# RapidIO™ Interconnect Specification Part 8: Error Management Extensions Specification

Rev. 1.3, 06/2005




**RapidIO Trade Association**

# Revision History

| Revision | Description | Date |
|---|---|---|
| 1.2 | First public release | 09/13/2002 |
| 1.3 | Technical changes: the following errata showings:<br>04-02-00002.001<br>and the following new features showings:<br>04-09-00022.002<br>Converted to ISO-friendly templates | 02/23/2005 |
| 1.3 | Removed confidentiality markings for public release | 06/07/2005 |



Questions regarding the RapidIO Trade Association, specifications, or membership should be forwarded to:

RapidIO Trade Association
Suite 325, 3925 W. Braker Lane
Austin, TX 78759
512-305-0070 Tel.
512-305-0009 FAX.

# Table of Contents

## Chapter 1 Error Management Extensions



## Chapter 2 Error Management Registers

# Table of Contents

## Annex A Error Management Discussion (Informative)

# List of Tables

# List of Tables

Blank page

# Chapter 1 Error Management Extensions

## 1.1 Introduction

The error management extensions describe added requirements in all physical and logical layers. These extensions add definitions to bits that were previously reserved in the Port *n* Control CSR and add new registers that are contained within the Error Management Extended Features Block. This chapter describes the behavior of a device when an error is detected and how the new registers and bits are managed by software and hardware.

## 1.2 Physical Layer Extensions

The following registers and register bit extensions allow software to monitor and control the reporting of transmission errors:

- (Extensions to the) Port *n* Control CSR defined in Section 2.2
- (Extensions to the) Port *n* Error and Status CSR defined in Section 2.2
- Port-write Target deviceID CSR defined in Section 2.3.2.8
- Port *n* Error Detect CSR defined in Section 2.3.2.10
- Port *n* Error Rate Enable CSR defined in Section 2.3.2.11
- Port *n* Attributes Capture CSR defined in Section 2.3.2.12
- Port *n* Packet/Control Symbol Capture 0 CSR defined in Section 2.3.2.13
- Port *n* Packet Capture 1-3 CSRs defined in Section 2.3.2.14 through Section 2.3.2.16
- Port *n* Error Rate CSR defined in Section 2.3.2.17
- Port *n* Error Rate Threshold CSR defined in Section 2.3.2.18

### 1.2.1 Port Error Detect, Enable and Capture CSRs

The occurrence of a transmission error shall be logged by hardware by setting the appropriate error indication bit in the Port *n* Error Detect CSR. Transmission errors that are enabled for error capture and error counting will have the corresponding bit set by software in the Port *n* Error Rate Enable CSR. When the Capture Valid Info status bit is not set in the Port *n* Error Capture Attributes CSR, information about the next enabled transmission error shall be saved to the Port *n* Error Capture CSRs. The Info Type and Error Type fields shall be updated and the Capture Valid Info status

bit shall be set by hardware in the Port *n* Error Capture Attributes CSR to lock the error capture registers. The first 16 bytes of the packet header or the 4 bytes of the control symbol that have a detected error are saved in the capture CSRs. Packets smaller than 16 bytes are captured in their entirety. The Port *n* Error Capture CSRs and the Port *n* Error Capture Attributes CSR are not overwritten by hardware with error capture information for subsequent errors until software writes a zero to the Capture Valid Info bit.

The Port *n* Error Detect CSR does not lock so subsequent error indications shall also be logged there by hardware. By reading the register, software may see the types of transmission errors that have occurred. The Port *n* Error Detect CSR is cleared by writing it with all logic 0s.

### 1.2.2 Error Reporting Thresholds

Transmission errors are normally hidden from system software since they may be recovered with no loss of data and without software intervention. Two thresholds are defined in the Port *n* Error Rate Threshold CSR which can be set to force a report to system software when the link error rate reaches a level that is deemed by the system to be either degraded or unacceptable. The two thresholds are respectively the Degraded Threshold and the Failed Threshold. These thresholds are used as follows.

When the error rate counter is incremented, the Error Rate Degraded Threshold Trigger provides a threshold value that, when equal to or exceeded by the value in the Error Rate Counter in the Port *n* Error Rate register, shall cause the error reporting logic to set the Output Degraded-encountered bit in the Port *n* Error and Status CSR, and notify the system software as described in Section 1.4.

The Error Rate Failed Threshold Trigger, if enabled, shall be larger than the degraded threshold trigger. It provides a threshold value that, when equal to or exceeded by the value in the Error Rate Counter, shall trigger the error reporting logic to set the Output Failed-encountered bit in the Port *n* Error and Status CSR, and notify system software as described in Section 1.4.

No action shall be taken if the Error Rate Counter continues to exceed either threshold value after initial notification when additional errors are detected. No action shall be taken when the Error Rate Counter drops below either threshold.

### 1.2.3 Error Rate Control and Status

The fields in the Port *n* Error Rate CSR are used to monitor the error rate of the link connected to port *n*.

The Error Rate Bias field determines the rate at which the Error Rate Counter is decremented and defines the acceptable error rate of the link for error reporting purposes. In the absence of additional counted link errors, this mechanism allows the system to recover from both Failed and Degraded levels of operation without a

software reset of the Error Rate Counter. If the link error rate is less than the decrement rate specified in the Error Rate Bias field, the value of the Error Rate counter will rarely be greater than 0x01 or 0x02.

The Error Rate Counter shall increment when a physical layer error is detected whose associated enable bit is set in the Port *n* Error Rate Enable register. The Error Rate Counter shall decrement at the rate specified by the Error Rate Bias field of the Port *n* Error Rate CSR. The Error Rate Counter shall not underflow (shall not decrement when equal to 0x00) and shall not overflow (shall not increment when equal to 0xFF). The incrementing and decrementing of the Error Rate Counter are in no way affected by the values in the Degraded and Failed thresholds. Software may reset the Error Rate Counter at any time.

The Error Rate Recovery field defines how far above the Error Rate Failed Threshold Trigger in the Port *n* Error Rate Threshold Register the Error Rate Counter is allowed to count. In the absence of additional counted errors, this allows software to control the length of time required for the value of the Error Rate Counter to drop below both the Failed and Degraded Thresholds.

The Peak Error Rate field shall contain the largest value encountered by the Error Rate Counter. This field is loaded whenever the current value of the Peak Error Rate field is exceeded by the value of the Error Rate Counter.

### 1.2.4 Port Behavior When Error Rate Failed Threshold is Reached

The behavior of a port when the Error Rate Counter in the Port *n* Error Rate CSR reaches the Error Rate Failed Threshold and the threshold is enabled depends upon the values of the Stop on Port Failed-encountered Enable and the Drop Packet Enable bits in the Port *n* Control CSR. The Table 1-1 below defines the required behavior.

**Table 1-1. Port Behavior when Error Rate Failed Threshold has been hit**

| Stop on Port Failed Encountered Enable | Drop Packet Enable | Port Behavior | Comments |
|---|---|---|---|
| 0 | 0 | The port shall continue to attempt to transmit packets to the connected device if the Output Failed-encountered bit is set and/or if the Error Rate Failed threshold has been met or exceeded. | All devices |
| 0 | 1 | The port shall discard packets that receive a Packet-not-accepted control symbol when the Error Rate Failed Threshold has been met or exceeded. Upon discarding a packet, the port shall set the Output Packet-dropped bit in the Port *n* Error and Status CSR. If the output port "heals", the Error Rate Counter falls below the Error Rate Failed Threshold, the output port shall continue to attempt to forward all packets. | Switch Device Only |
| 1 | 0 | The port shall stop attempting to send packets to the connected device when the Output Failed-encountered bit is set. The output port will congest. | All devices. |
| 1 | 1 | The port shall discard all output packets without attempting to send when the port's Output Failed-encountered bit is set. Upon discarding a packet, the port shall set Output Packet-dropped bit in the Port *n* Error and Status CSR. | All devices. |

### 1.2.5 Packet Timeout Mechanism in a Switch Device

In some systems, it is either desirable or necessary to bound the length of time a packet can remain in a switch. To enable this functionality, a switch shall monitor the length of time each packet accepted by one of its ports has been in the switch. The acceptance of a packet by a port is signaled by the port issuing a packet-accepted control symbol for the packet. The timing begins when the port accepts the packet.

If a packet remains in a switch longer than the Time-to-Live time specified by the Time-to-Live field of the Packet Time-to-live CSR as defined in Section 2.3.2.9, the packet shall be discarded rather than forwarded, the Output Packet-Dropped bit shall be set in the Port *n* Error and Status CSR and the system shall be notified as described in Section 1.4.

## 1.3 Logical and Transport Layer Extensions

While the RapidIO link may be working properly, an end point processing element may encounter logical or transport layer errors, or other errors unrelated to its RapidIO ports, while trying to complete a transaction. The "ERROR" status response transaction is the mechanism for the target device to indicate to the source that there is a problem completing the request. Experiencing a time-out waiting for a response is also a symptom of an end point or switch fabric with a problem. These

types of errors are logged and reporting enabled with a set of registers that are separate from those used for the Physical Layer errors.

- Logical/Transport Layer Error Detect CSR defined in Section 2.3.2.2
- Logical/Transport Layer Error Enable CSR defined in Section 2.3.2.3
- Logical/Transport Layer Capture CSRs defined in Section 2.3.2.4 to Section 2.3.2.7

### 1.3.1 Logical/Transport Error Detect, Enable and Capture CSRs

When a logical or transport layer error is detected, the appropriate error bit shall be set by the hardware in the Logical/Transport Layer Error Detect CSR. If the corresponding bit is also set in the Logical/Transport Layer Error Enable CSR, the detect register shall lock, the appropriate information is saved in the Logical/Transport Layer Capture registers, all resources held by the transaction are freed, and system software is notified of the error as described in Section 1.4. If multiple enabled errors occur during the same clock cycle, multiple bits will be set in the detect register and the contents of the Logical/Transport Layer Capture registers are implementation dependent. Once locked, subsequent errors will not set another error detect bit. The contents of the Logical/Transport Capture CSRs are valid if the bitwise AND of the Logical/Transport Layer Error Detect CSR and the Logical/Transport Layer Error Detect Enable CSR is not equal to zero (0x00000000).

Software shall write the Logical/Transport Detect register with all logic 0s to clear the error detect bits or a corresponding enable bit to unlock the register. Any other recovery actions associated with these types of errors are system dependent and outside the scope of this specification.

### 1.3.2 Message Passing Error Detection

Message passing is a special case of logical layer error recovery requiring error detection at both the source and destination ends of the message. The source of the message has the request-to-response time-out (defined in the Port Response Time-out Control CSR in the RapidIO Physical Layer specifications) to detect lost request or response packets in the switch fabric. However, in order to not hang the recipient mailbox in the case of a lost request packet for a multiple packet message, the recipient mailbox shall have an analogous response-to-request time-out. This time-out is for sending a response packet to receiving the next request packet of a given message operation, and has the same value as the request-to-response time-out that is already specified. The Logical/Transport Layer Control Capture CSR contains the ‘msg info’ field to capture the critical information of the last received (or sent) message segment before time-out.

## 1.4 System Software Notification of Error

System software is notified of logical, transport, and physical layer errors in two ways. An interrupt is issued to the local system by a device, the method of which is not defined in this specification, or a Maintenance port-write operation is issued by a device. Maintenance port-write operations are sent to a predetermined system host (defined in the Port-write Target deviceID CSR in Section 2.3.2.8). The sending device sets the Port-write Pending status bit in the Port *n* Error and Status CSR. A 16 byte data payload of the Maintenance Port-write packet contains the contents of several CSRs, the port on the device that encountered the error condition (for port-based errors), and some optional implementation specific additional information as shown in Table 1-2. Software indicates that it has seen the port-write operation by clearing the Port-write Pending status bit.

The Component Tag CSR is defined in the *RapidIO Part 3: Common Transport Specification*, and is used to uniquely identify the reporting device within the system. A Port ID field, the Logical/Transport Layer Detect CSR defined in Section 2.3.2.2, and the Port *n* Error Detect CSR defined in Section 2.3.2.10 are used to describe the encountered error condition.

**Table 1-2. Port-write Packet Data Payload for Error Reporting**

| Data Payload Byte Offset | Word | |
|---|---|---|
| 0x0 | Component Tag CSR | |
| 0x4 | Port *n* Error Detect CSR | |
| 0x8 | Implementation specific | Port ID (byte) |
| 0xC | Logical/Transport Layer Error Detect CSR | |

## 1.5 Mechanisms for Software Debug

In most systems, it is difficult to verify the error handling software. The Error management extensions make some registers writable for easier debug.

The Logical/Transport Layer Error Detect register and the Logical/Transport Layer Error Capture registers are writable by software to allow software debug of the system error recovery mechanisms. For software debug, software must write the Logical/Transport Layer Error capture registers with the desired address and device id information then write the Logical/Transport Layer Error Detect register to set an error bit and lock the registers. When an error detect bit is set, the hardware will inform the system software of the error using its standard error reporting mechanism. After the error has been reported, the system software may read and clear registers as necessary to complete its error handling protocol testing.

The Port *n* Error Detect register and the Port *n* Error Capture registers are also

writable by software to allow software debug of the system error recovery and thresholding mechanism. For debug, software must write the Port *n* Attributes Error Capture CSR to set the Capture Valid Info bit and then the packet/control symbol information in the other capture registers. Each write of a non-zero value to the Port *n* Error Detect CSR shall cause the Error Rate Counter to increment if the corresponding error bit is enabled in the Port *n* Error Rate Enable CSR. When a threshold is reached, the hardware will inform the system software of the error using its standard error reporting mechanism. After the error has been reported, the system software may read and clear registers as necessary to complete its error handling protocol testing.

Blank page

# Chapter 2 Error Management Registers

## 2.1 Introduction

This section describes the Error Management Extended Features block, and adds a number of new bits to the existing standard physical layer registers. ‘End-point only’ and ‘switch only’ register bits shall be considered reserved when the registers are implemented on devices for which these bits are not required.

## 2.2 Additions to Existing Registers

The following bits are added to the parallel and serial logical layer specification Port *n* Control CSRs.

**Table 2-1. Bit Settings for Port *n* Control CSRs**

| Bit | Name | Reset Value | Description |
|---|---|---|---|
| 28 | Stop on Port Failed-encountered Enable | 0b0 | This bit is used with the Drop Packet Enable bit to force certain behavior when the Error Rate Failed Threshold has been met or exceeded. See Section 1.2.4 of the Part 8: Error Management Extensions for detailed requirements. |
| 29 | Drop Packet Enable | 0b0 | This bit is used with the Stop on Port Failed-encountered Enable bit to force certain behavior when the Error Rate Failed Threshold has been met or exceeded. See Section 1.2.4 of the Part 8: Error Management Extensions for detailed requirements. |
| 30 | Port Lockout | 0b0 | When this bit is cleared, the packets that may be received and issued are controlled by the state of the Output Port Enable and Input Port Enable bits in the Port *n* Control CSR.<br>When this bit is set, this port is stopped and is not enabled to issue or receive any packets; the input port can still follow the training procedure and can still send and respond to link-requests; all received packets return packet-not-accepted control symbols to force an error condition to be signaled by the sending device |

The following bits are added to the parallel and serial specification Port *n* Error and Status CSRs.

**Table 2-2. Bit Settings for Port *n* Error and Status CSRs**

| Bit | Name | Reset Value | Description |
|---|---|---|---|
| 5 | Output Packet-dropped | 0b0 | Output port has discarded a packet. Once set remains set until written with a logic 1 to clear. |
| 6 | Output Failed-encountered | 0b0 | Output port has encountered a failed condition, meaning that the port's failed error threshold has been reached in the Port *n* Error Rate Threshold register. Once set remains set until written with a logic 1 to clear. |
| 7 | Output Degraded-encountered | 0b0 | Output port has encountered a degraded condition, meaning that the port's degraded error threshold has been reached in the Port *n* Error Rate Threshold register. Once set remains set until written with a logic 1 to clear. |

## 2.3 New Error Management Registers

This section describes the Extended Features block (EF_ID=0h0007) that allows an external processing element to manage the error status and reporting for a processing element. This chapter only describes registers or register bits defined by this extended features block. All registers are 32-bits and aligned to a 32-bit boundary.

Table 2-3 describes the required behavior for accesses to reserved register bits and reserved registers for the RapidIO Extended Features register space,

**Table 2-3. Extended Feature Space Reserved Access Behavior**

| Byte Offset | Space Name | Item | Initiator behavior | Target behavior |
|---|---|---|---|---|
| 0x100–FFFC | Extended Features Space | Reserved bit | read - ignore returned value[1] | read - return logic 0 |
| | | | write - preserve current value[2] | write - ignored |
| | | Implementation-defined bit | read - ignore returned value unless implementation-defined function understood | read - return implementation-defined value |
| | | | write - preserve current value if implementation-defined function not understood | write - implementation-defined |
| | | Reserved register | read - ignore returned value | read - return logic 0s |
| | | | write - | write - ignored |

[1]Do not depend on reserved bits being a particular value; use appropriate masks to extract defined bits from the read value.

[2]All register writes shall be in the form: read the register to obtain the values of all reserved bits, merge in the desired values for defined bits to be modified, and write the register, thus preserving the value of all reserved bits.

## 2.3.1 Register Map

Table 2-4 shows the register map for the error management registers. This register map is currently only defined for devices with up to 16 RapidIO ports, but can be extended or shortened if more or less port definitions are required for a device. For example, a device with four RapidIO ports is only required to use register map space corresponding to offsets [EF_PTR+0x00] through [EF_PTR+0x13C]. Register map offset [EF_PTR+0x140] can be used for another Extended Features block.

**Table 2-4. Error Management Extensions Register Map**

| | Block Byte Offset | Register Name |
|---|---|---|
| General | 0x0 | Error Management Extensions Block Header |
| | 0x4 | Reserved |
| | 0x8 | Logical/Transport Layer Error Detect CSR |
| | 0xC | Logical/Transport Layer Error Enable CSR |
| | 0x10 | Logical/Transport Layer High Address Capture CSR |
| | 0x14 | Logical/Transport Layer Address Capture CSR |
| | 0x18 | Logical/Transport Layer Device ID Capture CSR |
| | 0x1C | Logical/Transport Layer Control Capture CSR |
| | 0x20-24 | Reserved |
| | 0x28 | Port-write Target deviceID CSR |
| | 0x2C | Packet Time-to-live CSR |
| | 0x30-3C | Reserved |
| Port 0 | 0x40 | Port 0 Error Detect CSR |
| | 0x44 | Port 0 Error Rate Enable CSR |
| | 0x48 | Port 0 Attributes Capture CSR |
| | 0x4C | Port 0 Packet/Control Symbol Capture 0 CSR |
| | 0x50 | Port 0 Packet Capture 1 CSR |
| | 0x54 | Port 0 Packet Capture 2 CSR |
| | 0x58 | Port 0 Packet Capture 3 CSR |
| | 0x5C-64 | Reserved |
| | 0x68 | Port 0 Error Rate CSR |
| | 0x6C | Port 0 Error Rate Threshold CSR |
| | 0x70-7C | Reserved |

**Table 2-4. Error Management Extensions Register Map**

| | Block Byte Offset | Register Name |
|---|---|---|
| Port 1 | 0x80 | Port 1 Error Detect CSR |
| | 0x84 | Port 1 Error Rate Enable CSR |
| | 0x88 | Port 1 Attributes Capture CSR |
| | 0x8C | Port 1 Packet/Control Symbol Capture 0 CSR |
| | 0x90 | Port 1 Packet Capture 1 CSR |
| | 0x94 | Port 1 Packet Capture 2 CSR |
| | 0x98 | Port 1 Packet Capture 3 CSR |
| | 0x9C-A4 | Reserved |
| | 0xA8 | Port 1 Error Rate CSR |
| | 0xAC | Port 1 Error Rate Threshold CSR |
| | 0xB0-BC | Reserved |
| Ports 2-14 | 0xC0–3FC | Assigned to Port 2-14 CSRs |
| Port 15 | 0x400 | Port 15 Error Detect CSR |
| | 0x404 | Port 15 Error Rate Enable CSR |
| | 0x408 | Port 15 Attributes Capture CSR |
| | 0x40C | Port 15 Packet/Control Symbol Capture 0 CSR |
| | 0x410 | Port 15 Packet Capture 1 CSR |
| | 0x414 | Port 15 Packet Capture 2 CSR |
| | 0x418 | Port 15 Packet Capture 3 CSR |
| | 0x41C-424 | Reserved |
| | 0x428 | Port 15 Error Rate CSR |
| | 0x42C | Port 15 Error Rate Threshold CSR |
| | 0x430-43C | Reserved |

## 2.3.2 Command and Status Registers (CSRs)

Refer to Table 2-3 for the required behavior for access to reserved registers and register bits.

#### 2.3.2.1 Error Management Extensions Block Header (Block Offset 0x0)

The error management extensions block header register contains the EF_PTR to the next EF_BLK and the EF_ID that identifies this as the error management extensions block header.

**Table 2-5. Bit Settings for Error Management Extensions Block Header**

| Bit | Name | Reset Value | Description |
|---|---|---|---|
| 0-15 | EF_PTR | | Hard wired pointer to the next block in the data structure, if one exists |
| 16-31 | EF_ID | 0x0007 | Hard wired Extended Features ID |

#### 2.3.2.2 Logical/Transport Layer Error Detect CSR (Block Offset 0x08)

This register indicates the error that was detected by the Logical or Transport logic layer. Multiple bits may get set in the register if simultaneous errors are detected during the same clock cycle that the errors are logged.

**Table 2-6. Bit Settings for Logical/Transport Layer Error Detect CSR**

| Bit | Name | Reset Value | Description |
|---|---|---|---|
| 0 | IO error response | 0b0 | Received a response of 'ERROR' for an IO Logical Layer Request.<br>(end point device only) |
| 1 | Message error response | 0b0 | Received a response of 'ERROR' for an MSG Logical Layer Request.<br>(end point device only) |
| 2 | GSM error response | 0b0 | Received a response of 'ERROR' for a GSM Logical Layer Request.<br>(end point device only) |
| 3 | Message Format Error | 0b0 | Received MESSAGE packet data payload with an invalid size or segment (MSG logical)<br>(end point device only) |
| 4 | Illegal transaction decode | 0b0 | Received illegal fields in the request/response packet for a supported transaction (IO/MSG/GSM logical)<br>(switch or endpoint device) |
| 5 | Illegal transaction target error | 0b0 | Received a packet that contained a destination ID that is not defined for this end point. End points with multiple ports and a built-in switch function may not report this as an error (Transport)<br>(end point device only) |
| 6 | Message Request Time-out | 0b0 | A required message request has not been received within the specified time-out interval (MSG logical)<br>(end point device only) |
| 7 | Packet Response Time-out | 0b0 | A required response has not been received within the specified time out interval (IO/MSG/GSM logical)<br>(end point device only) |
| 8 | Unsolicited Response | 0b0 | An unsolicited/unexpected Response packet was received (IO/MSG/GSM logical; only Maintenance response for switches)<br>(switch or endpoint device) |

**Table 2-6. Bit Settings for Logical/Transport Layer Error Detect CSR**

| Bit | Name | Reset Value | Description |
|---|---|---|---|
| 9 | Unsupported Transaction | 0b0 | A transaction is received that is not supported in the Destination Operations CAR (IO/MSG/GSM logical; only Maintenance port-write for switches) (switch or endpoint device) |
| 10-23 | — | | Reserved |
| 24-31 | Implementation Specific error | 0x00 | An implementation specific error has occurred. (switch or end point device) |

### 2.3.2.3 Logical/Transport Layer Error Enable CSR (Block Offset 0x0C)

This register contains the bits that control if an error condition locks the Logical/Transport Layer Error Detect and Capture registers and is reported to the system host.

**Table 2-7. Bit Settings for Logical/Transport Layer Error Enable CSR**

| Bit | Name | Reset Value | Description |
|---|---|---|---|
| 0 | IO error response enable | 0b0 | Enable reporting of an IO error response. Save and lock original request transaction information in all Logical/Transport Layer Capture CSRs. (end point device only) |
| 1 | Message error response enable | 0b0 | Enable reporting of a Message error response. Save and lock original request transaction information in all Logical/Transport Layer Capture CSRs. (end point device only) |
| 2 | GSM error response enable | 0b0 | Enable reporting of a GSM error response. Save and lock original request transaction capture information in all Logical/Transport Layer Capture CSRs. (end point device only) |
| 3 | Message Format Error enable | 0b0 | Enable reporting of a message format error. Save and lock transaction capture information in Logical/Transport Layer Device ID and Control Capture CSRs. (end point device only) |
| 4 | Illegal transaction decode enable | 0b0 | Enable reporting of an illegal transaction decode error Save and lock transaction capture information in Logical/Transport Layer Device ID and Control Capture CSRs. (switch or end-point device) |
| 5 | Illegal transaction target error enable | 0b0 | Enable reporting of an illegal transaction target error. Save and lock transaction capture information in Logical/Transport Layer Device ID and Control Capture CSRs. (end point device only) |
| 6 | Message Request time-out enable | 0b0 | Enable reporting of a Message Request time-out error. Save and lock transaction capture information in Logical/Transport Layer Device ID and Control Capture CSRs for the last Message request segment packet received. (end point device only) |
| 7 | Packet Response Time-out error enable | 0b0 | Enable reporting of a packet response time-out error. Save and lock original request address in Logical/Transport Layer Address Capture CSRs. Save and lock original request Destination ID in Logical/Transport Layer Device ID Capture CSR. (end point device only) |

**Table 2-7. Bit Settings for Logical/Transport Layer Error Enable CSR**

| Bit | Name | Reset Value | Description |
|---|---|---|---|
| 8 | Unsolicited Response error enable | 0b0 | Enable reporting of an unsolicited response error. Save and lock transaction capture information in Logical/Transport Layer Device ID and Control Capture CSRs.<br>(switch or end-point device) |
| 9 | Unsupported Transaction error enable | 0b0 | Enable reporting of an unsupported transaction error. Save and lock transaction capture information in Logical/Transport Layer Device ID and Control Capture CSRs.<br>(switch or end-point device) |
| 10-23 | — | | Reserved |
| 24-31 | Implementation Specific error enable | 0x00 | Enable reporting of an implementation specific error has occurred. Save and lock capture information in appropriate Logical/Transport Layer Capture CSRs. |

### 2.3.2.4 Logical/Transport Layer High Address Capture CSR (Block Offset 0x10)

This register contains error information. It is locked when a Logical/Transport error is detected and the corresponding enable bit is set. This register is only required for end point devices that support 66 or 50 bit addresses.

**Table 2-8. Bit Settings for Logical/Transport Layer High Address Capture CSR**

| Bit | Name | Reset Value | Description |
|---|---|---|---|
| 0-31 | address[0-31] | All 0s | Most significant 32 bits of the address associated with the error (for requests, for responses if available) |

### 2.3.2.5 Logical/Transport Layer Address Capture CSR (Block Offset 0x14)

This register contains error information. It is locked when a Logical/Transport error is detected and the corresponding enable bit is set.

**Table 2-9. Bit Settings for Logical/Transport Layer Address Capture CSR**

| Bit | Name | Reset Value | Description |
|---|---|---|---|
| 0-28 | address[32-60] | All 0s | Least significant 29 bits of the address associated with the error (for requests, for responses if available) |
| 29 | — | | Reserved |
| 30-31 | xamsbs | 0b00 | Extended address bits of the address associated with the error (for requests, for responses if available) |

### 2.3.2.6 Logical/Transport Layer Device ID Capture CSR (Block Offset 0x18)

This register contains error information. It is locked when an error is detected and the corresponding enable bit is set.

**Table 2-10. Bit Settings for Logical/Transport Layer Device ID Capture CSR**

| Bit | Name | Reset Value | Description |
|---|---|---|---|
| 0-7 | MSB destinationID | 0x00 | Most significant byte of the destinationID associated with the error (large transport systems only) |
| 8-15 | destinationID | 0x00 | The destinationID associated with the error |
| 16-23 | MSB sourceID | 0x00 | Most significant byte of the sourceID associated with the error (large transport systems only) |
| 24-31 | sourceID | 0x00 | The sourceID associated with the error |

### 2.3.2.7 Logical/Transport Layer Control Capture CSR (Block Offset 0x1C)

This register contains error information. It is locked when a Logical/Transport error is detected and the corresponding enable bit is set.

**Table 2-11. Bit Settings for Logical/Transport Layer Control Capture CSR**

| Bit | Name | Reset Value | Description |
|---|---|---|---|
| 0-3 | ftype | 0x0 | Format type associated with the error |
| 4-7 | ttype | 0x0 | Transaction type associated with the error |
| 8-15 | msg info | 0x00 | letter, mbox, and msgseg for the last Message request received for the mailbox that had an error (Message errors only) |
| 16-31 | Implementation specific | 0x0000 | Implementation specific information associated with the error |

### 2.3.2.8 Port-write Target deviceID CSR (Block Offset 0x28)

This register contains the target deviceID to be used when a device generates a Maintenance port-write operation to report errors to a system host.

**Table 2-12. Bit Settings for Port-write Target deviceID CSR**

| Bit | Name | Reset Value | Description |
|---|---|---|---|
| 0-7 | deviceID_msb | 0x00 | This is the most significant byte of the port-write target deviceID (large transport systems only) |
| 8-15 | deviceID | 0x00 | This is the port-write target deviceID |
| 16 | large_transport | 0b0 | deviceID size to use for a port-write<br>0b0 - use the small transport deviceID<br>0b1 - use the large transport deviceID |
| 17-31 | — | | Reserved |

### 2.3.2.9 Packet Time-to-live CSR (Block Offset 0x2C)

The Packet Time-to-live register specifies the length of time that a packet is allowed to exist within a switch device. The maximum value of the Time-to-live variable (0xFFFF) shall correspond to 100 msec. +/-34%. The resolution (minimum step size) of the Time-to-live variable shall be (maximum value of Time-to-live)/($2^{16}$-1). The reset value is all logic 0s, which disables the Time-to-live function so that a packet never times out. This register is not required for devices without switch functionality.

**Table 2-13. Bit Settings for Packet Time-to-live CSR**

| Bit | Name | Reset Value | Description |
|---|---|---|---|
| 0-15 | Time-to-live value | 0x0000 | Maximum time that a packet is allowed to exist within a switch device |
| 16-31 | — | | Reserved |

### 2.3.2.10 Port *n* Error Detect CSR (Block Offset 0x40, 80,..., 400)

The Port *n* Error Detect Register indicates transmission errors that are detected by the hardware.

**Table 2-14. Bit Settings for Port *n* Error Detect CSR**

| Bit | Name | Reset Value | Description |
|---|---|---|---|
| 0 | Implementation specific error | 0b0 | An implementation specific error has been detected |
| 1-7 | — | | Reserved |
| 8 | Received S-bit error | 0b0 | Received a packet/control symbol with an S-bit parity error (parallel) |
| 9 | Received corrupt control symbol | 0b0 | Received a control symbol with a bad CRC value (serial)<br>Received a control symbol with a true/complement mismatch (parallel) |
| 10 | Received acknowledge control symbol with unexpected ackID | 0b0 | Received an acknowledge control symbol with an unexpected ackID (packet-accepted or packet_retry) |
| 11 | Received packet-not-accepted control symbol | 0b0 | Received packet-not-accepted acknowledge control symbol |
| 12 | Received packet with unexpected ackID | 0b0 | Received packet with unexpected ackID value - out-of-sequence ackID |
| 13 | Received packet with bad CRC | 0b0 | Received packet with a bad CRC value |
| 14 | Received packet exceeds 276 Bytes | 0b0 | Received packet which exceeds the maximum allowed size |
| 15-25 | — | | Reserved |
| 26 | Non-outstanding ackID | 0b0 | Link_response received with an ackID that is not outstanding |
| 27 | Protocol error | 0b0 | An unexpected packet or control symbol was received |
| 28 | Frame toggle edge error | 0b0 | FRAME signal toggled on falling edge of receive clock (parallel) |

**Table 2-14. Bit Settings for Port *n* Error Detect CSR**

| Bit | Name | Reset Value | Description |
|---|---|---|---|
| 29 | Delineation error | 0b0 | FRAME signal toggled on non-32-bit boundary (parallel)<br>Received unaligned /SC/ or /PD/ or undefined code-group (serial) |
| 30 | Unsolicited acknowledge control symbol | 0b0 | An unexpected acknowledge control symbol was received |
| 31 | Link time-out | 0b0 | An acknowledge or link-response control symbol is not received within the specified time-out interval |

### 2.3.2.11 Port *n* Error Rate Enable CSR (Block Offset 0x44, 84,..., 404)

This register contains the bits that control when an error condition is allowed to increment the error rate counter in the Port *n* Error Rate Threshold Register and lock the Port *n* Error Capture registers.

**Table 2-15. Bit Settings for Port *n* Error Rate Enable CSR**

| Bit | Name | Reset Value | Description |
|---|---|---|---|
| 0 | Implementation specific error enable | 0b0 | Enable error rate counting of implementation specific errors |
| 1-7 | — | | Reserved |
| 8 | Received S-bit error enable | 0b0 | Enable error rate counting of a packet/control symbol with an S-bit parity error (parallel) |
| 9 | Received control symbol with bad CRC enable | 0b0 | Enable error rate counting of a corrupt control symbol |
| 10 | Received out-of-sequence acknowledge control symbol enable | 0b0 | Enable error rate counting of an acknowledge control symbol with an unexpected ackID |
| 11 | Received packet-not-accepted control symbol enable | 0b0 | Enable error rate counting of received packet-not-accepted control symbols |
| 12 | Received packet with unexpected ackID enable | 0b0 | Enable error rate counting of packet with unexpected ackID value - out-of-sequence ackID |
| 13 | Received packet with Bad CRC enable | 0b0 | Enable error rate counting of packet with a bad CRC value |
| 14 | Received packet exceeds 276 Bytes enable | 0b0 | Enable error rate counting of packet which exceeds the maximum allowed size |
| 15-25 | — | | Reserved |
| 26 | Non-outstanding ackID enable | 0b0 | Enable error rate counting of link-responses received with an ackID that is not outstanding |
| 27 | Protocol error enable | 0b0 | Enable error rate counting of protocol errors |
| 28 | Frame toggle edge error enable | 0b0 | Enable error rate counting of frame toggle edge errors |

**Table 2-15. Bit Settings for Port *n* Error Rate Enable CSR**

| Bit | Name | Reset Value | Description |
|---|---|---|---|
| 29 | Delineation error | 0b0 | Enable error rate counting of delineation errors |
| 30 | Unsolicited acknowledge control symbol | 0b0 | Enable error rate counting of unsolicited acknowledge control symbol errors |
| 31 | Link time-out | 0b0 | Enable error rate counting of link time-out errors |

### 2.3.2.12 Port *n* Attributes Capture CSR (Block Offset 0x48, 88,..., 408)

The error capture attribute register indicates the type of information contained in the Port *n* error capture registers. In the case of multiple detected errors during the same clock cycle one of the errors must be reflected in the Error type field. The error that is reflected is implementation dependent.

**Table 2-16. Bit Settings for Port *n* Attributes Capture CSR**

| Bit | Name | Reset Value | Description |
|---|---|---|---|
| 0-1 | Info type | 0b00 | Type of information logged<br>00 - packet<br>01 - control symbol (only error capture register 0 is valid)<br>10 - implementation specific (capture register contents are implementation specific)<br>11 - undefined (S-bit error), capture as if a packet (parallel physical layer only) |
| 2 | — | | Reserved |
| 3-7 | Error type | 0x00 | The encoded value of the bit in the Port *n* Error Detect CSR that describes the error captured in the Port *n* Error Capture CSRs. |
| 8-27 | Implementation Dependent | All 0s | Implementation Dependent Error Information |
| 28-30 | — | | Reserved |
| 31 | Capture valid info | 0b0 | This bit is set by hardware to indicate that the Packet/control symbol capture registers contain valid information. For control symbols, only capture register 0 will contain meaningful information. |

### 2.3.2.13 Port *n* Packet/Control Symbol Capture 0 CSR (Block Offset 0x4C, 8C,..., 40C)

Captured control symbol information includes the true and complement of the control symbol. This is exactly what arrives on the RapidIO interface with bits 0-7 of the capture register containing the least significant byte of the 32-bit quantity. This register contains the first 4 bytes of captured packet symbol information.

**Table 2-17. Bit Settings for Port *n* Packet/Control Symbol Capture 0 CSR**

| Bit | Name | Reset Value | Description |
|---|---|---|---|
| 0-31 | Capture 0 | All 0s | True and Complement of Control Symbol (parallel) or Control Character and Control Symbol (serial) or Bytes 0 to 3 of Packet Header |

### 2.3.2.14 Port *n* Packet Capture 1 CSR (Block Offset 0x50, 90,..., 410)

Error capture register 1 contains bytes 4 through 7 of the packet header.

**Table 2-18. Bit Settings for Port *n* Packet Capture 1 CSR**

| Bit | Name | Reset Value | Description |
|---|---|---|---|
| 0-31 | Capture 1 | All 0s | Bytes 4 thru 7 of the packet header. |

### 2.3.2.15 Port *n* Packet Capture 2 CSR (Block Offset 0x54, 94,..., 414)

Error capture register 2 contains bytes 8 through 11 of the packet header.

**Table 2-19. Bit Settings for Port *n* Packet Capture 2 CSR**

| Bit | Name | Reset Value | Description |
|---|---|---|---|
| 0-31 | Capture 2 | All 0s | Bytes 8 thru 11of the packet header. |

### 2.3.2.16 Port *n* Packet Capture 3 CSR (Block Offset 0x58, 98,..., 418)

Error capture register 3 contains bytes 12 through 15 of the packet header.

**Table 2-20. Bit Settings for Port *n* Packet Capture 3 CSR**

| Bit | Name | Reset Value | Description |
|---|---|---|---|
| 0-31 | Capture 3 | All 0s | Bytes 12 thru 15 of the packet header. |

### 2.3.2.17 Port *n* Error Rate CSR (Block Offset 0x68, A8,..., 428)

The Port *n* Error Rate register is a 32-bit register used with the Port *n* Error Rate Threshold register to monitor and control the reporting of transmission errors, shown in Table 2-21.

**Table 2-21. Bit Settings for Port *n* Error Rate CSR**

| Bit | Name | Reset Value | Description |
|---|---|---|---|
| 0-7 | Error Rate Bias | 0x80 | These bits provide the error rate bias value<br>0x00 - do not decrement the error rate counter<br>0x01 - decrement every 1ms (+/-34%)<br>0x02 - decrement every 10ms (+/-34%)<br>0x04 - decrement every 100ms (+/-34%)<br>0x08 - decrement every 1s (+/-34%)<br>0x10 - decrement every 10s (+/-34%)<br>0x20 - decrement every 100s (+/-34%)<br>0x40 - decrement every 1000s (+/-34%)<br>0x80 - decrement every 10000s (+/-34%)<br>other values are reserved |
| 8-13 | — | | Reserved |
| 14-15 | Error Rate Recovery | 0b00 | These bits limit the incrementing of the error rate counter above the failed threshold trigger.<br>0b00 - only count 2 errors above<br>0b01 - only count 4 errors above<br>0b10 - only count 16 error above<br>0b11 - do not limit incrementing the error rate count |
| 16-23 | Peak Error Rate | 0x00 | This field contains the peak value attained by the error rate counter. |
| 24-31 | Error Rate Counter | 0x00 | These bits maintain a count of the number of transmission errors that have been detected by the port, decremented by the Error Rate Bias mechanism, to create an indication of the link error rate. |

### 2.3.2.18 Port *n* Error Rate Threshold CSR (Block Offset 0x6C, AC,..., 42C)

The Port *n* Error Rate Threshold register is a 32-bit register used to control the reporting of the link status to the system host.

**Table 2-22. Bit Settings for Port *n* Error Rate Threshold CSR**

| Bit | Name | Reset Value | Description |
|---|---|---|---|
| 0-7 | Error Rate Failed Threshold Trigger | 0xFF | These bits provide the threshold value for reporting an error condition due to a possibly broken link.<br>0x00 - Disable the Error Rate Failed Threshold Trigger<br>0x01 - Set the error reporting threshold to 1<br>0x02 - Set the error reporting threshold to 2<br>...<br>0xFF - Set the error reporting threshold to 255 |

**Table 2-22. Bit Settings for Port *n* Error Rate Threshold CSR**

| Bit | Name | Reset Value | Description |
|---|---|---|---|
| 8-15 | Error Rate Degraded Threshold Trigger | 0xFF | These bits provide the threshold value for reporting an error condition due to a degrading link.<br>0x00 - Disable the Error Rate Degraded Threshold Trigger<br>0x01 - Set the error reporting threshold to 1<br>0x02 - Set the error reporting threshold to 2<br>...<br>0xFF - Set the error reporting threshold to 255 |
| 16-31 | — | | Reserved |

# Annex A Error Management Discussion (Informative)

## A.1 Introduction

This section is intended to provide useful information/background on the application of the error management capabilities. This section is a guideline, not part of the specification.

## A.2 Limitations of Error Management Discussion

The RapidIO hardware that implements the Error Management extensions is able to log transmission errors and errors that occur at a higher level. Some error scenarios require no software intervention and recovery procedures are done totally by the hardware.

Some error scenarios detected require fault management software for recovery to be successful. For example, some types of logical layer errors on a Read or Write operation may be recoverable by killing the software process using the affected memory space and removing the memory space from the available system resource pool. It may also be possible for software to retry the operation, possibly through a different path in the switch fabric. Since such fault management software is typically tightly coupled to a particular system and/or implementation, it is considered outside of the scope of this specification.

Another area of fault recovery that requires fault management software to be implemented is correcting of system state after an error during an atomic operation. The swap style Atomic operations are possibly recoverable through software and require software convention to uniquely identify attempts to take locks. For example, if the request is lost and times out, software can examine the current lock value to determine if the request or the associated response was the transaction that was lost in the switch fabric. For all other Atomic operations (such as the Atomic set operation), it is impossible to correct the system state in the presence of a ‘lost packet’ type of error.

The use of RapidIO message packets relies on the use of higher layer protocols for error management. Since end points that communicate via messaging are typically running a variety of higher layer protocols, error reporting of both request and response time-outs is done locally by the message queue management controller.

Note that side effect errors can occur, for example, ERROR responses or RETRY responses during an active (partially completed) message, which may complicate the recovery procedure. The recovery strategies for messages lost in this manner are outside of the scope of this specification.

Globally Shared Memory systems that encounter a logical or transport layer error are typically not recoverable by any mechanism as this usually means that the processor caches are no longer coherent with the main memory system. Historically, recovery from such errors requires a complete reboot of the machine after the component that caused the error is repaired or replaced.

## A.3 Hot-insertion/extraction Discussion

Hot-insertion can be regarded as an error condition in which a new part of the system is detected, therefore, hot-insertion of a Field Replaceable Unit (FRU) can be handled utilizing the above described mechanisms. This section describes two approaches for hot insertion. The first generally applies to high availability systems, or systems where FRUs need to brought into the system in a controlled manner. The second generally applies to systems where availability is less of a concern, for example, a trusted system or a system without a system host.

At system boot time, the system host identifies all of the unattached links in the machine through system discovery and puts them in a locked mode, whereby all incoming packets are to be rejected, leaving the drivers and receivers enabled. This is done by setting the Discovered bit in the Port General Control CSR and the Port Lockout bit in the Port *n* Control CSR. Note that whenever an FRU is removed, the port lockout bit should be used to ensure that whatever new FRU is inserted cannot access the system until the system host allows it. When a FRU is hot-inserted connecting to a switch device, the now connected link will automatically start the training sequence. When training is complete (the Port OK bit in the Port *n* Error and Status CSR is now set), the locked port generates a Maintenance port-write operation to notify the system host of the new connection, and sets the Port-write Pending bit.

On receipt of the port-write, the system host is responsible for bringing the inserted FRU into the system in a controlled manner. The system host can communicate with the inserted FRU using Maintenance operations after clearing all error conditions, if any, clearing the Port Lockout bit and clearing the Output and Input Port Enable bits in the Port *n* Control CSR. This procedure allows the system host to access the inserted FRU safely, without exposing itself to incorrect behavior by the inserted FRU.

In order to issue Maintenance operations to the inserted FRU, the system host must first make sure that the ackID values for both ends are consistent. Since the inserted FRU has just completed a power-up reset sequence, both it’s transmit and receive ackID values are the reset value of 0x00. The system host can set the switch device’s

transmit and receive ackID values to also be 0x00 through the Port *n* Local ackID Status CSR if they are not already in that state, and can then issue packets normally.

The second method for hot insertion would allow the replaced FRU to bring itself into the system, which is necessary for a system in which the FRU is the system host itself. In this approach, the Port Lockout bit is not set and instead the Output and Input Port Enable bits are set for any unconnected port, allowing inserted FRUs free access to the system without reliance on a system host. Also, a port-write operation is not generated when the training sequence completes and the link is active, so a host is not notified of the event. However, this method leaves the system vulnerable to corruption from a misbehaving hot-inserted FRU.

As with the first case, the system host must make the ackID values for both link partners match in order to begin sending packets. In order to accomplish this, the system host generates a link-request/link-status to the attached device to obtain it's expected receiver value using the Port *n* Link Maintenance Request and Response CSRs. It can then set its transmit ackID value to match. Next, the system host generates a Maintenance write operation to set the attached device's Port *n* Local ackID Status CSR to set the transmit ackID value to match the receive ackID value in the system host. Upon receipt of the maintenance write, the attached device sets it's transmit ackID value as instructed, and generates the maintenance response using the new value. Packet transmission can now proceed normally.

Hot extraction from a port's point of view behaves identically to a very rapidly failing link and therefore can utilize the above described error reporting mechanism. Hot extraction is ideally done in a controlled fashion by taking the FRU to be removed out of the system as a usable resource through the system management software so that extraction does not cause switch fabric congestion or result in a loss of data.

The required mechanical aspects of hot-insertion and hot-extraction are not addressed in this specification.

## A.4 Port-write Discussion

The error management specification includes only one destination for port-write operations, while designers of reliable systems would assume that two is the minimum number. This section explains the rationale for only having one port-write destination.

It is assumed that in the event of an error on a link that both ends of the link will see the error. Thus, there are two parties who can be reporting on any error. In the case that the sole link between an end point and a switch fails completely, the switch is expected to see and report the error. When one of a set of redundant links between an end point and a switch device fails, it is expected that the switch and possibly the end point will report the failure.

When a link between two switches fails, it is assumed that there are multiple paths to the controlling entity available for the port-write to travel. The switches will be able to send at least one, and possibly two, reports to the system host. It is assumed that it is possible to set up a switch's routing parameters such that the traffic to the system host will follow separate paths from each switch.

In some reliable systems, the system host is implemented as multiple redundant subsystems. It is assumed in RapidIO that only one subsystem is actually in control at any one time, and so should be the recipient of all port-writes. If the subsystem that should be in control is detected to be insane, it is the responsibility of the rest of the control subsystem to change the destination for port-writes to be the new subsystem that is in control.

## A.5 Physical Layer Fatal Error Recovery Discussion

Recovery from a fatal error under software control at the physical layer may be possible under certain circumstances. An example of this would be if the transmitter and receiver have lost synchronization of their ackIDs. This could occur if one end of the link experienced a spurious reset. In this case a loss of packets may occur as there may be outstanding unacknowledged packets between the transmitter and the receiver.

Such an event would cause an error to be detected given the appropriate initial conditions at the transmitter, and, eventually a port-write to the system host to be generated if the system is properly configured:

- The reset state of the Input Port Enable bit in the Port *n* Control CSR set to disabled throughout the system.
- All defined errors in the Port *n* Error Detect CSRs are enabled and will increment the error rate counter throughout the system.

If a device experiences a reset event, numerous errors will be detected by the transmitter over time, and eventually an error threshold is reached as described in Section 1.2.2, "Error Reporting Thresholds", and the system host is notified as described in Section 1.4, "System Software Notification of Error". The most likely errors that will be detected are bits 12 (Received packet-not-accepted control symbol) and 26 (Non-outstanding ackID) in the Port *n* Error Detect CSRs, but others could be encountered depending upon the state of the link at the time of the reset event.

Re-synchronizing the ackIDs must be done from the transmitter side as it is not possible to communicate with the receiver with maintenance transactions in this situation. This can be done by resetting some of the physical layer state by writing the Port *n* Local ackID Status CSR with the appropriate ackID values. It may be necessary to have the transmitter drop outstanding packets using that CSR as well, depending upon the situation. It may not be desirable, or it might not be possible, to resend the packets, depending upon the state of the overall system and the

transmitter implementation.

Therefore, the following sequence of events occur:

1. The system is configured as described above and is operating.
2. The receiver of a transmitter/receiver pair experiences a reset.
3. The transmitter enters error recovery mode and attempts to re-train the link.
4. Eventually the receiver comes back and link re-training completes.
5. The transmitter starts the error recovery sequence and begins to encounter large numbers of errors due to a bad ackID for a link-response (which may immediately cause a port-write transaction to be sent to the system host to report the condition) or having all packets receive packet-not-accepted control symbols. As noted earlier, other errors may also be detected.
6. At some point, an error threshold is reached and the system host is sent a port-write maintenance transaction to report the condition, if one has not already been sent.
7. The system host cleans up the machine using maintenance transactions, including resetting ackIDs in the transmitter and rediscovering and reconfiguring the lost portion of the machine. This may be a very complex and time-consuming task.

Note that it may be useful to implement resetting the ackIDs and restarting the link in hardware for lab debug or for applications where frequent resets are expected and software intervention is not required.

## A.6 Persistence of Error Management Registers

Under some conditions, a device may be unable to accept any packets because it is in an undefined, ‘broken’ condition. It is unable to accept Maintenance packets to access any of its Error Detect and capture registers so it cannot be queried by software. Only a device ‘reset’ is able to bring the device back. The meaning of a link-request/reset condition may be modified for some implementations of the Error Management Extensions to be a ‘soft reset’ condition. A device that supports a soft reset will still cause a hardware reset however the Port *n* Error Detect register, the Port *n* Error Capture registers, the Logical/Transport Error Detect register, and the Logical/Transport Error Capture registers may retain their previous values.

Blank page

# Glossary of Terms and Abbreviations

The glossary contains an alphabetical list of terms, phrases, and abbreviations used in this book.

**D**

**Degraded threshold**. Bits 8-15 of the Port n Error Rate Threshold CSR. An application-specific level that indicates an unacceptable error rate resulting in degraded throughput, when equal to the error rate count.

**F**

**Failed threshold**. Bits 0-7 of the Port n Error Rate Threshold CSR. An application-specific level that indicates an error rate due to a broken link, when equal to the error rate count.

**H**

**Hot-insertion**. Hot-insertion is the insertion of a processing element into a powered-up system.

**Hot-extraction**. Hot-extraction is the removal of a processing element from a powered-up system.

**L**

**Logical/Transport error**. A logical/transport error is one that cannot be resolved using the defined transmission error recovery sequence, results in permanent loss of data or causes system corruption. Recovery may possible under software control.

**N**

**Non-reporting processing element**. A non-reporting processing element depends upon an attached device (usually a switch) to report its logged errors to the system host on its behalf.

**O**

**Operation**. A set of transactions between end point devices in a RapidIO system (requests and associated responses) such as a read or a write.

**Ownership**. A processing element has the only valid copy of a coherence granule and is responsible for returning it to home memory.

**P**

**Physical error**. A physical error occurs only in the physical layer.

**Port healing**. The process whereby software resets the error rate count, or allows it to decrement as required by the error rate bias field of the Port n Error Rate CSR.

---

**R**

**Read operation**. An operation used to obtain a globally shared copy of a coherence granule.

**Reporting processing element**. A reporting processing element is capable of reporting its logged errors to the system host.

---

**S**

**Switch processing element**. One of three processing elements, a switch processing element, or switch, is capable of logging and reporting errors to the host system.

---

**T**

**Transmission error**. A transmission error is one that can be resolved using the defined transmission error recovery sequence, results in no permanent loss of data and does not cause system corruption. Recovery may also be possible under software control using mechanisms outside of the scope of this specification.

Blank page

Blank page