PCI-SIG ENGINEERING CHANGE NOTIFICATION

TITLE:	Enhanced DPC (eDPC)
DATE:	Introduced: April 4, 2012; Last Updated: November 14, 2012 Final PWG Approval November 15, 2012
AFFECTED DOCUMENTS:	PCI Express Base Specification Revision 3.0 Downstream Port Containment (DPC) ECN
SPONSORS:	Joe Cowan, HP; Mahesh Wagh, Intel

Part I

1. Summary of the Functional Changes

This optional normative ECN defines enhancements to the Downstream Port Containment (DPC) ECN, an ECN that enabled automatic disabling of the Link below a Downstream Port following an uncorrectable error. The DPC ECN defined functionality for both Switch Downstream Ports and Root Ports. This ECN mostly defines functionality that is specific to Root Ports, functionality referred to as "RP Extensions for DPC".

Key enhancements for Root Ports include fine-grained controls for managing Programmed I/O (PIO) errors, and Completion synthesis to avoid Completion Timeouts for outstanding Non-Posted Requests. A new mechanism for both Root Ports and Switch Downstream Ports enables software triggering of DPC.

2. Benefits as a Result of the Changes

Beyond the benefits provided by the DPC ECN, this ECN mainly standardizes additional DPC functionality that's specific to Root Ports. This enables a more widespread and consistent implementation of that functionality, while reducing the amount of platform-specific software.

3. Assessment of the Impact

Root Ports that implement RP Extensions for DPC are most heavily impacted. Root Ports and Switches that implement the other optional extensions in this ECN are modestly impacted.

4. Analysis of the Hardware Implications

RP Extensions for DPC increases the size of the DPC Capability structure, as well as using certain bits and field values that were previously Reserved. RPs that implemented original DPC and proprietary implementations of any functionality now architected by this Enhanced DPC ECN will have to maintain those proprietary implementations if they wish to maintain compatibility with old software that uses that functionality.

5. Analysis of the Software Implications

Software that supports the original DPC functionality but is unaware of this new functionality will continue to work unmodified. New or updated software is required to enable and utilize this new functionality.

6. Analysis of the C&I Test Implications

If C&I tests are developed, then the associated extended capability as well as error trigger event will need to be developed. This may entail developing a DPC-focused DUT that can be instructed to simulate an async removal event and trigger DPC within a Downstream Port (a port that is physically upstream from the DUT).

Part II

Detailed Description of the change

Note to Editor & Reviewers: Purple text in this ECN is text that was added or modified by the original Downstream Port Containment (DPC) ECN or its erratum.

Modify Terms and Acronyms as shown:

Terms and Acronyms

RP PIO — Root Port Programmed I/O. See Section 6.2.10.3.

Modify Section 2.7.2.2 as shown:

2.7.2.2. Rules For Use of Data Poisoning

- ❑ Support for TLP poisoning in a Transmitter is optional.
- ❑ Data poisoning applies only to the data within a Write Request (Posted or Non-Posted), a Message with Data, an AtomicOp Request, a Read Completion, or an AtomicOp Completion.
  - Poisoning of a TLP is indicated by a 1b value in the EP bit
  - Transmitters are permitted to set the EP bit to 1b only for TLPs that include a data payload. The behavior of the receiver is not specified if the EP bit is set for any TLP that does not include a data payload.
- ❑ If a Transmitter supports data poisoning, TLPs that are known to the Transmitter to include bad data must use the poisoning mechanism defined above.
- ❑ If a Downstream Port supports Poisoned TLP Egress Blocking, the Poisoned TLP Egress Blocking Enable bit is Set, and a poisoned TLP targets going out the Egress Port, the Port must handle the TLP as a Poisoned TLP Egress Blocked error unless there is a higher precedence error. See Sections 6.2.3.2.3, 6.2.5, and 7.xx.2. Further:
  - The Port must not transmit the TLP.
  - If the Poisoned TLP Egress Blocked error is unmasked and DPC is enabled, DPC must be triggered and the Port must behave as described in Section 2.9.3.
  - If DPC is not triggered and the TLP is a Non-Posted Request, the Port must return a Completion the same as if the TLP had triggered DPC. See Section 2.9.3.

Add the following to the end of Section 2.8

IMPLEMENTATION NOTE

Completion Timeout Prefix/Header Log Capable

The prefix/header of the Request TLP associated with a Completion Timeout may optionally be recorded by Requesters that implement the Advanced Error Reporting Capability. Support for

recording of the prefix/header is indicated by the value of the Completion Timeout Prefix/Header Log Capable bit in the Advanced Error Capabilities and Control register.

A Completion Timeout may be the result of improper configuration, system failure, or Async Removal (see Section 6.7.5). In order for host software to distinguish a Completion Timeout error after which continued normal operation is not possible (e.g., after one caused by improper configuration or a system failure) from one where continued normal operation is possible (e.g., after an Async Removal), it is strongly encouraged that Requesters log the Request TLP prefix/header associated with the Completion Timeout.

Add Section 2.9.3:

2.9.3. Transaction Layer Behavior During Downstream Port Containment

During Downstream Port Containment (DPC), the LTSSM associated with the Downstream Port is directed to the Disabled state. Once it reaches the Disabled state, it remains there as long as the DPC Trigger Status bit in the DPC Status register is Set. See Section 6.2.10 for requirements on how long software must leave the Downstream Port in DPC. This section specifies the Transaction Layer's behavior once DPC has been triggered, and as long as the Downstream Port remains in DPC.

- Once DPC has been triggered, no additional (Upstream) TLPs are accepted from the Data Link Layer.
- If the condition that triggered DPC was associated with an Upstream TLP, any subsequent Upstream TLPs that were already accepted from the Data Link Layer must be discarded silently.

The Downstream Port handles (Downstream) TLPs submitted by the device core in the following manner.

- If the condition that triggered DPC was associated with a Downstream TLP, any prior Downstream TLPs are permitted to be dropped silently or transmitted before the Link goes down. Otherwise, the following rules apply.
- For each Non-Posted Request, the Port must return a Completion and discard the Request silently. The Completer ID field must contain the value associated with the Downstream Port.
  - If the DPC Completion Control bit is Set in the DPC Control register, then Completions are generated with Unsupported Request (UR) Completion Status.
  - If the DPC Completion Control bit is Clear, Completions are generated with Completer Abort (CA) Completion Status.
  - ~~If DPC occurs while any Non-Posted Requests are still outstanding, the associated Requesters will encounter Completion Timeouts. The solution stack should comprehend and account for this possibility.~~
- The Port must terminate any PME_Turn_Off handshake Requests targeting the Port in such a way that the Port is considered to have acknowledged the PME_Turn_Off Request (see the Implementation Note in Section 5.3.3.2.1).

- ❑ The Port must handle Vendor Defined Message Requests as described in Section 2.2.8.6. (e.g., silently discard Vendor Defined Type 1 Message Requests that it is not designed to receive) since the DL_Down prevents the Request from reaching its targeted Function.
- ❑ For all other Posted Requests and Completions, the Port must silently discard the TLP.

For any outstanding Non-Posted Requests where DPC being triggered prevents their associated Completions from being returned, the following apply:

- ❑ For Root Ports that support RP Extensions for DPC, the Root Port may track certain Non-Posted Requests, and when DPC is triggered, synthesize a Completion for each tracked Request. This helps avoid Completion Timeouts that would otherwise occur as a side-effect of DPC being triggered. Each synthesized Completion must have a UR or CA Completion Status as determined by the DPC Completion Control bit. The set of Non-Posted Requests that get tracked is implementation-specific, but it is strongly recommended that all Non-Posted Requests that are generated by host processor instructions (e.g., "read", "write", "load", "store", or one that corresponds to an AtomicOp) be tracked. Other candidates for tracking include peer-to-peer Requests coming from other Root Ports and Requests coming from Root Complex Integrated Endpoints.
- ❑ Otherwise, the associated Requesters may encounter Completion Timeouts. The software solution stack should comprehend and account for this possibility.

Modify Section 6.2.3.2.3 as shown:

6.2.3.2.3. Error Pollution

…

For errors detected in the Transaction layer and Uncorrectable Internal Errors, it is permitted and recommended that no more than one error be reported for a single received TLP, and that the following precedence (from highest to lowest) be used:

- ❑ Uncorrectable Internal Error
- ❑ Receiver Overflow
- ❑ …
- ❑ Unsupported Request (UR), Completer Abort (CA), or Unexpected Completion[1]
- ❑ Poisoned TLP Received or Poisoned TLP Egress Blocked

The Completion Timeout error is not in the above precedence list, since it is not detected by processing a received TLP. Errors listed under the same bullet are mutually exclusive, so their relative order does not matter.

[1] ~~These are mutually exclusive errors, so their relative order does not matter.~~

Modify Section 6.2.3.2.4 as follows:

6.2.3.2.4. Advisory Non-Fatal Error Cases

…

If software wishes for an agent with AER to handle what would normally be an Advisory Non-Fatal Error case as being more serious, software can escalate the severity of the uncorrectable error to fatal, in which case the agent (if enabled) will signal the error with ERR_FATAL.

This section covers Advisory Non-Fatal Error handling for errors managed by the PCI Express Extended Capability and AER. Section 6.2.10.3 covers the RP PIO error handling mechanism for Root Ports that support RP Extensions for DPC. RP PIO Advisory Non-Fatal Errors are similar in concept to AER Advisory Non-Fatal Errors, but apply to different error cases and are managed by different controls.

…

6.2.3.2.4.2. Intermediate Receiver

When a Receiver that's not serving as the ultimate PCI Express destination for a TLP detects[2] a non-fatal error with the TLP, this "intermediate" Receiver must handle this case as an Advisory Non-Fatal Error.[3] A Receiver with AER signals the error (if enabled) by sending an ERR_COR Message. A Receiver without AER sends no error Message for this case. An exception to the intermediate Receiver case for Root Complexes (RCs) is noted below.

An example where the intermediate Receiver case occurs is a Switch that detects poison or bad ECRC in a TLP that it is routing. Even though this was an uncorrectable (but non-fatal) error at this point in the TLP's route, the intermediate Receiver handles it as an Advisory Non-Fatal Error, so that the ultimate Receiver of the TLP (i.e., the Completer for a Request TLP, or the Requester for a Completion TLP) is not precluded from handling the error more appropriately according to its error settings. For example, a given Completer that detects poison in a Memory Write Request[4] might have the error masked (and thus go unsignaled), whereas a different Completer in the same hierarchy might signal that error with ERR_NONFATAL.

A Poisoned TLP Egress Blocked error is never handled as an intermediate Receiver case since it is not detected as a part of processing a received TLP.

…

[2] If the Receiver does not implement ECRC Checking or ECRC Checking is not enabled, the Receiver will not detect an ECRC Error.

[3] If the severity is fatal, the error is not an Advisory Non-Fatal Error, and must be signaled (if enabled) with ERR_FATAL.

[4] See Section 2.7.2.2 for special rules that apply for poisoned Memory Write Requests.

6.2.3.2.4.4. Requester with Completion Timeout

This section applies to Requesters other than Root Ports performing programmed I/O (PIO). See Section 6.2.10.3 for related RP PIO functionality in Root Ports that support RP Extensions for DPC.

When the Requester of a Non-Posted Request times out while waiting for the associated Completion, the Requester is permitted to attempt to recover from the error by issuing a separate subsequent Request. The Requester is permitted to attempt recovery zero, one, or multiple (finite) times, but must signal the error (if enabled) with an uncorrectable error Message if no further recovery attempt will be made.

If the severity of the Completion Timeout is non-fatal, and the Requester elects to attempt recovery by issuing a new request, the Requester must first handle the current error case as an Advisory Non-Fatal Error.[5] A Requester with AER signals the error (if enabled) by sending an ERR_COR Message. A Requester without AER sends no error Message for this case.

Note that automatic recovery by the Requester from a Completion Timeout is generally possible only if the Non-Posted Request has no side-effects, but may also depend upon other considerations outside the scope of this specification.

Modify Section 6.2.3.2.5 as follows:

6.2.3.2.5. Requester Receiving a Completion with UR/CA Status

When a Requester receives back a Completion with a UR/CA Status, generally the Completer has handled the error as an Advisory Non-Fatal Error, assuming the error severity was non-fatal at the Completer (see Section 6.2.3.2.4.1). The Requester must determine if any error recovery action is necessary, what type of recovery action to take, and whether or not to report the error.

If the Requester needs to report the error, the Requester must do so solely through a Requester-specific mechanism. For example, many devices have an associated device driver that can report errors to software. As another important example, the Root Complex on some platforms returns all 1's to software if a Configuration Read Completion has a UR/CA Status.

Section 6.2.10.3 covers RP PIO controls for Root Ports that support RP Extensions for DPC. Outside of the RP PIO mechanisms, ~~The~~ Requesters ~~is~~ are not permitted to report the error using PCI Express logging and error Message signaling.

[5] If the severity is fatal, the error is not an Advisory Non-Fatal Error, and must be signaled (if enabled) with ERR_FATAL. The Requester is strongly discouraged from attempting recovery since sending ERR_FATAL will often result in the entire hierarchy going down.

The following modification to Figure 6-3 was made by an erratum against the original Downstream Port Containment ECN, and is shown here for reference:

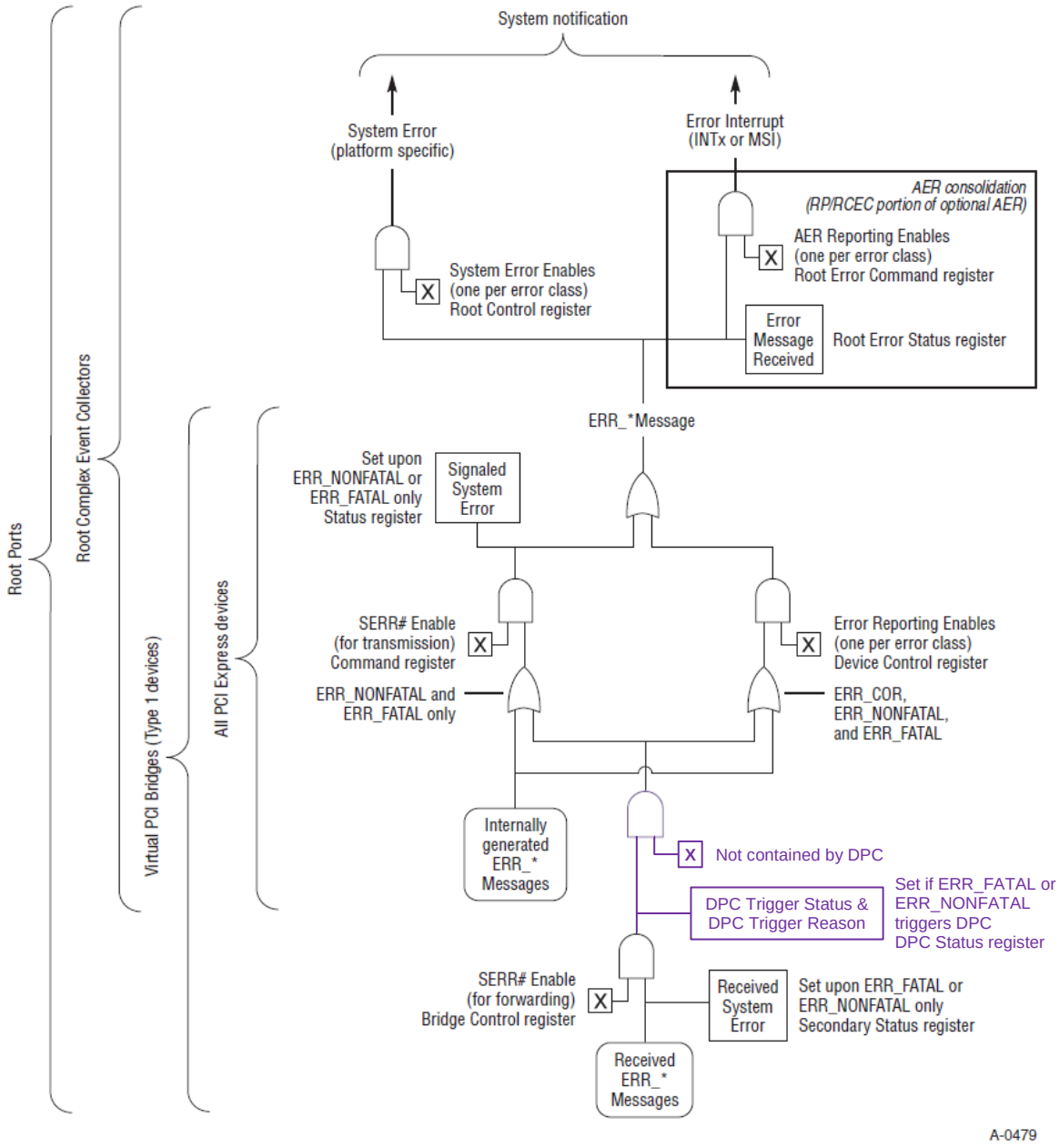

Figure 6-3: Pseudo Logic Diagram for Error Message Controls

Modify Table 6-5 as shown:

Table 6-5: Transaction Layer Error List

Error Name	Error Type (Default Severity)	Detecting Agent Action[74]	References
Poisoned TLP Received	Uncorrectable (Non-Fatal)	*Receiver:* Send ERR_NONFATAL to Root Complex or ERR_COR for the Advisory Non-Fatal Error cases described in Sections 6.2.3.2.4.2 and 6.2.3.2.4.3. Log the prefix/header of the Poisoned TLP.	Section 2.7.2.2
Poisoned TLP Egress Blocked		*Downstream Port Transmitter:* Send ERR_NONFATAL to Root Complex. Log the prefix/header of the poisoned TLP.	Section 2.7.2.2
…		…	…
Completion Timeout		*Requester:* *Send ERR_NONFATAL to Root Complex or ERR_COR for the Advisory Non-Fatal Error case described in Section 6.2.3.2.4.4.* If the Completion Timeout Prefix/Header Log Capable bit is Set in the Advanced Error Capabilities and Control register, log the prefix/header of the Request TLP that encountered the error.	Section 2.8

Add Section 6.2.10:

6.2.10. Downstream Port Containment (DPC)

Downstream Port Containment (DPC) is an optional normative feature of a Downstream Port. DPC halts PCI Express traffic below a Downstream Port after an unmasked uncorrectable error is detected at or below the Port, avoiding the potential spread of any data corruption, and permitting error recovery if supported by software. A Downstream Port indicates support for DPC by implementing a DPC Extended Capability structure, which contains all DPC control and status bits. See Section 7.xx.

DPC is disabled by default, and cannot be triggered unless enabled by software using the DPC Trigger Enable field. When the DPC Trigger Enable field is set to 01b, DPC is enabled and is triggered when the Downstream Port detects an unmasked uncorrectable error or when the Downstream Port receives an ERR_FATAL Message. When the DPC Trigger Enable field is set to 10b, DPC is enabled and is triggered when the Downstream Port detects an unmasked uncorrectable error or when the Downstream Port receives an ERR_NONFATAL or

ERR_FATAL Message. In addition to uncorrectable errors of the type managed by the PCI Express Extended Capability and Advanced Error Reporting (AER), RP PIO errors can be handled as uncorrectable errors. See Section 6.2.10.3. There is also a mechanism described in Section 6.2.10.4 for software or firmware to trigger DPC.

When DPC is triggered due to receipt of an uncorrectable error Message, the Requester ID from the Message is recorded in the DPC Error Source ID register and that Message is discarded and not forwarded Upstream. When DPC is triggered by an unmasked uncorrectable error, that error will not be signaled with an uncorrectable error Message, even if otherwise enabled. However, when DPC is triggered, DPC can signal an interrupt or send an ERR_COR Message if enabled. See Sections 6.2.10.1 and 6.2.10.2.

When DPC is triggered, the Downstream Port immediately Sets the DPC Trigger Status bit and DPC Trigger Reason field to indicate the triggering condition (unmasked uncorrectable error, ERR_NONFATAL, ~~or~~ ERR_FATAL, RP PIO error, or software triggered), and disables its Link by directing the LTSSM to the Disabled state. Once the LTSSM reaches the Disabled state, it remains in that state until the DPC Trigger Status bit is Cleared. To ensure that the LTSSM has time to reach the Disabled state or at least to bring the Link down under a variety of error conditions, software must leave the Downstream Port in DPC until the Data Link Layer Link Active bit in the Link Status register reads 0b. See Section 7.8.8. See Section 2.9.3 for other important details on Transaction Layer behavior during DPC.

After DPC has been triggered in a Root Port that supports RP Extensions for DPC, the Root Port may require some time to quiesce and clean up its internal activities, such as those associated with DMA read Requests. When the DPC Trigger Status bit is Set and the DPC RP Busy bit is Set, software must leave the Root Port in DPC until the DPC RP Busy bit reads 0b.

After software releases the Downstream Port from DPC, the associated Link will normally attempt to retrain. Software can use Data Link Layer State Changed interrupts, DL_Active ERR_COR signaling, or both, to signal when the Link reaches the DL_Active state again. See Sections 6.7.3.3 and 6.2.10.5.

IMPLEMENTATION NOTE

Data Value of All ~~Ones~~1's

Many platforms, including those supporting RP Extensions for DPC, can return a data value of all ~~ones~~ 1's to software when an error is associated with a PCI Express Configuration, I/O, or Memory Read Request. During DPC, the Downstream Port discards Requests destined for the Link and completes them with an error (i.e., either with an Unsupported Request (UR) or Completer Abort (CA) Completion Status). By ending a series of MMIO or configuration space operations with a read to an address with a known data value not equal to all ~~ones~~1's, software may determine if a Completer has been removed or DPC has been triggered.

Also see the Implementation Note "Use of RP PIO Advisory Non-Fatal Errors".

IMPLEMENTATION NOTE

Selecting Non-Posted Request Response During DPC

The DPC Completion Control bit determines how a Downstream Port responds to a Non-Posted Request (NPR) received during DPC. The selection needs to take into account how the rest of the platform handles PCI Express uncorrectable error recovery.

While specific PCI Express uncorrectable error recovery mechanisms in a platform are outside the scope of this specification, here are some guidelines based on general considerations.

If the platform or drivers do not support a PCI Express uncorrectable error recovery strategy, there's no envisioned benefit to enabling DPC, and thus no need to select the NPR response.

If the PCI Express uncorrectable error recovery strategy relies on software detecting containment by looking for all 1's returned by PIO reads, then a UR Completion may be the more appropriate selection, assuming the RP synthesizes an all 1's return value for PIO reads that return UR Completions. The all 1's synthesis would need to occur for PIO reads that target Configuration Space, Memory Space, and perhaps I/O Space.

If the PCI Express uncorrectable error recovery strategy utilizes a mechanism that handles UR and CA Completions differently for PIO reads, then a CA Completion might be the more appropriate selection. CA Completions coming back from a PCI Express device normally indicate a device programming model violation, which may need to trigger Root containment and error recovery.

IMPLEMENTATION NOTE

Selecting the DPC Trigger Condition

Non-Fatal Errors are uncorrectable errors that indicate that a particular TLP was unreliable, and in general the associated Function should not continue its normal operation. Fatal errors are uncorrectable errors that indicate that a particular Link and its related hardware are unreliable, and in general the entire hierarchy below that Link should not continue normal operation. This distinction between Non-Fatal and Fatal errors together with the Root Port error containment capabilities can sometimes be used to select the appropriate DPC trigger condition. The following assumes that there is no peer-to-peer traffic between devices.

Some RCs implement a proprietary feature that will be referred to generically as "Function Level Containment" (FLC). This is not an architected feature of PCI Express. A Root Port that implements FLC is capable of containing the traffic associated with a specific Function when a Non-Fatal Error is detected in that traffic. Switch Downstream Ports below a Root Port with FLC should be configured to trigger DPC when the Downstream Port detects an unmasked uncorrectable error itself or when the Downstream Port receives an ERR_FATAL Message. Under this mode, the Switch Downstream Port passes ERR_NONFATAL Messages it receives Upstream without triggering DPC. This enables Root Port FLC to handle Non-Fatal Errors that render a specific Function unreliable and Switch Downstream Port DPC to handle errors that render a sub-tree of the hierarchy domain unreliable. The Downstream Port still needs to trigger DPC for all unmasked uncorrectable errors it detects, since an ERR_NONFATAL it generates will have its own

Requester ID, and the FLC hardware in the Root Port would not be able to determine which specific Function below the Switch Downstream Port was responsible for the Non-Fatal Error.

Switch Downstream Ports below a Root Port without FLC should be configured to trigger DPC when the Switch Downstream Port detects an unmasked uncorrectable error or when the Switch Downstream Port receives an ERR_NONFATAL or ERR_FATAL Message. This enables DPC to contain the error to the affected hierarchy below the Link and allow continued normal operation of the unaffected portion of the hierarchy domain.

IMPLEMENTATION NOTE

Software Polling the DPC RP Busy Bit

The DPC RP Busy bit is a means for hardware to indicate to software that the RP needs to remain in DPC containment while the RP does some internal cleanup and quiescing activities. While the details of these activities are implementation specific, the activities will typically complete within a few microseconds or less. However, under worst-case conditions such as those that might occur with certain internal errors in large systems, the busy period might extend substantially, possibly into multiple seconds. If software is unable to tolerate such lengthy delays within the current software context, software may need to rely on using timer interrupts to schedule polling under interrupt.

IMPLEMENTATION NOTE

Determination of DPC Control

DPC may be controlled in some configurations by platform firmware and in other configurations by the operating system. DPC functionality is strongly linked with the functionality in Advanced Error Reporting. To avoid conflicts over whether platform firmware or the operating system have control of DPC, it is recommended that platform firmware and operating systems always link the control of DPC to the control of Advanced Error Reporting.

6.2.10.1. DPC Interrupts

A DPC-capable Downstream Port must support the generation of DPC interrupts. DPC interrupts are enabled by the DPC Interrupt Enable bit in the DPC Control register. DPC interrupts are indicated by the DPC Interrupt Status bit in the DPC Status register.

If the Port is enabled for level-triggered interrupt signaling using INTx messages, the virtual INTx wire must be asserted whenever and as long as the following conditions are satisfied:

- The value of the Interrupt Disable bit in the Command register is 0b.
- The value of the DPC Interrupt Enable bit is 1b.
- The value of the DPC Interrupt Status bit is 1b.

Note that all other interrupt sources within the same Function will assert the same virtual INTx wire when requesting service.

If the Port is enabled for edge-triggered interrupt signaling using MSI or MSI-X, an interrupt message must be sent every time the logical AND of the following conditions transitions from FALSE to TRUE:

- The associated vector is unmasked (not applicable if MSI does not support PVM).
- The value of the DPC Interrupt Enable bit is 1b.
- The value of the DPC Interrupt Status bit is 1b.

The Port may optionally send an interrupt message if interrupt generation has been disabled, and the logical AND of the above conditions is TRUE when interrupt generation is subsequently enabled.

The interrupt message will use the vector indicated by the DPC Interrupt Message Number field in the DPC Capability register. This vector may be the same or may be different from the vectors used by other interrupt sources within this Function.

6.2.10.2. DPC ERR_COR Signaling

A DPC-capable Downstream Port must support ERR_COR signaling, independent of whether it supports Advanced Error Reporting (AER) or not. DPC ERR_COR signaling is enabled by the DPC ERR_COR Enable bit in the DPC Control register. DPC triggering is indicated by the DPC Trigger Status bit in the DPC Status register. DPC ERR_COR signaling is managed independently of DPC interrupts, and it is permitted to use both mechanisms concurrently.

If the DPC ERR_COR Enable bit is Set, and the Correctable Error Reporting Enable bit in the Device Control register is Set, the Port must send an ERR_COR Message each time the DPC Trigger Status bit transitions from Clear to Set. DPC ERR_COR signaling must not Set the Correctable Error Detected bit in the Device Status register, since this event is not handled as an error.

For a given DPC trigger event, if a Port is going to send both an ERR_COR Message and an MSI/MSI-X transaction, then the Port must send the ERR_COR Message prior to sending the MSI/MSI-X transaction. There is no corresponding requirement if the INTx mechanism is being used to signal DPC interrupts, since INTx Messages won't necessarily remain ordered with respect to ERR_COR Messages when passing through routing elements.

IMPLEMENTATION NOTE

Use of DPC ERR_COR Signaling

It is recommended that operating systems use DPC interrupts for signaling when DPC has been triggered. While DPC ERR_COR signaling indicates the same event, DPC ERR_COR signaling is primarily intended for use by platform firmware, when it needs to be notified in order to do its own logging of the event or provide "firmware first" services.

6.2.10.3. Root Port Programmed I/O (RP PIO) Error Controls

The RP PIO error control registers enable fine-grained control over what happens when Non-Posted Requests that are tracked by the Root Port encounter certain uncorrectable errors or Advisory Non-Fatal Errors. See Section 2.9.3 for a description of which Non-Posted Requests are tracked. A set of control and status bits exists for receiving Completion with Unsupported Request status (UR Cpl), receiving Completion with Completer Abort status (CA Cpl), and Completion Timeout (CTO) errors. Independent sets of these error bits exist for Configuration Requests, I/O Requests, and Memory Requests. This finer granularity enables more precise error handling for this subset of uncorrectable errors (UR Cpl, CA Cpl, and CTO). As a key example, UR Cpl errors with Memory Read Requests can be configured to trigger DPC for proper containment and error handling, while UR Cpl errors with Configuration Requests can be configured to return all 1's (without trigging DPC) for normal probing and enumeration.

A UR or CA error logged in AER is the result of the Root Port operating in the role of a Completer, and for a received Non-Posted Request, returning a Completion. In contrast, a UR Cpl or CA Cpl error logged as an RP PIO error is the result of the Root Port operating in the role of a Requester, and for an outstanding Non-Posted Request, receiving a Completion. CTO errors logged in both AER and RP PIO are the result of the Root Port operating in the role of a Requester, though the RP PIO error controls support per-space granularity. Depending upon the control register settings, CTO errors can be logged in AER registers, in RP PIO registers, or both. If software unmasks CTO errors in RP PIO, it is recommended that software mask CTO errors in AER in order to avoid unintended interactions.

The RP PIO Header Log, RP PIO ImpSpec Log, and RP PIO TLP Prefix Log registers are referred to collectively as the RP PIO log registers. The RP PIO Header Log must be implemented; the RP PIO ImpSpec Log and RP PIO TLP Prefix Log are optional. The RP PIO Log Size field indicates how many DWORDs are allocated for the RP PIO log registers, and from this the allocated size for the RP PIO TLP Prefix Log can be calculated. See Section 7.xx.2.

The RP PIO Status, Mask, and Severity registers behave similarly to the Uncorrectable Error Status, Mask, and Severity registers in AER. See Sections 7.10.2, 7.10.3, and 7.10.4. When an RP PIO error is detected while it is unmasked, the associated bit in the RP PIO Status register is Set, and the error is recorded in the RP PIO log registers (assuming that RP PIO error logging resources are available). When an RP PIO error is detected while it is masked, the associated bit is still Set in the RP PIO Status register, but the error does not trigger DPC and the error is not recorded in the RP PIO log registers.

Each unmasked RP PIO error is handled either as an uncorrectable error or an Advisory Non-Fatal Error, as determined by the value of the corresponding bit in the RP PIO Severity register. If the associated Severity bit is Set, the error is handled as uncorrectable, triggering DPC (assuming that DPC is enabled) and signaling this event with a DPC interrupt and/or ERR_COR (if enabled). If the associated Severity bit is Clear, the error is handled as an Advisory Non-Fatal Error (without triggering DPC) and signaled with ERR_COR (if enabled).

IMPLEMENTATION NOTE

Use of RP PIO Advisory Non-Fatal Errors

Each RP PIO error can be handled either as uncorrectable or an Advisory Non-Fatal error. Uncorrectable error handling usually logs the error, triggers DPC, and signals the event either with a DPC interrupt, an ERR_COR, or both. Advisory Non-Fatal Error handling usually logs the error and signals the event with ERR_COR.

RP PIO Advisory Non-Fatal Errors can be used by software in certain cases to handle RP PIO errors robustly without incurring the disruption caused if DPC is triggered in the RP. If an RP PIO Exception is not enabled for a given error, an all 1's value must be returned whenever the error occurs. If the error does not trigger DPC, software may be uncertain if the all 1's value returned by a given PIO read is the actual data value returned by the Completion versus indicating that an error occurred with that PIO read. If software enables Advisory Non-Fatal Error handling for that error, instances of that error will be logged, enabling software to distinguish the two cases.

The use of RP PIO Advisory Non-Fatal Errors is notably beneficial if DPC is triggered in a Switch Downstream Port, and that causes one or more Completion Timeouts in the RP as a side-effect, as described in Section 2.9.3. If the RP handles Completion Timeout errors as Advisory Non-Fatal, this avoids DPC being triggered in the RP, permitting continued operation with the other Switch Downstream Ports.

The RP PIO First Error Pointer, RP PIO Header Log, and RP PIO TLP Prefix Log behave similarly to the First Error Pointer, Header Log, and TLP Prefix Log in AER. The RP PIO First Error Pointer is defined to be valid when its value indicates a bit in the RP PIO Status register that is Set. When the RP PIO First Error Pointer is valid, the RP PIO log registers contain the information associated with the indicated error. The RP PIO ImpSpec Log, if implemented, contains implementation-specific information, e.g., the source of the Request TLP.

In contrast to AER, where the recording of CTO error information in the AER log registers is optional, RP PIO implementations must support recording RP PIO CTO error information in the RP PIO log registers.

The RP PIO SysError register provides a means to generate a System Error when an RP PIO error occurs. If an unmasked RP PIO error is detected while its associated bit in the RP PIO SysError register is Set, a System Error is generated.

The RP PIO Exception register provides a means to generate a synchronous processor exception[6] when an RP PIO error occurs with certain tracked Non-Posted Requests that are generated by a processor instruction. See Section 2.9.3. This exception must support all such tracked read Requests, and may optionally support Configuration write, I/O write, and AtomicOp Requests. If an RP PIO error with an exception-supported Non-Posted Request is detected while its associated bit in the RP PIO Exception register is Set, the processor instruction that generated the Non-Posted Request must take a synchronous exception. This is independent of whether the RP PIO error is masked or not.

[6] "Exception" is used as a generic term for a variety of mechanisms used by processors, including interrupts, traps, machine checks, instruction aborts, etc.

The details of a processor instruction taking a synchronous exception are processor-specific, but at a minimum, the mechanism must be able to interrupt the normal processor instruction flow either before completion of the instruction that generated the Non-Posted Request, or immediately following that instruction. The intent is that exception handling routines in system firmware, the operating system, or both, can examine the cause of the exception and take corrective action if necessary.

If an RP PIO error occurs with a processor-generated read or AtomicOp Request, and the RP PIO Exception register value does not cause an exception, a value of all 1's must be returned for the instruction that generated the Request.

IMPLEMENTATION NOTE

Synchronous Exception Implementation

The exact mechanism for implementing synchronous exceptions is processor and platform specific. One possible implementation is poisoning the data returned to the processor for a read or AtomicOp Request that encounters an error. While this approach is likely to work with those Requests, it might not work with Configuration and I/O write Requests since they return no data.

Another possible implementation is marking the response transaction for processor-generated Non-Posted Requests with some other type of indication of the Request having failed, e.g., a "hard fail" response. This approach is more likely to work with all processor-generated Non-Posted Requests.

IMPLEMENTATION NOTE

RP PIO Mask Bit Behavior and Rationale

For a given RP PIO error, the associated mask bit in the RP PIO Mask register affects its associated status bit setting, error logging, and error signaling in a manner that closely parallels the behavior of mask bits in AER.

SysError generation for a given RP PIO error is primarily controlled by the associated bit in the RP PIO SysError register, but is also contingent upon the associated RP PIO mask bit being Clear. This behavior was chosen for consistency with AER, and also since it is poor practice to generate a SysError without logging the reason.

Exception generation for a given RP PIO error is independent of the associated RP PIO mask bit value. Usage Models are envisioned where an RP PIO error needs to generate an Exception without logging an RP PIO error or triggering DPC.

If a Root Port has outstanding tracked Non-Posted Requests when DPC is triggered, the Root Port is required to synthesize Completions for those tracked Requests, although this may be a logical synthesis, and not the actual creation of Completion TLPs. See Section 2.9.3. These synthesized Completions shall have a UR or CA Completion Status, as determined by the value of the DPC Completion Control bit. The UR/CA Completion Status, combined with the associated Request's targeted Space (Configuration, I/O, or Memory), then determines which RP PIO error control bits are used to govern the remainder of the error handling for each outstanding tracked Request. For

example, a Root Port handling the case of a Configuration Read receiving a synthesized UR Completion would use the Cfg UR Cpl bit (bit 0) in the RP PIO SysError and RP PIO Exception registers to determine whether to generate a System Error, generate a processor instruction exception, or both.

Root Port error handling for tracked Non-Posted Requests with errors other than receiving UR and CA Completions is governed by a combination of AER and RP PIO error controls. Examples are CTO[7], Poisoned TLP Received, and Malformed TLP. For a given error managed by AER, the associated AER Mask and Severity bits partially or completely determine if the error must be handled as an uncorrectable error, handled as an Advisory Non-Fatal Error, or handled as a masked error.

- If the AER-managed error is to be handled as an uncorrectable error (see Section 6.2.2.2), DPC is triggered. As described earlier, Completions are synthesized for any outstanding tracked Non-Posted Requests, and for each Request, the RP PIO SysError and RP PIO Exception bits associated with the Request type and Completion Status apply.
- If the AER-managed error is to be handled as an Advisory Non-Fatal Error (see Section 6.2.3.2.4) or a masked error (see Section 6.2.3.2.2), DPC is not triggered, so the RP PIO SysError and RP PIO Exception bits do not apply.
- One notable example is the case of a Root Port with an outstanding PIO read receiving an associated Completion that is poisoned. If the AER Severity bit is Set, forcing the error to be handled as an uncorrectable error, the RP PIO Exception bit determines if all 1's is returned versus generating an exception. If the error is to be handled as an Advisory Non-Fatal Error as described in Section 6.2.3.2.4.3, the poisoned data is propagated internally by the Root Port, and from there is handled in a platform specific manner. If the error is to be handled as a masked error, the poisoned data is handled in a platform specific manner.

6.2.10.4. Software Triggering of DPC

If the DPC Software Triggering Supported bit in the DPC Capability register is Set, then software can trigger DPC by writing a 1b to the DPC Software Trigger bit in the DPC Control register, assuming that DPC is enabled and the Port isn't currently in DPC. This mechanism is envisioned to be useful for software and/or firmware development and testing. It also supports usage models where software or firmware examines RP PIO Exceptions or RP PIO Advisory Non-Fatal Errors, and decides to trigger DPC based upon the situation.

When this mechanism triggers DPC, the DPC Trigger Reason and DPC Trigger Reason Extension fields in the DPC Status register will indicate this as the reason.

If a Port is already in DPC when a 1b is written to the DPC Software Trigger bit, the Port remains in DPC, and the DPC Trigger Reason and DPC Trigger Reason Extension fields are not modified.

[7] CTO errors have status and mask bits in both AER and RP PIO, though RP PIO has independent sets of bits for each of the 3 spaces. Other errors in AER have no equivalent errors in RP PIO.

IMPLEMENTATION NOTE

Avoid Disable Link and Hot-Plug Surprise Use With DPC

It is recommended that software not Set the Link Disable bit in the Link Control register while DPC is enabled but not triggered. Setting the Link Disable bit will cause the Link to be directed to DL_Down, invoking some semantics similar to those in DPC, but lacking others. The subsequent arrival of Posted Requests will likely trigger DPC soon, anyway.

Similarly, it is recommended that a Port that supports DPC not Set the Hot-Plug Surprise bit in the Slot Capabilities register. Having this bit Set blocks the reporting of Surprise Down errors, preventing DPC from being triggered by this important error, greatly reducing the benefit of DPC.

6.2.10.5. DL_Active ERR_COR Signaling

Support for this feature is indicated by the DL_Active ERR_COR Signaling Supported bit in the DPC Capability register. The feature is enabled by the DL_ACTIVE ERR_COR Enable bit in the DPC Control register. The DL_ACTIVE state is indicated by the Data Link Layer Link Active bit in the Link Status register. DL_ACTIVE ERR_COR signaling is managed independently of Data Link Layer State Changed interrupts, and it is permitted to use both mechanisms concurrently.

If the DL_ACTIVE ERR_COR Enable bit is Set, and the Correctable Error Reporting Enable bit in the Device Control register is Set, the Port must send an ERR_COR Message each time the Link transitions into the DL_ACTIVE state. DL_ACTIVE ERR_COR signaling must not Set the Correctable Error Detected bit in the Device Status register, since this event is not handled as an error. In contrast to Data Link Layer State Changed interrupts, DL_Active ERR_COR signaling only indicates the Link enters the DL_Active state, not when the Link exits the DL_Active state.

For a given DL_ACTIVE event, if a Port is going to send both an ERR_COR Message and an MSI/MSI-X transaction, then the Port must send the ERR_COR Message prior to sending the MSI/MSI-X transaction. There is no corresponding requirement if the INTx mechanism is being used to signal DL_ACTIVE interrupts, since INTx Messages won't necessarily remain ordered with respect to ERR_COR Messages when passing through routing elements.

IMPLEMENTATION NOTE

Use of DL_ACTIVE ERR_COR Signaling

It is recommended that operating systems use Data Link Layer State Changed interrupts for signaling when DL_ACTIVE changes state. While DL_ACTIVE ERR_COR signaling indicates a subset of the same events, DL_ACTIVE ERR_COR signaling is primarily intended for use by platform firmware, when it needs to be notified in order to do Downstream Port configuration or provide "firmware first" services.

Modify Section 7.10 as shown:

7.10.2. Uncorrectable Error Status Register (Offset 04h)

…

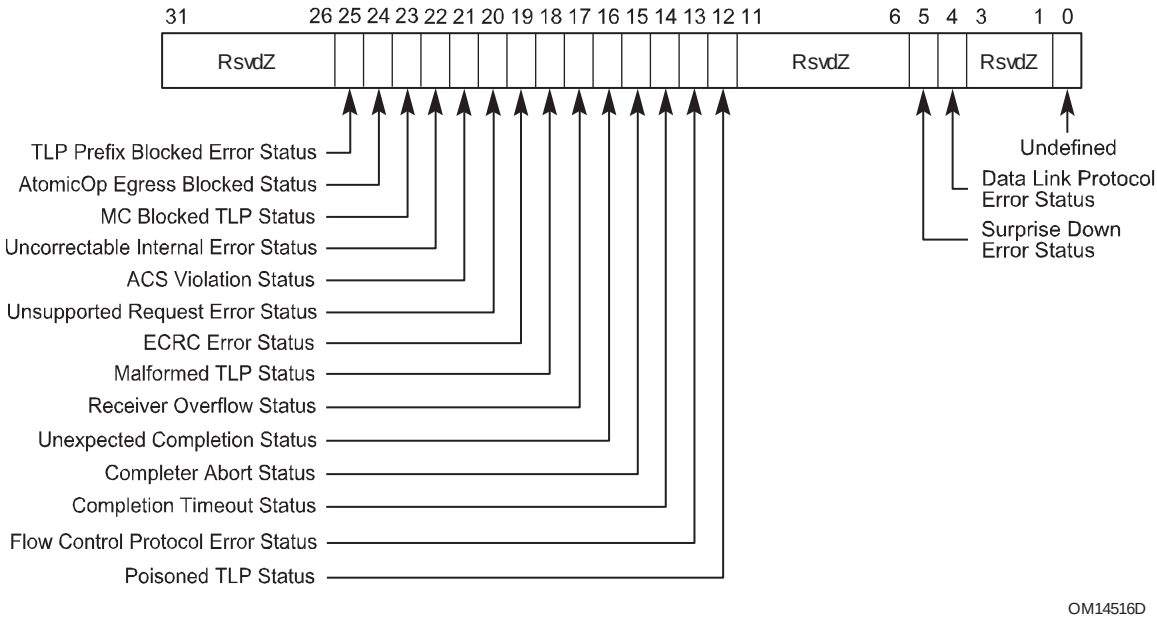

Figure 7-34: Uncorrectable Error Status Register

Note to Editor: modify the above figure to indicate the bit that is defined in the table below.

Table 7-31: Uncorrectable Error Status Register

Bit Location	Register Description	Attributes	Default
…	…	…	…
12	**Poisoned TLP Received Status**	RW1CS	0b
…	…	…	…
26	**Poisoned TLP Egress Blocked Status** (Optional)	RW1CS	0b

7.10.3. Uncorrectable Error Mask Register (Offset 08h)

…

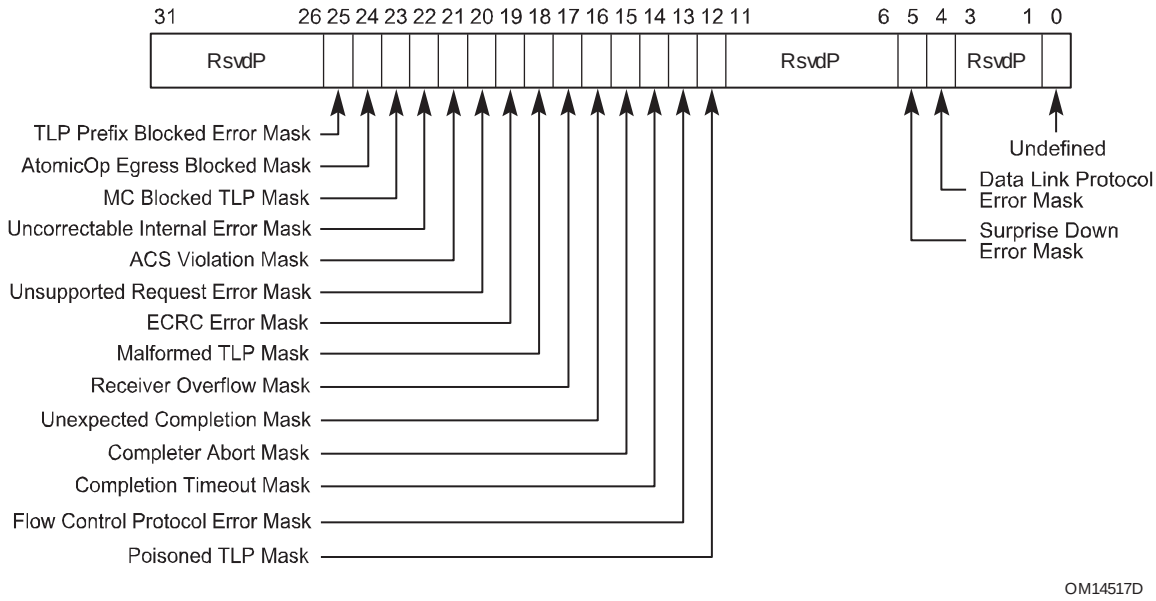

Figure 7-35: Uncorrectable Error Mask Register

Note to Editor: modify the above figure to indicate the bit that is defined in the table below.

Table 7-32: Uncorrectable Error Mask Register

Bit Location	Register Description	Attributes	Default
…	…	…	…
12	**Poisoned TLP Received Mask**	RWS	0b
…	…	…	…
26	**Poisoned TLP Egress Blocked Mask** (Optional)	RWS	1b

7.10.4. Uncorrectable Error Severity Register (Offset 0Ch)

…

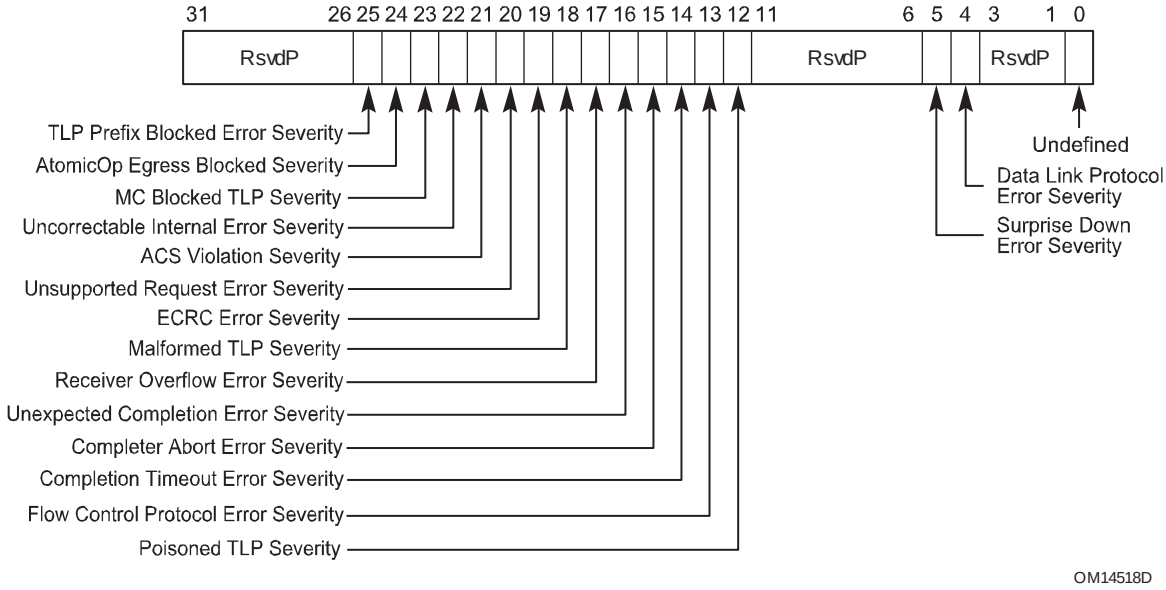

Figure 7-36: Uncorrectable Error Severity Register

Note to Editor: modify the above figure to indicate the bit that is defined in the table below.

Table 7-33: Uncorrectable Error Severity Register

Bit Location	Register Description	Attributes	Default
…	…	…	…
12	**Poisoned TLP Received Severity**	RWS	0b
…	…	…	…
26	**Poisoned TLP Egress Blocked Severity** (Optional)	RWS	0b

Add Section 7.xx:

7.xx. DPC Extended Capability

The Downstream Port Containment (DPC) Extended Capability is an optional normative capability that provides a mechanism for Downstream Ports to contain uncorrectable errors and enable software to recover from them. See Section 6.2.10. This capability may be implemented by a Root Port or a Switch Downstream Port. It is not applicable to any other Device/Port type.

If a Downstream Port implements the DPC Extended Capability, that Port must also be capable of reporting the DL_Active state, and indicate so by Setting the Data Link Layer Link Active Reporting Capable bit in the Link Capabilities register. See Section 7.8.6.

The various RP PIO registers must be implemented only by Root Ports that support RP Extensions for DPC, as indicated by the RP Extensions for DPC bit in the DPC Capability register.

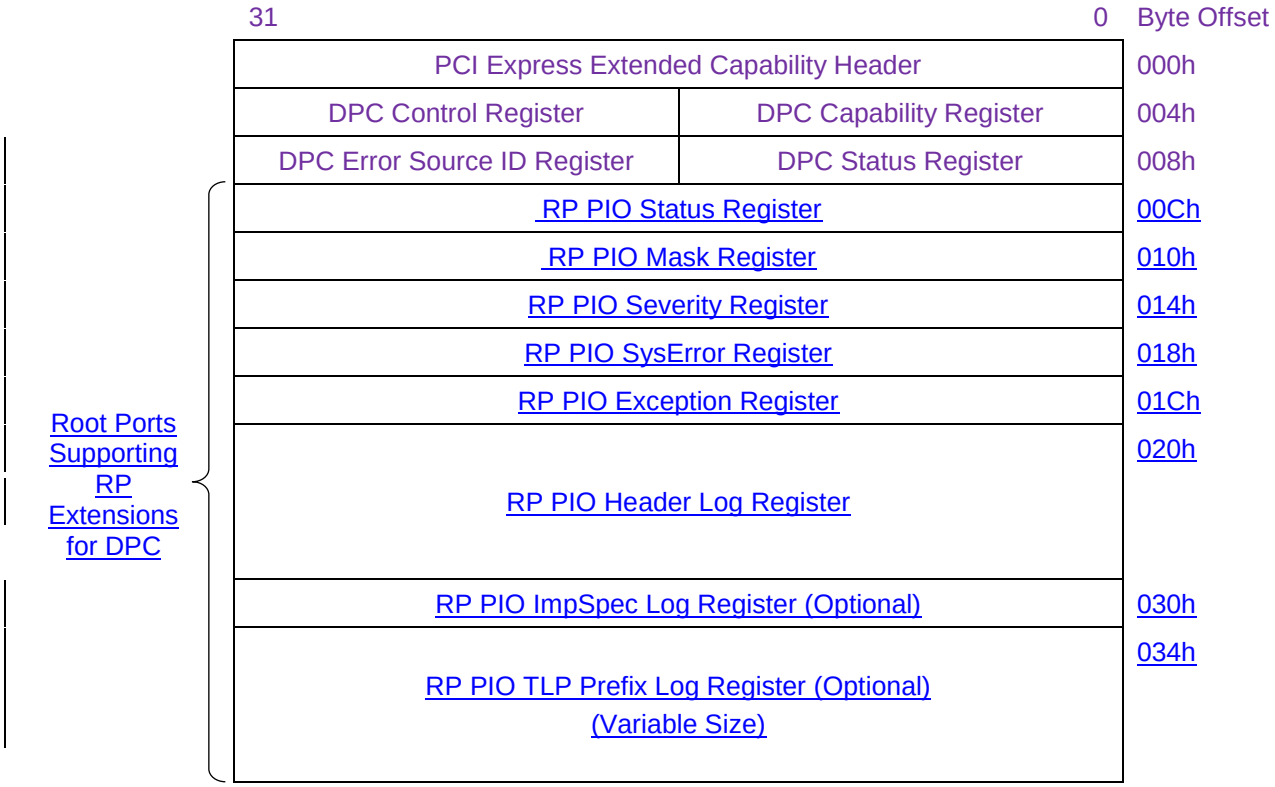

Figure 7-xx DPC Extended Capability

7.xx.1. DPC Extended Capability Header (Offset 00h)

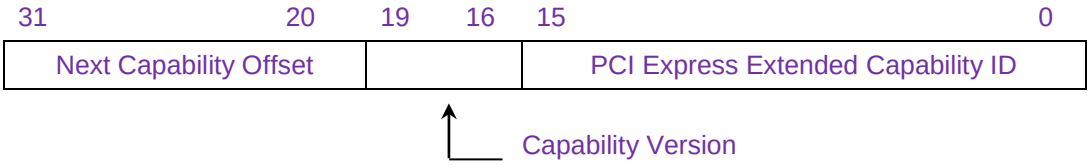

Figure 7-xx DPC Extended Capability Header

Table 7-xx DPC Extended Capability Header

Bit Location	Register Description	Attributes
15:0	**PCI Express Extended Capability ID** – This field is a PCI-SIG defined ID number that indicates the nature and format of the extended capability. PCI Express Extended Capability ID for the DPC Extended Capability is 001Dh.	RO
19:16	**Capability Version** – This field is a PCI-SIG defined version number that indicates the version of the capability structure present. Must be 1h for this version of the specification.	RO
31:20	**Next Capability Offset** – This field contains the offset to the next PCI Express Extended Capability structure or 000h if no other items exist in the linked list of capabilities.	RO

7.xx.2. DPC Capability Register (Offset 04h)

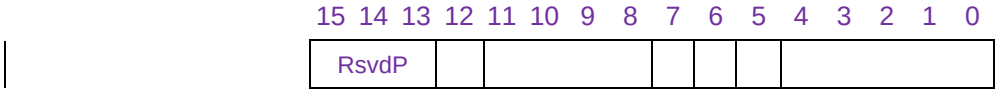

Figure 7-xx DPC Capability Register

Note to Editor: modify the above figure to indicate the fields that are defined in the table below.

Table 7-xx DPC Capability Register

Bit Location	Register Description	Attributes
4:0	**DPC Interrupt Message Number** – This field indicates which MSI/MSI-X vector is used for the interrupt message generated in association with the DPC Capability structure. For MSI, the value in this field indicates the offset between the base Message Data and the interrupt message that is generated. Hardware is required to update this field so that it is correct if the number of MSI Messages assigned to the Function changes when software writes to the Multiple Message Enable field in the MSI Message Control register. For MSI-X, the value in this field indicates which MSI-X Table entry is used to generate the interrupt message. The entry must be one of the first 32 entries even if the Function implements more than 32 entries. For a given MSI-X implementation, the entry must remain constant. If both MSI and MSI-X are implemented, they are permitted to use different vectors, though software is permitted to enable only one mechanism at a time. If MSI-X is enabled, the value in this field must indicate the vector for MSI-X. If MSI is enabled or neither is enabled, the value in this field must indicate the vector for MSI. If software enables both MSI and MSI-X at the same time, the value in this field is undefined.	RO
5	**RP Extensions for DPC** – If Set, this bit indicates that a Root Port supports a defined set of DPC Extensions that are specific to Root Ports. Switch Downstream Ports must not Set this bit.	RO
6	**Poisoned TLP Egress Blocking Supported** – If Set, this bit indicates that the Root Port or Switch Downstream Port supports the ability to block the transmission of a poisoned TLP from its Egress Port. Root Ports that support RP Extensions for DPC must Set this bit.	RO
7	**DPC Software Triggering Supported** – If Set, this bit indicates that a Root Port or Switch Downstream Port supports the ability for software to trigger DPC. Root Ports that support RP Extensions for DPC must Set this bit.	RO

11:8	**RP PIO Log Size** – This field indicates how many DWORDs are allocated for the RP PIO log registers, comprised by the RP PIO Header Log, the RP PIO ImpSpec Log, and RP PIO TLP Prefix Log. If the Root Port supports RP Extensions for DPC, the value of this field must be 4 or greater; otherwise, the value of this field must be 0. See Sections 7.xx.11, 7.xx.12, and 7.xx.13.	RO
12	**DL_Active ERR_COR Signaling Supported –** If Set, this bit indicates that the Root Port or Switch Downstream Port supports the ability to signal with ERR_COR when the Link transitions to the DL_Active state. Root Ports that support RP Extensions for DPC must Set this bit.	RO

7.xx.3. DPC Control Register (Offset 06h)

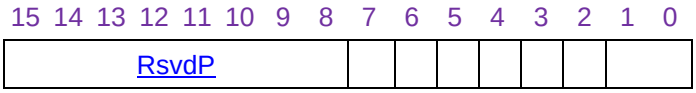

Figure 7-xx DPC Control Register

Note to Editor: modify the above figure to indicate the bits/fields that are defined in the table below.

Table 7-xx DPC Control Register

Bit Location	Register Description	Attributes
1:0	**DPC Trigger Enable** – This field enables DPC and controls the conditions that cause DPC to be triggered. Defined encodings are: 00b DPC is disabled 01b DPC is enabled and is triggered when the Downstream Port detects an unmasked uncorrectable error or when the Downstream Port receives an ERR_FATAL Message. 10b DPC is enabled and is triggered when the Downstream Port detects an unmasked uncorrectable error or when the Downstream Port receives an ERR_NONFATAL or ERR_FATAL Message. 11b Reserved Default value of this field is 00b.	RW
2	**DPC Completion Control** – This bit controls the Completion Status for Completions formed during DPC. See Section 2.9.3. Defined encodings are: 0b Completer Abort (CA) Completion Status 1b Unsupported Request (UR) Completion Status Default value of this bit is 0b.	RW
3	**DPC Interrupt Enable** – When Set, this bit enables the generation of an interrupt to indicate that DPC has been triggered. See Section 6.2.10.1. Default value of this bit is 0b.	RW

4	**DPC ERR_COR Enable** – When Set, this bit enables the sending of an ERR_COR Message to indicate that DPC has been triggered. See Section 6.2.10.2. Default value of this bit is 0b.	RW
5	**Poisoned TLP Egress Blocking Enable** – This bit must be RW if the Poisoned TLP Egress Blocking Supported bit is Set; otherwise, it is permitted to be hardwired to 0b. Software must not Set this bit unless the Poisoned TLP Egress Blocking Supported bit is Set. When Set, this bit enables the associated Egress Port to block the transmission of poisoned TLPs. See Section 2.7.2.2. Default value of this bit is 0b.	RW / RO
6	**DPC Software Trigger** – This bit must be RW if the DPC Software Triggering Supported bit is Set; otherwise, it is permitted to be hardwired to 0b. If DPC is enabled and the DPC Trigger Status bit is Clear, when software writes 1b to this bit, DPC is triggered. Otherwise, software writing a 1b to this bit has no effect. It is permitted to write 1b to this bit while simultaneously writing updated values to other fields in this register, notably the DPC Trigger Enable field. For this case, the DPC Software Trigger semantics are based on the updated value of the DPC Trigger Enable field. This bit always returns 0b when read.	RW / RO
7	**DL_Active ERR_COR Enable –** This bit must be RW if the DL_Active ERR_COR Signaling Supported bit is Set; otherwise, it is permitted to be hardwired to 0b. Software must not Set this bit unless the DL_Active ERR_COR Signaling Supported bit is Set. When Set, this bit enables the associated Downstream Port to signal with ERR_COR when the Link transitions to the DL_Active state. See Section 6.2.10.5. Default value of this bit is 0b.	RW / RO

7.xx.4. DPC Status Register (Offset 08h)

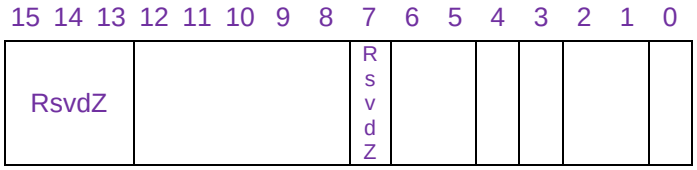

Figure 7-xx DPC Status Register

Note to Editor: modify the above figure to indicate the bits/fields that are defined in the table below.

Table 7-xx DPC Status Register

Bit Location	Register Description	Attributes
0	**DPC Trigger Status** – When Set, this bit indicates that DPC has been triggered. DPC is event triggered. While this bit is Set, hardware must direct the LTSSM to the Disabled State. This bit must be cleared before the LTSSM can be released from the Disabled State. See Section 6.2.10 for requirements on how long software must leave the Downstream Port in DPC. Once these requirements are met, software is permitted to clear this bit regardless of the state of other status bits associated with the triggering event. After clearing this bit, software must honor timing requirements defined in Section 6.6.1 with respect to the first Configuration Read following a Conventional Reset. Default value of this bit is 0b.	RW1CS
2:1	**DPC Trigger Reason** – This field indicates why DPC has been triggered. Defined encodings are: 00b DPC was triggered due to an unmasked uncorrectable error. 01b DPC was triggered due to receiving an ERR_NONFATAL. 10b DPC was triggered due to receiving an ERR_FATAL. 11b ~~Reserved~~ DPC was triggered due to a reason that is indicated by the DPC Trigger Reason Extension field. This field is valid only when the DPC Trigger Status bit is Set; otherwise the value of this field is undefined.	ROS
3	**DPC Interrupt Status** – This bit is Set if DPC is triggered while the DPC Interrupt Enable bit is Set. This may cause the generation of an interrupt. See Section 6.2.10.1. Default value of this bit is 0b.	RW1CS

4	**DPC RP Busy –** When the DPC Trigger Status bit is Set and this bit is Set, the Root Port is busy with internal activity that must complete before software is permitted to Clear the DPC Trigger Status bit. If software Clears the DPC Trigger Status bit while this bit is Set, the behavior is undefined. This field is valid only when the DPC Trigger Status bit is Set; otherwise the value of this field is undefined. This bit is applicable only for Root Ports that support RP Extensions for DPC, and is Reserved for Switch Downstream Ports. Default value of this bit is undefined.	RO / RsvdZ
6:5	**DPC Trigger Reason Extension** – This field serves as an extension to the DPC Trigger Reason field. When that field is valid and has a value of 11b, this field indicates why DPC has been triggered. Defined encodings are: 00b DPC was triggered due to an RP PIO error. 01b DPC was triggered due to the DPC Software Trigger bit. 10b Reserved 11b Reserved This field is valid only when the DPC Trigger Status bit is Set and the value of the DPC Trigger Reason field is 11b; otherwise the value of this field is undefined.	ROS
12:8	**RP PIO First Error Pointer** – The value of this field identifies a bit position in the RP PIO Status register, and this field is considered valid when that bit is Set. When this field is valid, and software writes a 1b to the indicated RP PIO Status bit (thus clearing it), this field must revert to its default value. This field is applicable only for Root Ports that support RP Extensions for DPC, and otherwise is Reserved. If this field is not Reserved, its default value is 11111b, indicating a permanently Reserved RP PIO Status bit, thus guaranteeing that this field is not considered valid.	ROS / RsvdZ

7.xx.5. DPC Error Source ID Register (Offset 0Ah)

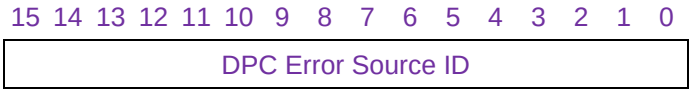

Figure 7-xx DPC Error Source ID Register

Table 7-xx DPC Error Source ID Register

Bit Location	Register Description	Attributes
15:0	**DPC Error Source ID** – When the DPC Trigger Reason field indicates that DPC was triggered due to the reception of an ERR_NONFATAL or ERR_FATAL, this register contains the Requester ID of the received Message. Otherwise, the value of this register is undefined.	ROS

7.xx.6. RP PIO Status Register (Offset 0Ch)

This register is present only in Root Ports that support RP Extensions for DPC. See Section 6.2.10.3.

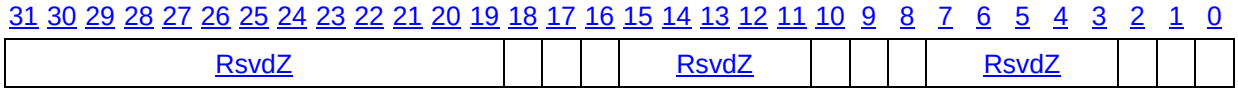

Figure 7-xx RP PIO Status Register

Note to Editor: modify the above figure to indicate the bits that are defined in the table below.

Table 7-xx RP PIO Status Register

Bit Location	Register Description	Attributes	Default
0	**Cfg UR Cpl –** Configuration Request received UR Completion	RW1CS	0b
1	**Cfg CA Cpl –** Configuration Request received CA Completion	RW1CS	0b
2	**Cfg CTO –** Configuration Request Completion Timeout	RW1CS	0b
8	**I/O UR Cpl –** I/O Request received UR Completion	RW1CS	0b
9	**I/O CA Cpl –** I/O Request received CA Completion	RW1CS	0b
10	**I/O CTO –** I/O Request Completion Timeout	RW1CS	0b
16	**Mem UR Cpl –** Memory Request received UR Completion	RW1CS	0b
17	**Mem CA Cpl –** Memory Request received CA Completion	RW1CS	0b
18	**Mem CTO –** Memory Request Completion Timeout	RW1CS	0b
31	Permanently Reserved, since the default RP PIO First Error Pointer field value points to it.	RsvdZ	0b

7.xx.7. RP PIO Mask Register (Offset 10h)

This register is present only in Root Ports that support RP Extensions for DPC. See Section 6.2.10.3.

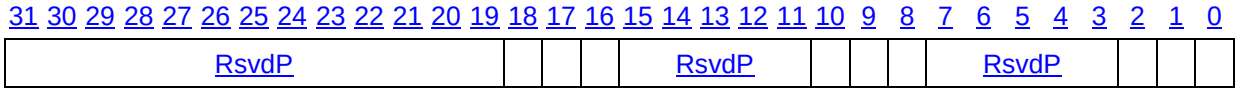

Figure 7-xx RP PIO Mask Register

Note to Editor: modify the above figure to indicate the bits that are defined in the table below.

Table 7-xx RP PIO Mask Register

Bit Location	Register Description	Attributes	Default
0	**Cfg UR Cpl –** Configuration Request received UR Completion	RWS	1b
1	**Cfg CA Cpl –** Configuration Request received CA Completion	RWS	1b
2	**Cfg CTO –** Configuration Request Completion Timeout	RWS	1b
8	**I/O UR Cpl –** I/O Request received UR Completion	RWS	1b
9	**I/O CA Cpl –** I/O Request received CA Completion	RWS	1b
10	**I/O CTO –** I/O Request Completion Timeout	RWS	1b
16	**Mem UR Cpl –** Memory Request received UR Completion	RWS	1b
17	**Mem CA Cpl –** Memory Request received CA Completion	RWS	1b
18	**Mem CTO –** Memory Request Completion Timeout	RWS	1b

7.xx.8. RP PIO Severity Register (Offset 14h)

This register is present only in Root Ports that support RP Extensions for DPC. See Section 6.2.10.3.

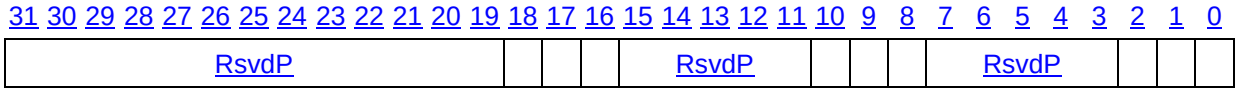

Figure 7-xx RP PIO Severity Register

Note to Editor: modify the above figure to indicate the bits that are defined in the table below.

Table 7-xx RP PIO Severity Register

Bit Location	Register Description	Attributes	Default
0	**Cfg UR Cpl –** Configuration Request received UR Completion	RWS	0b
1	**Cfg CA Cpl –** Configuration Request received CA Completion	RWS	0b
2	**Cfg CTO –** Configuration Request Completion Timeout	RWS	0b
8	**I/O UR Cpl –** I/O Request received UR Completion	RWS	0b
9	**I/O CA Cpl –** I/O Request received CA Completion	RWS	0b
10	**I/O CTO –** I/O Request Completion Timeout	RWS	0b
16	**Mem UR Cpl –** Memory Request received UR Completion	RWS	0b
17	**Mem CA Cpl –** Memory Request received CA Completion	RWS	0b
18	**Mem CTO –** Memory Request Completion Timeout	RWS	0b

7.xx.9. RP PIO SysError Register (Offset 18h)

This register is present only in Root Ports that support RP Extensions for DPC. See Section 6.2.10.3.

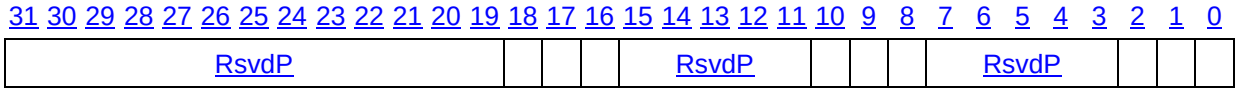

Figure 7-xx RP PIO SysError Register

Note to Editor: modify the above figure to indicate the bits that are defined in the table below.

Table 7-xx RP PIO SysError Register

Bit Location	Register Description	Attributes	Default
0	**Cfg UR Cpl –** Configuration Request received UR Completion	RWS	0b
1	**Cfg CA Cpl –** Configuration Request received CA Completion	RWS	0b
2	**Cfg CTO –** Configuration Request Completion Timeout	RWS	0b
8	**I/O UR Cpl –** I/O Request received UR Completion	RWS	0b
9	**I/O CA Cpl –** I/O Request received CA Completion	RWS	0b
10	**I/O CTO –** I/O Request Completion Timeout	RWS	0b
16	**Mem UR Cpl –** Memory Request received UR Completion	RWS	0b
17	**Mem CA Cpl –** Memory Request received CA Completion	RWS	0b
18	**Mem CTO –** Memory Request Completion Timeout	RWS	0b

7.xx.10. RP PIO Exception Register (Offset 1Ch)

This register is present only in Root Ports that support RP Extensions for DPC. See Section 6.2.10.3.

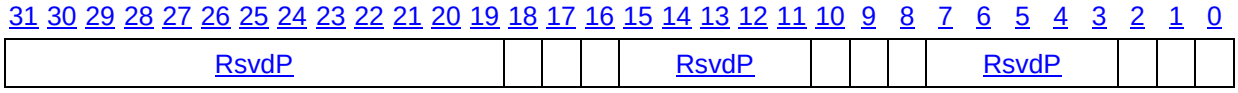

Figure 7-xx RP PIO Exception Register

Note to Editor: modify the above figure to indicate the bits that are defined in the table below.

Table 7-xx RP PIO Exception Register

Bit Location	Register Description	Attributes	Default
0	**Cfg UR Cpl –** Configuration Request received UR Completion	RWS	0b
1	**Cfg CA Cpl –** Configuration Request received CA Completion	RWS	0b
2	**Cfg CTO –** Configuration Request Completion Timeout	RWS	0b
8	**I/O UR Cpl –** I/O Request received UR Completion	RWS	0b
9	**I/O CA Cpl –** I/O Request received CA Completion	RWS	0b
10	**I/O CTO –** I/O Request Completion Timeout	RWS	0b
16	**Mem UR Cpl –** Memory Request received UR Completion	RWS	0b
17	**Mem CA Cpl –** Memory Request received CA Completion	RWS	0b
18	**Mem CTO –** Memory Request Completion Timeout	RWS	0b

7.xx.11. RP PIO Header Log Register (Offset 20h)

This register is implemented only in Root Ports that support RP Extensions for DPC. The RP PIO Header Log register contains the header from the TLP associated with a recorded RP PIO error. Refer to Section 6.2.10.3 for further details. This register is 16 bytes and is formatted identically to the Header Log register in AER. See Section 7.10.8.

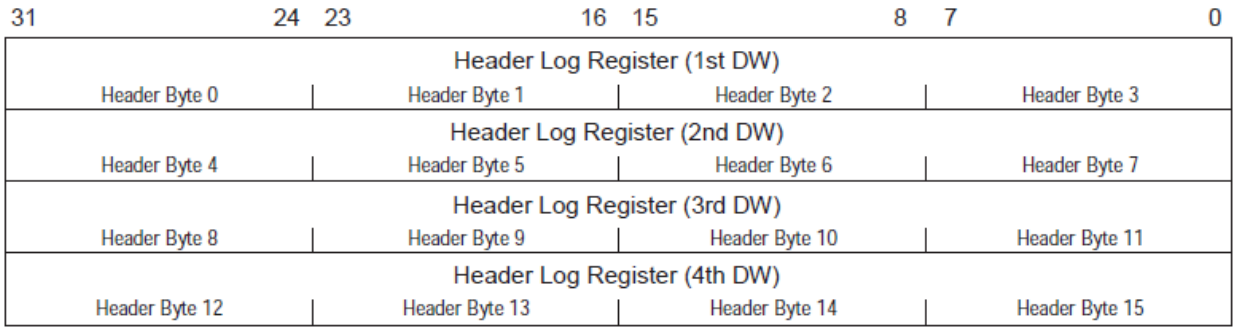

Figure 7-xx: RP PIO Header Log Register

Note to Editor: modify the above figure to indicate the RP PIO Header Log register.

Table 7-xx: RP PIO Header Log Register

Bit Location	Register Description	Attributes	Default
127:0	Header of TLP associated with error	ROS	0

7.xx.12. RP PIO ImpSpec Log Register (Offset 30h)

This register is permitted to be implemented only in Root Ports that support RP Extensions for DPC. The RP PIO ImpSpec Log register, if implemented, contains implementation-specific information associated with the recorded error, e.g., indicating the source of the Request TLP. Space is allocated for this register if the value of the RP PIO Log Size field is 5 or greater. If space is allocated for the register, but the register is not implemented, the bits must be hardwired to 0b.

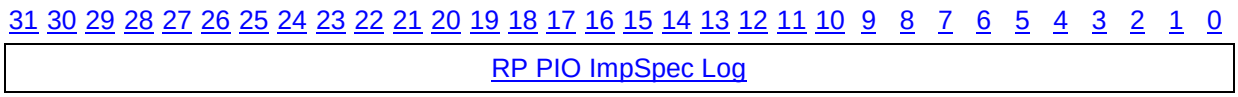

Figure 7-xx: RP PIO ImpSpec Log Register

Table 7-xx: RP PIO ImpSpec Log Register

Bit Location	Register Description	Attributes	Default
31:0	**RP PIO ImpSpec Log**	ROS	0

7.xx.13. RP PIO TLP Prefix Log Register (Offset 34h)

This register is permitted to be implemented only in Root Ports that support RP Extensions for DPC. The RP PIO TLP Prefix Log register contains any End-End TLP Prefixes from the TLP corresponding to a recorded RP PIO error. Refer to Section 6.2.10.3 for further details.

If the Root Port supports tracking Non-Posted Requests that contain End-End TLP Prefixes, this register must be implemented, and must be of sufficient size to record the maximum number of End-End TLP Prefixes for any tracked Request. See Section 2.9.3. The allocated size in DWORDs of the RP PIO TLP Prefix Log register is the RP PIO Log Size minus 5 if the RP PIO Log Size is 9 or less, or 4 if the RP PIO Log Size is greater than 9. The implemented size of the TLP Prefix Log must be less than or equal to the Root Port's Max End-End TLP Prefixes field value. For the case where the Root Port never transmits Non-Posted Requests containing End-End TLP Prefixes, the allocated and implemented size of the TLP Prefix Log is permitted to be 0. Any DWORDs allocated but not implemented must be hardwired to zero.

This register is formatted identically to the TLP Prefix Log register in AER, although this register's allocated size is variable, whereas the register in AER is always 4 DWORDs. See Section 7.10.12. The First TLP Prefix Log register contains the first End-End TLP Prefix from the TLP, the Second TLP Prefix Log register contains the second End-End TLP Prefix, and so forth. If the TLP contains fewer TLP Prefixes than this register accommodates, any remaining TLP Prefix Log registers must contain zero.

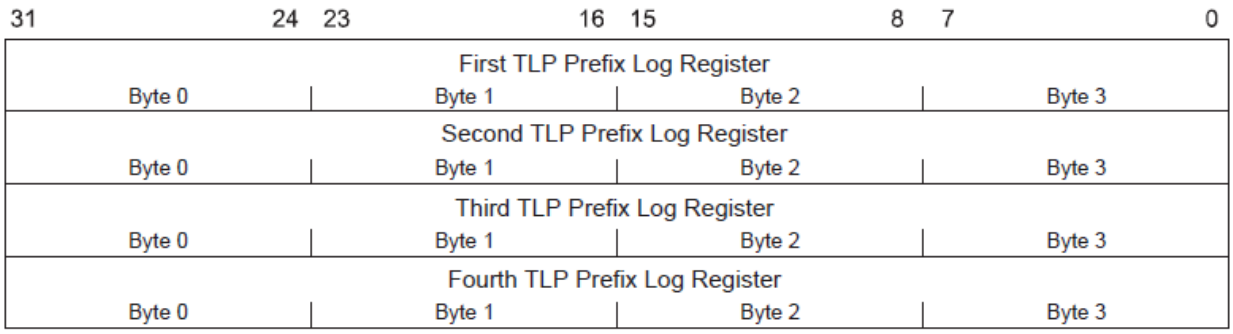

Figure 7-xx: RP PIO TLP Prefix Log Register

Table 7-xx: RP PIO TLP Prefix Log Register

Bit Location	Register Description	Attributes	Default
127:0	**RP PIO TLP Prefix Log**	ROS	0