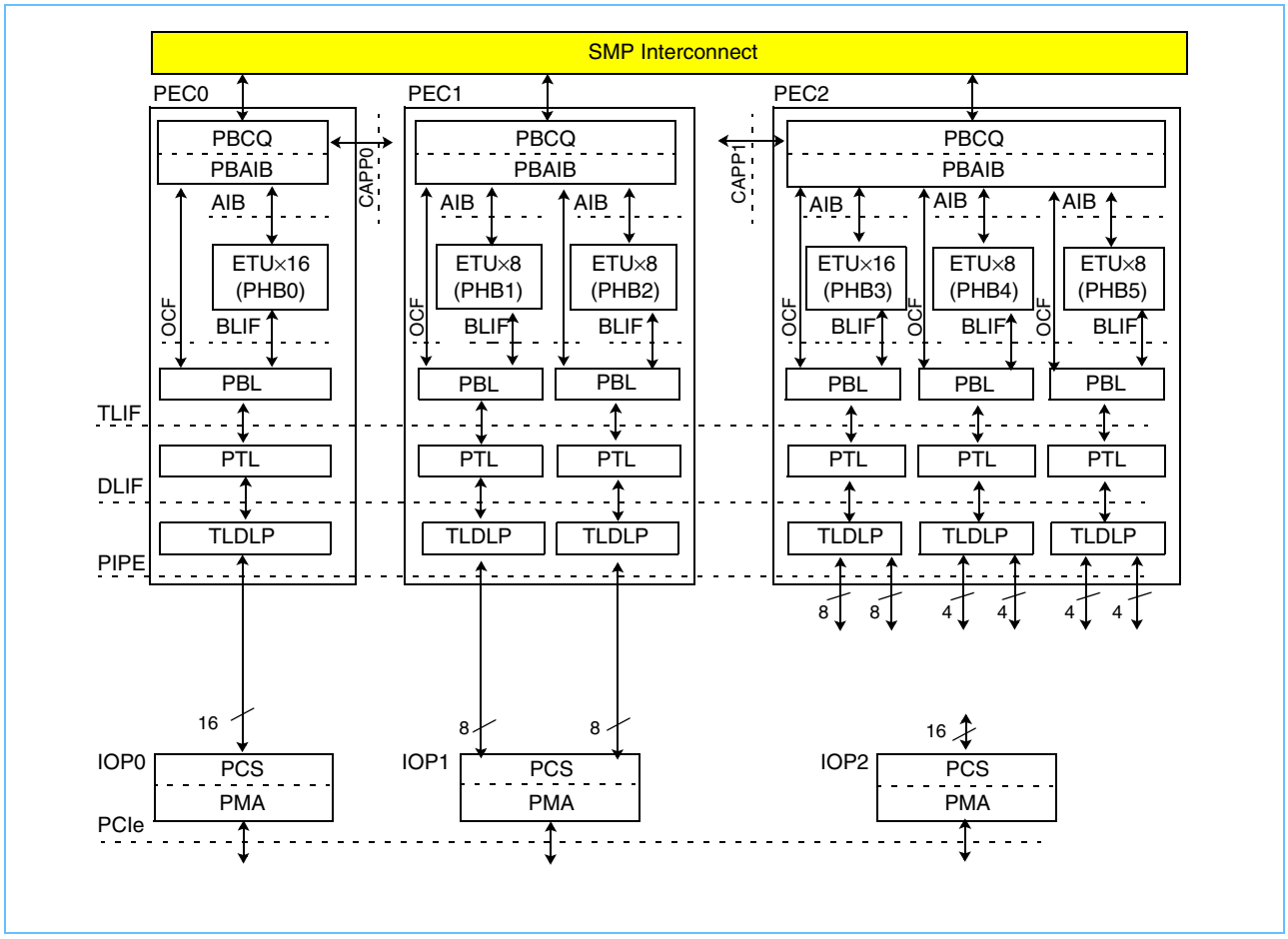
1.2 POWER9 Configurations

The POWER9 processor chip has three PCIe controllers of 16 lanes each for a total of 48 lanes of PCIe Gen4 I/O. The PECs can be configured as follows:

- PEC0: One ×16 lanes
- PEC1: Two ×8 lanes (bifurcation)
- PEC2: One ×16 lanes, two ×8 lanes (bifurcation), or one ×8 and two ×4 lanes (trifurcation)

Each grouping of lanes is called a stack. Each stack has dedicated ETU and PCIe blocks. Each set of 16 lanes has only one PBAIB and PBCQ pair to interface to the SMP interconnect. The resources of the PBCQ are shared between the stacks that it services. If all of the stacks in a PEC are not used, stacks are used left to right (see *Figure 1-2* on page 18). For example, if only one stack is going to be used in PEC1, PHB1 is used. Within a stack grouping, lanes can be swapped to facilitate board wiring.

Figure 1-2. POWER9 PCIe High-Level Diagram



1.3 Reliability, Availability, and Serviceability (RAS)

1.3.1 Bit-Level RAS

- End-to-end data protection from the SMP interconnect ECC to the PCI packet LCRC/ECRC
- Arrays have SEC/DED ECC
- Register files have parity (some have SEC/DED)
- Support all SMP interconnect parity/ECC
- Major control registers have parity protection

1.3.2 Enhanced Error Handling (EEH)

If an error can be isolated to an endpoint, this endpoint is blocked from introducing new transactions until the error can be resolved.

1.3.3 Freeze Mode

An error that requires a reset of a stack enters freeze mode. Freeze mode blocks all new transactions to and from the stack. Outstanding operations on the SMP interconnect run to completion, marking data as bad if required. Reset and initialization can be performed on the stack without a checkstop of the chip. A freeze on a stack does not affect the actions of another stack even if they share a PEC.

2. Processor Bus Common Queue

The processor bus common queue (PBCQ) is the PCIe unit's interface to the SMP interconnect. The PBCQ supports 1 - 3 PCIe stacks.

When supporting one PCIe stack, there are enough resources to maintain full bandwidth of one 16-lane Gen4 PCIe stack, which can be bifurcated to support two 8-lane Gen4 PCIe stacks or trifurcated to support one 8-lane and two 4-lane Gen4 PCIe stacks. When supporting more than one stack, the default is to split the queue resources proportionally to the lanes supported. The default resource assignment can be overridden by the registers read-stack override, write-stack override, and store-stack override. Each stack maintains a dedicated programming interface. Errors are isolated to a stack, and the stacks can go into and recover from a freeze independently.

The PBCQ has two fundamental modes of operation: I/O mode and CAPP mode. The CAPP Control Register controls the mode of operation. The I/O mode is the traditional PHB root-complex operation. In CAPP mode, the PBCQ behaves as a data transport layer under the control of the SMP interconnect-attached CAPP unit.

2.1 I/O Mode Features

- Supports three programmable I/O ranges per PCI stack (one PHB Register and two PCI-bus memory)
- 32 DMA write queues per PBCQ, each with 128-byte data buffer to support the following activities:
  - DMA writes
  - Atomic operations
  - Interrupt requests
- 64 DMA read queues per PBCQ; each with a 128-byte data buffer
  - PEC0 has 128 DMA read address queues
- 16 cache-inhibited (CI) loads per PBCQ, each with a 32-byte data buffer
- 16 cache-inhibited store queues per PBCQ, each with a 128-byte data buffer to support:
  - MMIO stores
  - Peer-to-peer stores
- Freeze-mode support

Table 2-1. PBCQ I/O Mode Default Resource Allocation by Stack

Resource	Dedicated	Bifurcated		Trifurcated		
	Stack 0	Stack 0	Stack 1	Stack 0	Stack 1	Stack 2
DMA write queues	32	16	16	16	8	8
DMA read queues	64 (PEC0 128)	32	32	32	16	16
Cache-inhibited store queues	16	8	8	8	4	4
Cache-inhibited load queues	16	8	8	8	4	4

Note: PEC0 is a dedicated-only stack. PEC1 is a bifurcated-only stack, and PEC2 can be programmed to be dedicated, bifurcated, or trifurcated.

2.2 CAPP Mode Features

- Supports three programmable I/O ranges per PCI stack (one PHB register and two PCI-bus memory)
- 32 inbound write queues per PBCQ, each with a 128-byte data buffer to support the following:
  - CAPP packets
  - DMA writes
  - Atomic operations
  - Interrupt requests
- 64 read queues per PBCQ (PEC0 has128 read queues), each with a 128-byte data buffer to support:
  - Up to 48 read queues can be used for CAPP APC master machine support
  - DMA read queues.
  - Support for NMMU responses
- 32 NMMU operation queues
- 16 cache-inhibited load queues per PBCQ, each with a 32-byte data buffer
- 16 cache-inhibited store queues per PBCQ, each with a 128-byte buffer to support
  - Up to 14 six-CAPP message queues
  - At least two cache-inhibited stores
- Freeze-mode support

Table 2-2. PBCQ CAPP Mode Resource Allocation by Stack

Resource	Dedicated	Bifurcated		Trifurcated		
	Stack 0	Stack 0	Stack 1	Stack 0	Stack 1	Stack 2
DMA write queues (shared with all write packet types)	32	16	16	16	8	8
DMA read queues	Minimum 16	Minimum 1	32	Minimum 1	16	16
CAPP APC	Maximum 48	Maximum 31	N/A	Maximum 31	N/A	N/A
Cache-inhibited store queues	Minimum 1	Minimum 1	8	Minimum 1	4	4
CAPP MSG	Maximum 15	Maximum 7	N/A	Maximum 7	N/A	N/A

Notes:
1. PEC0 is a dedicated only stack, PEC1 is a bifurcated only stack, and PEC2 can be programmed to be dedicated, bifurcated, or trifurcated.
2. Resources assigned to CAPP are controlled by the CAPP Control Register.
3. CAPP can only be connected to PEC0/stack 0 (PHB0) and PEC2/stack 0 (PHB3).

2.3 PBCQ Outbound Interface

The PBCQ uses the SMP interconnect reflected command interface to monitor for address/ttypes destined for the PCIe bus or PHB.

Each stack has three Base Address Registers (BARs). *Table 2-3* lists the SMP interconnect commands that the PCI unit accepts. The reflected command is made up of four ports, but a single stack can accept at most two commands per cycle across the four ports.

Table 2-3. Outbound SMP interconnect Reflected Command Summary (Page 1 of 2)

SMP Interconnect Command	Accept Condition	Partial Response	AIB Command
Cache-inhibited (CI) partial read (**ci_pr_rd**) Aligned sizes of 1, 2, 4, 8, 16, or 32 bytes	I/O BAR hit	lpc_ack if queue is available lpc_ack and rty_other if all load queues are full	ReadReq.CI
Cache-Inhibited partial out of order write (**ci_pr_ooo_w**) Aligned sizes of 1, 2, 4, 8, 16, 32, 64, or 128 bytes	I/O BAR hit	lpc_ack if queue is available lpc_ack and rty_other if all store queues are full	WriteReq.CI
Cache-inhibited (CI) partial write (**ci_pr_w**) Aligned sizes of 1, 2, 4, 8, 16, 32, 64, or 128 bytes	I/O BAR hit	lpc_ack if queue is available lpc_ack and rty_other if all store queues are full	WriteReq.CI
Cache-line DMA write (**cl_dma_w**) Aligned sizes of 1, 2, 4, 8, 16, 32, 64, or 128 bytes	I/O BAR hit and peer-to-peer mode	lpc_ack if queue is available lpc_ack and rty_other if all store queues are full	WriteReq.CI
Partial cache-line DMA write (**pr_dma_w**) Aligned sizes of 1, 2, 4, 8, 16, 32, 64, or 128 bytes	IO BAR hit and peer-to-peer mode	lpc_ack if queue is available lpc_ack and rty_other if all store queues are full	WriteReq.CI

1. Report hang commands are used for the SMP interconnect hang protocol.

Table 2-3. Outbound SMP interconnect Reflected Command Summary (Page 2 of 2)

SMP Interconnect Command	Accept Condition	Partial Response	AIB Command
Cache-line DMA inject (**cl_dma_inj**) Aligned sizes of 1, 2, 4, 8, 16, 32, 64, or 128 bytes	IO BAR hit and peer-to-peer mode	lpc_ack if queue is available lpc_ack and rty_other if all store queues are full	WriteReq.CI
Partial cache-line DMA inject (**pr_dma_inj**) Aligned sizes of 1, 2, 4, 8, 16, 32, 64, or 128 bytes	IO BAR hit and peer-to-peer mode	lpc_ack if queue is available lpc_ack and rty_other if all store queues are full	WriteReq.CI
Report hang[1] (**rpt_hang.***)	–	–	–

1. Report hang commands are used for the SMP interconnect hang protocol.

2.3.1 Outbound Ordering Rules

Table 2-4 lists the ordering rules for commands sent outbound from the SMP interconnect to the AIB.

Table 2-4. Outbound Ordering Rules

Row Pass Column?	CI Store	CI OOO Store	Peer-to-Peer	CI Load
CI Store	No	No	Yes	Yes
CI OOO Store	Yes	Yes	Yes	Yes
Peer-to-Peer	Yes	Yes	No	Yes
CI Load	No	No	Yes	Yes

2.3.2 Outbound SMP Interconnect Tag Identifiers

Table 2-5 defines the PEC acknowledge tags.

Table 2-5. PEC Acknowledge Tag Definition

Acknowledge Tag(7:21)									Description
7:13	14	15	16	17	18	19	20	21	
11101UU	0	0	1	0	S	S	S	S	SSSS = Cache-inhibited store queue 0 - 15
Note: UU = 00 for PEC0; UU = 01 for PEC1; UU = 10 for PEC2; tTag (0:6) = tc_pe_group_id_dc(0:3) & tc_pe_chip_id_dc(0:2)									

Table 2-6 defines the PEC route tags.

Table 2-6. PEC Route Tag Definition (Page 1 of 2)

RTag(7:21)									Description
7:13	14	15	16	17	18	19	20	21	
11101UU	0	0	0	W	W	W	W	W	WWWWW= DMA write queue 0 - 31 (for atomics)
11101UU	0	0	1	0	S	S	S	S	SSSS = Cache-inhibited store queue 0 - 15
11101UU	0	0	1	1	0	M	M	M	MMM = NMMU checkout response 0 - 7
Note: tTag (0:6) = tc_pe_group_id_dc(0:3) & tc_pe_chip_id_dc(0:2)									

Table 2-6. PEC Route Tag Definition (Page 2 of 2)

RTag(7:21)									Description
7:13	14	15	16	17	18	19	20	21	
11101UU	0	0	1	1	1	-	-	-	Reserved
11101UU	0	1	-	-	-	-	-	-	Reserved
11101UU	1	R	R	R	R	R	R	R	RRRRRRR = DMA read queue 0 - 127 for PEC0, 0 - 63 for PEC1 and PEC2
Note: tTag (0:6) = tc_pe_group_id_dc(0:3) & tc_pe_chip_id_dc(0:2)									

2.4 Outbound Operation Summary

The following sections list the steps involved in the outbound operations.

2.4.1 Cache-Inhibited Read

A valid reflected command (RCMD), with ttype = ci_pr_rd, whose address matches one of the BARs is accepted if a load queue is available. If no load queues are available, a partial responses of **rty_lpc** is returned.

If the command is accepted, the following responses occur on the SMP interconnect:

- A partial response of lpc_ack.
- An INFO type aTag is provided.

The following information is stored for later use:

- Address and size in the PBCQCRF.
- Transfer tag (tTAG) in the TTAGCRF.
- Ticket captured in the TkTCAM.
- Ordering prerequisites are captured.

The combined response (cresp) bus is monitored until the stored ticket is matched with a clean cresp. If the cresp is of type *retry*, the queue is released and the command is forgotten. After all the prerequisites are met, the request is sent to the AIB bus through the speed matching buffer (SMB). When the AIB bus returns the load data, the inbound logic signals the load queue that the data is ready. The returned data is scheduled to be sent on the SMP interconnect using the tTAG stored in the TTAGCRF as the data routing tag (DTAG).

2.4.2 Cache-Inhibited Write and Peer-to-Peer Write

A valid RCMD with ttype = ci_pr_wr (if peer-to-peer ttypes can also include: **cl_dma_w**, **cl_dma_inj**, **pr_dma_w**, and **pr_dma_inj**) whose address matches one of the BARs is accepted if a store queue is available. If no store queues are available, a partial responses of **rty_lpc** is returned.

If the command is accepted, the following responses occur on the SMP interconnect:

- A partial response of lpc_ack.
- A routing tag type aTag is provided.

The following is stored for later use:

- Address and size in the PBCQCRF.
- Ticket captured in the TkTCAM.
- Ordering prerequisites are captured.

The cresp bus is monitored until the stored ticket is matched with a clean cresp. If the cresp is of type *retry*, the queue is released and the command is forgotten. After all the prerequisites are met and the data is received, the request is sent to the AIB through the SMB.

2.4.3 CAPP Message Write

If the command is accepted, the following responses occur on the SMP interconnect:

1. A valid message is receive on the PEC/CAPP message sideband interface.
2. A store queue is allocated.
3. The prerequisites are captured.
4. After all of the prerequisites are met (including receiving all data), the request is sent to the AIB.
5. When the message has been completed on the AIB and the state machine is freed, the CAPP is signaled that the message buffer can be reused with the pe_cxa_msg_free_v signal.

2.4.4 APC Master Reads

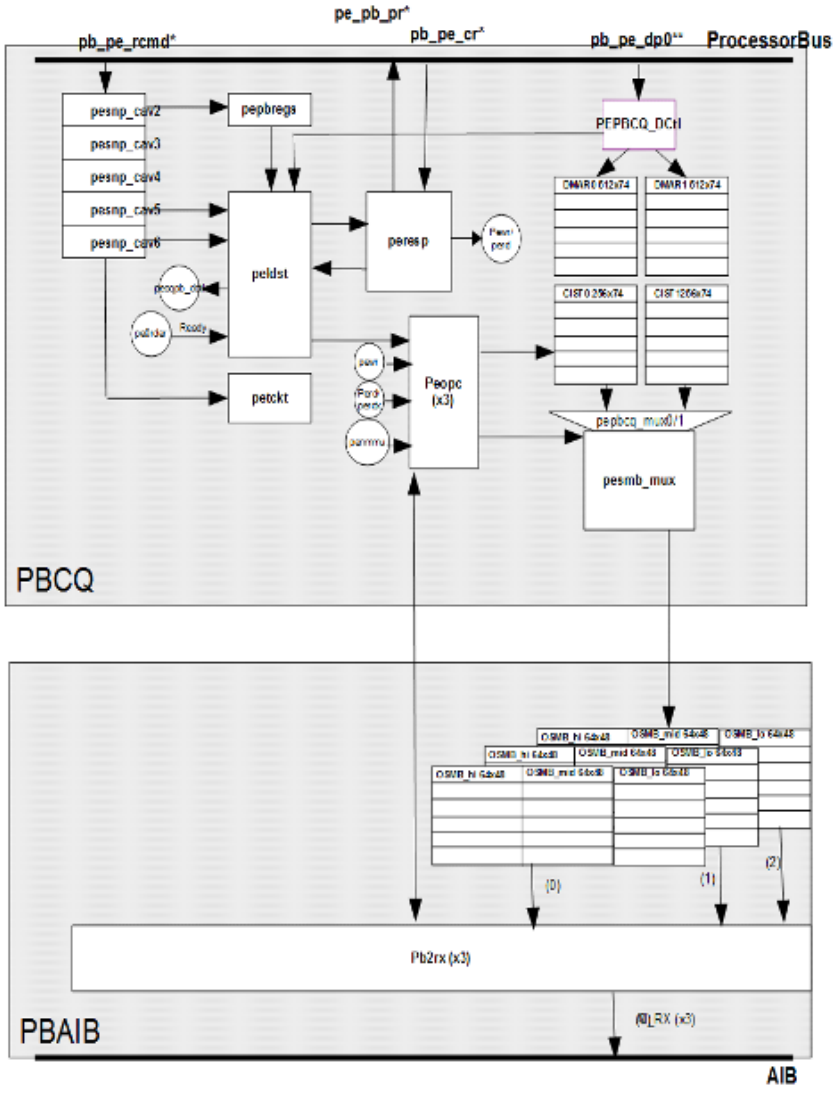
If the command is accepted, the following responses occur on the SMP interconnect:

1. An outbound SMP interconnect data transfer is received with an rTag matching a read queue under control of the APC.
2. APC master reads are always 128 bytes. There are no ordering requirements for the data transfers.
3. As the data is received, sfStat information is reflected back on the PEC/CAPP sfStat interface.
4. After all the data has been received for this request, it is sent to the AIB as a write.
5. When the operation has completed on the AIB, the CAPP is signaled that the data buffer can be reused.

2.5 Outbound Data Flow Diagram

Figure 2-1 shows the outbound data flow logic.

Figure 2-1. Outbound Second-Level Diagram



2.6 PBCQ Inbound Interface

When in I/O mode, the PBCQ uses the SMP interconnect command interface to initiate requests for the PHB. *Table 2-7* summarizes the SMP interconnect ttypes that the PHB initiates.

Table 2-7. Inbound SMP Interconnect Command Summary

AIB Command	SMP Interconnect Transaction	Comment
ReadReq.DMA[1]	Cache Line (CL) DMA read RWNITC (**cl_rd_nc**)	All read lengths are requested as a cache line to improve the chances of intervention data.
ReadReq.TCE[1]	Cache Line (CL) DMA read RWNITC (**cl_rd_nc**)	All read lengths are requested as a cache line to improve the chances of intervention data.
WriteReq.DMA[2]	Cache Line (CL) DMA Write – I (**cl_dma_w_i**), Cache Line (CL) DMA Cache Inject (**cl_dma_inj**) Partial Cache Line (CL) DMA Write (**dma_pr_w**)	
Tunnel Command: Atomic	Atomic Command (**armw***/**armwf***)	SMP interconnect command type is based on tunnel command metadata. If **armwf,*** a response is sent back as an outbound write.
Tunnel Command: ASB_Notify	ASB notify command (**asb_notify**)	A response is sent back as an outbound write.
WriteReq.CI[2]	Interrupt Notify (**pr_dma_w**)	
N/A	Background Kill (**bkill**)	In response to an **addr_ack_bk_*** combined response for a command sourced by a DMAW or Atomic state machine.

1. All other AIB read request sub-commands are not supported.
2. All other AIB write request sub-commands are not supported.

2.7 Inbound Ordering Rules

2.7.1 PCIe-Only Mode

Table 2-8 lists the inbound order rules.

Table 2-8. Inbound Order Rules[1]

Row/Pass Column	Write	Read	Tunnel	Interrupt Notify[4]
Write	No	Yes	Yes	Yes
Read	No	Yes	Yes[2]/No[3]	Yes
Load Response	No	Yes	Yes	Yes
Tunnel	No	Yes	Yes	Yes
Interrupt Notify[4]	No	Yes	Yes	Yes

1. Order rules only apply to packets with the same PE number. The exception is the read versus tunnel rules, which apply across all reads.
2. If read versus tunnel order mode is for only on zero byte reads, the reads greater than zero bytes can pass tunnel packets.
3. If read versus tunnel order mode is for all size reads, the reads cannot pass tunnels.
4. PCIe interrupt ordering rules are in effect to the interrupt source controller in the PHB. The interrupt notify is the trigger to the interrupt presentation controller.

Typically, a command is considered “complete” from an ordering standpoint when a clean combined response is received. The following two configurable mode bits change the ordering behavior:

- Strict Ordering Mode: Used only for debug. Commands are sent to the SMP interconnect in the order they arrive at the AIB.
- Channel Streaming Mode: DMA write operations do not wait for a clean combined response. Instead, they use the SMP interconnect architected channel streaming tag for DMA write operations.

2.7.2 CAPP Mode

In CAPP mode, all I/O ordering rules apply for I/O packets. CAPP are not considered I/O packets, do not push any I/O packets, and I/O packets do not push CAPP packets.

2.8 Command Scope

Table 2-9 describes the minimum scope attempted by the PBCQ for each command.

Table 2-9. Minimum Command Scope

Command	Minimum Scope
Write: **cl_dma_w**, **cl_dma_inj**, **pr_dma_w**, **pr_dma_inj**	Group
Read: **cl_dma_rd**	Node
Atomic: **armw***, **armwf***	Group
asb_notify	Vector Group (all 1's)
Interrupt Notify: **ci_pr_w**	Local Node

2.9 Inbound Transaction Identifiers

Table 2-10 defines how the PEC assigns the tTags.

Table 2-10. PEC Transaction ID Definition

Transaction Tag(7:21)									Description
7:13	14	15	16	17	18	19	20	21	
11101UU	0	0	0	W	W	W	W	W	WWWWW = DMA write queue 0 - 31
11101UU	0	0	1	–	–	–	–	–	Reserved
11101UU	0	1	–	–	–	–	–	–	Reserved
11101UU	1	R	R	R	R	R	R	R	RRRRRRR = DMA read queue 0 - 127 for PEC0 and 0 - 63 for PEC1 and PEC2
Note: tTag (0:6) = tc_pe_group_id_dc(0:3) & tc_pe_chip_id_dc(0:2)									

2.10 DMA Write Cache Inject Mode

Cache injection of write commands are controlled by the PBCQ Hardware Configuration Register bits 34:35. The four modes on when to issue a cache inject are:

1. Initial attempt as an inject. This mode attempts a DMA write as a cache inject. If the cache does not contain the line, the memory controller accepts the command instead of a retry occurring.
2. The choice to cache inject is inherited from the stream. If the command is the first in a stream, PBCQ Hardware Configuration Register bit 36 indicates whether to start with an inject or a non-inject command.
3. The choice to inject is determined by the TLP hint bits. If either hint bit equals '1', the command attempts a cache inject.
4. No cache inject.

2.11 Inbound Operations Summary

2.11.1 DMA Read

A DMA read command consists of the following steps:

1. A **ReadReq.DMA** command is received from the AIB.
2. A DMA read command along with the address, size, and node ID are written into the SMB. An asynchronous handshake to the nest clock domain occurs.
3. The SMB is read, the packet is assigned to a fragmentation engine, and ordering requirements are recorded.
4. With each fragment of the packet, a free DMA read queue is selected and the DMA read information is written to the inbound array.
5. After all ordering prerequisites are met and the command works its way to the head of the request queue, the command is sent to the SMP interconnect.
6. The combined response bus is monitored until this command's cresp is received. A retry cresp causes the command to be resent. A clean cresp causes the state machine to wait for the returned data.
7. After the data is received from the SMP interconnect, an **AIB ReadResponse.DMA** command is built and sent through the SMB to the AIB.
8. The DMA read queue is freed.

2.11.2 DMA Write

A DMA write command consists of the following steps:

1. A **WriteReq.DMA** command is received from the AIB.
2. A DMA write command along with the address, size, and node ID are written into the SMB. An asynchronous handshake to the nest clock domain occurs.
3. The SMB is read, a free DMA write queue is selected, and the DMA write information is written to the inbound array.
4. Ordering with pre-existing commands in flight is determined and recorded.
5. After all ordering prerequisites are met, and the command works its way to the head of the request queue, the command is sent to the SMP interconnect.
6. The combined response bus is monitored until this command's cresp is received. A retry cresp causes the command to be resent. A clean cresp causes the state machine to send data to the SMP interconnect.
7. For writes, this completes the command and the DMA write state machine is freed.

2.11.3 Interrupt Notify

An interrupt notify command is comprised of the following steps:

1. A **WriteReq.CI** command is received from the AIB.
2. An interrupt notify command along with the interrupt information and node ID are written into the SMB. An asynchronous handshake to the nest clock domain occurs.
3. The SMB is read, a free write queue is selected, and the Interrupt information is written to the inbound array.
4. Ordering with pre-existing commands in flight is determined and recorded.
5. After all ordering prerequisites are met and the command works its way to the head of the request queue, the command is sent to the SMP interconnect.
6. The combined response bus is monitored until this command's cresp is received. A retry cresp causes the command to be resent.
7. For interrupts, this completes the command and the write state machine is freed.

2.11.4 Inbound CAPP Write

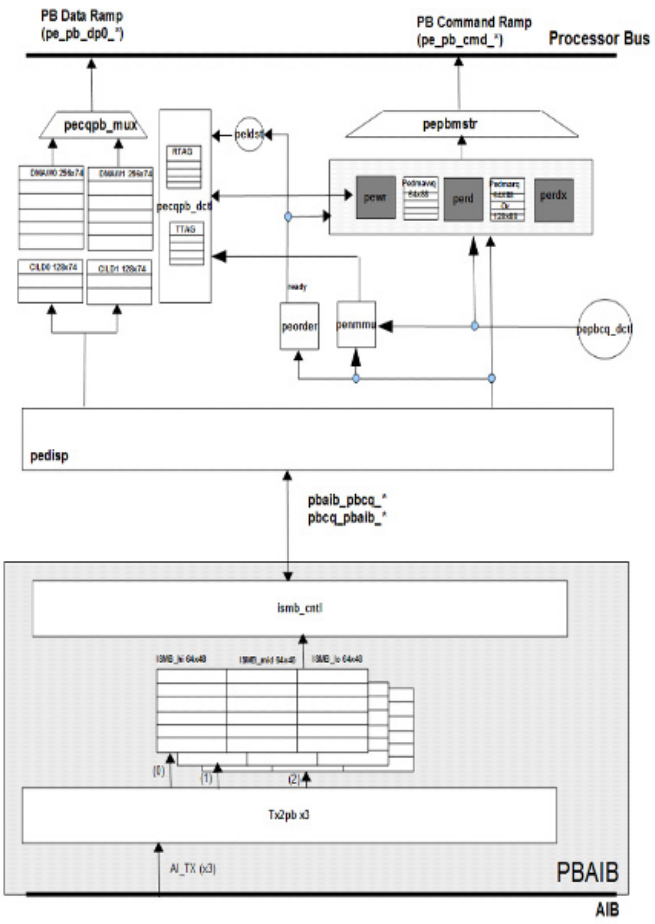
An inbound CAPP write command is comprised of the following steps:

1. A **WriteReq.DMA** command is received from the AIB.
2. A DMA write command along with the address, size, and node ID are written into the SMB. The asynchronous handshake to the nest clock domain occurs.
3. The SMB is read, a free DMA write queue is selected, and the DMA write information is written to the inbound CAM.
4. If F = 0 in the inbound PCI address, the data is scheduled for immediate transfer to the SMP interconnect data ramp using the rTag in the PCI address. If F = 1, the PEC/CAP rTag interface is used to request the rTag for this transaction.
5. After the rTag is available, all ordering prerequisites are met, the command works its way to the head of the request queue, and the data is sent to the SMP interconnect.
6. When the data transfer is complete, the DMA write state machine is freed.

2.12 Inbound Data Flow Diagram

Figure 2-2 illustrates the inbound second-level data flow.

Figure 2-2. Inbound Second-Level Diagram



2.13 Peer-to-Peer Mode

This section describes the peer-to-peer mode.

2.13.1 Outbound Operation

A peer-to-peer (P2P) command is defined as a DMA write command that hits the MMIO address bar. It uses the same outbound resources as an MMIO write and has the same size limitations of 1, 2, 4, 8, 16, 32, 64, and 128 bytes. The P2P packets stay in order, but do not have any order relationship with other outbound operations.

2.14 PBCQ Error Handling

There are four types of errors in the PBCQ. Errors are collected in the FIR register (NFIR) and the action taken on individual errors is programed in the Action 0/1 Registers (NFIRAction0 and NFIRAction1).

For the NFIRAction0 and NFIRAction1 registers, the available actions are:

- No action (‘10’).
- A recoverable error interrupt (‘01’) is sent to the service processor for handling. No other hardware action is taken (INF error).
- A checkstop error (‘00’) is sent to the chip logic to shut down the chip. No other hardware action is taken (fatal error).
- Freeze the PCI interface (‘11’). See *Section 2.14.1. PBCQ Freeze Behavior* for full details (PHB recoverable error).

2.14.1 PBCQ Freeze Behavior

When a freeze is detected, the following events occur:

- A **ai_rx_raise_fence** command is activated on the AIB and stops the ETU from sending data.
- All outbound CI stores are dropped (SMP interconnect commands receive presp lpc_ack but are discarded in the PEC).
- All outbound CI loads return all ones data to the SMP interconnect.
- All previously started DMA writes and interrupt operations will complete on the SMP interconnect. The data sent is determined by the Data Freeze Register.
- All previously started DMA reads will complete on the SMP interconnect and the data discarded.
- Any commands that have not started on the SMP interconnect are dropped.
- In CAPP mode, all outbound messages and APC Master reads are dropped. All inbound commands are discarded, including any commands waiting on rTag.
- In CAPP mode, pe_cxa_linkdown is raised.

The following sequence is required to exit a freeze:

- Reset the ETU (Bit 0 of the PHB Reset Register).
- Wait for all SMP interconnect traffic to complete by polling the inbound and outbound active signals (CQStat bits 0:1).
- Clear the FIR errors, this causes the freeze condition to be cleared.
- Take the ETU our of reset and re-initialize the link.

2.14.2 PHB EEH Support

When an enhanced error handling (EEH) error occurs, DMA reads must be returned with a completer abort.

If the PHB detects the EEH is in error before sending the DMA read to the PBCQ, it marks it with a completer abort flag (bit 32 of the AIB command, which is bit 0 of the address field). When the PBCQ detects this bit, it does not present this command to the SMP interconnect and returns the completion response on the outbound completion forwarding interface (OCF) with a completer abort status.

A PE EEH error can occur after a DMA read has been sent to the PBCQ. In this situation, the PHB sends the EEH state via the PE state table (PEST) interface.

The PBCQ only uses the EEH status information on read completion packets that are destined for the OCF. If the EEH bit is active, the first completion after detecting a bad EEH state is returned with a completer abort. Any other fragments of a large packet after the completer abort are thrown away before presenting to the OCF. There is no attempt to block the sending of subsequent fragments to the SMP interconnect. A completer abort response is a header only completion transaction with an abort completion status.

2.14.3 PBCQ SMP Interconnect Hang Behavior

The PBCQ follows the SMP interconnect hang protocol. The PEC has a configurable hang poll divider (see *Section 5.1.1 PBCQ Hardware Configuration Register* on page 47) that enables the PEC hang rate to be adjusted from the rate the SMP interconnect is using.

Table 2-11 describes the hang window for each command.

Table 2-11. PEC Bus Master Command Hang Behavior

Command	Command Hang Window Duration	Action on rpt_hang.poll	rpt_hand.(f)check
cl_rd_nc	From address request to data arrival	Idle → Busy (rtp_hang.poll) Busy → Hang Hang → Drive rty_other	Hang → drive rty_other
ci_pr_w **cl_dma_w_i** **cl_dma_inj** **dma_pr_w** **dma_pr_inj** **armw_***	W0: From address request to cresp W1: From cresp to data sent		
armwf_*	W0: From address request to cresp W1: From cresp to data arrival		

Table 2-12 summarizes the PEC data hang behavior.

Table 2-12. PEC Data Hang Behavior

Command	Command Hang Window Duration	Action on rpt_hang.poll
ci_pr_w **ci_pr_ooo_w**	From cresp to data arrival	Idle → Busy (rtp_hang.data) Busy → Hang Hang → Set data hang FIR
Note: No other master or slave operations participate in the data hang protocol.		

3. Tunnel Operations

The PCIe controller (PEC) unit provides a tunnelling ability that supports atomic and ASB notify SMP interconnect commands from a PCIe device.

A tunnel operation is initiated by a posted DMA write that matches the most-significant 16 bits of the PCIe address with that of the PHB ASN Compare/Mask Register. In addition to any data required by the tunnel operation, the posted write also passes a 16-byte metadata field preceding the operation data. This metadata field defines the type of tunnel operation along with other control information required to perform the operation.

Table 3-1 describes the tunnel metadata field.

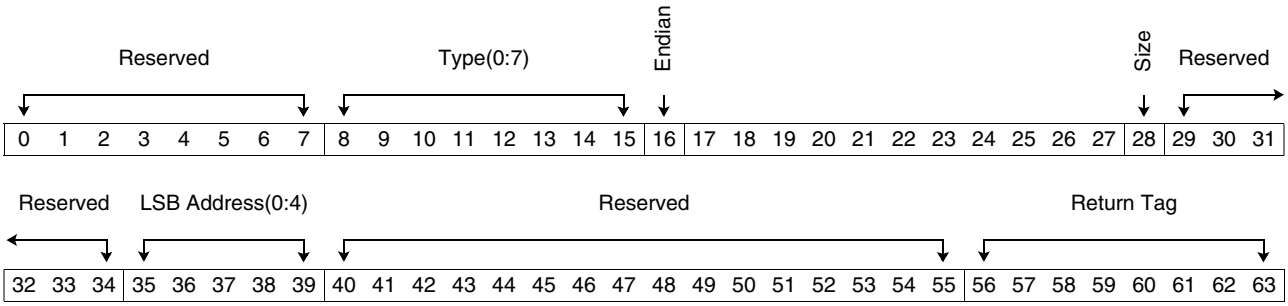


Table 3-1. Tunnel Metadata Field

Bits	Field Name	Description
0:7	Reserved	Reserved. Fields should be zeros.
8:15	Type(0:7)	Type(0:1) = '00': Atomic operation Type(0:1) = '01': ASB notify operation Type(2:7) is defined in the *Table 3-8 Atomic Type* on page 40.
16	Endian	0 Big endian. 1 Little endian (only used for atomic operations)
28	Size	0 4 -byte operation. 1 8-byte operation (only used for atomic o.perations).
29:34	Reserved	Reserved. Fields should be zeros.
35:39	LSB Address(0:4)	Five bits to replace the lower five bits of the posted write address. Because of the metadata, the lower five bits of the address might not reflect the lower five bits required on the SMP interconnect operation. These five bits replace the lower address bits of the posted write. **Note:** LSB_Address(3:4) should always be zeros.
40:55	Reserved	Reserved. Field should be zeros.
56:63	Return Tag	Tag used to correlate the inbound operation with the response.

The tunnelled command is ordered behind the posted write commands that precede it. All commands, including other tunnelled commands, can pass tunnelled commands. If the tunnelled command has a response, it is returned via a posted MMIO write initiated by the PEC unit. The base address of the response is defined in the Tunnel Bar Register.

Table 3-2 describes the tunnel response address.

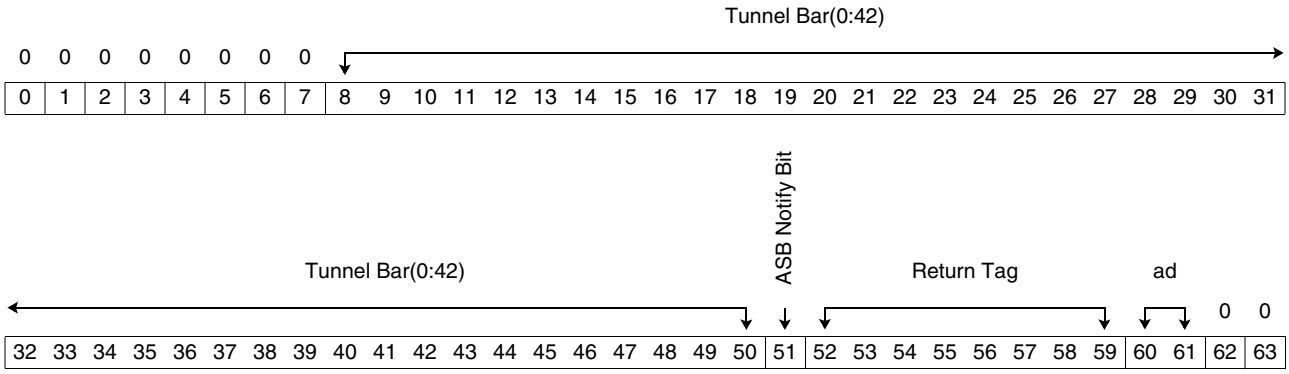


Table 3-2. Tunnel Response Address

Bits	Field Name	Description
0:7	Reserved	Reserved. Field should be zeros.
8:50	Tunnel Bar(0:42)	The Tunnel Bar must map into the MMIO space of the PHB.
51	ASB Notify Bit	0 Response is atomic. 1 Response is ASB notify.
52:59	Return Tag	This is the return tag from the metadata.
60:61	ad	LSB_Address(1:2) from metadata for an atomic response. 00 Value for an ASB_notify response.
62:63	Reserved	Reserved. Field should be zeros.

3.1 ASB Notify

The ASB notify command on the SMP interconnect is an address-only command. The only data passed via the posted write is the metadata; thus, the posted write command is a 16-byte write. The ASN match must be defined over the most-significant eight bits of the address, because the ASB notify operation on the SMP interconnect defines the address bits 8:15 as zeros. The LPAR is a 12-bit field and defined in the least-significant bits of the address; thus, the lower five bits of the LPAR are in the metadata.

Table 3-3 shows the ASB notify posted-write address fields.

Table 3-3. ASB Notify Posted Write Address Field

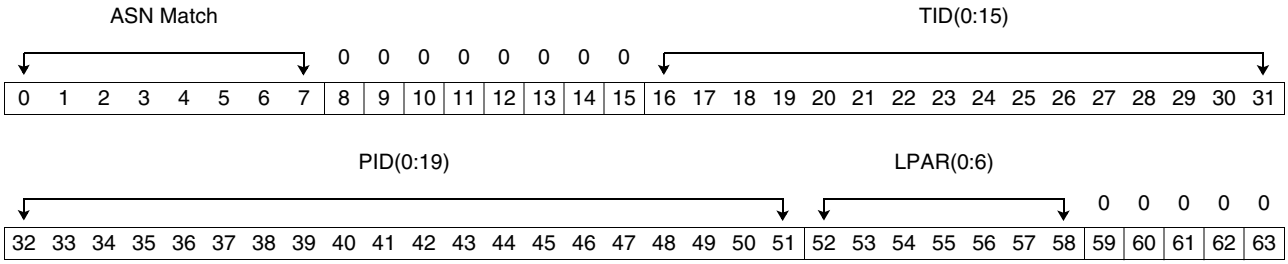


Table 3-4 describes the ASB notify posted-write metadata fields.

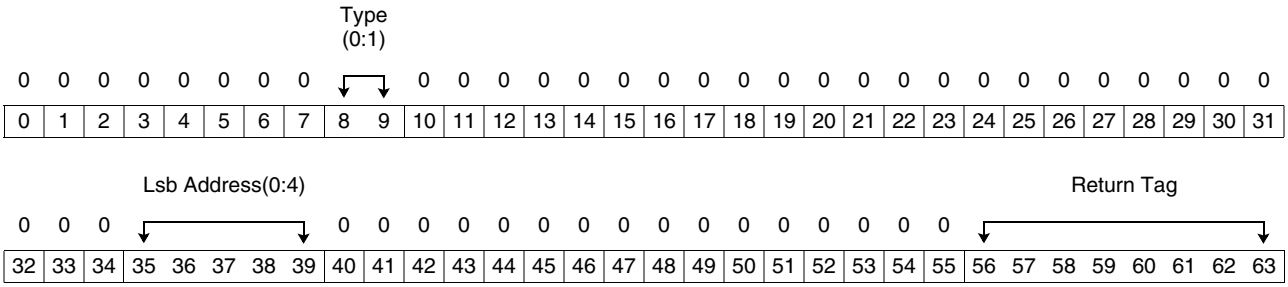


Table 3-4. ASB Notify Posted Write Metadata

Bits	Field Name	Description
8:9	Type(0:1)	ASB notify operation
35:39	Lsb Address(0:4)	The five bits to replace the lower five bits of the posted write address. Because of the metadata, the lower five bits of the address might not reflect the lower five bits required on the SMP interconnect operation. These five bits replace the lower address bits of the posted write. **Note:** LSB_Address(3:4) should always be zeros.
56:63	Return Tag	Tag used to correlate the inbound operation with the response.

The response of the ASB notify command is a 1-byte write. The field is all zeros for a successful operation and all ones for a failed operation. *Table 3-5* describes the ASB notify response address.

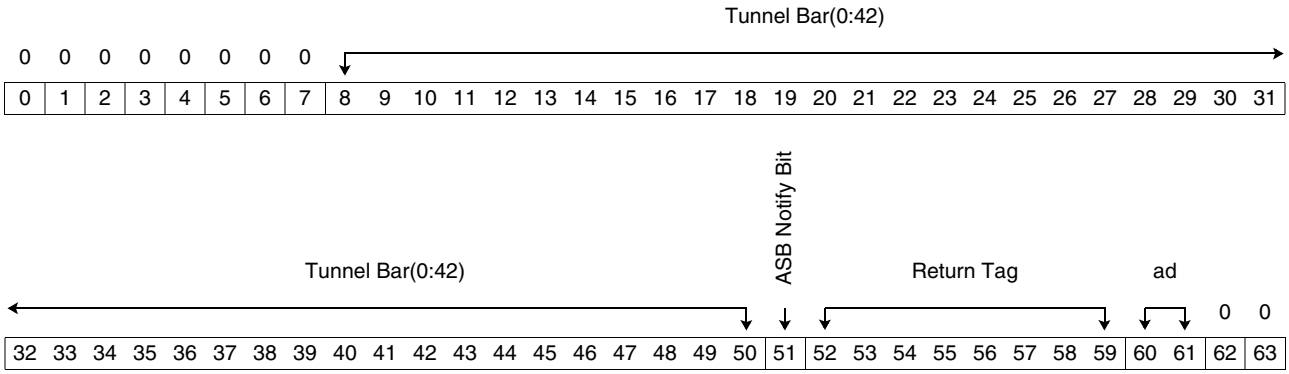


Table 3-5. ASB Notify Response Address

Bits	Field Name	Description
8:50	Tunnel Bar(0:42)	The Tunnel Bar must map into the MMIO space of the PHB.
51	ASB Notify Bit	1 Response is ASB notify.
52:59	Return Tag	This is the Return tag from the metadata.
60:61	ad	‘00’ Lsb_addr(1:2) from the metadata.

Table 3-6 summarizes the ASB notify response data.

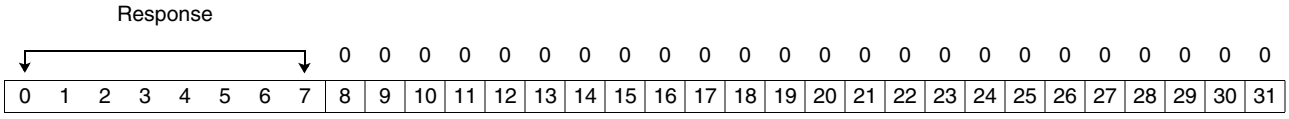


Table 3-6. ASB Notify Response Data

Bits	Field Name	Description
0:7	Response	‘00000000’ Successful ‘11111111’ Unsucessful

3.2 Atomics

Atomic operations are presented as 32-byte posted writes. The first 16 bytes are the metadata and the second 16 bytes are the atomic operands, which are dependent on the size and type of the atomic operation. The address of the posted write is the destination address of the atomic operation with the exception of the least-significant five bits, which are passed through the metadata.

Atomic operations can be on a 4-byte or 8-byte field, have one or two operands, and have a response based on the type. If a response is returned, it is a posted write of either four or eight bytes with the alignment as defined by the Lsb address bits.

Figure 3-1. Atomic Posted Write Address Field

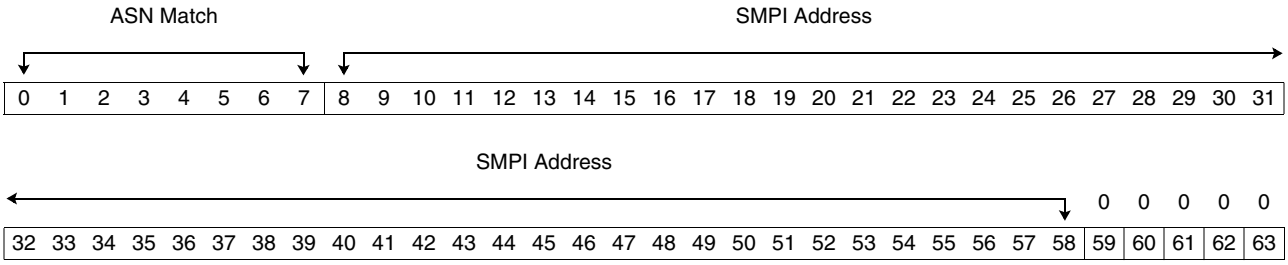


Figure 3-2. Atomic Posted Write Metadata

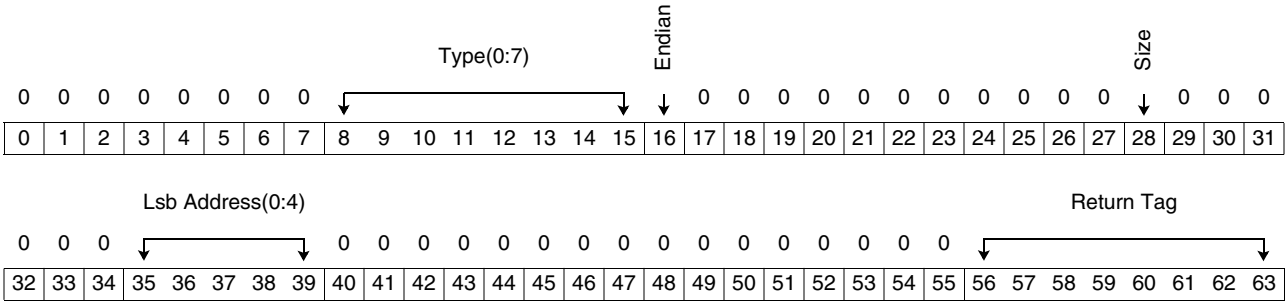


Table 3-7. Atomic Posted Write Metadata

Bits	Field Name	Description
8:15	Type(0:7)	Type(0:1) = '00'. See *Table 3-8 Atomic Type* on page 40 for Type(2:7) definition.
16	Endian	0 Big endian. 1 Little endian.
28	Size	0 4-byte operation. 1 8-byte operation (only used for atomic operations).
35:39	Lsb Address(0:4)	Five bits to replace the lower five bits of the posted write address. Because of the metadata, the lower five bits of the address might not reflect the lower five bits required on the SMP interconnect operation. These five bits replace the lower address bits of the posted write. **Note:** Lsb_Address(3:4) should always be zeros.
56:63	Return Tag	Tag used to correlate the inbound operation with the response.

The Atomic type and 16-byte data are defined in *Table 3-8 Atomic Type*, *Table 3-9 Alignment of 16-Byte Data for Atomic Posted Write* on page 40, and *Table 3-10 Atomic with Fetch Response Address* on page 41.

Table 3-8. Atomic Type

Value	Command	Response
000000	Fetch and ADD	Y
000001	Fetch and XOR	Y
000010	Fetch and OR	Y
000011	Fetch and AND	Y
000100	Fetch and Maximum Unsigned	Y
000101	Fetch and Maximum Signed	Y
000110	Fetch and Minimum Unsigned	Y
000111	Fetch and Minimum Signed	Y
010000	Compare and Swap Not Equal	Y
010001	Compare and Swap Equal	Y
001000	Compare and Swap Unconditional	Y
011000	Fetch and Increment Bounded	Y
011001	Fetch and Increment Equal	Y
011100	Fetch and decrement Bounded	Y
100000	Store ADD	N
100001	Store XOR	N
100010	Store OR	N
100011	Store AND	N
100100	Store Maximum Unsigned	N
100101	Store Maximum Signed	N
100110	Store Minimum Unsigned	N
100111	Store Minimum Signed	N
111000	Store Twin	N

Table 3-9 shows the legal operand alignment and data placement for atomic operands.

Table 3-9. Alignment of 16-Byte Data for Atomic Posted Write

Address 60:63	Byte															
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0000	V								W							
	V								W							
0100					V								W			
1000	W								V							
	W								V							
1100					W								V			

The response of the Atomic with fetch command is either a 4-byte or 8 byte write depending on the ‘s’ field in the metadata. *Figure 3-3* and *Table 3-10* detail the Atomic fetch with response address.

Figure 3-3. Atomic with Fetch Response Address

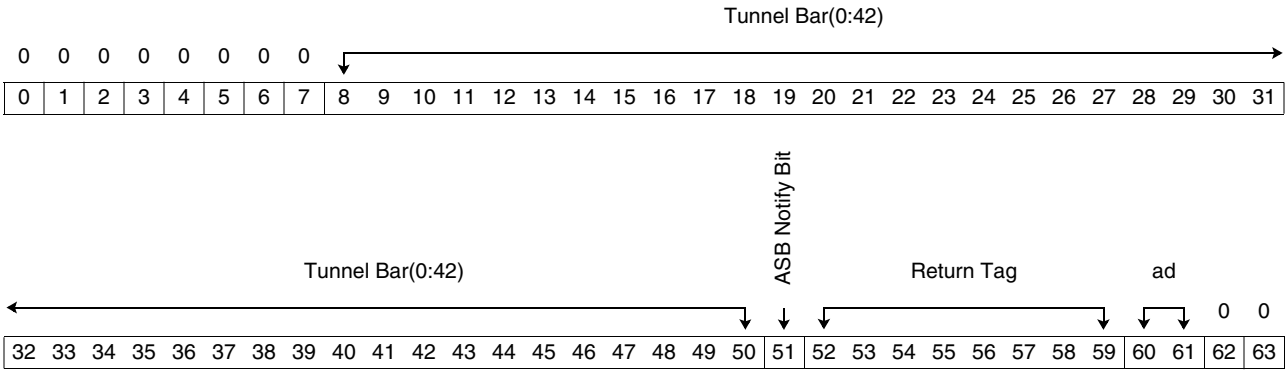


Table 3-10. Atomic with Fetch Response Address

Bits	Field Name	Description
8:50	Tunnel Bar(0:42)	The Tunnel Bar must map into the MMIO space of the PHB.
51	ASB Notify Bit	0 Response is atomic.
52:59	Return Tag	This is the return tag from the metadata.
60:61	ad	lsb_addr(1:2) from metadata.

Figure 3-4 describes the 4-byte atomic response data and *Figure 3-5* describes the 8-byte atomic response data.

Figure 3-4. Atomic Response Data (4-Byte Data)

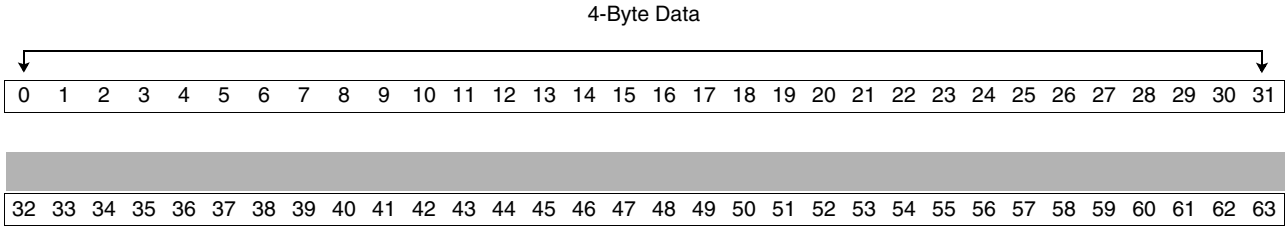
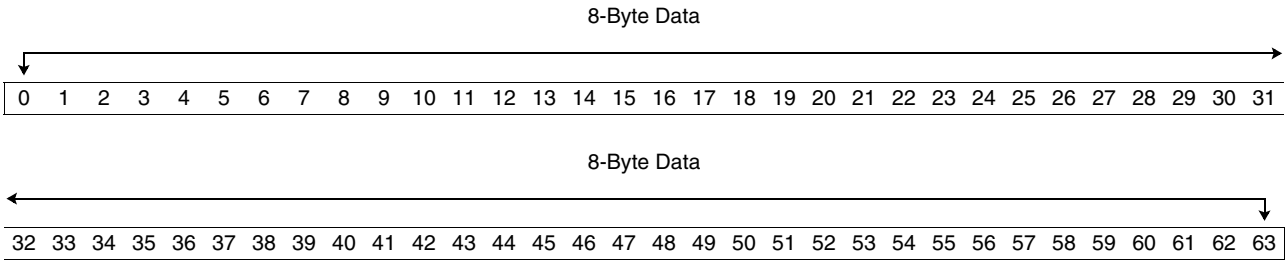


Figure 3-5. Atomic Response Data (8-Byte Data)



4. PBCQ CAPP Features

4.1 CAPP Mode

In CAPP mode, the PBCQ is a data transport layer for CAPP. There is an inbound path, where a CAPI device sends packets to the processor chip and an outbound path where the CAPP unit directs packets to the external CAPI device. The PEC shares its existing inbound and outbound buffers and controllers with the CAPP unit.

4.1.1 Inbound CAPP Traffic

The CAPI device can send both CAPP packets and I/O packets. A PCIe packet is identified as a CAPP packet in the PHB if the PCIe address matches either the CAPI Compare/Mask Register or its non-blocking writes (NBW) Compare/Mask Register. Details of these registers are in the *Power Systems Host Bridge 4 (PHB4) Specification*.

Most CAPP packets are PCIe-posted write packets. Therefore, they share the DMA write resources of the PEC. Normal CAPP write packets are data-only transfers on the SMP interconnect to either an rTag included in the overloaded address or an rTag that the CAPP unit provides on a side-band interface. The CAPP traffic, I/O or CAPP packets, dynamically share the PEC DMA write buffers and controllers.

The NMMU response packet is the only type of CAPP read defined.

4.1.2 Outbound CAPP Traffic

There are two types of outbound CAPP traffic: APC and MSG. To move APC-type traffic, some of the PEC read buffers and controllers are controlled via the CAPP engine using a sideband interface. MSG-type traffic uses the PEC MMIO store buffers and controllers. When these resources are assigned to CAPP, they are not available for normal I/O function. The number of read and MMIO store resources assigned to the CAPP is controlled by the CAPP Control Register.

4.2 Special CAPP Modes

4.2.1 Non-Blocking Write

If a CAPP packet is designated as NBW, it is guaranteed a non-blocking path through the PHB and PEC. This is done in the PEC by guaranteeing at least one of the DMA write buffers and controllers will be reserved for the NBW. The DMA write resources are dynamically assigned for I/O, normal CAPP, and NBW CAPP. If there is more traffic of one type, it uses the resource that it requires with the only reservation that at least one buffer/controller is available for use for NBW packets.

Additional setup is required to use NBW CAPP packets. In the PHB, the NBW Compare and Mask must be initialized. In the PEC, the PBAIB TX Command Credit Register and PBAIB TX Data Credit Register must be set up to use AIB channel 3, which is the channel that the NBW packets use.

4.2.2 NMMU Packets

The NMMU checkout and response requests are CAPP packets. A checkout transaction consists of a checkout request followed by a response request. The PEC supports 32 checkout transaction, which are identified by a tag in the request commands. A checkout request is held in the PEC unit and is not sent to the NMMU until its corresponding response request occurs. If multiple checkout requests occur to a tag before a response request, the last checkout request will be used in the transaction. Checkout requests and responses of different transactions can be interleaved. Checkout requests are sent to the NMMU in the order that the response requests occur. There is no presumed order for the return of response data to the CAPI device.

An NMMU checkout request is a 32-byte CAPP write packet.

An NMMU response request is a 16-byte CAPP read packet in which the response data is the response from the NMMU.

Figure 4-1 shows the address format of the checkout request and *Figure 4-2* shows the address format of the response request.

Figure 4-1. CAPI NMMU Request Address Format (Checkout Request)

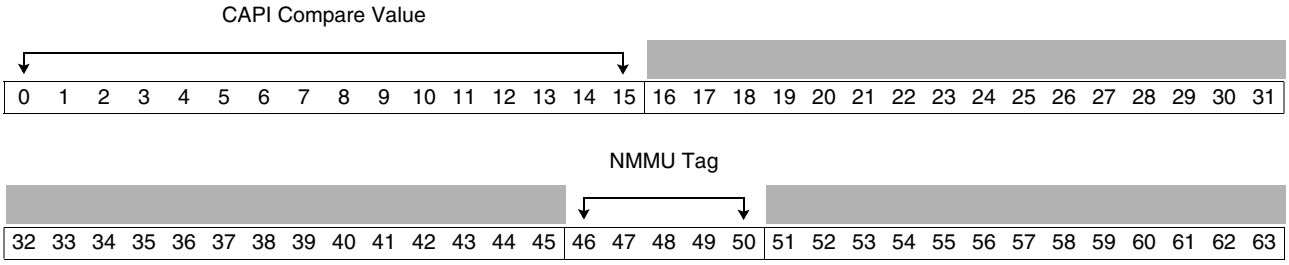
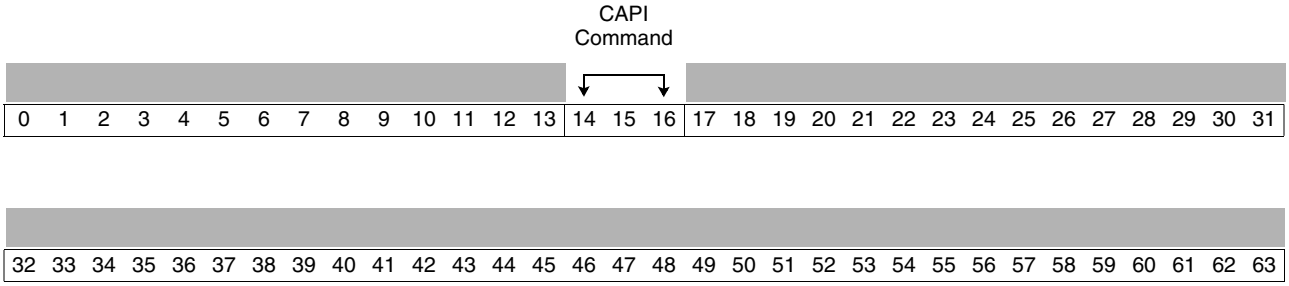


Figure 4-2. CAPI NMMU Request Address Format (Response Request)



Bits	Field Name	Description
0:15	CAPI Compare Value	Value must match the CAPI Compare/Mask Register in the *Power Systems Host Bridge 4 (PHB4) Specification* to indicate that this is a CAPI command. **Notes:** 1. Bits 14:15 are a part of the command field and must be '10'. 2. Bits 61:60 cannot equal '01'. This value indicates that the posted write is an MSI interrupt.
14:16	CAPI Command	'101' NMMU request.
46:50	NMMU Tag	Identifies the transaction of the request.

5. Processor Bus Common Queue Registers

All PBCQ and PBAIB registers are accessed via SCOM, there are no MMIO registers.

All registers are protected by even parity. For proper parity generation, all unimplemented bits must be set to '0' when written.

Table 5-1 summarizes the PBCQ registers.

Table 5-1. PBCQ Register Summary (Page 1 of 2)

SCOM Offset	Description	Page
NestBase+0x00	PBCQ Hardware Configuration Register	Page 47
NestBase+0x01	Drop Priority Control Register	Page 51
NestBase+0x02	PBCQ Error Inject Control Register	Page 52
NestBase+0x03	PCI Nest Clock Trace Control Register	Page 53
NestBase+0x04	PBCQ Performance Monitor Control Register	Page 57
NestBase+0x05	PBCQ Processor Bus Address Extension Mask	Page 59
NestBase+0x06	PBCQ Predictive Vector Timeout Register	Page 60
NestBase+0x07	CAPP Control Register	Page 61
NestBase+0x08	PBCQ Read Stack Override Register	Page 61
NestBase+0x09	PBCQ Write Stack Override Register	Page 62
NestBase+0x0A	PBCQ Store Stack Override Register	Page 62
NestBase+0x0B	PBCQ Retry Backoff Control Register	Page 63
NestBase+0x0C:0x1F	Reserved	Reserved
NestBase+StackBase+0x0 NestBase+StackBase+0x1 NestBase+StackBase+0x2	PCI Nest Fault Isolation Register (FIR), Clear, Set	Page 64
NestBase+StackBase+0x3 NestBase+StackBase+0x4 NestBase+StackBase+0x5	PCI Nest FIR Mask, Clear, Set	Page 68
NestBase+StackBase+0x6	PCI Nest FIR Action 0	Page 66
NestBase+StackBase+0x7	PCI Nest FIR Action 1	Page 67
NestBase+StackBase+0x8	PCI Nest FIR WOF	Page 68
NestBase+StackBase+0xA	Error Report Register 0	Page 69
NestBase+StackBase+0xB	Error Report Register 1	Page 71
NestBase+StackBase+0xC	PBCQ General Status Register	Page 72
NestBase+StackBase+0xD	PBCQ Mode Register	Page 72
NestBase+StackBase+0xE	MMIO Base Address Register 0	Page 72
NestBase+StackBase+0xF	MMIO Base Address Register Mask 0	Page 73
NestBase+StackBase+0x10	MMIO Base Address Register 1	Page 73
NestBase+StackBase+0x11	MMIO Base Address Register Mask 1	Page 74
NestBase+StackBase+0x12	PHB Register Base Address Register	Page 74

Table 5-1. PBCQ Register Summary (Page 2 of 2)

SCOM Offset	Description	Page
NestBase+StackBase+0x13	Interrupt Base Address Register	Page 75
NestBase+StackBase+0x14	Base Address Enable Register	Page 75
NestBase+StackBase+0x15	Data Freeze Type Register	Page 76
NestBase+StackBase+0x16	PBCQ Tunnel Bar Register	Page 77
PCIBase +0x00	PBAIB Hardware Control Register	Page 77
PCIBase +0x02	PCIe Read Stack Override Register Copy	Page 79
PCIBase+StackBase+0x00 PCIBase+StackBase+0x01 PCIBase+StackBase+0x02	PCI Fault Isolation Register (PFIR), Clear, Set PCI FIR Set	Page 80
PCIBase+StackBase+0x03 PCIBase+StackBase+0x04 PCIBase+StackBase+0x05	PCI FIR Mask, Clear, Set	Page 81
PCIBase+StackBase+0x06	PCI FIR Action 0	Page 80
PCIBase+StackBase+0x07	PCI FIR Action 1	Page 81
PCIBase+StackBase+0x08	PCI FIR WOF	Page 82
PCIBase+StackBase +0x0A	ETU Reset Register	Page 82
PCIBase+StackBase +0x0B	PBAIB Error Report Register	Page 83
PCIBase+StackBase +0x0C	Reserved	
PCIBase+StackBase +0x0D	PBAIB TX Command Credit Register	Page 84
PCIBase+StackBase +0x0E	PBAIB TX Data Credit Register	Page 85
NestBase = 4010C00 for PEC0 = 4011000 for PEC1 = 4011400 for PEC2 PCIBase = D010800 for PE0 = E010800 for PE1 = F010800 for PE2 Stackoffset = 00000040 for Stack0 = 00000080 for Stack1 = 000000C0 for Stack2		
PHB0 = PCIBase PE0 + 0x100 PHB1 = PCIBase PE1 + 0x100 PHB2 = PCIBase PE1 + 0x100 + 0x040 PHB3 = PCIBase PE2 + 0x100 PHB4 = PCIBase PE2 + 0x100 + 0x040 PHB5 = PCIBase PE2 + 0x100 + 0x080		

Table 5-2 defines the terminology used in the register descriptions.

Table 5-2. Register Access Legend

Type	Description
R/W	**Read-Write.** The value in the register can be either read or written. There are some exceptions, but generally the register holds the last value written.
R/O	**Read-Only.** Reads to the register are allowed and returns the current value in the register. A write operation is not allowed and does not affect the register value. If the write operation requires a status, the status always indicates a successful write.
R/W1C	**Read-Write 1 to Clear.** Reads to the register are allowed and returns the current value in the register. A write operation with a value of '1' clears the register bit to '0'.
W/O	**Write-Only.** The value in the register can only be written, never read. A read of this register causes an SCOM error.
W/O0C	**Write-Only 0 to Clear.** The value in the register can only be written, never read. Writing a '0' causes the bit to clear, a '1' leaves the bit unchanged.
W/O1S	**Write-Only 1 to Set.** The value in the register can only be written, never read. Writing a '1' causes the bit to be set, a '0' leaves the bit unchanged.
W/C	**Write-Clear.** The value in the register can only be written, never read. Any value written clears the entire register contents.

5.1 Processor Bus Common Queue Registers Descriptions

5.1.1 PBCQ Hardware Configuration Register

This register contains control bits to set the PEC's PBCQ hardware configuration.

Mnemonic PbcqHwCfg

Address Offset NestBase+0x00

Bit	Field Mnemonic	Type	Reset Value	Description
0:3	hang_poll_scale	R/W	0	Number of SMP interconnect (PB) hang polls that must be detected to indicate a PB hang poll to the logic.
4:7	hang_data_scale	R/W	0	Number of PB data polls that must be detected to indicate a PB data poll to the logic.
8:11	hang_pe_scale	R/W	0	Number of PB data polls that must be detected to indicate a PEC poll to the logic.
12	block_cqpb_pb_init	R/W	0	When set, the PCI-to-PB data movement ignores the PB initialization signal. When not set, the PCI-to-PB data movement stops on the cache-line boundaries.
13	Reserved	R/W	0	Implemented but reserved.
14	hang_sm_on_are	R/W	0	Controls SMP interconnect master state machines when an address error is received. 0 State machine returns to idle. DMA write/interrupts drop a command silently. DMA reads send an address error (ARE) indication back to the AIB. 1 State machines hang on receiving an ARE and wait for a flush reset.

Bit	Field Mnemonic	Type	Reset Value	Description
15	disable_pci_clk_check	R/W	0	Controls the logic that checks FIR bit 15. 0 PCI clock testing enabled. 1 Valid PCI clock is not verified.
16	enable_lfsr_arb	R/W	0	When set, use the LSFR to choose the next outbound command type. When not set, use the round-robin method to choose the next outbound command type.
17	enable_dmaw_iopacing	R/W	0	Allow I/O pacing scheme for DMA write operations.
18	enable_dmar_iopacing	R/W	0	Allow I/O pacing scheme for DMA read operations.
19	adr_bar_mode	R/W	0	SMP interconnect architected bit to distinguish between address mappings.
20	stq_allocation	R/W	0	Store queue allocation: 0 One queue is reserved for MMIO store and peer-to-peer and the rest are shared. 1 Only one queue is used for all traffic.
21	disable_lpc_cmds	R/W	0	Disable LPC ACK for commands that the PEC does not service.
22	disable_ooo_mode	R/W	0	PEC orders o-o-o type CI stores like a normal CI store command. **Note:** This is set to '1' during initialization and should not be changed.
23:26	osmb_early_start	R/W	0's	Determines how much overlap of reading/writing the OSMB enables: 0000 Most conservative, full packet written before signaling the PCI side. 0001 Most conservative, full packet written in before signaling the PCI side. 0010 Start 2 writes from the end of the packet. 0011 Start 3 writes from the end of the packet. 0100 Start 4 writes from the end of the packet. 0101 Start 5 writes from the end of the packet. 0110 Start 6 writes from the end of the packet. 0111 Start 7 writes from the end of the packet. 1000 Most aggressive. Start 8 writes from the end of the packet, before data is written. 1111 Extra aggressive. Start before any data written (when command is written).
27:28	qfifo_hold_mode	R/W	0	Determines how much overlap of reading/writing the OSMB enables: 00 Stop sending outbound commands when QFIFO is full 01 Stop sending outbound commands when QFIFO has 1 entry empty 10 Stop sending outbound commands when QFIFO has 3 entry empty 11 Ignore QFIFO count when sending outbound packets
29:31	Reserved	R/W	0	Implemented but reserved.
32	wr_strict_order	R/W	0	Strictly-order inbound write commands. Independent of node ID.
33	channel_streaming_en	R/W	0	Enable channel Tag streaming on the SMP interconnect.
34:35	wr_cache_inject_mode	R/W	0	Selects cache inject on the SMP interconnect: 00 No cache injection. 01 POWER7-style cache injection. 10 Use PCI TLP hint bits to start/continue cache injection. 11 Initial attempt as inject (POWER9 style).
36	en_new_flow_cache_inject	R/W	0	Enable all new flows on PEC to send cache-inject commands.

Bit	Field Mnemonic	Type	Reset Value	Description
37	disable_inj_on_resend	R/W	0	Controls cache inject on resent combined responses. 0 A cache-line inject(**cl_dmaw_inj**) is sent in response to a resent cresp. 1 A normal (**cl_dma_w_i**) DMA write is sent in response to a resent cresp.
38	force_dis_ctag_to_use_flow	R/W	0	When channel streaming is disabled, this bit forces DMA writes to still use flows for nonordering reasons (drop the priority).
39:40	Reserved	R/W	00	Implemented but reserved.
41	disable_wr_Vg(pred)	R/W	0	If Vg is disabled, do Vg(sys). 0 Default is to use the predicted target 1 Use the all 1's target only.
42	disable_wr_scope_group	R/W	0	Disable DMA write group scope. 0 Default is to use SMP interconnect rules based on the address. 1 Do not allow group scope.
43	disable_intwr_Vg(pred)	R/W	0	If Vg is disabled, do Vg(sys). 0 Default is to use the predicted target. 1 Use the all 1's target only.
44	disable_intwr_scope_group	R/W	0	Disable interrupt write group scope: 0 Default is to use SMP interconnect rules based on address. 1 Do not allow group scope.
45	disable_intwr_scope_node	R/W	0	Disable interrupt write node scope: 0 Default is to use SMP interconnect rules based on address. 1 Do not allow node scope.
45	Reserved	R/W	0	Implemented but reserved.
46:48	inject_threshold_dec_rate	R/W	0's	000 Off. 001 Inject rate of approximately 16 cycles. 011 Inject rate of approximately 32 cycles. 111 Inject rate of approximately 64 cycles.
49	Reserved	R/W	0	Implemented but reserved.
50	disable_rd_scope_nodal	R/W	0	Force starting DMA read operations scope to: 0 Default is to use SMP interconnect rules based on address. 1 Force starting scope to system.
51	disable_rd_scope_group	R/W	0	Disable DMA read group scope: 0 Default is to use SMP interconnect rules based on address. 1 Do not allow group scope.
52	disable_rd_scope_RnNn	R/W	0	Disable DMA read scope Rn and An: 0 Default is to use SMP interconnect rules based on address. 1 Do not allow An or Rn scope.
53	enabled_rd_skip_group	R/W	0	Skip over group scope in DMA read progression. 0 Default is to not skip group scope. 1 Skip group scope during scope progression.
54	disable_rd_Vg(pred)	R/W	0	If Vg is disabled, do Vg(sys). 0 Default is to use the predicted target. 1 Use the all 1's target only.
55	disable_tce_scope_nodal	R/W	0	Force starting DMA read operations scope to: 0 Default is to use SMP interconnect rules based on address. 1 Force starting scope to system.

Bit	Field Mnemonic	Type	Reset Value	Description
56	disable_tce_scope_group	R/W	0	Disable DMA read group scope: 0 Default is to use the SMP interconnect rules based on address. 1 Do not allow Group scope.
57	disable_tce_scope_Rn_Nn	R/W	0	Disable DMA TCE group Rn and Nn: 0 Default is to use SMP interconnect rules based on address. 1 Do not allow Nn or Rn scope.
58	enabled_tce_skip_group	R/W	0	Skip over group scope in DMA read progression. 0 Default is to not skip group scope. 1 Skip group scope during scope progression.
59	disable_tce_Vg(pred)	R/W	0	If Vg is disabled, do Vg(sys). 0 Default is to use the predicted target. 1 Use the all 1's target only.
60	disable_pb_tce_arbitration	R/W	0	Disable preferential TCE read arbitration. This bit can only be '0' for a ×16 stack. A ×8 stack must set this bit to '0'.
61	disable_cq_tce_arbitration	R/W	0	Disable preferential TCE read response arbitration.
62	enable_mc_prefetch	R/W	0	Controls setting the PADE bits in the **cl_rd_nc** command. 1 Prefetch based on the address of the command (assumes 4 KB pages). 0 No prefetch indications (PADE= '0000').
63	ignore_sfstat	R/W	0	Controls the DMA read state machine behavior when receiving SFSTAT. 0 When sfSTAT is set, the read address is retried using the next higher scope. 1 When sfSTAT is set, the read address is not retried.

5.1.2 Drop Priority Control Register

This register controls the behavior of the SMP interconnect drop priority logic.

Mnemonic DrpPriCtl

Address Offset NestBase+0x01

Bit	Field Mnemonic	Type	Reset Value	Description
0:5	DropPriorityMask	R/W	0	Mask value to determine when a **rty_drp** will cause the priority to increment.
6	disable_ctag_drop_priority	R/W	0	Enable drop priority based on channel stream. 0 Default is to have all commands in a stream have the same priority. 1 Each command in a stream starts at the lowest priority, and only the **rty_drp** command's priority is changed.
07	enable_IO_cmd_pacing	R/W	0	Enable I/O command pacing for drop priority: 0 Default is to keep the command rate as fast as possible and increase priority. 1 Keep drop priority, and lower command rate, until the pacing count is reached.
8:16	DropPaceCount	R/W	0	Initial value to use when determining if the drop priority should be increased when I/O command pacing is enabled.
17:22	DropPaceInc	R/W	0	The value to increment pacing count when a **rty_drp** cresp is received.
23:25	RtyDropDivider	R/W	0	Value used to "divide down" **rty_drp** combined responses before invoking a drop priority. 000/001 First retry drop combined response invokes a drop priority mechanism. 010 Second retry drop combined response invokes a drop priority mechanism. 011 Third retry drop combined response invokes a drop priority mechanism. … 111 Seventh retry drop combined response invokes a drop priority mechanism.

5.1.3 PBCQ Error Inject Control Register

This register contains control bits to enable injecting errors.

Mnemonic PbcqEinj

Address Offset NestBase+0x02

Bit	Field Mnemonic	Type	Reset Value	Description
0:1	inject_type	R/W	00	Determines the type of error injected. 01 Correctable error/parity error. 10 Uncorrectable error. 11 Special uncorrectable error (see notes 1, 2, and 3).
2	cq_ecc_inject_enable	R/W	0	Enable ECC inject on CQ data arrays. Uses bits 0:1 to determine the type of injection.
3:6	cq_sram_array	R/W	0000	Determines which CQ SRAM array to force the ECC error into. 0000 CI store 0:63 0001 CI store 64:127 0010 DMA read 0:63 0011 DMA read 64:127 010x Reserved 1000 DMA write 0:63 1001 DMA write 64:127 1010 CI load 0:63 1011 CI load 64:127 11xx Reserved 1111 OSMB data
7	cq_par_inject_enable	R/W	0	Enable parity inject on CQ arrays.
8:10	cq_register_array	R/W	000	Determines which CQ register array to force the parity error if bit 7 is enabled. 000 PBCQ array 001 TTAG 010 RTAG 011 OSMB Command 100 CQPB DMAR 101 CQPB DMAW 110-111 Reserved
11	Constant_error_inj	R/W	0	0 Error to occur one time. Error injection is re-triggered with a new write to this register. 1 Error to occur until bit 11 is changed to 0 or the injection is disabled.

1. SUE is not injected on CI load data. Value 0:6 = ‘111101x’ does not have any affect.
2. SUE on OSMB data is not useful and has not been verified.
3. SUE injected on arrays moving data toward the processor bus will not be detected in the NFIR.

5.1.4 PCI Nest Clock Trace Control Register

This register controls the nest clock trace bus selection.

Mnemonic NestTrc

Address Offset NestBase+0x03

Bit	Field Mnemonic	Type	Reset Value	Description
0:3	TraceSelA	R/W	0000	Mux select for bits (0:21)
4:7	TraceSelB	R/W	0000	Mux select for bits (22:43)
8:11	TraceSelC	R/W	0000	Mux select for bits (44:65)
12:15	TraceSelD	R/W	0000	Mux select for bits (66:87)
16	EnableNestTrace	R/W	0	Enable nest clock domain tracing
Mux Select: 0000 : perd_trace(0:21) 00:05 rd_cmd_queue_dout 06:12 rd_cmd_size_dout 13:21 rd_cmd_addr_dout				
Mux Select: 0001 : perd_trace(22:43) 22:24 rd_cmd_type_dout 25:27 rd_cmd_frag_type_dout 28:34 rd_cmd_frag_prev_q_dout 35 rd_cmd_abort_dout 36 rd_cmd_valid_d1_dout 37 rdq_addr_req_dout 38:43 rdq_addr_queue_dout				
Mux Select: 0010 : perdx_trace(0:21) 00:05 rd_cmd_queue_dout 06:12 rd_cmd_size_dout 13:21 rd_cmd_addr_dout				
Mux Select: 0011 : perdx_trace(22:43) 22:24 rd_cmd_type_dout 25:27 rd_cmd_frag_type_dout 28:34 rd_cmd_frag_prev_q_dout 35 rd_cmd_abort_dout 36 rd_cmd_valid_d1_dout 37 rdq_addr_req_dout 38:43 rdq_addr_queue_dout				
Mux Select: 0100 : peorder_trace 00 pedisp_ld_reply 01 pedisp_dmarq 02 pedisp_dmawq 03:07 pedisp_queue_addr 08 pedisp_no_order 09 pedisp_no_post_order 10:11 pedisp_stk 12:20 pedisp_flow 21 disp_new_ctag_dout				

Mux Select: 0101 : pewr_trace(0:21)	
00:04	wr_cmd_queue_dout
05:11	wr_cmd_size_dout
12:20	wr_cmd_queue_dout
21	wr_cmd_hint_dout
Mux Select: 0110 : pewr_trace(22:43)	
22	wr_cmd_etuwr_dout
23	wr_cmd_intwr_dout
24	wr_cmd_p2pwr_dout
25	wr_cmd_last_frag_dout
26	wr_cmd_cap_fbit_dout
27	wr_cmd_tunnel_dout
28	wr_cmd_no_post_order_dout
29	wr_cmd_nbw_dout
30	wr_cmd_valid_d1_dout
31	wr_addr_req_dout
32:36	wr_addr_queue_dout
37	wr_data_req_dout
38:42	wr_data_queue_dout
43	wr_data_freeze_dout
Mux Select: 0111 : peldst_trace	
00:04	ldq_sm_dout(0)
05:08	stq_sm_dout(0)
09	OR_REDUCE(final_load_req_cav4(*))
10	OR_REDUCE(final_store_req_cav4(*))
11:21	reserved for future definition
Mux Select: 1000 : pepbcq_dctl_trace	
00:07	pb_txid_cav23_dout
08:09	pb_ow_id_cav23_dout
10	pb_dval_cav23_dout
11:21	reserved for future definition
Mux Select: 1001 : peopc_trace (stack 0)	
00:02	reserved for future definition
03:05	rtdsm_encode
06:07	wrsm_dout
08	reserved for future definition
09:11	rtdnsm_dout
12	smb_space_avail
13	resp_ack_available
14	resp_retry_available
15	wtd_act
16:20	command_type_dout
21	reserved for future definition

Mux Select: 1010 : peopc_trace (stack 1)	
00:02	reserved for future definition
03:05	rtdsm_encode
06:07	wrsm_dout
08	reserved for future definition
09:11	rtdnsm_dout
12	smb_space_avail
13	resp_ack_available
14	resp_retry_available
15	wtd_act
16:20	command_type_dout
21	reserved for future definition
Mux Select: 1011 : peopc_trace (stack 2)	
00:02	reserved for future definition
03:05	rtdsm_encode
06:07	wrsm_dout
08	reserved for future definition
09:11	rtdnsm_dout
12	smb_space_avail
13	resp_ack_available
14	resp_retry_available
15	wtd_act
16:20	command_type_dout
21	reserved for future definition
Mux Select: 1100 : pecqpb_dctl_trace	
00:08	data_pipe_dout
09:14	dreq_pipe_dout
15	trace_ld_done
16	trace_wr_done
17	trace_rtag_we
18	trace_ttag_we
19:20	freeze_type_dout
21	reserved for future definition
Mux Select: 1101 : pedisp_trace	
00	ismb_cmd_val_dout
01:02	ismb_cmd_stk_enc
03:10	ismb_cmd_dout(0:7)
11	ismb_cmd_val_dout
12:13	ismb_data_stk_dout
14	ismb_data_first_dout
15:17	pbdata_wren_dout
18	ismb_data_last
19:21	reserved for future definition
Mux Select: 1110 : penmmu_trace	
00	nmmu_pb_req_dout
01:03	nmmu_pb_rtag_dout
04:08	nmmu_pb_queue_dout
09	cq_nmmu_req_dout
10:14	cq_queue_info_dout
15:17	cq_nmmu_queue_dout
18	pedisp_pbdata_wren
19:21	pedisp_pbdata_wraddr(5:7)

Mux Select: 1111 : pepbmst_trace	
00	cmd_ctagv_dout
01:07	cmd_ctag_dout
08	cmd_pipe_needed
09	pace_allows_cmd
10:11	cmd_pipe_dout
12:14	cmd_scope_dout
15:16	cmd_req_winner_enc
17:21	reserved for future definition

5.1.5 PBCQ Performance Monitor Control Register

This register contains control bits to select the PCI performance monitor.

Mnemonic PMonCtl

Address Offset NestBase+0x04

Bit	Field Mnemonic	Type	Reset Value	Description
0:15	pe_pmon_en	R/W	0	Enable performance monitor outputs per bit.
16	Reserved	R/W	0	Reserved.
17	pe_pmon_all_engines	R/W	0	Monitor all engines or dedicated engine per stack. Dedicated engine is the first engine assigned to a stack in the default stack assignment.
18:19	pe_pmon_stk	R/W	0	Which stack to monitor.
20:21	pe_pmon_rd_type	R/W	0	Determines how TCEs are handled when counting DMA read events: 00 No monitor or reads. 01 Monitor TCE. 10 Monitor DMA. 11 Monitor DMA and TCE.
22:24	pe_pmon_mux_Byte 0	R/W	0	Controls data sent to performance monitor bits 0:7 000 Group 0. 001 Group 1. 010 Group 2. 011 Group 3. 100 Group 4. 101 Group 5. 110 - 111: Reserved. Behavior is unpredictable.
25:27	pe_pmon_mux_Byte 1	R/W	0	Controls data sent to performance monitor bits 8:15. 000 Group 0. 001 Group 1. 010 Group 2. 011 Group 3. 100 Group 4. 101 Group 5. 110-111: Reserved. Behavior is unpredictable.

Table 5-3. Performance Monitor Group Definition (Page 1 of 2)

Group	Bit	Description
Group 0	0	Successful system DMA read
	1	Successful system DMA write
	2	Successful group DMA read
	3	Successful group DMA write
	4	Successful nodal DMA read
	5	Successful cache inject
	6	32 bytes send to SMP interconnect
	7	32 bytes received from SMP interconnect
Group 1	0	Retried system DMA read
	1	Retried system DMA write
	2	Retried group DMA read
	3	Retried group DMA write
	4	Retried nodal DMA read
	5	Failed cache inject
	6	Reserved
	7	Reserved
Group 2	0	DMA read average machine usage (all engines or default engine)
	1	DMA write average machine usage (all engines or default engine)
		DMA read default engine PB command request latency (command request to grant)
		DMA write default engine PB command request latency (command request to grant)
	4	DMA read default engine command request to data arrival latency
	5	DMA write default engine PB data grant latency (data request to data grant)
	6	Reserved
	7	Reserved
Group 3	0	Retry increment nodal DMA read
	1	Retry increment group DMA read
	2	Retry increment system DMA read
	3	Retry increment system DMA write
	4	Retry increment group DMA write
	5	Reserved
	6	Reserved
	7	Reserved

Table 5-3. Performance Monitor Group Definition (Page 2 of 2)

Group	Bit	Description
Group 4	0	DMA read data was from local (L2/L3) cache
	1	DMA read data was from the local memory controller
	2	DMA read data was from the near (L2/L3) cache
	3	DMA read data was from the near memory controller
	4	DMA read data was from remote (L2/L3) cache
	5	DMA read data was from the remote memory controller
	6	Reserved
	7	Reserved
Group 5	0	Successful CI store
	1	Successful CI load
	2	Retried CI store
	3	Retried CI load
	4	Reserved
	5	Reserved
	6	Reserved
	7	Successful partial DMA write

5.1.6 PBCQ Processor Bus Address Extension Mask

Chip Address Extension Mask Enable as defined by the SMP interconnect architecture.

Mnemonic AddrExtMask

Address Offset NestBase+0x05

Bit	Field Mnemonic	Type	Reset Value	Description
0:6	AddrExtMask	R/W	0	Mask applied to addresses 15:21 in the Group and Chip ID address match detection as defined by the SMP interconnect.

5.1.7 PBCQ Predictive Vector Timeout Register

This register contains the upper address bits on returned tunneled packets.

Mnemonic predv

Address Offset NestBase+0x06

Bit	Field Mnemonic	Type	Reset Value	Description
0:7	Rd_prev_timeout_mask	R/W	0xFC	A mask that controls the number of cycles upon which to reset the Vg scope predictor for read commands. Values of interest are: 0xFF 64K cycles 0xFE 32K cycles 0xFC 16K cycles 0xF8 8K cycles 0x80 512 (simulation/stress mode)
8:15	wr_prev_timeout_mask	R/W	0xFC	A mask that controls the number of cycles upon which to reset the Vg scope predictor for write commands. Values of interest are: 0xFF 64K cycles 0xFE 32K cycles 0xFC 16K cycles 0xF8 8K cycles 0x80 512 (simulation/stress mode)

Note: For specific bit alignments, contact the design team.

5.1.8 CAPP Control Register

This register is used to enable coherently attached processor proxy (CAPP) mode.

Mnemonic CAPP_CNTL

Address Offset NestBase+0x07

Bit	Field Mnemonic	Type	Reset Value	Description
0	capp_en	R/W	0	Enable CAPP mode of operation. 0 PEC works in traditional I/O mode. 1 PEC works in CAPP mode.
01	Reserved	R/W	0	Reserved.
02	enable_load_throttle	R/W	0	Only assigns seven MMIO load engines to stk0 (CAPP stack), so that an outbound non-blocking channel will exist.
03:11	Reserved	R/W	0's	Reserved.
12:15	capp_num_msg_eng	R/W	0's	Number of STQ engines that have been given to the CAPP for use. If PBCQ is operating as a ×8 stack, the maximum number of engines given to CAPP is six and assigned in the order of 7 to 2. If PBCQ is operating as a ×16 stack, the maximum number of engines given to CAPP is 14 and assigned in the order of STQ 15 to 2. The value of this field must match the setup of the CAPP unit.
16:63	capp_APC_engine(0:47)	R/W	0's	Indicates DMA read engines that have been given to the CAPP for use. If PBCQ is operating as a ×8 stack, only engines 0 - 30 are available for CAPP use. If PBCQ is operating as a ×16 stack, engines 0 - 47 are available for CAPP use. The value of this field must match the setup of the CAPP unit.

5.1.9 PBCQ Read Stack Override Register

This register overrides the read engine assignment when bifurcated or trifurcated. If this register is used, the PBAIBTXCCR and PBAIBTXDCR Registers must also be set up.

Mnemonic Nrdstkovr

Address Offset NestBase+0x08

Bit	Field Mnemonic	Type	Reset Value	Description
0:31	Stack0 Override	R/W	0	Additional read engines from 32:63 that are assigned to stack0 when enabled. Engines 0:31 are always assigned to stack0.
32:47	Stack1 Override	R/W	0	Additional read engines from 48:63 that are assigned to stack1 when enabled and not assigned to stack0. Engines 32:47 are assigned to stack1 if not assigned to stack0.
48	Override Enable	R/W	0	Enable override.
Note: This register must match the PCI version, prdstkovr.				

5.1.10 PBCQ Write Stack Override Register

This register overrides the write engine assignment when bifurcated or trifurcated. If this register is used, the PBAIBTXCCR and PBAIBTXDCR Registers must also be set up.

Mnemonic nwrstkovr

Address Offset NestBase+0x09

Bits	Field Mnemonic	Type	Reset Value	Description
0:15	Stack0 Override	R/W	0	Additional read engines from 16:31 that are assigned to stack0 when enabled. Engines 0:15 are always assigned to stack0.
16:23	Stack1 Override	R/W	0	Additional read engines from 24:31 that are assigned to stack1 when enabled and not assigned to stack0. Engines 16:23 are assigned to stack1 if not assigned to stack0
24	Override Enable	R/O	0's	Enable override.

5.1.11 PBCQ Store Stack Override Register

This register overrides the store engine assignment when bifurcated or trifurcated.

Mnemonic nstqstkovr

Address Offset NestBase+0x0A

Bits	Field Mnemonic	Type	Reset Value	Description
0:7	Stack0 Override	R/W	0	Additional store engines from 8:15 that are assigned to stack0 when enabled. Engines 0:7 are always assigned to stack0.
8:11	Stack1 Override	R/W	0	Additional store engines from 12:15 that are assigned to stack1 when enabled and not assigned to stack0. Engines 8:11 are assigned to stack1 if not assigned to stack0
12	Override Enable	R/O	0's	Enable override.

Note: Some engines are reserved for a given stack if that stack is enabled.

Stack0: Engine 0 is reserved for CI stores and Engine 1 is reserved for peer-to-peer, if enabled.

Stack1: Engine 8 is reserved for CI stores and Engine 9 is reserved for peer-to-peer, if enabled.

Stack2: Engine 12 is reserved for CI stores and Engine 13 is reserved for peer-to-peer, if enabled.

If an engine is reserved for a stack, it cannot be assigned to another stack if the function that the engine is reserved for can occur during operation.

5.1.12 PBCQ Retry Backoff Control Register

This register is a configuration register for the master retry backoff mechanism.

Mnemonic rtyboctl

Address Offset NestBase+0x0B

Bits	Field Mnemonic	Type	Reset Value	Description
0	grp_all_notjustown_dis	R/W	0	0 Count all retry cresps seen on the PB. 1 Configure the backoff mechanism to only count cresps for commands that the PEC masters.
1	grp_syn_adj_dis	R/W	0	0 If the backoff level increases, the sample window duration also increases. 1 Disable dynamic readjustment of the sample window in which the mechanism counts retries before resetting.
2	grp_dyn_adj_lvl_dis	R/W	0	0 If the backoff level increases, the threshold for incrementing the backoff level also increases. 1 Disable dynamic readjustment of the backoff level.
3	grp_all_rty_cntr_div2_en	R/W	0	0 The count is reset at the end of the sample window. 1 Enable division by two of the retry count at the end of the sample window.
4:7	grp_all_max_lvl_cnt_qual	R/W	1111	Sets the highest backoff level allowed. If this field is set to 0x0, the master retry backoff function is disabled.
8:13	grp_all_cnt_threshold1_qual	R/W	000011	Sets the lower retry count threshold, below which the backoff level will be decremented.
14:19	grp_all_cnt_threshold2_qual	R/W	001111	Sets the upper retry count threshold, above which the backoff level is incremented.
20	grp_mstr_rty_backoff_en	R/W	0	Enable group scope master retry backoff.
21	sys_all_notjustown_dis	R/W	0	0 Count all retry cresps seen on the PB. 1 Configure the backoff mechanism to only count cresps for commands that the PEC masters.
22	sys_syn_adj_dis	R/W	0	0 If the backoff level increases, the sample window duration also increases. 1 Disable dynamic readjustment of the sample window in which the mechanism counts retries before resetting.
23	sys_dyn_adj_lvl_dis	R/W	0	0 If the backoff level increases, the threshold for incrementing the backoff level also increases. 1 Disable dynamic readjustment of the backoff level.
24	sys_all_rty_cntr_div2_en	R/W	0	0 The count is reset at the end of the sample window. 1 Enable division by two of the retry count at the end of the sample window.
25:28	sys_all_max_lvl_cnt_qual	R/W	1111	Sets the highest backoff level allowed. If this field is set to 0x0, the master retry backoff function is disabled.
29:34	sys_all_cnt_threshold1_qual	R/W	000011	Sets the lower retry count threshold, below which the backoff level is decremented.
35:40	sys_all_cnt_threshold2_qual	R/W	001111	Sets the upper retry count threshold, above which the backoff level is incremented.
41	sys_mstr_rty_backoff_en	R/W	0	Enable system scope master retry backoff.

5.1.13 PCI Nest Fault Isolation Register (FIR), Clear, Set

This section describes the PCI Nest FIR Register, Clear, and Set.

Mnemonic NFIR

Address Offset NestBase+StackBase+0x0
NestBase+StackBase+0x1
NestBase+StackBase+0x2

Bits	Field Mnemonic	Type	Reset Value	Description
0	bar_pe	See Note 1	0	One of the BARs or BAR Mask Register parity error.
1	nonbar_pe		0	Any non-BAR parity error.
2	PB_to_PEC_ce		0	ECC correctable error off of outbound SMP interconnect.
3	PB_to_PEC_ue		0	ECC uncorrectable error off of outbound SMP interconnect.
4	PB_to_PEC_sue		0	ECC special uncorrectable error off of outbound SMP interconnect
5	ary_ecc_ce		0	ECC correctable error on an internal array.
6	ary_ecc_ue		0	ECC uncorrectable error on an internal array.
7	ary_ecc_sue		0	ECC special uncorrectable error on an internal array.
8	register_array_pe		0	Parity error on an internal register file.
9	pb_interface_pe		0	Parity error on the PB interface (address/aTag/tTag/rTAG).
10	pb_data_hang_errors		0	Any SMP interconnect data hang poll error (only checked for CI stores).
11	pb_hang_errors		0	Any SMP interconnect command hang error (domestic address range).
12	rd_are_errors		0	SMP interconnect address error (ARE) detected by a DMA read.
13	nonrd_are_errors		0	SMP interconnect address error detected by a DMA write or an interrupt engine.
14	pci_hang_error		0	PBCQ detected that the PCI load, store, EOI, or DMA read response did not make forward progress.
15	pci_clock_error		0	PBCQ has detected that the PCI clock has stopped.
16	PFIR_freeze		0	This is the freeze signal from the PFIR freeze output.
17	hw_errors		0	Any miscellaneous hardware error.
18	UnsolicitiedPBData		0	The PEC received data with an rTAG matching a queue that was not expecting data or too much data was received.
19	UnExpectedCResp		0	PEC received an unexpected combined response.
20	InvalidCResp		0	PEC received an invalid combined response.
21	PBUnsupportedSize		0	PEC received a CI load/store that hits a BAR but is an unsupported size or address alignment. CI load size ≠ 1, 2, 4, 8, 16, 32 with the address naturally aligned. CI store size: PB tsize encoding cannot generate an invalid size, but the CI store address must still be naturally aligned.

1. Address NestBase + 0x00 is read/write (R/W).
   Address NestBase + 0x01 is bitwise clear (a “new error” overrides the clear function) (W/O0C).
   Address NestBase + 0x02 is bitwise set (W/O1S).

Bits	Field Mnemonic	Type	Reset Value	Description
22	PBUnsupported Cmd	See Note 1	0	PEC receives a CI command that matches one of the BARs but is an unsupported ttype. Supported ttypes are **ci_pr_rd, ci_pr_ooo_w**, **ci_pr_w,** and **ris.**
23	Secure Address Err		0	Atomic command hits a secure address range
24	cxa_pe_capp_error		0	In CAPP mode, the CAPP unit indicates an error that causes the link to go down.
25	Tunnel Error		0	An error in a tunnelled packet.
26	Software defined		0	
27	Pec scom_err		0	Error bit from PEC SCOM engine, which is set because of an SCOM protocol error.
28:29	scomfir_error		0	Error bit from SCOM FIR engine, which is set because of an SCOM protocol error or if the FIR Mask Action 0 or Action 1 Register has a parity error.

1. Address NestBase + 0x00 is read/write (R/W).
   Address NestBase + 0x01 is bitwise clear (a “new error” overrides the clear function) (W/O0C).
   Address NestBase + 0x02 is bitwise set (W/O1S).

5.1.14 PCI Nest FIR Action 0

This register contains FIR action control.

Mnemonic NFIRAction0

Address Offset NestBase+StackBase+0x6

Bits	Field Mnemonic	Type	Reset Value	Description
0:27	Action0	R/W		Action select for corresponding bit in FIR: (Action0, Action1) = Action Select 00 Checkstop error 01 Recoverable error 10 Reserved (no action taken) 11 Freeze
0	bar_pe		Checkstop	
1	nonbar_pe		Freeze	
2	PB_to_PEC_ce		Recoverable	
3	PB_to_PEC_ue		Freeze	
4	PB_to_PEC_sue		Recoverable	
5	ary_ecc_ce		Recoverable	
6	ary_ecc_ue		Freeze	
7	ary_ecc_sue		Recoverable	
8	register_array_pe		Freeze	
9	pb_interface_pe		Checkstop	
10	pb_data_hang_errors1		Recoverable	
11	pb_hang_errors1		Recoverable	
12	rd_are_errors1		Freeze	
13	nonrd_are_errors1		Freeze	
14	pci_hang_error		Freeze	
15	pci_clock_error		Freeze	
16	AIB_Fence		Freeze	
17	hw_errors		Checkstop	
18	UnsolicitiedPBData1		Checkstop	
19	UnExpectedCResp1		Checkstop	
20	InvalidCResp1		Checkstop	
21	PBUnsupportedSize1		Checkstop	
22	PBUnsupported Cmd1		Checkstop	
23	Secure Address Err		Freeze	
24	cxa_pe_capp_error		Freeze	
25	Tunnel Error		Freeze	
26	Software defined		Freeze	
27	pec_scom_err		Recoverable	
28:29	scomfir_error		Recoverable	

5.1.15 PCI Nest FIR Action 1

This register contains FIR action control.

Mnemonic NFIRAction1

Address Offset NestBase+StackBase+0x7

Bits	Field Mnemonic	Type	Reset Value	Description
0:27	Action1	R/W		Action select for corresponding bit in FIR: (Action0, Action1) = Action Select 00 Checkstop error 01 Recoverable error 10 Reserved (no action taken) 11 Freeze
0	bar_pe		Checkstop	
1	nonbar_pe		Freeze	
2	PB_to_PEC_ce		Recoverable	
3	PB_to_PEC_ue		Freeze	
4	PB_to_PEC_sue		Recoverable	
5	ary_ecc_ce		Recoverable	
6	ary_ecc_ue		Freeze	
7	ary_ecc_sue		Recoverable	
8	register_array_pe		Freeze	
9	pb_interface_pe		Checkstop	
10	pb_data_hang_errors1		Recoverable	
11	pb_hang_errors1		Recoverable	
12	rd_are_errors1		Freeze	
13	nonrd_are_errors1		Freeze	
14	pci_hang_error		Freeze	
15	pci_clock_error		Freeze	
16	AIB_Fence		Freeze	
17	hw_errors		Checkstop	
18	UnsolicitiedPBData1		Checkstop	
19	UnExpectedCResp1		Checkstop	
20	InvalidCResp1		Checkstop	
21	PBUnsupportedSize1		Checkstop	
22	PBUnsupported Cmd1		Checkstop	
23	Secure Address Err		Freeze	
24	cxa_pe_capp_error		Freeze	
25	Tunnel Error		Freeze	
26	Software defined		Freeze	
27	pec_scom_err		Recoverable	
28:29	scomfir_error		Recoverable	

5.1.16 PCI Nest FIR Mask, Clear, Set

This register contains the Nest FIR Mask bits

Mnemonic NFIRMask

Address Offset NestBase+StackBase+0x3
NestBase+StackBase+0x4
NestBase+StackBase+0x5

Bits	Field Mnemonic	Type	Reset Value	Description
0:27	Mask	See Note 1	All ‘1’s	0 Corresponding FIR bit is not masked. 1 Corresponding FIR bit is masked. When a FIR is masked, the corresponding FIR bit is still set in the FIR, but the programmed action is not taken.

1. Address NestBase+0x03 is read/write (R/W)
   Address NestBase+0x04 is bitwise clear (W/O0C)
   Address NestBase+0x05 is bitwise set (W/O1S)

5.1.17 PCI Nest FIR WOF

This register contains the Nest FIR “who’s on first” bits.

Mnemonic NFIRWOF

Address Offset NestBase+StackBase+0x8

Bits	Field Mnemonic	Type	Reset Value	Description
0:27	See NFIR	W/C	0	Corresponds to the bits in the NFIR. Captures the first error (for example, no new FIR bits are set) after a system checkstop condition. Multiple recoverable errors can set the FIR.

5.1.18 Error Report Register 0

This register holds some of the individual error status bits that make up the FIR.

Mnemonic CerrRpt0

Address Offset NestBase+StackBase+0xA

Bits	Field Mnemonic	Type	Reset Value	Description
0	pbcqhwcfg_pe	R/O	0	FIR(01)
1	drpprictl_pe			
2	pbeinj_pe			
3	nesttrc_pe			
4	pmonctl_pe			
5	cqstat_pe			
6	pbcqmode_pe			
7	tunnel_pe			
8	predv_pe			
9	pepbcq_dctl_ecc_ce			FIR(02)
10	pepbcq_dctl_ecc_ue			FIR(03)
11	pepbcq_dctl_ecc_sue			FIR(04)
12	pepbcq0_mux_data_ce			FIR(05)
13	pepbcq1_mux_data_ce			
14	pecqpb_mux_ecc_ce			
15	pecarb_ismb_ce			
16	pepbcq0_mux_data_ue			FIR(06)
17	pepbcq1_mux_data_ue			
18	pecqpb_mux_ecc_ue			
19	pecarb_ismb_ue			
20	pepbcq0_mux_data_sue			FIR(07)
21	pepbcq1_mux_data_sue			
22	pecqpb_mux_ecc_sue (cannot hit)			
23	pecarb_capp_fsue			
24	pesmb_mux_pbcq_pe			FIR(08)
25	pesmb_mux_cqpb_pe			
26	pecqpb_dctl_rtag_perr			
27	peopc_mode_bus0_qfifo_pe			
28	Reserved			
29	pcarb_ismb_pe			

Bits	Field Mnemonic	Type	Reset Value	Description
30	cr0_atag_pe	R/O	0	FIR(09)
31	cr1_atag_pe			
32	cr2_atag_pe			
33	cr3_atag_pe			
34	rcmd0_addr_pe			
35	rcmd1_addr_pe			
36	rcmd2_addr_pe			
37	rcmd3_addr_pe			
38	rcmd0_ttag_pe			
39	rcmd1_ttag_pe			
40	rcmd2_ttag_pe			
41	rcmd3_ttag_pe			
42	cr0_ttag_pe			
43	cr1_ttag_pe			
44	cr2_ttag_pe			
45	cr3_ttag_pe			
46	pepbcq_dctl_rtag_pe			
47	peldst_data_hang_error			FIR(10)
48	perd_pb_hang_error			FIR(11)
49	pewr_pb_hang_error			
50	perd_are			FIR(12)
51	Reserved			
52	pewr_are			FIR(13)
53	peldst_pe_hang_error			FIR(14)
54	perd_pe_hang_error			
55	pewr_pe_hang_error			
56	perd_unexp_data			FIR(18)
57	peldst_unexp_data			
58	pepbcq_dctl_unexp_data			
59	nmmu_freeze			FIR(25)
60	qfifo_cnt_err			FIR(17)
61	Reserved			FIR(25)
62	aib_clkchk_error			FIR(15)
63	PFIR_Freeze			FIR(16)

5.1.19 Error Report Register 1

This register holds some of the individual error status bits that make up the FIR.

Mnemonic CerrRpt1

Address Offset NestBase+StackBase+0xB

Bits	Field Mnemonic	Type	Reset Value	Description
0	perd_unexp_cresp	R/O	0	FIR(19)
1	pewr_unexp_cresp			
2	perd_invalid_cresp			FIR(20)
3	pewr_invalid_cresp			
4	peldst_invalid_cresp			
5	peldst_invalid_tsize			FIR(21)
6	peldst_invalid_ttype			FIR(22)
7	reserved			
8	cxa_capp_error			FIR(24)
9	capp_cntl_pe			FIR(00)
10	mmiobar0_parity_error			
11	mmiobar1_parity_error			
12	phbbar_parity_error			
13	intbar_parity_error			
14	mmiobar0_mask_parity_error			
15	mmiobar1_mask_parity_error			
16	bare_parity_error			
17	dfreeze_pe			FIR(01)
18	nrdstkovr_pe			
19	nwrstkovr_pe			
20	nststkovr_pe			
21	perd_ack_dead			FIR(12)
22	pewr_ack_dead			FIR(13)
23	rtyboctl_pe			FIR(01)
24	secure_addr_err			FIR(23)
25	addrextmask_pe			FIR(01)
26	dmawq_overrun			FIR(17)
27	frag_overrun			FIR(17)
29:63	Reserved			N/A

5.1.20 PBCQ General Status Register

This register contains general status information.

Mnemonic CQStat

Address Offset NestBase+StackBase+0xC

Bits	Field Mnemonic	Type	Reset Value	Description
0	inbound_active	R/O	0	Inbound state machines are active.
1	outbound_active	R/O	0	Outbound state machines are active.

5.1.21 PBCQ Mode Register

This register contains control bits to set the PEC's SMP interconnect modes of operation.

Mnemonic PbcqMode

Address Offset NestBase+StackBase+0xD

Bits	Field Mnemonic	Type	Reset Value	Description
0	peer to peer mode	R/W	0	0 Disable peer-to-peer operations. 1 Enable peer-to-peer operations.
1:7	Reserved	R/W	0's	Implemented but reserved.
8:63	Reserved	R/O	0's	Read only.

5.1.22 MMIO Base Address Register 0

This register contains the Base Address Register of the PCIe memory. This space can be subdivided by the M32 and M64 bars in the PHB.

Mnemonic MMIOBAR0

Address Offset NestBase+StackBase+0xE

Bits	Field Mnemonic	Type	Reset Value	Description
0:39	BAR	R/W	0	Base Address 0 of PCIe memory.

5.1.23 MMIO Base Address Register Mask 0

This register contains the BAR mask used to set the size of the address space represented by the MMIO Base Address Register 0.

Note: The formula for a BAR hit is: If (SMP interconnect address(8:47) AND BAR Mask) = (BAR AND BAR Mask), the address range is a match.

Mnemonic MMIOBAR0_MASK

Address Offset NestBase+StackBase+0xF

Bits	Field Mnemonic	Type	Reset Value	Description
0:39	Mask	R/W	0	Mask value: FFFFFFFFF 64 KB address range FFFFFFFFE 128 KB address range FFFFFFFFC 256 KB address range … 8000000000 32 PB address range
Note: Address spaces must be a power of 2.				

5.1.24 MMIO Base Address Register 1

This register contains a second Base Address Register of PCIe memory. This space can be subdivided by the M32 and M64 bars in the PHB.

Mnemonic MMIOBAR1

Address Offset NestBase+StackBase+0x10

Bits	Field Mnemonic	Type	Reset Value	Description
0:39	BAR	R/W	0	Base Address 1 of PCIe memory.

5.1.25 MMIO Base Address Register Mask 1

This register contains the BAR Mask used to set the size of the address space represented by the MMIO Base Address Register 1.

Note: The formula for a BAR hit is: If (SMP interconnect Address(8:47) AND BAR Mask) = (BAR AND BAR Mask), the address range is a match.

Mnemonic MMIOBAR1_MASK

Address Offset NestBase+StackBase+0x11

Bits	Field Mnemonic	Type	Reset Value	Description
0:39	Mask	R/W	0	Mask value: FFFFFFFFFF 64 KB space FFFFFFFFFE 128K space FFFFFFFFFC 256 KB space … 8000000000 32 PB space
Note: Address spaces must be a power of 2.				

5.1.26 PHB Register Base Address Register

This register contains the SMP interconnect Base Address Register used to capture PB commands destined for the PHB. This BAR is intended to target the PHB Register space.

Note: The formula for a BAR hit is: If (SMP interconnect Address(8:49)) = (BAR), the address range is a match.

Mnemonic PHBBAR

Address Offset NestBase+StackBase+0x12

Bits	Field Mnemonic	Type	Reset Value	Description
0:41	BAR	R/W	0	BAR of PHB Register space.
Note: This is a fixed 16 KB space; therefore, a mask is not required.				

5.1.27 Interrupt Base Address Register

This register is the Base Address Register of the address space used for MSI interrupts.

Note: The formula for a BAR hit is: If (SMP interconnect Address(8:35)) = (BAR), the address range is a match.

Mnemonic INTBAR

Address Offset NestBase+StackBase+0x13

Bits	Field Mnemonic	Type	Reset Value	Description
0:27	BAR	R/W	0	Base address value of MSI address space.
Note: This space is fixed at 256 MB and does not have a mask.				

5.1.28 Base Address Enable Register

This register contains the enable bits for the SMP interconnect Base Address Registers.

Mnemonic BARE

Address Offset NestBase+StackBase+0x14

Bits	Field Mnemonic	Type	Reset Value	Description
0	MMIO_BAR0_EN	R/W	0	MMIO Base Address Register 0 BAR enable. 0 Addresses that match the BAR are not accepted on the SMP interconnect. 1 Addresses that match the BAR are accepted on the SMP interconnect.
1	MMIO_BAR1_EN	R/W	0	MMIO Base Address Register 1 BAR enable. 0 Addresses that match the BAR are not accepted on the SMP interconnect. 1 Addresses that match the BAR are accepted on the SMP interconnect.
2	PHB_BAR_EN	R/W	0	PHB Register Base Address Register BAR enable. 0 Addresses that match the BAR are not accepted on the SMP interconnect. 1 Addresses that match the BAR are accepted on the SMP interconnect.
3	INT_BAR_EN	R/W	0	MSI Register Base Address Register BAR enable. 0 Addresses that match the BAR are not accepted on the SMP interconnect. 1 Addresses that match the BAR are accepted on the SMP interconnect.

5.1.29 Data Freeze Type Register

This register controls the type of data freeze that occurs.

Mnemonic DFREEZE

Address Offset NestBase+StackBase+0x15

Bits	Field Mnemonic	Type	Reset Value	Description
0:27	Action0	R/W		1 POWER8-style freeze. All write DATA is sent with a SUE if a freeze is active. 0 POWER9-style freeze. Only write data received after a freeze is sent with SUE. Data that arrived before the freeze is sent as normal. This register only applies if the corresponding FIR bit is set to freeze.
0	bar_pe		1	
1	nonbar_pe		1	
2	PB_to_PEC_ce		1	
3	PB_to_PEC_ue		1	
4	PB_to_PEC_sue		1	
5	ary_ecc_ce		1	
6	ary_ecc_ue		1	
7	ary_ecc_sue		1	
8	register_array_pe		1	
9	pb_interface_pe		1	
10	pb_data_hang_errors		1	
11	pb_hang_errors		1	
12	rd_are_errors		1	
13	nonrd_are_errors		1	
14	pci_hang_error		1	
15	pci_clock_error		1	
16	PFIR_Freeze		1	
17	hw_errors		1	
18	UnsolicitiedPBData		1	
19	UnExpectedCResp		1	
20	InvalidCResp		1	
21	PBUnsupportedSize		1	
22	PBUnsupported Cmd		1	
23	Reserved		1	
24	cxa_pe_capp_error		1	
25	Tunnel Error		1	
26	Software defined		Freeze	

5.1.30 PBCQ Tunnel Bar Register

This register appends the upper address bits on return tunneled packets.

Mnemonic Tunnelbar

Address Offset NestBase+StackBase+0x16

Bits	Field Mnemonic	Type	Reset Value	Description
0:42	Tunnel_bar	R/W	0	Addresses 8:50 append to tunneled packets upon response.

5.1.31 PBAIB Hardware Control Register

This register contains controls that are in the PCI clock domain.

Mnemonic PbaibHwCfg

Address Offset PCIBase+0x00

Bits	Field Mnemonic	Type	Reset Value	Description
0:15	Reserved	R/W	0	Implemented but reserved for future hardware configuration bits.
16:18	osmb_datastart_mode	R/W	0	Mode bits for allowing over commit of the outbound speed matching buffer. 000 Start next packet after four data beats. 100 Start next packet after five data beats. 101 Start next packet after six data beats. 110 Start next packet after seven data beats. 111 Start next packet after eight data beats. All others are reserved.
19	Reserved	R/W	0	Implemented but reserved for future hardware configuration bits.
20:23	tx_resp_hwm	R/W	1010	Mode bits for high water mark for stalling outbound commands.
24:27	tx_resp_lwm	R/W	0010	Mode bits for low water mark for stalling outbound commands.
28:29	osmb_earlyempty_mode	R/W	10	Mode bits for allowing sending outbound SMB empty early. 00 Do not send empty toggle early. 01 Send empty toggle three cycles early. 10 Send empty toggle four cycles early. 11 Send empty toggle five cycles early.
30	PCI Clock Trace enable	R/W	0	Enable debug tracing of PCI clock signals

Bits	Field Mnemonic	Type	Reset Value	Description
31:32	trace_stack	R/W	0	00 Stack0 01 Stack1 10 Stack2 11 Reserved Trace output: 0:15 pb2rx:cmd1_dout(0:15) 16 pb2rx:sm_cntl_sendcommand_dout 17:21 pb2rx:cntl_sm_dout 22 pb2rx:ocf_tag_valid_int 23 pb2rx:ocf_last_frag_int 24:31 pb2rx:cmd0_dout(80:87) 32 pb2rx:ocf_data_start_dout 33 pb2rx:ocf_data_end_dout 34 pb2rx:ocf_cmd_abort 35 pb2rx:ocf_cmd_drop 36:43 pb2rx:ocf_cmd_tag 44:67 tx2pb:cav1_tx_cmd_dout(00:23) 68 tx2pb:ccr_ch0_inc 69 tx2pb:ccr_ch0_dec 70 tx2pb:ccr_ch1_inc 71 tx2pb:ccr_ch1_dec 72 tx2pb:ccr_ch2_inc 73 tx2pb:ccr_ch2_dec 74 tx2pb:ismb_allow_inc_credit 75 tx2pb:dsmb_allow_inc_credit 76:83 tx2pb:pbaib_tag 84 tx2pb:cca_ch0_inc 85 tx2pb:cca_ch0_dec 86 tx2pb:cca_ch3_inc 87 tx2pb:cca_ch3_dec
34:35	inject_stack	R/W	0	00 Stack0 01 Stack1 10 Stack2 11 Reserved
36:38	Inbound SMB Error Inject	R/W	0	Mode bits for enabling injection of parity and ECC errors into the inbound speed matching buffer. Parity is carried on the control fields and ECC on data. 000 No error 100 Correctable ECC error 101 Uncorrectable ECC error 110 Special uncorrectable ECC error 111 Parity error
39	Constant_err_inject	R/W	0	0 One shot error inject 1 Constant error inject

Bits	Field Mnemonic	Type	Reset Value	Description
40:42	osmb_hol_blk_cnt	R/W	0	Mode bits for allowing outbound speed matching buffer to Head of Line (HOL) block for a programmable number of cycles. 0-- Disabled, no HOL blocking. If no AIB credit is available, reject the command. 100 Enabled. Wait four PCI clocks for a credit to become available before rejecting. 101 Enabled. Wait eight PCI clocks for a credit to become available before rejecting. 110 Enabled. Wait 16 PCI clocks for a credit to become available before rejecting. 111 Enabled. Wait 32 PCI clocks for a credit to become available before rejecting.

5.1.32 PCIe Read Stack Override Register Copy

Override read engine assignment when bifurcated or trifurcated.

Mnemonic prdstkovr

Address Offset NestBase+0x02

Bits	Field Mnemonic	Type	Reset Value	Description
0:31	Stack0 Override	R/W	0	Additional read engines from 32:63 that are assigned to stack0 when enabled. Engines 0:31 are always assigned to stack0.
32:47	Stack1 Override	R/W	0	Additional read engines from 48:63 that are assigned to stack1 when enabled and not assigned to stack0. Engines 32:47 are assigned to stack1 if not assigned to stack0.
48	Override Enable	R/W	0	Enable override.
Note: This register must make the nest version *nrdstkovr*.				

5.1.33 PCI Fault Isolation Register (PFIR), Clear, Set

Mnemonic PFIR

Address Offset PCIBase+StackBase+0x0
PCIBase+StackBase+0x1
PCIBase+StackBase+0x2

Bits	Field Mnemonic	Type	Reset Value	Description
0	register_pe	See Note 1	0	PBAIB register parity error.
1	hardware_error		0	Hardware error.
2	AIB_intf_error		0	AIB interface error.
3	ETU_Reset_error		0	ETU reset error.
4	PEC_scom_error		0	Common PEC SCOM error.
5:6	scomfir_error		0	

1. Address PCIBase + 0x00 is read/write (R/W).
   Address PCIBase + 0x01 is bitwise clear (a 'new error' overrides the clear function) (W/O0C).
   Address PCIBase + 0x02 is bitwise set (W/O1S).

5.1.34 PCI FIR Action 0

This register contains FIR action control.

Mnemonic pfirAction0

Address Offset PCIBase+StackBase+0x6

Bits	Field Mnemonic	Type	Reset Value	Description
0:27	Action0	R/W		Action select for corresponding bit in FIR: (Action0, Action1) = Action Select 00 Checkstop error 01 Recoverable error 10 Reserved (no action taken) 11 Freeze
0	register_pe		Freeze	
1	hardware_error		Checkstop	
2	AIB_intf_error		Freeze	
3	ETU_Reset_error		Freeze	
4	PEC_scom_error		Recoverable	
5:6	scomfir_error	R/O	Recoverable	Read only.

5.1.35 PCI FIR Action 1

This register contains FIR action control.

Mnemonic pfirAction1

Address Offset PCIBase+StackBase+0x7

Bits	Field Mnemonic	Type	Reset Value	Description
0:27	Action0	R/W		Action select for corresponding bit in FIR: (Action0, Action1) = Action Select 00 Checkstop error 01 Recoverable error 10 Reserved (no action taken) 11 Freeze
0	register_pe		Freeze	
1	hardware_error		Checkstop	
2	AIB_intf_error		Freeze	
3	ETU_Reset_error		Freeze	
4	PEC_scom_error		Recoverable	
5:6	scomfir_error	R/O	Recoverable	Read only.

5.1.36 PCI FIR Mask, Clear, Set

This register contains the PFIR Mask bits.

Mnemonic pfirMask

Address Offset PCIBase+StackBase+0x3
PCIBase+StackBase+0x4
PCIBase+StackBase+0x5

Bits	Field Mnemonic	Type	Reset Value	Description
0:27	Mask	See Note 1	All 1's	0 Corresponding FIR bit is not masked. 1 Corresponding FIR bit is masked. When a FIR is masked, the corresponding FIR bit is still set in the FIR Register, but the programmed action is not taken.

1. Address PCIBase+0x03 is read/write (R/W).
   Address PCIBase+0x04 is bitwise clear (W/O0C).
   Address PCIBase+0x05 is bitwise set (W/O1S).

5.1.37 PCI FIR WOF

This register contains the PFIR “Who's on First” bits.

Mnemonic pfirWOF

Address Offset PCIBase+StackBase+0x8

Bits	Field Mnemonic	Type	Reset Value	Description
0:27	See PFIR	W/C	0	Corresponds to the bits in the PFIR register. Captures the first error (that is, no new FIR bits are set) after a system checkstop condition. Multiple recoverable errors can set the FIR.

5.1.38 ETU Reset Register

This register resets the ETU and PHB functions.

Mnemonic ETUReset

Address Offset PCIBase+StackBase+0x0A

Bits	Field Mnemonic	Type	Reset Value	Description
0	PHB Reset	R/W	1	Fundamental reset of the PHB and AIB logic. If this bit is written from ‘0’ to a ‘1’, the PE unit must be put into freeze first.

5.1.39 PBAIB Error Report Register

This register holds some of the individual error status bits that make up the FIR bit from the PBAIB.

Mnemonic pbaibCerrRpt

Address Offset PCIBase+StackBase+0x0B

Bits	Field Mnemonic	Type	Reset Value	Description
0	capp_sec_bar_pe	R/O	0	PFIR(00)
1	pbaibhwcfg_pe			
2	etureset_pe			
3	pbaibtxccr_pe			
4	pbaibtxdcr_pe			
5	peaib_overrun			PFIR(01)
6	peaib_qcnt_err			
7	peaib_tx_dat_err			PFIR(02)
8	peaib_cmd_pe			
9	peaib_dat_pe			
10	peaib_cmd_crd_avail_pe			PFIR(00)
11	peaib_dat_crd_avail_pe			
12	peaib_cmd_crd_pe			PFIR(02)
13	peaib_dat_crd_pe			
14	etu_scom_access_err			PFIR(03)
15	async_error			PFIR(01)
16	prdstkovr_pe			PFIR(00)
17	aib_pec_scom_err			PFIR(04)
18	pbaib_fence_pcie			PFIR(02)
19	phbreset_not_freeze			PFIR(03)
20	tag_overrun			PFIR(01)

5.1.40 PBAIB TX Command Credit Register

This register controls the inbound AIB command credits for the four AIB channels. It is only used if the nonblocking write function or queue override registers are being used and can be left as zeros. If this value has any nonzero value, it overrides the hardware defaults assigned to each channel.

Mnemonic PBAIBTXCCR

Address Offset PCIBase+StackBase+0x0D

Bits	Field Mnemonic	Type	Reset Value	Description
0:2	Max Outstanding Channel 0 Credits	R/W	0	This value is the maximum number of credits that are allowed outstanding on channel 0. Used for DMA writes. For the POWER9 processor, this value is ‘001’.
3:9	Reserved	R/O	0	Reserved.
10:15	Available Channel 0 Credits	R/W	0	This value is the maximum number of credits available for channel 0. This value cannot exceed the number of write engines available to this stack. Used for DMA writes. **Note:** If NBW (ch3) is used, this pool of credits is also used for ch3.
16:18	Max Outstanding Channel 1 Credits	R/W	0	This value is the maximum number of credits that are allowed outstanding on channel 1. Used for load responses. **Note:** For POWER9, this value is ‘001’.
19:25	Reserved	R/O	0	Reserved.
26:31	Available Channel 1 Credits	R/W	0	Used for load responses. This value is the maximum number of credits available for channel 1. This value cannot exceed the number of load engines available to this stack.
32:34	Max Outstanding Channel 2 Credits	R/W	0	Used for DMA reads. This value is the maximum number of credits that are allowed outstanding on channel 2. **Note:** For POWER9, this value is ‘001’.
35:40	Reserved	R/O	0	Reserved.
41:47	Available Channel 2 Credits	R/W	0	This value is the maximum number of credits available for channel 2. This value cannot exceed the number of read fragmentation engines available to this stack. A stack should be given one credit (fragmentation engine) per eight read engines. The minimum an active stack can be given is one and the total of all stacks can not be greater than eight. Used for DMA reads.
48:50	Max Outstanding Channel 3 Credits	R/W	0	This value is the maximum number of credits that are allowed outstanding on channel 3. Used for NBW writes. **Note:** For POWER9, this value is ‘001’ if NBW is used and ‘000’ if NBW is not used.
51:63	Reserved	R/O	0	Reserved.

5.1.41 PBAIB TX Data Credit Register

This register controls the inbound AIB data credits for the four AIB channels. It is only used if the PBAIBTXCCR register is used.

Mnemonic PBAIBTXDCR

Address Offset PCIBase+StackBase+0x0E

Bits	Field Mnemonic	Type	Reset Value	Description
0:8	Reserved	R/W	0	Reserved.
9:15	Available Channel 0 Data Credits	R/W	0	Used for DMA writes. This value should match pbaibtxccr bits 10:15.
16:25	Reserved	R/O	0	Reserved.
26:31	Available Channel 1 data Credits	R/W	0	Used for Load Responses. This value should match pbaibtxccr bits 26:31.
32:57	Reserved	R/O	0	Reserved.
58:63	Available Channel 3 data Credits	R/W	0	Used for NBW reads. This value should match pbaibtxccr bits 10:15.

6. IOP-to-PE Unit Connectivity

The 48 lanes of PCIe I/O (IOP) are configured into three IOP interfaces (IOP0, IOP1, and IOP2). IPO0 connects to PEC0, which includes PHB0. IPO1 connects to PEC1, which includes PHB1 and PHB2. IPO2 connects to PEC2, which includes PHB3, PHB4, and PHB5. Each IOP-PEC pair has a lane configuration register to control the steering between the IOP and the PHBs in each PEC. *Table 6-1*shows the IOP-PHB combinations. Only IOP2 has different combinations of PHB assignments and is controlled by bits 0:1 of the PEC2 Lane Configuration bits in the pervasive Chiplet Config Register 1 at address 0x0F000009, bits 10:11.

Table 6-1. IOP - PHB Assignments

PECx	PHB Assignment	
PEC0 Lane Configuration (0:1)	IOP0 (16x)	
	0:15	
'00'	PHB(0:15)	

PEC1 Lane Configuration (0:3)	IOP1 (16x)	
	0:7	8:15
'0000'	PHB1(0:7)	PHB2(0:7)

PEC1 Lane Configuration (0:5)	IOP2 (16x)			
	0:3	4:7	8:11	12:15
000000	PHB3(0:7)		PHB3(8:15)	
010000	PHB3(0:7)		PHB4(0:3)	
100000	PHB3(0:7)		PHB4(0:3)	PHB5(0:3)

Note: Color legend: red = mode; blue = stack0 control and signals; green = stack1 control and signals; orange = stack2 control and signals.

6.1 PHB Board Wiring Combinations

Each PHB has wiring combinations that are controlled by its lane configuration bits and a PHB swap bit. The PHB swap and lane configuration bits can be found in the pervasive Chiplet Config Register for PCI0, PCI1, and PCI2 respectively (see *Table 6-2*).

Table 6-2. PHB Lane Configuration and Swap Bit Register Mapping

Function	PCI Chiplet Configuration Register
PEC0 Lane Configuration(0:1)	PCI0 Chiplet Config Register 1 at address 0x0D000009, bits 13:14
PHB0 Swap Bit	PCI0 Chiplet Config Register 1 at address 0x0D000009, bit 12
PEC1 Lane Configuration(0:3)	PCI1 Chiplet Config Register 1 at address 0x0E000009, bits 14:17
PHB1 Swap Bit	PCI1 Chiplet Config Register 1 at address 0x0E000009, bit 12
PHB2 Swap Bit	PCI1 Chiplet Config Register 1 at address 0x0E000009, bit 13
PEC2 Lane Configuration(0:5)	PCI2 Chiplet Config Register 1 at address 0x0F000009, bits 10:15
PHB3 Swap Bit	PCI2 Chiplet Config Register 1 at address 0x0F000009, bit 7
PHB4 Swap Bit	PCI2 Chiplet Config Register 1 at address 0x0F000009, bit 8
PHB5 Swap Bit	PCI2 Chiplet Config Register 1 at address 0x0F000009, bit 9

Table 6-3. PHB0 Wiring Combinations

PHB0 Swap Bit	PEC0 Lane Configuration (0:1)	IOP0 (16X)	
		0:7	8:15
0	0b00	PHB0(0:7)	PHB0(8:15)
0	0b01	PHB0(0:7)	PHB0(15:8)
0	0b10	PHB0(7:0)	PHB0(8:15)
0	0b11	PHB0(7:0)	PHB0(15:8)
1	0b00	PHB0(15:8)	PHB0(7:0)
1	0b01	PHB0(8:15)	PHB0(7:0)
1	0b10	PHB0(15:8)	PHB0(0:7)
1	0b11	PHB0(8:15)	PHB0(0:7)

Table 6-4. PHB1 Wiring Combinations (Page 1 of 2)

PHB1 Swap Bit	PEC1 Lane Config (0:1)	IOP1 (16X)	
		0:3	4:7
0	‘00’	PHB1(0:3)	PHB1(4:7)
0	‘01’	PHB1(0:3)	PHB1(7:4)
0	‘10’	PHB1(3:0)	PHB1(4:7)
0	‘11’	PHB1(3:0)	PHB1(7:4)
1	‘00’	PHB1(7:4)	PHB1(3:0)

Table 6-4. PHB1 Wiring Combinations (Page 2 of 2)

PHB1 Swap Bit	PEC1 Lane Config (0:1)	IOP1 (16X)	
		0:3	4:7
1	‘01’	PHB1(4:7)	PHB1(3:0)
1	‘10’	PHB1(7:4)	PHB1(0:3)
1	‘11’	PHB1(4:7)	PHB1(0:3)

Table 6-5. PHB2 Wiring Combinations

PHB2 Swap Bit	PEC1 Lane Configuration (2:3)	IOP1 (16X)	
		8:11	12:15
0	‘00’	PHB2(0:3)	PHB2(4:7)
0	‘01’	PHB2(0:3)	PHB2(7:4)
0	‘10’	PHB2(3:0)	PHB2(4:7)
0	‘11’	PHB2(3:0)	PHB2(7:4)
1	‘00’	PHB2(7:4)	PHB2(3:0)
1	‘01’	PHB2(4:7)	PHB2(3:0)
1	‘10’	PHB2(7:4)	PHB2(0:3)
1	‘11’	PHB2(4:7)	PHB2(0:3)

Table 6-6. PHB3 Wiring Combinations (Page 1 of 2)

PEC2 Lane Configuration (0:1)	**PHB3 Swap** Bit	PEC2 Lane Configuration (2:3)	IOP2 (16X)			
			0:3	4:7	8:11	12:15
‘00’	0	‘00’	PHB3(0:7)		PHB3(8:15)	
‘00’	0	‘01’	PHB3(0:7)		PHB3(15:8)	
‘00’	0	‘10’	PHB3(7:0)		PHB3(8:15)	
‘00’	0	‘11’	PHB3(7:0)		PHB3(15:8)	
‘00’	1	‘00’	PHB3(15:8)		PHB3(7:0)	
‘00’	1	‘01’	PHB3(8:15)		**PHB3(**7:0)	
‘00’	1	‘10’	PHB3(15:8)		PHB3(0:7)	
‘00’	1	‘11’	PHB3(8:15)		PHB3**(0:7)**	
‘01’	0	‘00’	PHB3(0:3)	PHB3(4:7)	PHB4	PHB4
‘01’	0	‘01’	PHB3(0:3)	PHB3(7:4)	PHB4	PHB4
‘01’	0	‘10’	PHB3(3:0)	PHB3(4:7)	PHB4	PHB4
‘01’	0	‘11’	PHB3(3:0)	PHB3(7:4)	PHB4	PHB4
‘01’	1	‘00’	PHB3(7:4)	PHB3(3:0)	PHB4	PHB4
‘01’	1	‘01’	PHB3(7:4)	PHB3**(0:3)**	PHB4	PHB4
‘01’	1	‘10’	PHB3(4:7)	PHB3(3:0)	PHB4	PHB4

Note: Color legend: red = mode; blue = stack0 control and signals; green = stack1 control and signals; orange = stack2 control and signals.

Table 6-6. PHB3 Wiring Combinations (Page 2 of 2)

PEC2 Lane Configuration (0:1)	**PHB3 Swap** Bit	PEC2 Lane Configuration (2:3)	IOP2 (16X)			
			0:3	4:7	8:11	12:15
'01'	1	'11'	PHB3(4:7)	PHB3(0:3)	PHB4	PHB4
'10'	0	'00'	PHB3(0:3)	PHB3(4:7)	PHB4	PHB5
'10'	0	'01'	PHB3(0:3)	PHB3(7:4)	PHB4	PHB5
'10'	0	'10'	PHB3(3:0)	PHB3(4:7)	PHB4	PHB5
'10'	0	'11'	PHB3(3:0)	PHB3(7:4)	PHB4	PHB5
'10'	1	'00'	PHB3(7:4)	PHB3(3:0)	PHB4	PHB5
'10'	1	'01'	PHB3(7:4)	PHB3(0:3)	PHB4	PHB5
'10'	1	'10'	PHB3(4:7)	PHB3(3:0)	PHB4	PHB5
'10'	1	'11'	PHB3(4:7)	PHB3(0:3)	PHB4	PHB5

Note: Color legend: red = mode; blue = stack0 control and signals; green = stack1 control and signals; orange = stack2 control and signals.

Table 6-7. PHB4 Wiring Combinations

PEC2 Lane Configuration (0:1)	**PHB4 Swap** Bit	PEC2 Lane Configuration (4:5)	IOP2 (16X)	
			8:11	12:15
'01'	0	'00'	PHB4(0:3)	PHB4(4:7)
'01'	0	'01'	PHB4(0:3)	PHB4(7:4)
'01'	0	'10'	PHB4(3:0)	PHB4(4:7)
'01'	0	'11'	PHB4(3:0)	PHB4(7:4)
'01'	1	'00'	PHB4(7:4)	PHB4(3:0)
'01'	1	'01'	PHB4(4:7)	PHB4(3:0)
'01'	1	'10'	PHB4(7:4)	PHB4(0:3)
'01'	1	'11'	PHB4(4:7)	PHB4(0:3)
'10'	0	'0x'	PHB4(0:3)	PHB5
'10'	0	'1x'	PHB4(3:0)	PHB5
'10'	1	'0x'	undefined	PHB5
'10'	1	'1x'	undefined	PHB5

Note: Color legend: red = mode; green = stack1 control and signals; orange = stack2 control and signals.

Table 6-8. PHB5 Wiring Combinations

PHB5 Swap Bit	PEC2 Lane Configuration (0:1)	PEC2 Lane Configuration (5)	IOP2 (16x)
			12:15
0	‘10‘	‘0‘	PHB5(0:3)
0	‘10‘	‘1‘	PHB5(3:0)
1	‘10‘	‘0‘	undefined
1	‘10‘	‘1‘	undefined

Note: Color legend: red = mode; orange = stack2 control and signals.

7. Error Handling

The PBCQ logic has implicit behavior when an error occurs that can be different depending on the error that occurs. There are specific actions that always take place based on the error class. In addition, there are also other operations that might occur, such as dropping a command or returning all ones data.

7.1 Recovery Classes

7.1.1 INF Class Error Handling

No special handling is performed for the INF class errors. The logic behaves as if they never occurred at all.

7.1.2 Freeze Class Error Handling

When a Freeze is detected, the following actions occur:

- The PHB is fenced from sending data.
- All outbound CI stores are dropped.
- All outbound CI loads return all ones data to the SMP interconnect.
- All previously started DMA writes and interrupt operations complete on the SMP interconnect.
- All previously started DMA reads complete on the SMP interconnect and the data is discarded.
- Any commands that have not started on the SMP interconnect are dropped.

7.1.3 Checkstop Class Error Handling

Checkstop errors act like INF errors in that the logic does not really do anything special except raise the checkstop signal to pervasive. It is expected that checkstop halts all operations, and either a full IPL or a memory preserving IPL (MPIPL) is required to restart the PEC unit.

7.2 Error Recovery

Error recovery consists of a sequence of operations that restore the PBCQ to the functional state that existed before the error occurred. As described in *Section 7.1 Recovery Classes* on page 93, there are three error classes and each one has its own sequence of operations for recovery. The classes are:

- Informative (INF)
- Freeze
- Checkstop

7.2.1 Recovery Sequences

This section describes the sequences to recover each of the error classes.

7.2.1.1 INF Class

An INF class error has been identified. These errors are configured to not put the PBCQ into a device error/freeze state or cause a fence. They are for informational purposes only. They have no mainline logic side effects. They cause the recovery error signal to be driven to the pervasive logic and are generally handled by the FSP.

- Use the SCOM interface for all steps unless stated otherwise.
- Term <value read> is the value read from the same register in a prior step.

Table 7-1. INF Recovery Sequence

Step	Access	Address	Write Data	Comment
Recov_1:	Rd	PCIBase+StackBase+0x00	N/A	Read PCI Clock Domain FIR
Recov_2:	Rd	PCIBase+StackBase+0x0B	N/A	Read pbaib_errRpt registers for specific error cause.
Recov_3:	Rd	NestBase+StackBase+0x00	N/A	Read Nest Clock Domain FIR
Recov_4:	Rd	NestBase+StackBase+0x0A NestBase+StackBase+0x0B	N/A	Read CerrRpt Registers for specific error cause.
• Based on the error in the FIR, firmware takes the appropriate RAS action. Typically for recoverable errors, a threshold mechanism is called out as a failure after a specific number of repeated correctable errors. • The contents of the CerrRpt Register is saved in case it is required by the hardware team for further debug.				
Recov_5:	Wr	NestBase+StackBase+0x01	Inverted <value read>	FIR clear address
• Clear the recoverable errors				
Recov_6	NOP	N/A	N/A	INF recovery sequence complete.

7.2.1.2 Freeze Class

There is no explicit indication to firmware that a freeze error has occurred (no interrupt or wire to the pervasive). Firmware detects a freeze and starts the recovery process.

The SCOM interface is used for all steps unless stated otherwise.

Table 7-2. Freeze Recovery Sequence

Step	Access	Address	Write Data	Comment
Recov_1	NOP	N/A	N/A	An EEH or freeze event is detected, typically and all ones data response to a CI load.
Recov_2	Rd	MMIO load to PHB register	N/A	Any read to a PHB MMIO register, typically searching for the EEH error.
• When an EEH event is detected, firmware attempts to read the PHB to determine the error cause. If the PHB read returns all ones data, it indicates that the PEC has entered the freeze state, and PEC freeze recovery should start.				
Recov_3	Rd	PCIBase+StackBase+0x00	N/A	Read PCI Clock Domain FIR (NFIR)
Recov_4	Rd	NestBase+StackBase+0x00	N/A	Read Nest Clock Domain FIR (NFIR)
• If NFIR(16) AIB Fence is set, this indicates that the PHB has had a fatal error. Firmware executes the PHB fatal error recovery procedure before returning to Recov_4. For all other errors, proceed to Recov_4.				
Recov_5	Wr	PCIBase+StackBase+0x0A	0x8000000000000000	Put the PHB in reset
Recov_6	Rd	NestBase+StackBase+0x0A, +0x0B If NFIR(23) = '1' read PBAIB Error Report: PCIBase+StackBase +0x0B	N/A	Read CerrRpt registers for specific error cause. If NFIR bit 23 = '1', the PBAIB error report must be read to determine the specific error.
• Based on the error in the FIR, firmware takes the appropriate RAS action. Typically for freeze errors, the PCI bus is recovered and the error logged with no other actions. • The contents of the CerrRpt Register is saved in case it is required by the hardware team for further debug.				
Recov_7	Rd	NestBase+StackBase+0x0C	N/A	Read the CQ status
• Bits 0:1 of the CQStat Register indicate if the PEC is actively working on the DMA/CI operation. • Repeat Recov_6 until 0:1 = '00'.				
Recov_8	Wr	PCIBase+StackBase+0x01	Inverted <value read>	PCI FIR clear address
Recov_9	Wr	NestBase+StackBase+0x01	Inverted <value read>	Nest FIR clear address
• Clear the freeze errors. • The PBCQ is now out of freeze mode.				
Recov_10	Wr	PCIBase+StackBase+0x0A	0x0000000000000000	Remove the PHB from reset
Recov_11	NOP	N/A	N/A	Freeze recovery sequence complete.
• After freeze recovery, the PCI bus must be re-initialized. Proceed to the PHB initialization sequence (Phase 4).				

7.2.1.3 Checkstop Class

A checkstop error has been identified. Typically no recovery by PowerVM is required. After the FSP has collected the required data, a full system IPL is required. If a checkstop occurs that can be recovered as a MPIPL, proceed to *Section 7.2.1.4. Memory Preserving IPL*.

7.2.1.4 Memory Preserving IPL

Use the SCOM interface for all steps unless stated otherwise.

Table 7-3. Memory Preserving IPL Recover Sequence

Step	Access	Address	Write Data	Comment
Recov_1	NOP	N/A	N/A	Detect a checkstop error can be recovered with MPIPL
Recov_2	Wr	NestBase+0x02	0x00000020_ 00000000	Force PEC into freeze mode by writing bit 26.
• Multiple bits help identify this as MPIPL instead of a random Freeze.				
Recov_3	Wr	PCIBase +0x0A	0x800000000 00000000	Put the PHB in reset.
Recov_4	Rd	NestBase+0x0F	N/A	Read the CQ status.
• CQStat Register bits 0:1 indicate if the PEC is actively working on the DMA/CI operation. • Repeat Recov_4 until 0:1 = '00'.				
Recov_5	Wr	NestBase+0x45	All 0's	Clear BARE register.
• Clear the BAR and IRSN enables to avoid addressing conflict post IPL.				
Recov_6	Wr	NestBase+0x01	Inverted <value read>	FIR clear address
• Clear the freeze errors. • The PBCQ is now out of freeze mode.				
Recov_7	NOP	N/A	N/A	MPIPL recovery wequence complete.
• After an MPIPL recovery, leave the PHB in reset for POWERVM to re-initialize the PCI bus.				

8. PEC Verification Register Initialization

These steps are written assuming the verification environment uses dials. To initialize with register operations, see the POWERVM initialization instructions.

The testcase-specific value (<TC specific value>) must be supplied by the RTX.

For basic I/O operation, the register defaults should be adequate for operation. The registers listed in *Table 8-1* must be configured.

8.1 PEC Verification Initialization Sequence

Table 8-1 shows the PEC verification initialization sequence.

Table 8-1. PEC Initialization Sequence

Step	Dial	Write Data	Comment
Init_1	N/A	N/A	Apply PEC flush resets.
• Assumption is simulator is called with initmode = normal, so that hardware reset values are applied.			
Init_2	pe_mmio_bar0	<TC specific value>	Bits 8:47 of the Base Address
• Typically used for PCIe memory region			
Init_3	pe_mmio_bar1	<TC specific value>	Bits 8:47 of the Base Address
• Typically used as a fail over region for a failing PHB PCIe memory region			
Init_4	pe_phb_bar	<TC specific value>	Bits 8:51 of the Base Address
• Must be used for the PHB internal MMIO register region (4 KB region)			
Init_5	pe_mmio_mask0	<TC specific value>	Specifies the size of the address range
• 64 KB to 32 PB region			
Init_6	pe_mmio_mask1	<TC specific value>	Specifies the size of the address range
• 64 KB to 32 PB region			
Init_7	pe_mmio_bar0_en	ON	Enables Bar 0 compare
Init_8	pe_mmio_bar1_en	ON	Enables Bar 1 compare
Init_9	pe_phb_bar_en	ON	Enables Bar 2 compare
Init_10	pe_etu_reset	OFF	Take ETU out of reset

Glossary

ACK	Acknowledge
AIB	ASIC interconnect bus
APC	Attached processor command
ARE	Address error
ASB	Accelerator switch board notify (also known as ASN)
ASIC	Application-specific integrated circuit
ASN	Accelerator switch board notify
aTag	Acknowledge tag
BAR	Base address register
BARE	Base address register enable
CAM	Content addressable memory
CAPI	Coherent Accelerator Processor Interface
CAPP	Coherently attached processor proxy
CFG	Configuration register core
CI	Cache inhibited
CL	Cache line
CQ	Common queue
CQPB	Common queue processor bus
CRESP	Combined response
DMA	Direct memory access
DTAG	Data routing tag
ECC	Error correction code
EEH	Enhanced error handling
EOI	End of interrupt
ETU	Express transaction unit
FIR	Fault isolation register
FSP	Flexible service processor

HOL	Head of line
INF	Informative
IPL	Initial program load
IRSN	Interrupt request source number
ISMB	Inbound speed matching buffer
KB	Kilobyte
LPC	Lowest point of coherency
Lsb	Least-significant bit
LSFR	Linear Feedback Shift Register
LSI	Level-sensitive interrupt
MB	Megabyte
MC	Memory controller
MMIO	Memory-mapped input/output
MPIPL	Memory-preserving initial program load
MSG	Message
MSI	Message-signaled interrupt
Mux	Multiplexer
NBW	Non-blocking writes
NFIRWOF	Nest FIR who’s on first
NMMU	Nest memory management unit
NOP	No operation
OCF	Outbound completion forwarding interface
OOO	Out of order
OSMB	Outbound speed matching buffer array
OW	Octword
PADE	Page mode, allocate, decrement, extend
PB	SMP interconnect
PCIASIC	Peripheral component interconnect application-specific integrated circuit
PCIe	Peripheral Component Interconnect Express

PCS	Physical coding sublayer
PEC	PCIe controller
PEST	PE state table
PHB	PCI host bridge
PHY	Physical I/O
PIPE	Physical I/O interface PCI express
PLL	Phase locked loop
PMA	Physical media access
POR	Power-on reset
presp	Partial response
R/O	Read-only
R/W	Read-write
R/W1C	Read-write ‘1’ to clear
RAS	Reliability, availability, serviceability
RCMD	Reflected command
REQ	Request
RIS	Request for interrupt service
RO	Read only
RTAG	Routing tag
RTX	Verification code
RWNITC	Processor bus command to read with no intent to modify
SCM	Single-chip module
SCOM	Serial communications
SERDES	Serializer/Deserializer
sfSTAT	Signal from the SMP interconnect that accompanies the data
SMB	Speed matching buffer
SMP	Symmetric multiprocessor
SRAM	Static random-access memory
STQ	Store queue

SUE	Special uncorrectable error
TC	Test case
TCE	Translation cache entry
TCE	Translation cache entry
TkTCAM	Ticket content addressable memory
TLDLP	Transaction and data link layer
TLP	Transaction layer protocol
TX	Transmit
UU	Unit number
W/C	Write-clear
W/O	Write-only
W/O0C	Write-only ‘0’ to clear
W/O1S	Write-only ‘1’ to set
WOF	Who’s on first