PCI-SIG ENGINEERING CHANGE NOTICE

TITLE:	Internal Error Reporting
DATE:	April 24, 2008
AFFECTED DOCUMENT:	PCI Express Base Specification, version 2.0
SPONSOR:	Integrated Device Technology, Hewlett-Packard, Sun, Intel, PLX Technology, NextIO

Part I

1. Summary of the Functional Changes

PCI Express (PCIe) defines error signaling and logging mechanisms for errors that occur on a PCIe interface and for errors that occur on behalf of transactions initiated on PCIe. It does not define error signaling and logging mechanisms for errors that occur within a component or are unrelated to a particular PCIe transaction.

A vendor specific device driver is associated with Endpoints that may be used to detect and report component errors detected within an Endpoint (e.g., via an interrupt). PCIe Switches are considered part of the platform infrastructure, and as a result are directly managed by the operating system. Thus, these devices typically have no vendor specific software that can be used to detect and report component errors.

PCIe is used in many applications, such as high-end severs and storage, that require the ability to detect and contain errors. As a result, PCIe switch vendors have developed proprietary and incompatible mechanisms to report internal component errors. The result is that in most systems today, these mechanisms are never used.

This ECN defines optional error signaling and logging mechanisms for all components except PCIe to PCI/PCI-X Bridges (i.e., Switches, Root Complexes, and Endpoints) to report internal errors that are associated with a PCI Express interface. Errors that occur within components but are not associated with PCI Express remain outside the scope of the specification.

In order to better facilitate error containment and recovery, this ECN also extends the error logging mechanisms provided by the First Error and Error Log registers defined in the Advanced Error Reporting enhanced capability to allow multiple error headers to be recorded.

2. Benefits as a Result of the Changes

1. An architected industry standard approach for signaling and reporting internal component errors associated with a PCI Express interface. This enables components and OSes/Hypervisors to implement common and interoperable error handling services.

2. Improved error containment and recovery by allowing multiple error headers to be recorded.

3. Even in implementations that record only a single error header, the ability to determine whether multiple errors of the same type have occurred. Currently Advanced Error Reporting defines only a single error status bit per error type and a first error header log. Thus, there is no way in software to distinguish the occurrence of a single error from multiple errors of the same type by examining Advanced Error Reporting registers.

3. Assessment of the Impact

All of the mechanisms defined in this ECN are optional and disabled by default. Therefore, components and software (e.g., OSes and Hypervisors) only take advantage of these capabilities if needed. Some increase in hardware and software complexity is required to take advantage of new capabilities.

- Components that implement the optional reporting of Corrected Internal Errors must implement the status and mask register bits associated with this capability and enhance AER to support this new error type.
- Components that implement the optional reporting of Uncorrectable Internal Errors must implement the status, mask, and severity register bits associated with this capability and enhance AER to support this new error type.
- Components that implement the optional ability to record multiple error headers must implement an error header buffer, logging capabilities as outlined in this ECN, as well as corresponding capability and control bits.
- Components that implement the optional reporting of header log overflows must implement the status and mask register bits associated with this capability and enhance AER to support this new error type.

4. Analysis of the Hardware Implications

No impact to hardware interoperability.

5. Analysis of the Software Implications

There is no impact to current software. All registers associated with capabilities outlined in this ECN default to a state that preserves functional compatibility with legacy operation. AER related software must be modified to take advantage of new capabilities.

Part II

Detailed Description of the change

Change Terms and Acronyms Section as follows:

Terms and Acronyms

Internal Error	An error associated with a PCI Express interface that occurs within a component and which may not be attributable to a packet or event on the PCI Express interface itself or on behalf of transactions initiated on PCI Express.

Update Section 6.2.1 as follows:

6.2.1. Scope

This section explains the error signaling and logging requirements for PCI Express components. This includes errors which occur on the PCI Express interface itself, ~~and~~ those errors which occur on behalf of transactions initiated on PCI Express, and errors which occur within a component and are related to the PCI Express interface. This section does not focus on errors which occur within the component ~~that are unrelated to a particular PCI Express transaction~~ that are unrelated to a PCI Express interface. This type of error signaling is better handled through proprietary methods employing device-specific interrupts.

PCI Express defines two error reporting paradigms: the baseline capability and the Advanced Error Reporting Capability. The baseline error reporting capabilities are required of all PCI Express devices and define the minimum error reporting requirements. The Advanced Error Reporting Capability is defined for more robust error reporting and is implemented with a specific PCI Express Capability structure (refer to Chapter 7 for a definition of this optional capability). This section explicitly calls out all error handling differences between the baseline and the Advanced Error Reporting Capability.

All PCI Express devices support existing, non-PCI Express-aware, software for error handling by mapping PCI Express errors to existing PCI reporting mechanisms, in addition to the PCI Express-specific mechanisms.

Update Section 6.2.2 as follows:

6.2.2. Error Classification

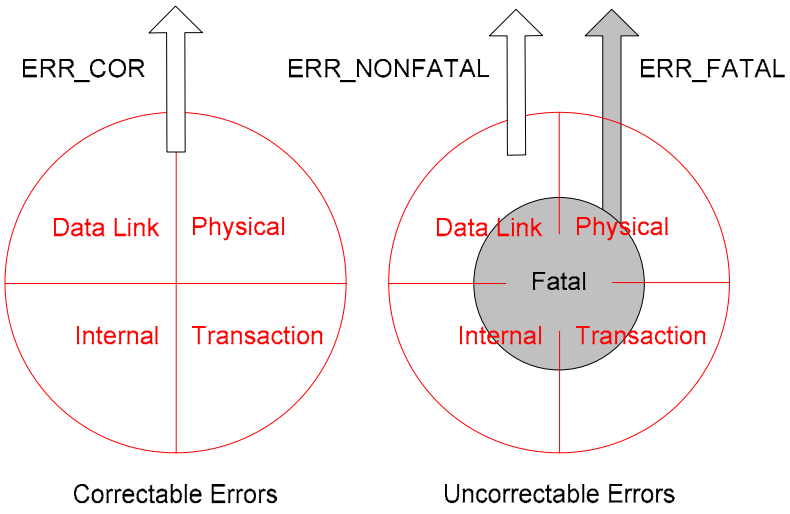

Figure 6-1: Error Classification

Update Section 6.2.3.2.3 as follows:

6.2.3.2.3. Error Pollution

Error pollution can occur if error conditions for a given transaction are not isolated to the most significant occurrence. For example, assume the Physical Layer detects a Receiver Error. This error is detected at the Physical Layer and an error is reported to the Root Complex. To avoid having this error propagate and cause subsequent errors at upper layers (for example, a TLP error at the Data Link Layer), making it more difficult to determine the root cause of the error, subsequent errors which occur for the same packet will not be reported by the Data Link or Transaction layers. Similarly, when the Data Link Layer detects an error, subsequent errors which occur for the same packet will not be reported by the Transaction Layer. This behavior applies only to errors that are associated with a particular packet – other errors are reported for each occurrence.

Corrected Internal Errors are errors whose effect has been masked or worked around by a component; refer to Section 6.2.9 for details. Therefore, Corrected Internal Errors do not contribute to error pollution and should be reported when detected.

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
- Flow Control Protocol Error
- ECRC Check Failed
- Malformed TLP
- Unsupported Request (UR), Completer Abort (CA), or Unexpected Completion[1]
- Poisoned TLP Received

The Completion Timeout error is not in the above precedence list, since it is not detected by processing a received TLP.

Here's an example of the rationale behind the precedence list. If an ECRC Check fails for a given TLP, the entire contents of the TLP including its header is potentially corrupt, so it makes little sense to report errors like Malformed TLP or Unsupported Request detected with the TLP.

Update Section 6.2.4 as follows:

6.2.4. Error Logging

Section 6.2.6 lists all the errors governed by this specification and for each error, the logging requirements are specified. Device Functions that do not support the Advanced Error Reporting Capability log only the Device Status register bits indicating that an error has been detected. Note that some errors are also reported using the reporting mechanisms in the PCI compatible (Type 00h and 01h) configuration registers. Section 7.5 describes how these register bits are affected by the different types of error conditions described in this section.

For device Functions supporting the Advanced Error Reporting Capability, each of the errors in Table 6-2, Table 6-3, and Table 6-4 -corresponds to a particular bit in the Uncorrectable Error Status register or Correctable Error Status register. These registers are used by software to determine more precisely which error and what severity occurred. For specific Transaction Layer errors and Uncorrectable Internal Errors the associated TLP

---

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

header is recorded. ~~in the Header Log register if the error is the first uncorrectable error detected (corresponding to the setting of the First Error Pointer register).~~

In a multi-Function device, PCI Express errors that are not related to any specific Function within the device, are logged in the corresponding status and logging registers of all Functions in that device.

The following PCI Express errors are not Function-specific:

- ❑ All Physical Layer errors
- ❑ All Data Link Layer errors
- ❑ These Transaction Layer errors:
  - ECRC Fail
  - UR, when caused by no Function claiming a TLP
  - Receiver Overflow
  - Flow Control Protocol Error
  - Malformed TLP
  - Unexpected Completion, when caused by no Function claiming a Completion
- ❑ Some Internal Errors
  - The determination of whether an Internal Error is Function-specific or not is implementation specific.

On the detection of one of these errors, a multi-Function device should generate at most one error reporting Message of a given severity, where the Message must report the Requester ID of a Function of the device that is enabled to report that specific type of error. If no Function is enabled to send a reporting Message, the device does not send a reporting Message. If all reporting-enabled Functions have the same severity level set for the error, only one error Message is sent. If all reporting-enabled Functions do not have the same severity level set for the error, one error Message for each severity level is sent. Software is responsible for scanning all Functions in a multi-Function device when it detects one of those errors.

Update Section 6.2.4.2 as follows:

6.2.4.2. Multiple Error Handling (Advanced Error Reporting Capability)

For the Advanced Error Reporting Capability, the Uncorrectable Error Status register and Correctable Error Status register accumulate the collection of errors which ~~occur on~~ correspond to ~~-~~that particular PCI Express interface. The bits remain set until explicitly cleared by software or reset. Since multiple bits might be set in the Uncorrectable Error Status register, the First Error Pointer register points to the unmasked uncorrectable error that occurred first. The First Error Pointer register is valid when the corresponding bit of

the Uncorrectable Error Status register is set. The First Error Pointer value is not meaningful when the corresponding bit of the Uncorrectable Error Status register is not set, or is an unimplemented or undefined bit.

~~For errors which require header logging, the Header Log register loaded according to the same rules as the First Error Pointer register (such that the Header Log register will correspond to the error indicated in the First Error Pointer register, when the First Error Pointer register is valid).~~

The Advanced Error Reporting Capability provides the ability to record headers for errors that require header logging. An implementation may support the recording of multiple headers, but at a minimum must support the ability of recording at least one. The ability to record multiple headers is indicated by the state of the Multiple Header Recording Capable bit and enabled by the Multiple Header Recording Enable bit of the Advanced Error Capabilities and Control register. When multiple header recording is supported and enabled, headers are recorded in the order in which the corresponding errors are detected.

When the First Error Pointer register is valid, the Header Log register contains the recorded header for the corresponding error. Writing a 1b to the bit of the Uncorrectable Error Status register to which a valid First Error Pointer register corresponds causes the space occupied by that recoded header to be released, the next recorded header to be reported in the Header Log register, and the First Error Pointer register to be updated to point to the corresponding Uncorrectable Error Status register bit. If no further recorded header is of the same error type as the last, then the bit in the Uncorrectable Error Status register to which the 1b was written is cleared; otherwise, it remains Set. When the last recorded header is reached and a 1b is written to the corresponding Uncorrectable Error Status register bit, then the First Error Pointer register becomes invalid (i.e., corresponds to an Uncorrectable Error Status register bit that is not Set, undefined, or unimplemented) and the value of the Header Log register is undefined.

~~Since the First Error Pointer and Header Log registers are only loaded for the first occurrence of an uncorrectable error,~~ Since an implementation only has the ability to record a finite number of headers, it is important that software services ~~these~~ the First Error Pointer and Header Log registers in a timely manner, to limit the risk of missing this information for subsequent errors. If an error requiring header logging is detected but the header cannot be recorded, then a Header Log Overflow error is reported by the Function. This occurs when an error that requires header logging is detected and the number of recorded headers supported by an implementation has been reached, or the Multiple Header Recording bit is not Set and the First Error Pointer register is valid.

Update Section 6.2.4.3 as follows:

6.2.4.3. Advisory Non-Fatal Error Logging

Section 6.2.3.2.4 describes Advisory Non-Fatal Error cases, under which an agent with AER detecting an uncorrectable error of non-fatal severity signals the error (if enabled) using ERR_COR instead of ERR_NONFATAL. For the same cases, an agent without AER sends no Error Message. The remaining discussion in this section is in the context of agents that do implement AER.

For Advisory Non-Fatal Error cases, since an uncorrectable error is signaled using the correctable Error Message, control/status/mask bits involving both uncorrectable and correctable errors apply. Figure 6.2 shows a flowchart of the sequence. Following are some of the unique aspects for logging Advisory Non-Fatal Errors.

First, the uncorrectable error needs to be of severity non-fatal, as determined by the associated bit in the Uncorrectable Error Severity register. If the severity is fatal, the error does not qualify as an Advisory Non-Fatal Error, and will be signaled (if enabled) with ERR_FATAL.

Next, the specific error case needs to be one of the Advisory Non-Fatal Error cases documented in Section 6.2.3.2.4. If not, the error does not qualify as an Advisory Non-Fatal Error, and will be signaled (if enabled) with an uncorrectable Error Message.

Next, the Advisory Non-Fatal Error Status bit is set in the Correctable Error Status register to indicate the occurrence of the advisory error, and the Advisory Non-Fatal Error Mask bit in the Correctable Error Mask register is checked to determine whether to proceed further with logging and signaling.

If the Advisory Non-Fatal Error Mask bit is clear, logging proceeds by setting the "corresponding" bit in the Uncorrectable Error Status register, based upon the specific uncorrectable error that's being reported as an advisory error and recording the header. If the "corresponding" uncorrectable error bit in the Uncorrectable Error Mask register is clear, the First Error Pointer and Header Log registers are updated to log the error, assuming they are not still "occupied" by a previous unserviced error.

Finally, an ERR_COR Message is sent if the Correctable Error Reporting Enable bit is set in the Device Control register.

Update Section 6.2.5 as follows:

6.2.5. Sequence of Device Error Signaling and Logging Operations

Figure 6-2 shows the sequence of operations related to signaling and logging of errors detected by a device.

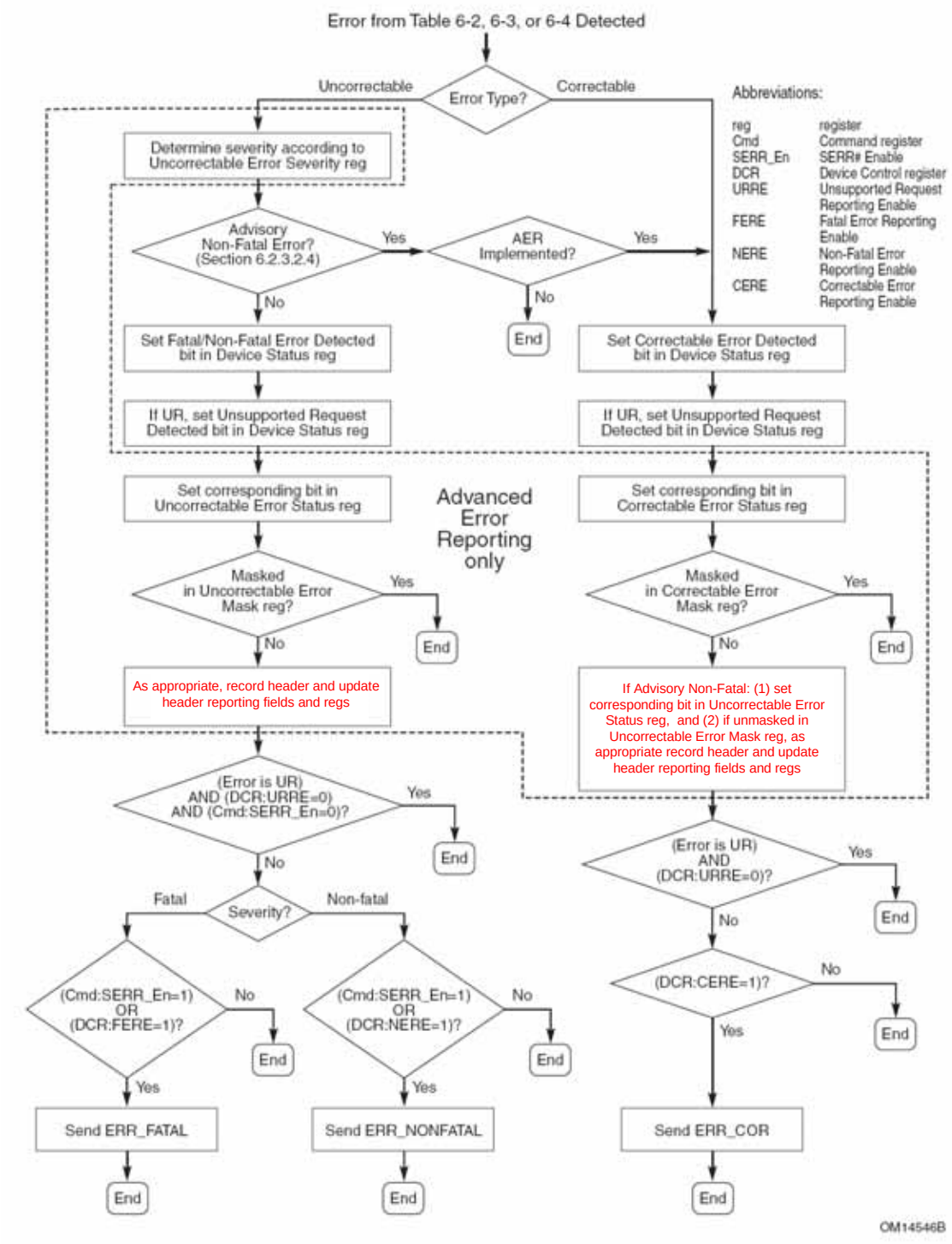

Figure 6-1: Flowchart Showing Sequence of Device Error Signaling and Logging Operations

Update Section 6.2.7 as follows and add new table before Table6-2:

6.2.7. Error Listing and Rules

Tables 6.x through 6.4 list all of the PCI Express errors that are defined by this specification. Each error is listed with a short-hand name, how the error is detected in hardware, the default severity of the error, and the expected action taken by the agent which detects the error. These actions form the rules for PCI Express error reporting and logging.

The Default Severity column specifies the default severity for the error without any software reprogramming. For device Functions supporting the Advanced Error Reporting Capability, the uncorrectable errors are programmable to Fatal or Non-fatal with the Error Severity register. Device Functions without Advanced Error Reporting Capability use the default associations and are not reprogrammable.

Table 6.x: General PCI Express Error List

Error Name	Error Type (Default Severity)	Detecting Agent Action[11]	References
Corrected Internal Error	Correctable (masked by default)	*Component:* Send ERR_COR to Root Complex.	*Section 6.2.9*
Uncorrectable Internal Error	Uncorrectable (Fatal and masked by default)	*Component:* Send ERR_FATAL to Root Complex. Optionally log the header of the first TLP associated with the error.	*Section 6.2.9*
Header Log Overflow	Correctable (masked by default)	*Component:* Send ERR_COR to Root Complex.	*Section 6.2.4.2*

[11] For these tables, detecting agent action is given as if all enable bits are set to "enable" and, for Advanced Error Handling, mask bits are disabled and severity bits are set to their default values. Actions must be modified according to the actual settings of these bits.

Insert new section following Section 6.2.8.1 as follows:

6.2.9. Internal Errors

An Internal Error is an error associated with a PCI Express interface that occurs within a component and which may not be attributable to a packet or event on the PCI Express interface itself or on behalf of transactions initiated on PCI Express. The determination of what is considered an Internal Error is implementation specific and is outside the scope of this specification.

Internal Errors may be classified as Corrected Internal Errors or Uncorrectable Internal Errors. A Corrected Internal Error is an error that occurs within a component that has been masked or worked around by hardware without any loss of information or improper operation. An example of a possible Corrected Internal Error is an internal packet buffer memory error corrected by an Error Correcting Code (ECC). An Uncorrectable Internal Error is an error that occurs within a component that results in improper operation of the component. An example of a possible Uncorrectable Internal Error is a memory error that cannot be corrected by an ECC. The only method of recovering from an Uncorrectable Internal Error is reset or hardware replacement.

Reporting of Internal Errors is optional. If Internal Errors are reported, then AER must be implemented.

Header logging is optional for Uncorrectable Internal Errors. When a header is logged, the header is that of the first TLP that was lost or corrupted by the Uncorrectable Internal Error. When header logging is not implemented or a header is not available, a header of all ones is recorded.

Internal Errors that can be associated with a specific PCI Express interface are reported by the Function(s) associated with that Port. Internal Errors detected within Switches that cannot be associated with a specific PCI Express interface are reported by the Upstream Port. Reporting of Internal Errors that cannot be associated with a specific PCI Express interface in all other multi-Port components (e.g., Root Complexes) is outside the scope of this specification.

Update Section 6.12.4 as follows:

6.12.4. ACS Violation Error Handling

ACS Violations may occur due to either hardware or software defects/failures. To assist in fault isolation and root cause analysis, it is recommended that AER be implemented in ACS components. The AER header recording and Header Log register ~~can log~~ may be used to determine the header of the offending Request. The ACS Violation Status, Mask, and

Severity bits provide positive identification of the error and increased control over error logging and signaling.

Update Section 7.10.2 as follows:

7.10.2 Uncorrectable Error Status Register (Offset 04h)

The Uncorrectable Error Status register indicates error detection status of individual errors on a PCI Express device Function. An individual error status bit that is Set indicates that a particular error was detected; software may clear an error status by writing a 1b to the respective bit. Refer to Section 6.2 for further details. Register bits not implemented by the Function are hardwired to 0b. Figure 7-32 details the allocation of register fields of the Uncorrectable Error Status register; Table 7-29 provides the respective bit definitions.

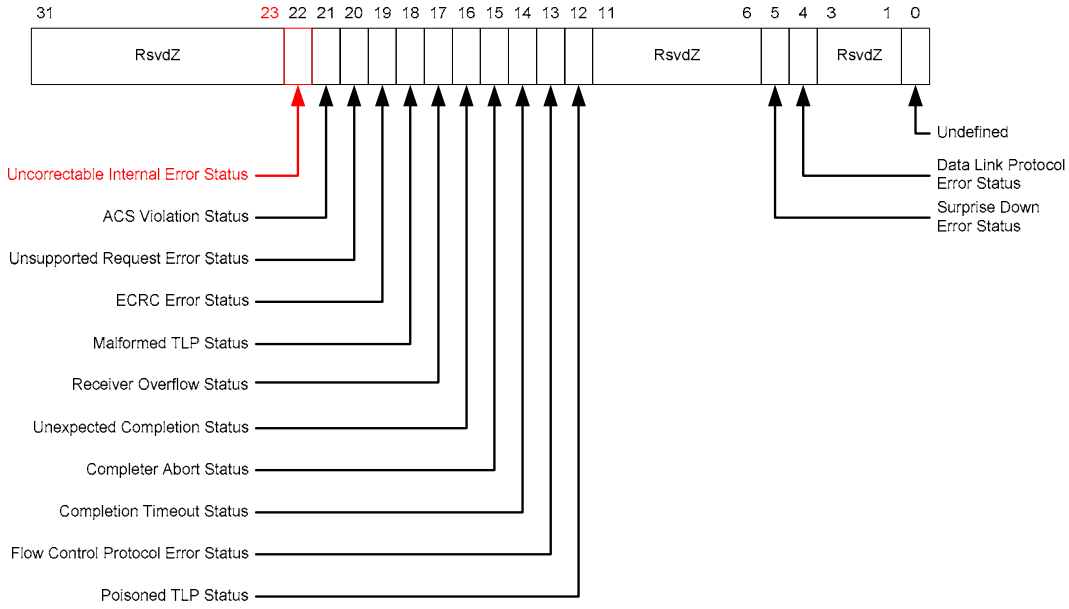

Figure 7-32: Uncorrectable Error Status Register

Table 7-29: Uncorrectable Error Status Register

Bit Location	Register Description	Attributes	Default
0	**Undefined** – The value read from this bit is undefined. In previous versions of this specification, this bit was used to indicate a Link Training Error. System software must ignore the value read from this bit. System software is permitted to write any value to this bit.	Undefined	Undefined

Bit Location	Register Description	Attributes	Default
4	**Data Link Protocol Error Status**	RW1CS	0b
5	**Surprise Down Error Status** (Optional)	RW1CS	0b
12	**Poisoned TLP Status**	RW1CS	0b
13	**Flow Control Protocol Error Status** (Optional)	RW1CS	0b
14	**Completion Timeout Status**[16]	RW1CS	0b
15	**Completer Abort Status** (Optional)	RW1CS	0b
16	**Unexpected Completion Status**	RW1CS	0b
17	**Receiver Overflow Status** (Optional)	RW1CS	0b
18	**Malformed TLP Status**	RW1CS	0b
19	**ECRC Error Status** (Optional)	RW1CS	0b
20	**Unsupported Request Error Status**	RW1CS	0b
21	**ACS Violation Status**	RW1CS	0b
22	**Uncorrectable Internal Error Status** (Optional)	RW1CS	0b

Update Section 7.10.3 as follows:

7.10.3 Uncorrectable Error Mask Register (Offset 08h)

The Uncorrectable Error Mask register controls reporting of individual errors by the device Function to the PCI Express Root Complex via a PCI Express Error Message. A masked error (respective bit Set in the mask register) is not ~~logged in the~~ recorded or reported in the Header Log register, does not update the First Error Pointer, and is not reported to the PCI Express Root Complex by this Function. Refer to Section 6.2 for further details. There is a mask bit per error bit of the Uncorrectable Error Status register. Register fields for bits not implemented by the Function are hardwired to 0b. Figure 7-33 details the allocation of register fields of the Uncorrectable Error Mask register; Table 7-30 provides the respective bit definitions.

[16] For Switch Ports, required if the Switch Port issues Non-Posted Requests on its own behalf (vs. only forwarding such Requests generated by other devices). If the Switch Port does not issue such Requests, then the Completion Timeout mechanism is not applicable and this bit must be hardwired to 0b.

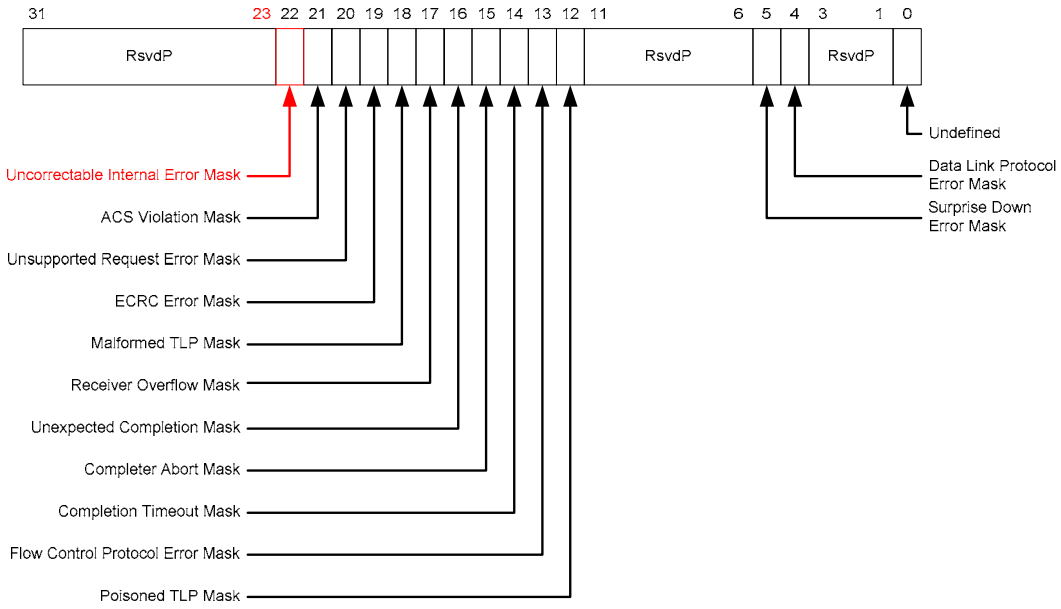

Figure 7-33: Uncorrectable Error Mask Register

Table 7-30: Uncorrectable Error Mask Register

Bit Location	Register Description	Attributes	Default
0	**Undefined** – The value read from this bit is undefined. In previous versions of this specification, this bit was used to mask a Link Training Error. System software must ignore the value read from this bit. System software must only write a value of 1b to this bit.	Undefined	Undefined
4	**Data Link Protocol Error Mask**	RWS	0b
5	**Surprise Down Error Mask** (Optional)	RWS	0b
12	**Poisoned TLP Mask**	RWS	0b
13	**Flow Control Protocol Error Mask** (Optional)	RWS	0b
14	**Completion Timeout Mask**[17]	RWS	0b
15	**Completer Abort Mask** (Optional)	RWS	0b
16	**Unexpected Completion Mask**	RWS	0b
17	**Receiver Overflow Mask** (Optional)	RWS	0b
18	**Malformed TLP Mask**	RWS	0b
19	**ECRC Error Mask** (Optional)	RWS	0b
20	**Unsupported Request Error Mask**	RWS	0b

[17] For Switch Ports, required if the Switch Port issues Non-Posted Requests on its own behalf (vs. only forwarding such Requests generated by other devices). If the Switch Port does not issue such Requests, then the Completion Timeout mechanism is not applicable and this bit must be hardwired to 0b.

Bit Location	Register Description	Attributes	Default
21	**ACS Violation Mask**	RWS	0b
22	**Uncorrectable Internal Error Mask** (Optional)	RWS	1b

Update Section 7.10.4 as follows:

7.10.4 Uncorrectable Error Severity Register (Offset 0Ch)

The Uncorrectable Error Severity register controls whether an individual error is reported as a Non-fatal or Fatal error. An error is reported as fatal when the corresponding error bit in the severity register is Set. If the bit is Clear, the corresponding error is considered non-fatal. Refer to Section 6.2 for further details. Register fields for bits not implemented by the Function are hardwired to the specified default value. Figure 7-34 details the allocation of register fields of the Uncorrectable Error Severity register; Table 7-31 provides the respective bit definitions.

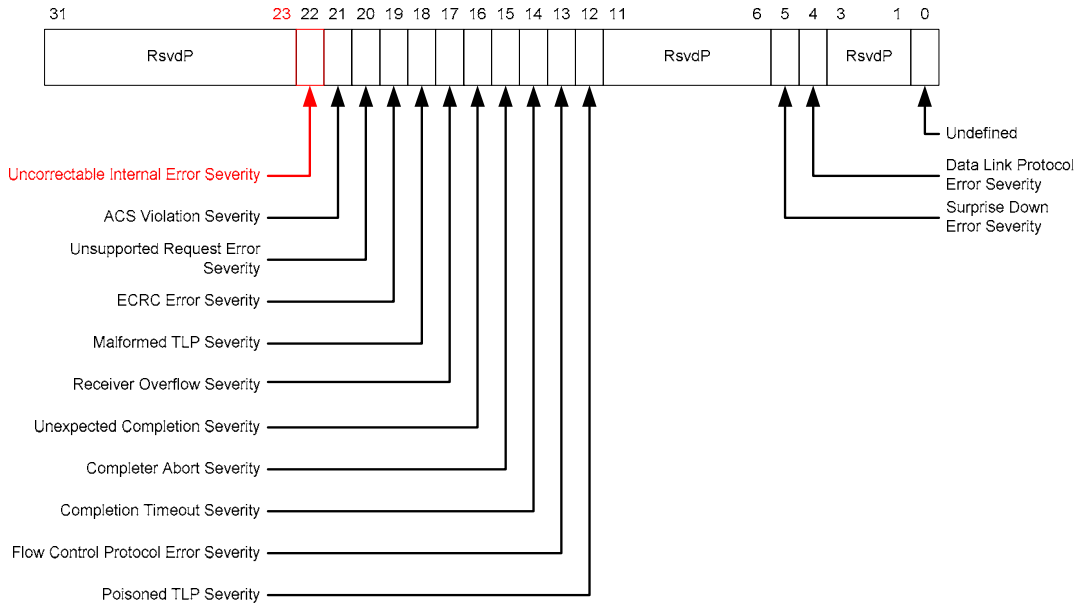

Figure 7-34: Uncorrectable Error Severity Register

Table 7-31: Uncorrectable Error Severity Register

Bit Location	Register Description	Attributes	Default
0	**Undefined** – The value read from this bit is undefined. In previous versions of this specification, this bit was used to Set the severity of a Link Training Error. System software must ignore the value read from this bit. System software is	Undefined	Undefined

	permitted to write any value to this bit.		
4	**Data Link Protocol Error Severity**	RWS	1b
5	**Surprise Down Error Severity** (Optional)	RWS	1b
12	**Poisoned TLP Severity**	RWS	0b
13	**Flow Control Protocol Error Severity** (Optional)	RWS	1b
14	**Completion Timeout Error Severity**[18]	RWS	0b
15	**Completer Abort Error Severity** (Optional)	RWS	0b
16	**Unexpected Completion Error Severity**	RWS	0b
17	**Receiver Overflow Error Severity** (Optional)	RWS	1b
18	**Malformed TLP Severity**	RWS	1b
19	**ECRC Error Severity** (Optional)	RWS	0b
20	**Unsupported Request Error Severity**	RWS	0b
21	**ACS Violation Severity**	RWS	0b
22	**Uncorrectable Internal Error Severity** (Optional)	RWS	1b

Update Section 7.10.5 as follows:

7.10.5 Correctable Error Status Register (Offset 10h)

The Correctable Error Status register reports error status of individual correctable error sources on a PCI Express device Function. When an individual error status bit is Set, it indicates that a particular error occurred; software may clear an error status by writing a 1b to the respective bit. Refer to Section 6.2 for further details. Register bits not implemented by the Function are hardwired to 0b. Figure 7-35 details the allocation of register fields of the Correctable Error Status register; Table 7-32 provides the respective bit definitions.

[18] For Switch Ports, required if the Switch Port issues Non-Posted Requests on its own behalf (vs. only forwarding such Requests generated by other devices). If the Switch Port does not issue such Requests, then the Completion Timeout mechanism is not applicable and this bit must be hardwired to 0b.

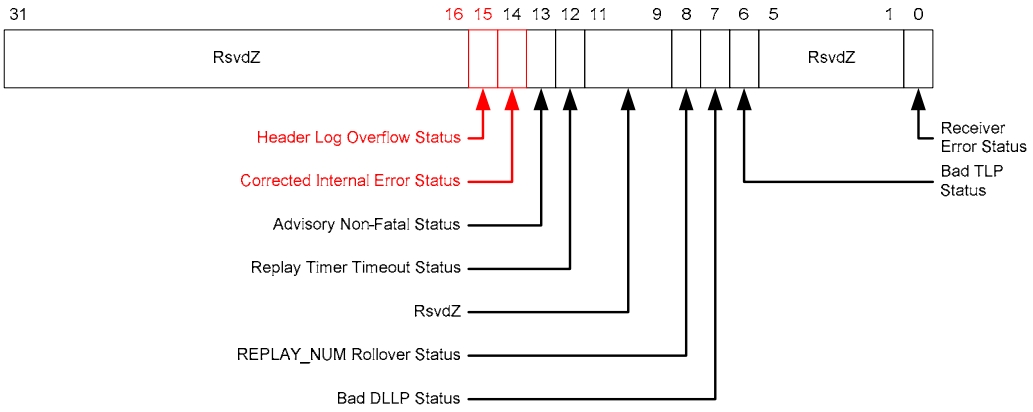

Figure 7-35: Correctable Error Status Register

Table 7-32: Correctable Error Status Register

Bit Location	Register Description	Attributes	Default
0	**Receiver Error Status**[19]	RW1CS	0b
6	**Bad TLP Status**	RW1CS	0b
7	**Bad DLLP Status**	RW1CS	0b
8	**REPLAY_NUM Rollover Status**	RW1CS	0b
12	**Replay Timer Timeout Status**	RW1CS	0b
13	**Advisory Non-Fatal Error Status**	RW1CS	0b
14	**Corrected Internal Error Status** (Optional)	RW1CS	0b
15	**Header Log Overflow Status** (Optional)	RW1CS	0b

Update Section 7.10.6 as follows:

7.10.6 Correctable Error Mask Register (Offset 14h)

The Correctable Error Mask register controls reporting of individual correctable errors by this Function to the PCI Express Root Complex via a PCI Express Error Message. A masked error (respective bit Set in the mask register) is not reported to the PCI Express Root Complex by this Function. Refer to Section6.2 for further details. There is a mask bit per error bit in the Correctable Error Status register. Register fields for bits not implemented by the Function are hardwired to 0b. Figure 7-36 details the allocation of register fields of the Correctable Error Mask register; Table 7-33 provides the respective bit definitions.

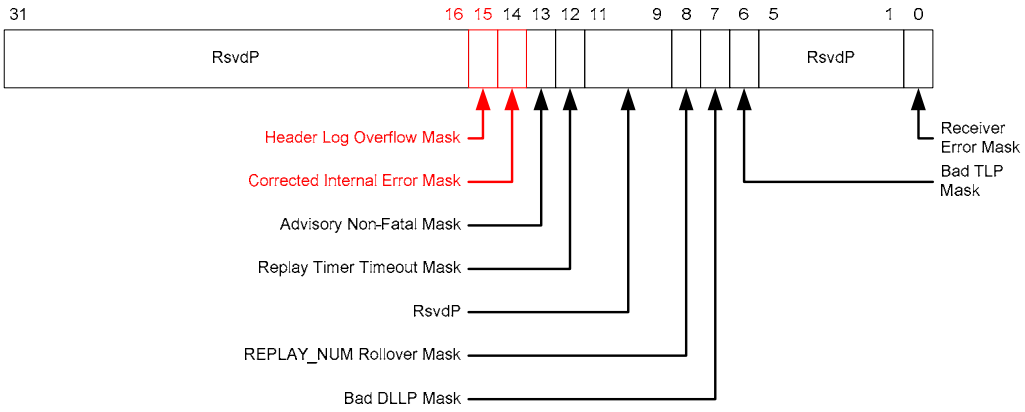

Figure 7-36: Correctable Error Mask Register

Table 7-33: Correctable Error Mask Register

Bit Location	Register Description	Attributes	Default
0	**Receiver Error Mask**[20]	RWS	0b
6	**Bad TLP Mask**	RWS	0b
7	**Bad DLLP Mask**	RWS	0b
8	**REPLAY_NUM Rollover Mask**	RWS	0b
12	**Replay Timer Timeout Mask**	RWS	0b
13	**Advisory Non-Fatal Error Mask** – This bit is Set by default to enable compatibility with software that does not comprehend Role-Based Error Reporting.	RWS	1b
14	**Corrected Internal Error Mask** (Optional)	RWS	1b
15	**Header Log Overflow Mask** (Optional)	RWS	1b

Update Section 7.10.7 as follows:

7.10.7 Advanced Error Capabilities and Control Register (Offset 18h)

Figure 7-37 details allocation of register fields in the Advanced Error Capabilities and Control register; Table 7-34 provides the respective bit definitions. Handling of multiple errors is discussed in Section 6.2.4.2.

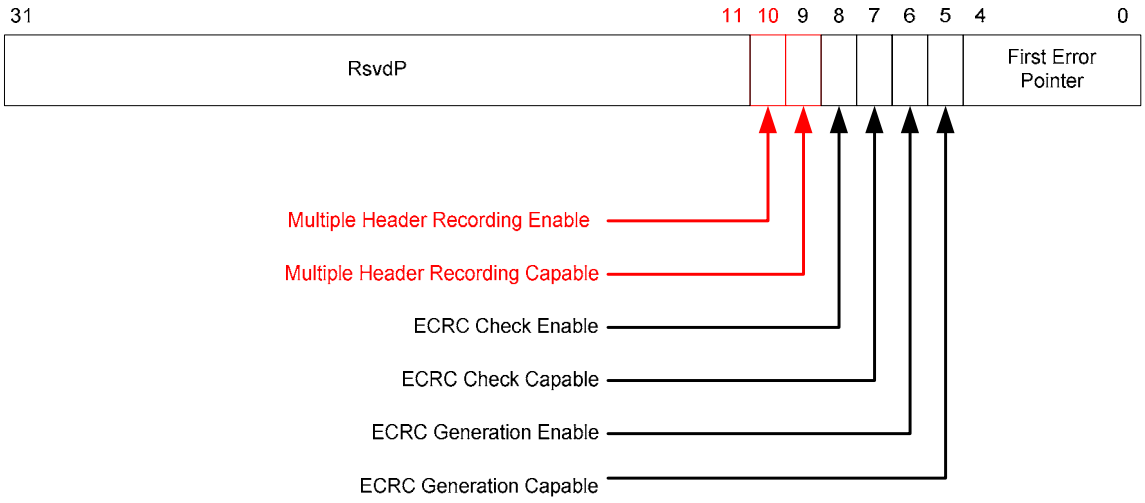

Figure 7-37: Advanced Error Capabilities and Control Register

Table 7-34: Advanced Error Capabilities and Control Register

Bit Location	Register Description	Attributes
4:0	**First Error Pointer** – The First Error Pointer is a field that identifies the bit position of the first error reported in the Uncorrectable Error Status register. Refer to Section 6.2 for further details.	ROS
5	**ECRC Generation Capable** – If Set, this bit indicates that the Function is capable of generating ECRC (see Section 2.7).	RO
6	**ECRC Generation Enable** – When Set, ECRC generation is enabled (see Section2.7). Functions that do not implement the associated mechanism are permitted to hardwire this bit to 0b. Default value of this bit is 0b.	RWS
7	**ECRC Check Capable** – If Set, this bit indicates that the Function is capable of checking ECRC (see Section 2.7). Functions that do not implement the associated mechanism are permitted to hardwire this bit to 0b.	RO
8	**ECRC Check Enable** – When Set, ECRC checking is enabled (see Section 2.7). Default value of this bit is 0b.	RWS
9	**Multiple Header Recording Capable** – If Set, this bit indicates that the Function is capable of recording more than one error header. Refer to Section 6.2 for further details.	RO

Bit Location	Register Description	Attributes
10	**Multiple Header Recording Enable** – When Set, this bit enables the function to record more than one error header. Functions that do not implement the associated mechanism are permitted to hardwire this bit to 0b. Default value of this bit is 0b.	RWS

Update Section 7.10.8 as follows:

7.10.8 Header Log Register (Offset 1Ch)

The Header Log register ~~captures~~ contains the header for the TLP corresponding to a detected error; refer to Section 6.2 for further details. Section 6.2 also describes the conditions where the packet header is ~~logged~~ recorded. This register is 16 bytes and adheres to the format of the headers defined throughout this specification.

The header is captured such that the fields of the header read by software in the same way the headers are presented in this document, when the register is read using DW accesses. Therefore, byte 0 of the header is located in byte 3 of the Header Log register, byte 1 of the header is in byte 2 of the Header Log register and so forth. For 12-byte headers, only bytes 0 through 11 of the Header Log register are used and values in bytes 12 through 15 are undefined.

Figure 7-38 details allocation of register fields in the Header Log register; Table 7-35 provides the respective bit definitions.

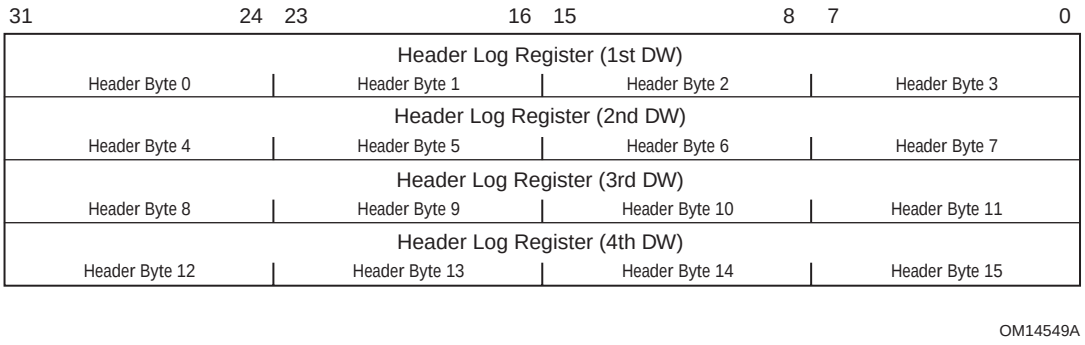

Figure 7-38: Header Log Register

Table 7-35: Header Log Register

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