RapidIO™ Interconnect Specification Part 6: 1x/4x LP-Serial Physical Layer Specification

Rev. 1.3, 06/2005

RapidIO Trade Association

Revision History

Revision	Description	Date
1.1	First release	12/17/2001
1.2	Technical changes: incorporate Rev. 1.1 errata rev. 1.1.1, errata 3	06/26/2002
1.3	Technical changes: incorporate Rev 1.2 errata 1 as applicable, the following errata showings: 03-03-00004.002, 03-07-00002.001, 03-12-00000.002, 03-12-00002.004, 04-02-00000.001, 04-05-00000.003, 04-05-00006.002 (partial), 04-05-00007.001 and the following new features showings: 02-03-0003.004, 02-06-00001.004, 04-08-00013.002, 04-09-00022.002 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 Overview

Chapter 2 Packets

Chapter 3 Control Symbols

Table of Contents

Chapter 4 PCS and PMA Layers

Table of Contents

Chapter 5 LP-Serial Protocol

Table of Contents

Chapter 6 LP-Serial Registers

Table of Contents

Table of Contents

Blank page

List of Figures

List of Figures

List of Tables

List of Tables

Chapter 1 Overview

1.1 Introduction

The *RapidIO Part 6: 1x/4x LP-Serial Physical Layer Specification* addresses the physical layer requirements for devices utilizing an electrical serial connection medium. This specification defines a full duplex serial physical layer interface (link) between devices using unidirectional differential signals in each direction. Further, it allows ganging of four serial links for applications requiring higher link performance. It also defines a protocol for link management and packet transport over a link.

RapidIO systems are comprised of end point processing elements and switch processing elements. The RapidIO interconnect architecture is partitioned into a layered hierarchy of specifications which includes the Logical, Common Transport, and Physical layers. The Logical layer specifications define the operations and associated transactions by which end point processing elements communicate with each other. The Common Transport layer defines how transactions are routed from one end point processing element to another through switch processing elements. The Physical Layer defines how adjacent processing elements electrically connect to each other. RapidIO packets are formed through the combination of bit fields defined in the Logical, Common Transport, and Physical Layer specifications.

The RapidIO 1x/4x LP-Serial specification defines a protocol for packet delivery between serial RapidIO devices including packet and control symbol transmission, flow control, error management, and other device to device functions. A particular device may not implement all of the mode selectable features found in this document. See the appropriate user's manual or implementation specification for specific implementation details of a device.

The 1x/4x LP-Serial physical layer specification has the following properties:

- Embeds the transmission clock with data using an 8B/10B encoding scheme.
- Supports one serial differential pair, referred to as one lane, or four ganged serial differential pairs, referred to as four lanes, in each direction.
- Allows switching packets between RapidIO 1x/4x LP-Serial ports and *RapidIO Part 4: 8/16 LP-LVDS Physical Layer Specification* ports without requiring packet manipulation.
- Employs similar retry and error recovery protocols as the RapidIO 8/16 LP-LVDS physical layer specification.

- Supports transmission rates of 1.25, 2.5, and 3.125 Gbaud (data rates of 1.0, 2.0, and 2.5 Gbps) per lane.

This specification first defines the individual elements that make up the link protocol such as packets, control symbols, and the serial bit encoding scheme. This is followed by a description of the link protocol. Finally, the control and status registers, signal descriptions, and electrical specifications are specified.

1.2 Packets

Chapter 2, “Packets” defines how a RapidIO 1x/4x LP-Serial packet is formed by prefixing a 10-bit physical layer header to the combined RapidIO transport and logical layer bit fields followed by an appended 16-bit CRC field.

This chapter shows the packet header format, the packet field definitions, the CRC error detection mechanism, and the packet alignment rules necessary to form LP-Serial packets.

1.3 Control Symbols

Chapter 3, “Control Symbols” defines the format of the two classes of control symbols (stype0 and stype1) used for packet acknowledgment, link utility functions, link maintenance, and packet delineation. A control symbol is a 24-bit entity (including a 5-bit CRC code). The control symbol is used for packet delineation and may also be embedded within a packet as well as sent when the link is idle.

Acknowledgment control symbols are used by processing elements to indicate packet transmission status. Utility control symbols are used to communicate buffer status and link recovery synchronization. Link maintenance control symbols are used by adjacent devices to communicate physical layer status, synchronization requests, and device reset.

1.4 PCS and PMA Layers

Chapter 4, “PCS and PMA Layers” describes the Physical Coding Sublayer (PCS) functionality as well as the Physical Media Attachment (PMA) functionality. The PCS layer functionality includes 8B/10B encoding scheme for embedding clock with data. It also gives transmission rules for the 1x and 4x interfaces and defines the link initialization sequence for clock synchronization.

The PMA (Physical Medium Attachment) function is responsible for serializing the 10-bit code-groups to and from the serial bitstream(s).

1.5 LP-Serial Protocol

Chapter 5, “LP-Serial Protocol” describes in detail how packets, control symbols, and the PCS/PMA layers are used to implement the physical layer protocol. This includes topics such as link initialization, link maintenance, error detection and recovery, flow control, and transaction delivery ordering.

1.6 LP-Serial Registers

Chapter 6, “LP-Serial Registers” describes the physical layer control and status register set. By accessing these registers a processing element may query the capabilities and status and configure another 1x/4x LP-Serial RapidIO processing element.

These registers utilize the Extended Features blocks and are accessed using *RapidIO Part 1: Input/Output Logical Specification* maintenance operations. Four types of RapidIO devices are defined in this section as follows:

- Generic End Point Processing Elements
- Generic End Point Processing Elements with software assisted error recovery
- Generic End Point Free Processing Elements (typically switch processing elements)
- Generic End Point Free Processing Elements with software assisted error recovery

1.7 Signal Descriptions

Chapter 7, “Signal Descriptions” contains the signal pin descriptions for a RapidIO LP-Serial end point device and shows connectivity between processing elements with 1x ports and processing elements with 4x ports.

1.8 AC Electrical Specifications

Chapter 8, “Electrical Specifications” describes the electrical specifications for the RapidIO 1x/4x LP-Serial device. This section defines two transmission types; short run and long run, as well as three speed grades (1.25 GHz, 2.5 GHz, and 3.125 GHz). This section also shows the required receiver eye diagrams for each link speed.

1.9 Interface Management

Annex A, “Interface Management (Informative)” contains information pertinent to interface management in a RapidIO system, including error recovery, link initialization, and packet retry state machines.

1.10 System Resources

Annex B, “Critical Resource Performance Limits (Informative)” contains a discussion on outstanding transactions and their relationship to transmission distance capability.

1.11 Manufacturability and Testability

Section Annex C, “Manufacturability and Testability (Informative)” recommends implementing to IEEE standard 1149.6 for improved manufacturing and manufacturing test.

Chapter 2 Packets

2.1 Introduction

This chapter specifies the LP-Serial packet format and the fields that are added by LP-Serial physical layer. These packets are fed into the PCS function explained in Chapter 4, “PCS and PMA Layers”.

2.2 Packet Field Definitions

This section specifies the bit fields added to a packet by the LP-Serial physical layer. These fields are required to implement the flow control, error management, and other specified system functions of the LP-Serial specification. The fields are specified in Table 2-1.

Table 2-1. Packet Field Definitions

Field	Description
ackID	Acknowledge ID is the packet identifier for acknowledgments back to the packet sender—see Section 5.4.2 for details concerning ackID functionality.
rsvd	The reserved bits are set to logic 0 when the packet is generated and ignored when a packet is received.
prio	Sets packet priority: 0b00 - lowest priority 0b01 - medium priority 0b10 - high priority 0b11 - highest priority See Section 5.4.3 for an explanation of prioritizing packets
CRF	Critical Request Flow is an optional bit that differentiates between flows of equal priority If Critical Request Flow is not supported, this bit is reserved See Section 5.4.3 for an explanation of prioritizing packets
CRC	Cyclic Redundancy Code used to detect transmission errors in the packet. See Section 2.4.1 for details on the CRC error detection scheme.

2.3 Packet Format

This section specifies the format of a LP-Serial packets. Figure 2-1 shows the format of the LP-Serial packet and how the physical layer ackID, rsvd, CRF, and prio fields are prefixed at the beginning of the packet and the 16-bit CRC field is appended to the end of the packet.

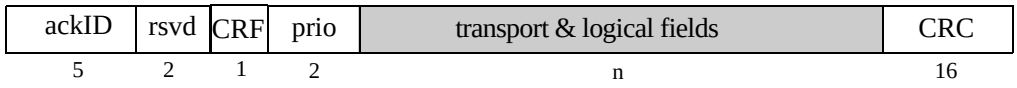


Figure 2-1. Packet Format

The unshaded fields are the fields added by the physical layer. The shaded field is the combined logical and transport layer bits and fields that are passed to the physical layer. The 2-bit rsvd field is required to make the packet length an integer multiple of 16 bits.

LP-Serial packets shall have a length that is an integer multiple of 32 bits. This sizing simplifies the design of port logic whose internal data paths are an integer multiple of 32 bits in width. Packets, as defined in this specification and the appropriate logical and transport layer specifications, have a length that is an integer multiple of 16 bits. This is illustrated in Figure 2-2. If the length of a packet defined by the above combination of specifications is an odd multiple of 16 bits, a 16-bit pad whose value is 0 (0x0000) shall be appended at the end of the packet such that the resulting padded packet is an integer multiple of 32 bits in length.

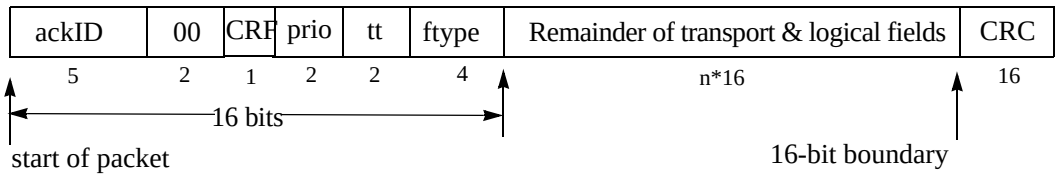


Figure 2-2. Packet Alignment

2.4 Packet Protection

A 16-bit CRC code is added to each packet by the LP-Serial physical layer to provide error detection. The code covers the entire packet except for the ackID field and one bit of the rsvd field, which are considered to be zero for the CRC calculations. Figure 2-3 shows the CRC coverage for the first 16 bits of the packet which contain the bits not covered by the code.

This structure allows the ackID to be changed on a link-by-link basis as the packet is transported across the fabric without requiring that the CRC be recomputed for each link. Since ackIDs on each link are assigned sequentially for each subsequent transmitted packet, an error in the ackID field is easily detected.

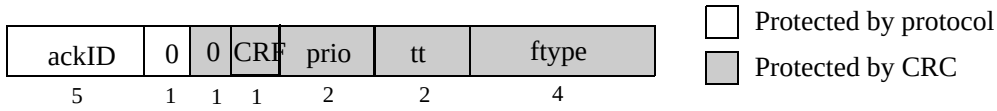


Figure 2-3. Error Coverage of First 16 Bits of Packet Header

2.4.1 Packet CRC Operation

The CRC is appended to a packet in one of two ways. For a packet whose length, exclusive of CRC, is 80 bytes or less, a single CRC is appended at the end of the logical fields. For packets whose length, exclusive of CRC, is greater than 80 bytes, a CRC is added after the first 80 bytes and a second CRC is appended at the end of the logical layer fields.

The second CRC value is a continuation of the first. The first CRC is included in the running calculation, meaning that the running CRC value is not reinitialized after it is inserted after the first 80 bytes of the packet. This allows intervening devices to regard the embedded CRC value as two bytes of packet payload for CRC checking purposes. If the CRC appended to the end of the logical layer fields does not cause the end of the resulting packet to align to a 32-bit boundary, a two byte pad of all logic 0s is postpended to the packet. The pad of logic 0s allows the CRC check to always be done at the 32-bit boundary. A corrupt pad may or may not cause a CRC error to be detected, depending upon the implementation.

The early CRC value can be used by the receiving processing element to validate the header of a large packet and start processing the data before the entire packet has been received, freeing up resources earlier and reducing transaction completion latency.

NOTE:

While the embedded CRC value can be used by a processing element to start processing the data within a packet before receiving the entire packet, it is possible that upon reception of the end of the packet the final CRC value for the packet is incorrect. This would result in a processing element that has processed data that may have been corrupted. Outside of the error recovery mechanism described in Section 5.11.2, the RapidIO Interconnect Specification does not address the occurrence of such situations nor does it suggest a means by which a processing element would handle such situations. Instead, the mechanism for handling this situation is left to be addressed by the device manufacturers for devices that implement the functionality of

early processing of packet data.

Figure 2-4 is an example of an unpadded packet of length less than or equal to 80 bytes.

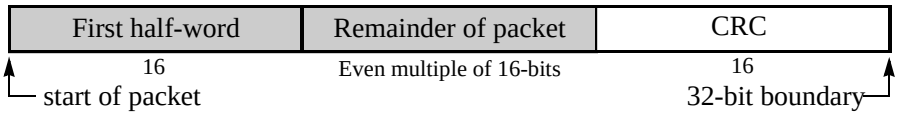


Figure 2-4. Unpadded Packet of Length 80 Bytes or Less

Figure 2-5 is an example of a padded packet of length less than or equal to 80 bytes.

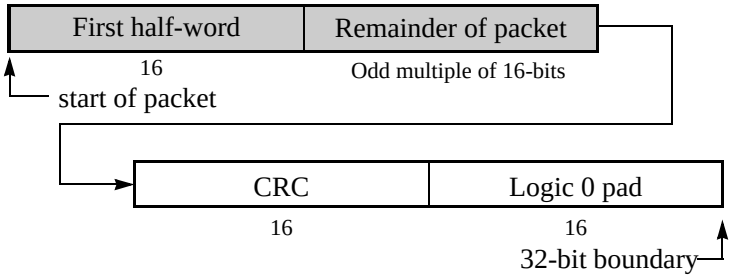


Figure 2-5. Padded Packet of Length 80 Bytes or Less

Figure 2-6 is an example of an unpadded packet of length greater than 80 bytes.

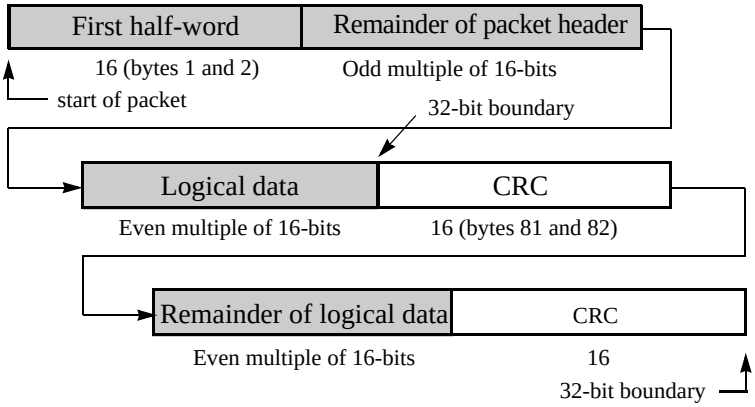


Figure 2-6. Unpadded Packet of Length Greater than 80 Bytes

Figure 2-7 is an example of a padded packet of length greater than 80 bytes.

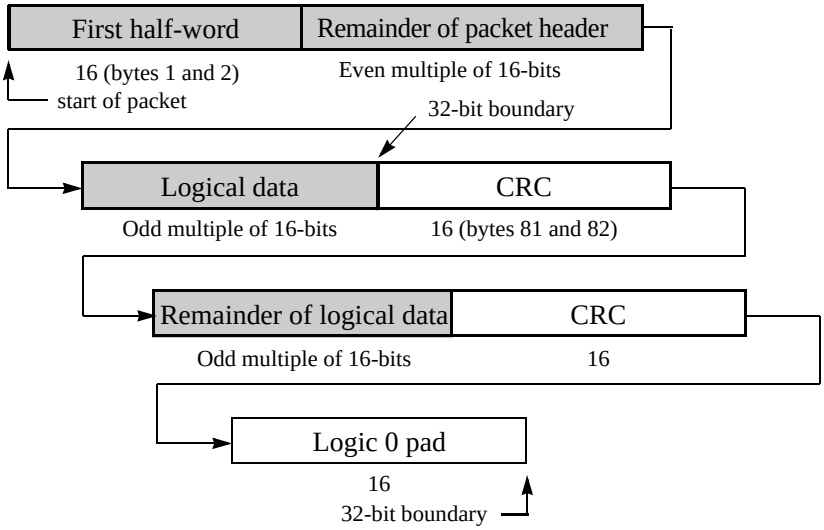


Figure 2-7. Padded Packet of Length Greater than 80 Bytes

2.4.2 16-Bit Packet CRC Code

The ITU polynomial $X^{16}+X^{12}+X^5+1$ shall be used to generate the 16-bit CRC for packets. The value of the CRC shall be initialized to 0xFFFF (all logic 1s) at the beginning of each packet. For the CRC calculation, the uncovered six bits are treated as logic 0s. As an example, a 16-bit wide parallel calculation is described in the equations in Table 2-2. Equivalent implementations of other widths can be employed.

Table 2-2. Parallel CRC Intermediate Value Equations

Check Bit	e00	e01	e02	e03	e04	e05	e06	e07	e08	e09	e10	e11	e12	e13	e14	e15
C00					x	x			x				x			
C01						x	x			x				x		
C02							x	x			x				x	
C03	x							x	x			x				x
C04	x	x			x	x				x						
C05		x	x			x	x				x					
C06	x		x	x			x	x				x				
C07	x	x		x	x			x	x				x			
C08	x	x	x		x	x			x	x				x		
C09		x	x	x		x	x			x	x				x	
C10			x	x	x		x	x			x	x				x
C11	x			x				x				x				
C12	x	x			x				x				x			

Table 2-2. Parallel CRC Intermediate Value Equations (Continued)

Check Bit	e00	e01	e02	e03	e04	e05	e06	e07	e08	e09	e10	e11	e12	e13	e14	e15
C13		x	x			x				x				x		
C14			x	x			x				x				x	
C15				x	x			x				x				x

where:

C00–C15 contents of the new check symbol

e00–e15 contents of the intermediate value symbol
e00 = d00 XOR c00
e01 = d01 XOR c01
through
e15 = d15 XOR c15

d00–d15 contents of the next 16 bits of the packet

c00–c15 contents of the previous check symbol

assuming the pipeline described in Figure 2-8

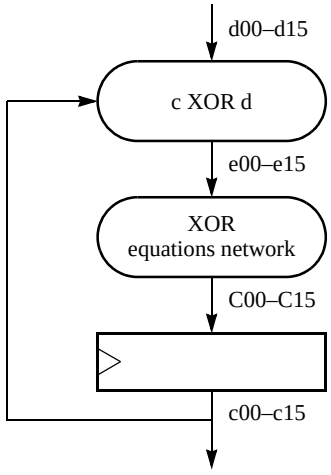


Figure 2-8. CRC Generation Pipeline

2.5 Maximum Packet Size

The maximum packet size permitted by the LP-Serial specification is 276 bytes. This includes all packet logical, transport, and physical layer header information, data payload, and required CRC bytes.

The maximum packet size of 276 bytes is achieved as shown below:

Table 2-3. Maximum Packet Size

Field	Size (bytes)	Layer	Notes
Header	2	Physical, Transport, Logical	
Source ID	2	Transport	
Destination ID	2	Transport	
Trans/wrsize	1	Logical	
srcTID	1	Logical	
Address	8	Logical	Includes Extended_address, Address, Wdptr, and Xambs
Payload	256	Logical	
CRC	4	Physical	Extra two CRC bytes for packets greater than 80 bytes
Total	276		

Blank page

Chapter 3 Control Symbols

3.1 Introduction

This chapter specifies RapidIO physical layer control symbols. Control symbols are the message elements used by ports connected by an LP-Serial link to manage all aspects of LP-Serial link operation. They are used for link maintenance, packet delimiting, packet acknowledgment, error reporting, and error recovery. For forward compatibility, control symbols received by a port with a reserved field encoding shall be ignored and not cause an error to be reported.

3.2 Control Symbol Field Definitions

This section describes the fields that make up the control symbols.

Table 3-1. Control Symbol Field Definitions

Field	Definition
stype0	Encoding for control symbols that make use of parameter0 and parameter1. Eight encodings are defined in Table 3-2.
parameter0	Used in conjunction with stype0 encodings. Reference Table 3-2 for the description of parameter0 encodings.
parameter1	Used in conjunction with stype0 encodings. Reference Table 3-2 for the description of parameter1 encodings.
stype1	Encoding for control symbols which make use of the cmd field. The eight encodings are defined in Table 3-6.
cmd	Used in conjunction with the stype1 field to define the link maintenance commands. Refer to Table 3-7 for the cmd field descriptions.
CRC	5-bit code used to detect transmission errors in control symbols. See Section 3.6 for details on the CRC error detection scheme.

3.3 Control Symbol Format

This section describes the general format of the LP-Serial control symbols. Figure 3-1 shows the control symbol format.

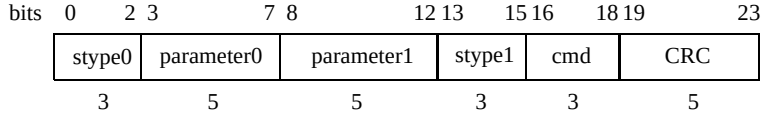


Figure 3-1. Packet-Retry Control Symbol Format

All control symbols follow the 24-bit control symbol format as detailed above. The

fields parameter0 and parameter1 are used by the functions encoded in the stype0 field. The cmd field is a modifier for the functions encoded in the stype1 field.

Control symbols can carry two functions, one encoded in the stype0 field and one encoded in the stype1 field. The functions encoded in stype0 are "status" functions that convey some type of status about the port transmitting the control symbol. The functions encoded in stype1 are requests to the receiving port or transmission delimiters.

A control symbol carrying one function is referred to using the name of the function it carries. A control symbol carrying two functions may be referred to using the name of either function that it carries. For example, a control symbol with stype0 set to packet-accepted and stype1 set to NOP is referred to a packet-accepted control symbol. A control symbol with stype0 set to packet-accepted and stype1 set to restart-from-retry is referred to as either a packet-accepted control symbol or a restart-from-retry control symbol depending on which name is appropriate for the context.

Control symbols are specified with the ability to carry two functions so that a packet acknowledgment and a packet delimiter can be carried in the same control symbol. Packet acknowledgment and packet delimiter control symbols constitute the vast majority of control symbol traffic on a busy link. Carrying an acknowledgment (or status) and a packet delimiter whenever possible in a single control symbol allows a significant reduction in link overhead traffic and an increase in the link bandwidth available for packet transmission.

3.4 Stype0 Control Symbols

The encoding and function of stype0 and the information carried in parameter0 and parameter1 for each stype0 encoding shall be as specified in Table 3-2.

Table 3-2. Stype0 Control Symbol Encoding

stype0	Function	Contents of		Reference
		Parameter0	Parameter1	
0b000	Packet-accepted	packet_ackID	buf_status	Section 3.4.1
0b001	Packet-retry	packet_ackID	buf_status	Section 3.4.2
0b010	Packet-not-accepted	packet_ackID	cause	Section 3.4.3
0b011	Reserved	-	-	-
0b100	Status	ackID_status	buf_status	Section 3.4.4
0b101	Reserved	-	-	-
0b110	Link-response	ackID_status	port_status	Section 3.4.5
0b111	Reserved	-	-	-

The status control symbol is the default stype0 encoding and is used when the control symbol does not convey another stype0 function. The following table

defines the parameters valid for stype0 control symbols.

Table 3-3. Stype0 Parameter Definitions

Parameter	Definition
packet_ackID	The ackID of the packet being acknowledged by an acknowledgment control symbol.
ackID_status	The value of ackID expected in the next packet the port receives. For example, a value of 0b00001 indicates the device is expecting to receive ackID 1.
buf_status	Specifies the number of maximum length packets that the port can accept without issuing a retry due to a lack of resources. The value of buf_status in a packet-accepted, packet-retry, or status control symbol is the number of maximum packets that can be accepted, inclusive of the effect of the packet being accepted or retried. Value 0-29: The encoding value specifies the number of new maximum sized packets the receiving device can receive. The value 0, for example, signifies that the downstream device has no available packet buffers (thus is not able to hold any new packets). Value 30: The value 30 signifies that the downstream device can receive 30 or more new maximum sized packets. Value 31: The downstream device can receive an undefined number of maximum sized packets, and relies on the retry protocol for flow control.

NOTE:

The following sections depict various control symbols. Since control symbols can contain one or two functions, shading in the figures is used to indicate which fields are applicable to that specific control symbol function.

3.4.1 Packet-Accepted Control Symbol

The packet-accepted control symbol indicates that the receiving device has taken responsibility for sending the packet to its final destination and that resources allocated by the sending device can be released. This control symbol shall be generated only after the entire packet has been received and found to be free of detectable errors. The packet-accepted control symbol format is displayed in Figure 3-2.

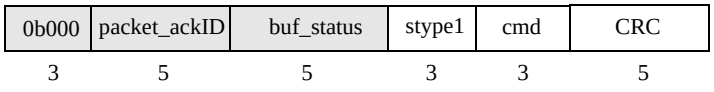


Figure 3-2. Packet-Accepted Control Symbol Format

3.4.2 Packet-Retry Control Symbol

A packet-retry control symbol indicates that the receiving device was not able to accept the packet due to some temporary resource conflict such as insufficient buffering and the sender should retransmit the packet. This control symbol format is displayed in Figure 3-3.

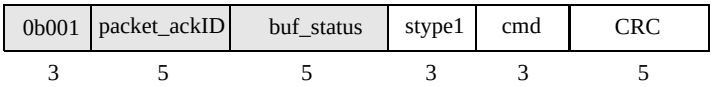


Figure 3-3. Packet-Retry Control Symbol Format

3.4.3 Packet-Not-Accepted Control Symbol

The packet-not-accepted control symbol is used to indicate to the sender of a packet why the packet was not accepted by the receiving port. As shown in Figure 3-4, the control symbol contains a cause field that indicates the reason for not accepting the packet and a packet_ackID field. If the receiving device is not able to specify the cause, or the cause is not one of defined options, the general error encoding shall be used.

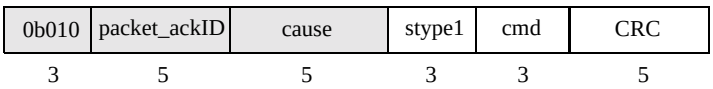


Figure 3-4. Packet-Not-Accepted Control Symbol Format

The cause field shall be used to display informational fields useful for debug. Table 3-4 displays the reasons a packet may not be accepted, indicated by the cause field.

Table 3-4. Cause Field Definition

Cause	Definition
0b00000	Reserved
0b00001	Received unexpected ackID on packet
0b00010	Received a control symbol with bad CRC
0b00011	Non-maintenance packet reception is stopped
0b00100	Received packet with bad CRC
0b00101	Received invalid character, or valid but illegal character
0b00110 - 0b11110	Reserved
0b11111	General error

3.4.4 Status Control Symbol

The status control symbol is the default stype0 encoding and is used when the control symbol does not convey another stype0 function. The status control symbol contains the ackID_status and the buf_status fields. The buf_status field indicates to the receiving port the number of maximum length packet buffers the sending port had available for packet reception at the time the control symbol was generated. The ackID_status field allows the receiving port to determine if it and the sending port are in sync with respect to the next ackID value the sending port expects to receive. The status control symbol format is shown in Figure 3-5 below.

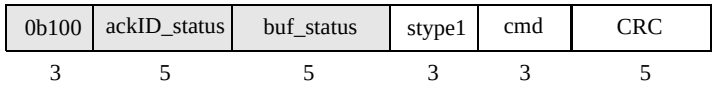


Figure 3-5. Status Control Symbol Format

3.4.5 Link-Response Control Symbol

The link-response control symbol is used by a device to respond to a link-request control symbol as described in the link maintenance protocol described in Section 5.5. The status reported in the status field is the status of the port at the time the associated input-status link-request control symbol was received.

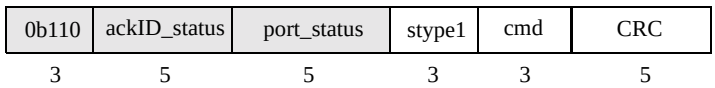


Figure 3-6. Link-Response Control Symbol Format

The port_status field of the link-response control symbol is defined in Table 3-5.

Table 3-5. Port_status Field Definitions

Port_status	Status	Description
0b00000		Reserved
0b00001		Reserved
0b00010	Error	The port has encountered an unrecoverable error and is unable to accept packets.
0b00011		Reserved
0b00100	Retry-stopped	The port has retried a packet and is waiting in the input retry-stopped state to be restarted.
0b00101	Error-stopped	The port has encountered a transmission error and is waiting in the input error-stopped state to be restarted.
0b00110 - 0b01111		Reserved
0b10000	OK	The port is accepting packets
0b10001 - 0b11111		Reserved

3.5 Stype1 Control Symbols

The encoding of stype1 and the function of the cmd field are defined in Table 3-6.

Table 3-6. Stype1 Control Symbol Encoding

stype1	stype1 Function	cmd	cmd Function	Packet Delimiter	Reference
0b000	Start-of-packet	0b000	-	yes	Section 3.5.1
0b001	Stomp	0b000	-	yes	Section 3.5.2
0b010	End-of-packet	0b000	-	yes	Section 3.5.3
0b011	Restart-from-retry	0b000	-	*	Section 3.5.4
0b100	Link-request	0b000 - 0b010	Reserved	*	-
		0b011	Reset-device		Section 3.5.5.1
		0b100	Input-status		Section 3.5.5.2
		0b101- 0b111	Reserved		-
0b101	Multicast-event	0b000	-	No	Section 3.5.6
0b110	Reserved	0b000	-	No	-
0b111	NOP (Ignore) **	0b000	-	No	-

Note: * denotes that restart-from-retry and link-request control symbols may only be packet delimiters if a packet is in progress.

Note: ** NOP (Ignore) is not defined as a control symbol, but is the default value when the control symbol does not convey another stype1 function.

NOTE:

The following sections depict various control symbols. Since control symbols can contain one or two functions, shading in the figures is used to indicate which fields are applicable to that specific control symbol function.

3.5.1 Start-of-Packet Control Symbol

The start-of-packet control symbol format is shown in Figure 3-7 below.

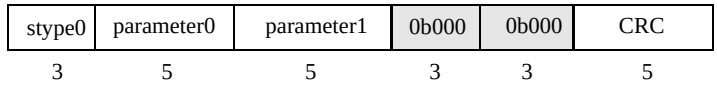


Figure 3-7. Start-of-Packet Control Symbol Format

3.5.2 Stomp Control Symbol

The stomp control symbol is used to cancel a partially transmitted packet. The protocol for packet cancelation is specified in Section 5.8. The stomp control symbol format is shown in Figure 3-8 below.

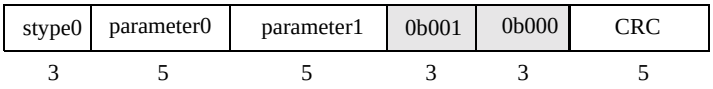


Figure 3-8. Stomp Control Symbol Format

3.5.3 End-of-Packet Control Symbol

The end-of-packet control symbol format is shown in Figure 3-9 below.

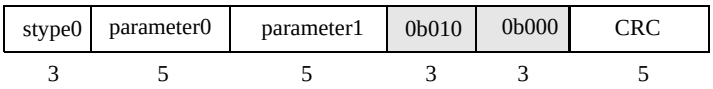


Figure 3-9. End-of-Packet Control Symbol Format

3.5.4 Restart-From-Retry Control Symbol

The restart-from-retry control symbol cancels a current packet and may also be transmitted on an idle link. This control symbol is used to mark the beginning of packet retransmission, so that the receiver knows when to start accepting packets after the receiver has requested a packet to be retried. The control symbol format is shown in Figure 3-10 below.

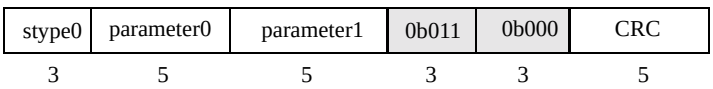


Figure 3-10. Restart-From-Retry Control Symbol Format

3.5.5 Link-Request Control Symbol

A link-request control symbol is used by a device to either issue a command to the connected device or request its input port status. A link-request control symbol always cancels a packet whose transmission is in progress and can also be sent between packets. Under error conditions, a link-request/input-status control symbol acts as a link-request/restart-from-error control symbol as described in Section 5.11.2.1, “Recoverable Errors.” This control symbol format is displayed in Figure 3-11.

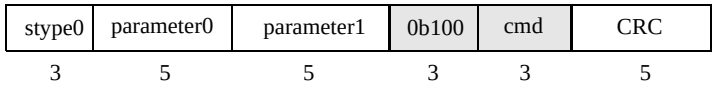


Figure 3-11. Link-Request Control Symbol Format

The cmd, or command, field of the link-request control symbol format is defined in Table 3-7 below.

Table 3-7. Cmd Field Definitions

cmd Encoding	Command Name	Description	Reference
0b000-0b010	-	Reserved	
0b011	Reset-device	Reset the receiving device	Section 3.5.5.1
0b100	Input-status	Return input port status; functions as a link request (restart-from-error) control symbol under error conditions	Section 3.5.5.2
0b101-0b111	-	Reserved	

3.5.5.1 Reset-Device Command

The reset-device command causes the receiving device to go through its reset or power-up sequence. All state machines and the configuration registers reset to the original power on states. The reset-device command does not generate a link-response control symbol.

Due to the undefined reliability of system designs it is necessary to put a safety lockout on the reset function of the link-request control symbol. A device receiving a reset-device command in a link-request control symbol shall not perform the reset function unless it has received four reset-device commands in a row without any other intervening packets or control symbols, except status control symbols. This will prevent spurious reset commands from inadvertently resetting a device.

When issuing a reset with four consecutive reset commands, care must be taken to account for all effects associated with the reset event. Consult *RapidIO Part 8: Error Management Extensions Specification* for more information.

3.5.5.2 Input-Status Command

The input-status command requests the receiving device to return the ackID value it expects to next receive from the sender on its input port and the current input port operational status for informational purposes. This command causes the receiver to flush its output port of all control symbols generated by packets received before the input-status command. Flushing the output port is implementation dependent and may result in either discarding the contents of the receive buffers or sending the control symbols on the link. The receiver then responds with a link-response control symbol.

3.5.6 Multicast-Event Control Symbol

The multicast-event control symbol differs from other control symbols in that it carries information not related to the link carrying the control symbol. The multicast-event control symbol allows the occurrence of a user-defined system event to be multicast throughout a system. Refer to Section 5.3.4 for more details on Multicast-Events.

The multicast-event control symbol format is shown in Figure 3-12 below.

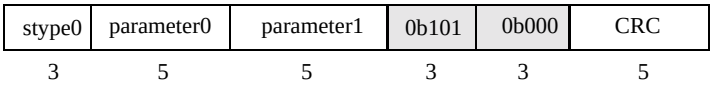


Figure 3-12. Multicast-Event Control Symbol Format

3.6 Control Symbol Protection

The 5-bit CRC shall be computed over control symbol bits 0 through 18 and provides 5-bit burst error detection for the entire 24-bit control symbol.

3.6.1 CRC-5 Code

The ITU polynomial $X^5+X^4+X^2+1$ shall be used to generate the 5-bit CRC for control symbols. The CRC check bits c0, c1, c2, c3, and c4 occupy the last 5 bits of a control symbol. It should be noted that the 5-bit CRC must be generated by each transmitter and verified by each receiver. Before the 5-bit CRC is computed, the CRC should be set to all 1's or 0b11111. In order to provide maximum implementation flexibility for all types of designs, a 20^{th} bit has been added. For all computations, the 20^{th} bit shall be the last bit applied and shall be set to a logic 0 (0b0).

3.6.2 CRC-5 Parallel Code Generation

Since it is often more efficient to implement a parallel CRC algorithm rather than a serial, examples of the equations for a complete, 19-bit single-stage parallel implementation is shown in shown in Table 3-8. Since only a single stage is used, the effect of both setting the initial CRC to all 1's (0b11111) and a 20^{th} bit set to logic 0 (0b0) have been included in the equations.

In Table 3-8, an "x" means that the data input should be an input to the Exclusive-OR necessary to compute that particular bit of the CRC. A "!x", means that bit 18 being applied to the CRC circuit must be inverted. Figure 3-13 shows the 19-bits that the CRC covers and how they should be applied to the circuit. As seen in Figure 3-13, bits are labeled with 0 on the left and 18 on the right. Bit 0, from the stype0 field, would apply to D0 in Table 3-8 and bit 18, from the cmd field, would apply to D18 in Table 3-8. Once completed, the 5-bit CRC is appended to the control symbol.

CRC Checksum	Control Symbol Data For CRC																	
Bits	**D 0**	**D 1**	**D 2**	**D 3**	**D 4**	**D 5**	**D 6**	**D 7**	**D 8**	**D 9**	**D 10**	**D 11**	**D 12**	**D 13**	**D 14**	**D 15**	**D 16**	**D 17**	**D 18**
4	x		x	x	x					x		x			x	x		x	x
3		x	x	x					x		x			x	x		x	x	!x
2		x		x	x			x				x	x	x	x		x		!x
1	x		x	x			x				x	x	x	x		x		x	!x
0	x	x		x	x	x					x		x			x	x		x

Table 3-8. Parallel CRC Equations

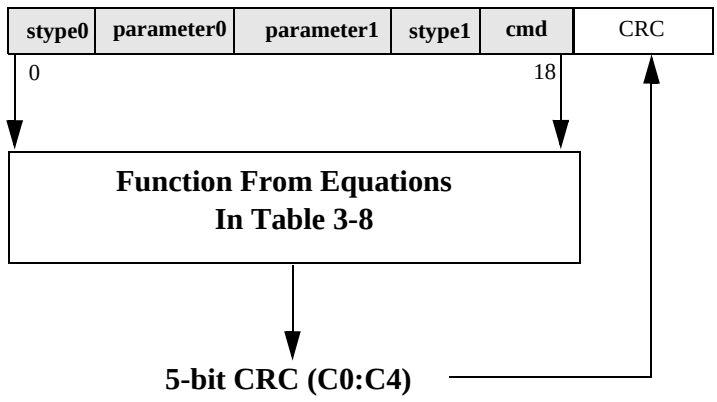


Figure 3-13. 5-bit CRC Implementation

Chapter 4 PCS and PMA Layers

4.1 Introduction

This chapter specifies the functions provided by the Physical Coding Sublayer (PCS) and Physical Media Attachment (PMA) sublayer. (The PCS and PMA terminology is adopted from IEEE 802.3). The topics include 8B/10B encoding, character representation, serialization of the data stream, code-groups, columns, link transmission rules, idle sequences, and link initialization.

The concept of lanes is used to describe the width of a LP-Serial link. A lane is defined as one unidirectional differential pair in each direction. RapidIO LP-Serial defines two link widths. The 1x LP-Serial link is a one-lane link and the 4x LP-Serial link is a 4-lane link. Wider links are possible, but are left for future work.

4.2 PCS Layer Functions

The Physical Coding Sublayer (PCS) function is responsible for idle sequence generation, lane striping, and encoding for transmission and decoding, lane alignment, and destriping on reception. The PCS uses an 8B/10B encoding for transmission over the link.

The PCS layer also provides mechanisms for determining the operational mode of the port as 4-lane or 1-lane operation, and means to detect link states. It provides for clock difference tolerance between the sender and receiver without requiring flow control.

The PCS layer performs the following transmit functions:

- Dequeues packets and delimited control symbols awaiting transmission as a character stream.
- Stripes the transmit character stream across the available lanes.
- Generates the idle sequence and inserts it into the transmit character stream for each lane when no packets or delimited control symbols are available for transmission.
- Encodes the character stream of each lane independently into 10-bit parallel code-groups.
- Passes the resulting 10-bit parallel code-groups to the PMA.

The PCS layer performs the following receive functions:

- Decodes the received stream of 10-bit parallel code-groups for each lane independently into characters.
- Marks characters decoded from invalid code-groups as invalid.
- If the link is using more than one lane, aligns the character streams to eliminate the skew between the lanes and reassembles (destripes) the character stream from each lane into a single character stream.
- Delivers the decoded character stream of packets and delimited control symbols to the higher layers.

4.3 PMA Layer Functions

The PMA (Physical Medium Attachment) function is responsible for serializing 10-bit parallel code-groups to/from a serial bitstream on a lane-by-lane basis. Upon receiving data, the PMA function provides alignment of the received bitstream to 10-bit code-group boundaries, independently on a lane-by-lane basis. It then provides a continuous stream of 10-bit code-groups to the PCS, one stream for each lane. The 10-bit code-groups are not observable by layers higher than the PCS.

4.4 Definitions

Definitions of terms used in this specification are provided below.

Byte: An 8-bit unit of information. Each bit of a byte has the value 0 or 1.

Character: A 9-bit entity comprised of an information byte and a control bit that indicates whether the information byte contains data or control information. The control bit has the value D or K indicating that the information byte contains respectively data or control information.

D-character: A character whose control bit has the value “D”.

K-character: A character whose control bit has the value “K”. Also referred to as a special character.

Code-group: A 10-bit entity that is the result of 8B/10B encoding a character.

Column: A group of four characters that are transmitted simultaneously on a 4x (4 lane) link.

Comma: A 7-bit pattern, unique to certain 8B/10B special code-groups, that is used by a receiver to determine code-group boundaries. See more in “Section 4.5.7.4, Sync (/K/)” on page 50 and Table 4-2, on page 48.

Idle sequence: The sequence of characters (code-groups after encoding) that is transmitted when a packet or control symbol is not being transmitted. The idle sequence allows the receiver to maintain bit synchronization and code-group alignment in between packets and control symbols.

Lane Alignment: The process of eliminating the skew between the lanes of a 4-lane LP-Serial link such that the characters transmitted as a column by the sender are output by

the alignment process of receiver as a column. Without lane alignment, the characters transmitted as a column might be scattered across several columns output by the receiver. The alignment process uses the columns of “A” special characters transmitted as part of the idle sequence.

Striping: The method used on a 4x link to send data across four lanes simultaneously. The character stream is *striped* across the lanes, on a character-by-character basis, starting with lane 0, to lane 1, to lane 2, to lane3, and wrapping back with the 5th character to lane 0.

4.5 8B/10B Transmission Code

The 8B/10B transmission code used by the PCS encodes 9-bit characters (8 bits of information and a control bit) into 10-bit code-groups for transmission and reverses the process on reception. Encodings are defined for 256 data characters and 12 special (control) characters.

The code-groups used by the code have either an equal number of ones and zeros (balanced) or the number of ones differs from the number of zeros by two (unbalanced). This selection of code-groups guarantees a minimum of two transitions, 0 to 1 or 1 to 0, within each code-group and it also eases the task of maintaining balance. Characters are encoded into either a single balanced code-group or a pair of unbalanced code-groups. The members of each code-group pair are the logical complement of each other. This allows the encoder, when selecting an unbalanced code-group, to select a code-group unbalanced toward ones or unbalanced toward zeros, depending on which is required to maintain the 0/1 balance of the encoder output code-group stream.

The 8B/10B code has the following properties.

- Sufficient bit transition density (3 to 8 transitions per code-group) to allow clock recovery by the receiver.
- Special code-groups that are used for establishing the receiver synchronization to the 10-bit code-group boundaries, delimiting control symbols and maintaining receiver bit and code-group boundary synchronization.
- Balanced. (can be AC coupled)
- Detection of single and some multiple-bit errors.

4.5.1 Character and Code-Group Notation

The description of 8B/10B encoding and decoding uses the following notation for characters, code-group and their bits.

The information bits ([0-7]) of an unencoded character are denoted with the letters “A” through “H” where the letter “H” denotes the most significant information bit (RapidIO bit 0) and the letter “A” denotes the least significant information bit (RapidIO bit 7). This is shown in Figure 4-1.

Each data character has a representation of the form Dx.y where x is the decimal value of the least significant 5 information bits EDCBA, and y is the decimal value of the most significant 3 information bits HGF as shown in Figure 4-1. Each special character has a similar representation of the form Kx.y.

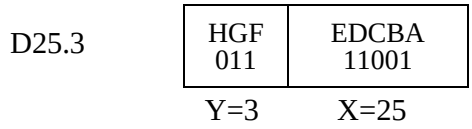


Figure 4-1. Character Notation Example (D25.3)

The output of the 8B/10B encoding process is a 10-bit code-group. The bits of a code-group are denoted with the letters “a” through “j”. The bits of a code-group are all of equal significance, there is no most significant or least significant bit. The ordering of the code-group bits is shown in Figure 4-2.

The code-groups corresponding to the data character Dx.y is denoted by /Dx.y/. The code-groups corresponding to the special character Kx.y is denoted by /Kx.y/.

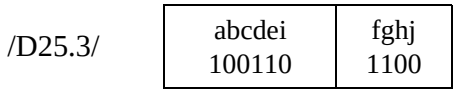


Figure 4-2. Code-Group Notation Example (/D25.3/)

4.5.2 Running Disparity

The 8B/10B encoding and decoding functions use a binary variable called running disparity. The variable can have a value of either positive (RD+) or negative (RD-). The encoder and decoder each have a running disparity variable for each lane which are all independent of each other.

The primary use of running disparity in the encoding process is to keep track of whether the decoder has output more ones or more zeros. The current value of encoder running disparity is used to select the which unbalanced code-group will be used when the encoding for a character requires a choice between two unbalanced code-groups.

The primary use of running disparity in the decoding process is to detect errors. Given a value of decoder running disparity, only (256 + 12) = 268 of the 1024 possible code-group values have defined decodings. The remaining 756 possible code-group values have no defined decoding and represent errors, either in that code-group or in an earlier code-group.

4.5.3 Running Disparity Rules

After power-up and before the port is operational, both the transmitter (encoder) and receiver (decoder) must establish current values of running disparity.

The transmitter shall use a negative value as the initial value for the running disparity for each lane.

The receiver may use either a negative or positive initial value of running disparity for each lane.

The following algorithm shall be used for calculating the running disparity for each lane. In the encoder, the algorithm operates on the code-group that has just been generated by the encoder. In the receiver, the algorithm operates on the received code-group that has just been decoded by the decoder.

Each code-group is divided to two sub-blocks as shown in Figure 4-2, where the first six bits (abcdei) form one sub-block (6-bit sub-block) and the second four bits (fghj) form a second sub-block (4-bit sub-block). Running disparity at the beginning of the 6-bit sub-block is the running disparity at the end of the previous code-group. Running disparity at the beginning of the 4-bit sub-block is the running disparity at the end of the 6-bit sub-block. Running disparity at the end of the code-group is the running disparity at the end of the 4-bit sub-block.

The sub-block running disparity shall be calculated as follows:

1. The running disparity is positive at the end of any sub-block if the sub-block contains more 1s than 0s. It is also positive at the end of a 4-bit sub-block if the sub-block has the value 0b0011 and at the end of a 6-bit sub-block if the sub-block has the value 0b000111.
2. The running disparity is negative at the end of any sub-block if the sub-block contains more 0s than 1s. It is also negative at the end of a 4-bit sub-block if the sub-block has the value 0b1100 and at the end of a 6-bit sub-block if the sub-block has the value 0b111000.
3. In all other cases, the value of the running disparity at the end of the sub-block is running disparity at the beginning of the sub-block (the running disparity is unchanged).

4.5.4 8B/10B Encoding

The 8B/10B encoding function encodes 9-bit characters into 10-bit code-groups.

The encodings for the 256 data characters (Dx.y) are specified in Table 4-1. The encodings for the 12 special characters (Kx.y) are specified in Table 4-2. Both tables have two columns of encodings, one marked RD- and one marked RD+. When encoding a character, the code-group in the RD- column is selected if the current value of encoder running disparity is negative and the code-group in the RD+ column is selected if the current value of encoder running disparity is positive.

Data characters (Dx.y) shall be encoded according to Table 4-1 and the current value of encoder running disparity. Special characters (Kx.y) shall be encoded according to Table 4-2 and the current value of encoder running disparity. After each character is encoded, the resulting code-group shall be used by the encoder to update the running disparity according to the rules in Section 4.5.3, “Running Disparity Rules.

4.5.5 Transmission Order

The parallel 10-bit code-group output of the encoder shall be serialized and transmitted with bit “a” transmitted first and a bit ordering of “abcdeifghj”. This is shown in Figure 4-3.

Figure 4-3 gives an overview of a character passing through the encoding, serializing, transmission, deserializing, and decoding processes. The left side of the figure shows the transmit process of encoding a character stream using 8B/10B encoding and the 10-bit serialization. The right side shows the reverse process of the receiver deserializing and using 8B/10B decoding on the received code-groups.

The dotted line shows the functional separation between the PCS layer, that provides 10-bit code-groups, and the PMA layer that serializes the code-groups.

The drawing also shows on the receive side the bits of a special character containing the comma pattern that is used by the receiver to establish 10-bit code-boundary synchronization.

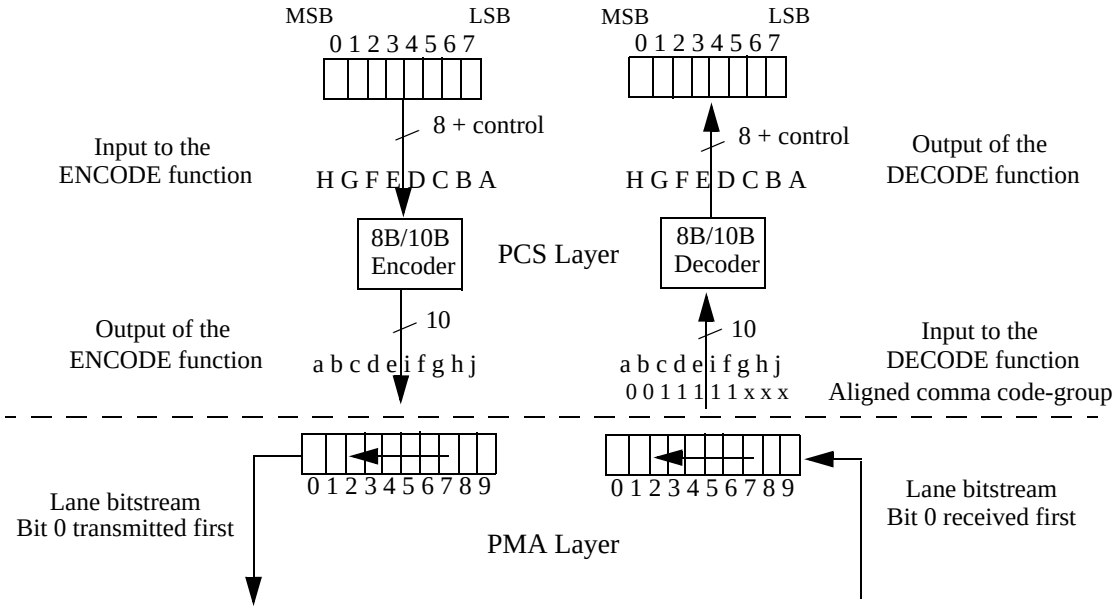


Figure 4-3. Lane Encoding, Serialization, Deserialization, and Decoding Process

4.5.6 8B/10B Decoding

The 8B/10B decoding function decodes received 10-bit code-groups into 9-bit characters, detects received code-groups that have no defined decoding and marks the resulting characters in the output stream of the decode as invalid character (INVALID).

The decoding function uses Table 4-1, Table 4-2 and the current value of the decoder running disparity. To decode a received code-group, the decoder shall select the RD- column of Table 4-1 and Table 4-2 if the current value of the decoder running disparity is negative or shall select the RD+ column if the value is positive. The decoder shall then compare the received code-group with the code-groups in the selected column of both tables. If a match is found, the code-group is decoded to the associated character. If no match is found, the code-group is decoded to a character that is flagged in some manner as invalid. After each code-group is decoded, the decoded code-group shall be used by the decoder to update the decoder running disparity according to the rules in Section 4.5.3, “Running Disparity Rules.

Table 4-1. Data Character Encodings

Character Name	Character Value (hex)	Character Bits HGF EDCBA	Current RD –	Current RD +
			abcdei fghj	abcdei fghj
D0.0	00	000 00000	100111 0100	011000 1011
D1.0	01	000 00001	011101 0100	100010 1011
D2.0	02	000 00010	101101 0100	010010 1011
D3.0	03	000 00011	110001 1011	110001 0100
D4.0	04	000 00100	110101 0100	001010 1011
D5.0	05	000 00101	101001 1011	101001 0100
D6.0	06	000 00110	011001 1011	011001 0100
D7.0	07	000 00111	111000 1011	000111 0100
D8.0	08	000 01000	111001 0100	000110 1011
D9.0	09	000 01001	100101 1011	100101 0100
D10.0	0A	000 01010	010101 1011	010101 0100
D11.0	0B	000 01011	110100 1011	110100 0100
D12.0	0C	000 01100	001101 1011	001101 0100
D13.0	0D	000 01101	101100 1011	101100 0100
D14.0	0E	000 01110	011100 1011	011100 0100
D15.0	0F	000 01111	010111 0100	101000 1011
D16.0	10	000 10000	011011 0100	100100 1011
D17.0	11	000 10001	100011 1011	100011 0100
D18.0	12	000 10010	010011 1011	010011 0100
D19.0	13	000 10011	110010 1011	110010 0100
D20.0	14	000 10100	001011 1011	001011 0100

Table 4-1. Data Character Encodings

Character Name	Character Value (hex)	Character Bits HGF EDCBA	Current RD –	Current RD +
			abcdei fghj	abcdei fghj
D21.0	15	000 10101	101010 1011	101010 0100
D22.0	16	000 10110	011010 1011	011010 0100
D23.0	17	000 10111	111010 0100	000101 1011
D24.0	18	000 11000	110011 0100	001100 1011
D25.0	19	000 11001	100110 1011	100110 0100
D26.0	1A	000 11010	010110 1011	010110 0100
D27.0	1B	000 11011	110110 0100	001001 1011
D28.0	1C	000 11100	001110 1011	001110 0100
D29.0	1D	000 11101	101110 0100	010001 1011
D30.0	1E	000 11110	011110 0100	100001 1011
D31.0	1F	000 11111	101011 0100	010100 1011
D0.1	20	001 00000	100111 1001	011000 1001
D1.1	21	001 00001	011101 1001	100010 1001
D2.1	22	001 00010	101101 1001	010010 1001
D3.1	23	001 00011	110001 1001	110001 1001
D4.1	24	001 00100	110101 1001	001010 1001
D5.1	25	001 00101	101001 1001	101001 1001
D6.1	26	001 00110	011001 1001	011001 1001
D7.1	27	001 00111	111000 1001	000111 1001
D8.1	28	001 01000	111001 1001	000110 1001
D9.1	29	001 01001	100101 1001	100101 1001
D10.1	2A	001 01010	010101 1001	010101 1001
D11.1	2B	001 01011	110100 1001	110100 1001
D12.1	2C	001 01100	001101 1001	001101 1001
D13.1	2D	001 01101	101100 1001	101100 1001
D14.1	2E	001 01110	011100 1001	011100 1001
D15.1	2F	001 01111	010111 1001	101000 1001
D16.1	30	001 10000	011011 1001	100100 1001
D17.1	31	001 10001	100011 1001	100011 1001
D18.1	32	001 10010	010011 1001	010011 1001
D19.1	33	001 10011	110010 1001	110010 1001
D20.1	34	001 10100	001011 1001	001011 1001
D21.1	35	001 10101	101010 1001	101010 1001
D22.1	36	001 10110	011010 1001	011010 1001
D23.1	37	001 10111	111010 1001	000101 1001
D24.1	38	001 11000	110011 1001	001100 1001

Table 4-1. Data Character Encodings

Character Name	Character Value (hex)	Character Bits HGF EDCBA	Current RD –	Current RD +
			abcdei fghj	abcdei fghj
D25.1	39	001 11001	100110 1001	100110 1001
D26.1	3A	001 11010	010110 1001	010110 1001
D27.1	3B	001 11011	110110 1001	001001 1001
D28.1	3C	001 11100	001110 1001	001110 1001
D29.1	3D	001 11101	101110 1001	010001 1001
D30.1	3E	001 11110	011110 1001	100001 1001
D31.1	3F	001 11111	101011 1001	010100 1001
D0.2	40	010 00000	100111 0101	011000 0101
D1.2	41	010 00001	011101 0101	100010 0101
D2.2	42	010 00010	101101 0101	010010 0101
D3.2	43	010 00011	110001 0101	110001 0101
D4.2	44	010 00100	110101 0101	001010 0101
D5.2	45	010 00101	101001 0101	101001 0101
D6.2	46	010 00110	011001 0101	011001 0101
D7.2	47	010 00111	111000 0101	000111 0101
D8.2	48	010 01000	111001 0101	000110 0101
D9.2	49	010 01001	100101 0101	100101 0101
D10.2	4A	010 01010	010101 0101	010101 0101
D11.2	4B	010 01011	110100 0101	110100 0101
D12.2	4C	010 01100	001101 0101	001101 0101
D13.2	4D	010 01101	101100 0101	101100 0101
D14.2	4E	010 01110	011100 0101	011100 0101
D15.2	4F	010 01111	010111 0101	101000 0101
D16.2	50	010 10000	011011 0101	100100 0101
D17.2	51	010 10001	100011 0101	100011 0101
D18.2	52	010 10010	010011 0101	010011 0101
D19.2	53	010 10011	110010 0101	110010 0101
D20.2	54	010 10100	001011 0101	001011 0101
D21.2	55	010 10101	101010 0101	101010 0101
D22.2	56	010 10110	011010 0101	011010 0101
D23.2	57	010 10111	111010 0101	000101 0101
D24.2	58	010 11000	110011 0101	001100 0101
D25.2	59	010 11001	100110 0101	100110 0101
D26.2	5A	010 11010	010110 0101	010110 0101
D27.2	5B	010 11011	110110 0101	001001 0101
D28.2	5C	010 11100	001110 0101	001110 0101

Table 4-1. Data Character Encodings

Character Name	Character Value (hex)	Character Bits HGF EDCBA	Current RD –	Current RD +
			abcdei fghj	abcdei fghj
D29.2	5D	010 11101	101110 0101	010001 0101
D30.2	5E	010 11110	011110 0101	100001 0101
D31.2	5F	010 11111	101011 0101	010100 0101
D0.3	60	011 00000	100111 0011	011000 1100
D1.3	61	011 00001	011101 0011	100010 1100
D2.3	62	011 00010	101101 0011	010010 1100
D3.3	63	011 00011	110001 1100	110001 0011
D4.3	64	011 00100	110101 0011	001010 1100
D5.3	65	011 00101	101001 1100	101001 0011
D6.3	66	011 00110	011001 1100	011001 0011
D7.3	67	011 00111	111000 1100	000111 0011
D8.3	68	011 01000	111001 0011	000110 1100
D9.3	69	011 01001	100101 1100	100101 0011
D10.3	6A	011 01010	010101 1100	010101 0011
D11.3	6B	011 01011	110100 1100	110100 0011
D12.3	6C	011 01100	001101 1100	001101 0011
D13.3	6D	011 01101	101100 1100	101100 0011
D14.3	6E	011 01110	011100 1100	011100 0011
D15.3	6F	011 01111	010111 0011	101000 1100
D16.3	70	011 10000	011011 0011	100100 1100
D17.3	71	011 10001	100011 1100	100011 0011
D18.3	72	011 10010	010011 1100	010011 0011
D19.3	73	011 10011	110010 1100	110010 0011
D20.3	74	011 10100	001011 1100	001011 0011
D21.3	75	011 10101	101010 1100	101010 0011
D22.3	76	011 10110	011010 1100	011010 0011
D23.3	77	011 10111	111010 0011	000101 1100
D24.3	78	011 11000	110011 0011	001100 1100
D25.3	79	011 11001	100110 1100	100110 0011
D26.3	7A	011 11010	010110 1100	010110 0011
D27.3	7B	011 11011	110110 0011	001001 1100
D28.3	7C	011 11100	001110 1100	001110 0011
D29.3	7D	011 11101	101110 0011	010001 1100
D30.3	7E	011 11110	011110 0011	100001 1100
D31.3	7F	011 11111	101011 0011	010100 1100
D0.4	80	100 00000	100111 0010	011000 1101

Table 4-1. Data Character Encodings

Character Name	Character Value (hex)	Character Bits HGF EDCBA	Current RD –	Current RD +
			abcdei fghj	abcdei fghj
D1.4	81	100 00001	011101 0010	100010 1101
D2.4	82	100 00010	101101 0010	010010 1101
D3.4	83	100 00011	110001 1101	110001 0010
D4.4	84	100 00100	110101 0010	001010 1101
D5.4	85	100 00101	101001 1101	101001 0010
D6.4	86	100 00110	011001 1101	011001 0010
D7.4	87	100 00111	111000 1101	000111 0010
D8.4	88	100 01000	111001 0010	000110 1101
D9.4	89	100 01001	100101 1101	100101 0010
D10.4	8A	100 01010	010101 1101	010101 0010
D11.4	8B	100 01011	110100 1101	110100 0010
D12.4	8C	100 01100	001101 1101	001101 0010
D13.4	8D	100 01101	101100 1101	101100 0010
D14.4	8E	100 01110	011100 1101	011100 0010
D15.4	8F	100 01111	010111 0010	101000 1101
D16.4	90	100 10000	011011 0010	100100 1101
D17.4	91	100 10001	100011 1101	100011 0010
D18.4	92	100 10010	010011 1101	010011 0010
D19.4	93	100 10011	110010 1101	110010 0010
D20.4	94	100 10100	001011 1101	001011 0010
D21.4	95	100 10101	101010 1101	101010 0010
D22.4	96	100 10110	011010 1101	011010 0010
D23.4	97	100 10111	111010 0010	000101 1101
D24.4	98	100 11000	110011 0010	001100 1101
D25.4	99	100 11001	100110 1101	100110 0010
D26.4	9A	100 11010	010110 1101	010110 0010
D27.4	9B	100 11011	110110 0010	001001 1101
D28.4	9C	100 11100	001110 1101	001110 0010
D29.4	9D	100 11101	101110 0010	010001 1101
D30.4	9E	100 11110	011110 0010	100001 1101
D31.4	9F	100 11111	101011 0010	010100 1101
D0.5	A0	101 00000	100111 1010	011000 1010
D1.5	A1	101 00001	011101 1010	100010 1010
D2.5	A2	101 00010	101101 1010	010010 1010
D3.5	A3	101 00011	110001 1010	110001 1010
D4.5	A4	101 00100	110101 1010	001010 1010

Table 4-1. Data Character Encodings

Character Name	Character Value (hex)	Character Bits HGF EDCBA	Current RD –	Current RD +
			abcdei fghj	abcdei fghj
D5.5	A5	101 00101	101001 1010	101001 1010
D6.5	A6	101 00110	011001 1010	011001 1010
D7.5	A7	101 00111	111000 1010	000111 1010
D8.5	A8	101 01000	111001 1010	000110 1010
D9.5	A9	101 01001	100101 1010	100101 1010
D10.5	AA	101 01010	010101 1010	010101 1010
D11.5	AB	101 01011	110100 1010	110100 1010
D12.5	AC	101 01100	001101 1010	001101 1010
D13.5	AD	101 01101	101100 1010	101100 1010
D14.5	AE	101 01110	011100 1010	011100 1010
D15.5	AF	101 01111	010111 1010	101000 1010
D16.5	B0	101 10000	011011 1010	100100 1010
D17.5	B1	101 10001	100011 1010	100011 1010
D18.5	B2	101 10010	010011 1010	010011 1010
D19.5	B3	101 10011	110010 1010	110010 1010
D20.5	B4	101 10100	001011 1010	001011 1010
D21.5	B5	101 10101	101010 1010	101010 1010
D22.5	B6	101 10110	011010 1010	011010 1010
D23.5	B7	101 10111	111010 1010	000101 1010
D24.5	B8	101 11000	110011 1010	001100 1010
D25.5	B9	101 11001	100110 1010	100110 1010
D26.5	BA	101 11010	010110 1010	010110 1010
D27.5	BB	101 11011	110110 1010	001001 1010
D28.5	BC	101 11100	001110 1010	001110 1010
D29.5	BD	101 11101	101110 1010	010001 1010
D30.5	BE	101 11110	011110 1010	100001 1010
D31.5	BF	101 11111	101011 1010	010100 1010
D0.6	C0	110 00000	100111 0110	011000 0110
D1.6	C1	110 00001	011101 0110	100010 0110
D2.6	C2	110 00010	101101 0110	010010 0110
D3.6	C3	110 00011	110001 0110	110001 0110
D4.6	C4	110 00100	110101 0110	001010 0110
D5.6	C5	110 00101	101001 0110	101001 0110
D6.6	C6	110 00110	011001 0110	011001 0110
D7.6	C7	110 00111	111000 0110	000111 0110
D8.6	C8	110 01000	111001 0110	000110 0110

Table 4-1. Data Character Encodings

Character Name	Character Value (hex)	Character Bits HGF EDCBA	Current RD –	Current RD +
			abcdei fghj	abcdei fghj
D9.6	C9	110 01001	100101 0110	100101 0110
D10.6	CA	110 01010	010101 0110	010101 0110
D11.6	CB	110 01011	110100 0110	110100 0110
D12.6	CC	110 01100	001101 0110	001101 0110
D13.6	CD	110 01101	101100 0110	101100 0110
D14.6	CE	110 01110	011100 0110	011100 0110
D15.6	CF	110 01111	010111 0110	101000 0110
D16.6	D0	110 10000	011011 0110	100100 0110
D17.6	D1	110 10001	100011 0110	100011 0110
D18.6	D2	110 10010	010011 0110	010011 0110
D19.6	D3	110 10011	110010 0110	110010 0110
D20.6	D4	110 10100	001011 0110	001011 0110
D21.6	D5	110 10101	101010 0110	101010 0110
D22.6	D6	110 10110	011010 0110	011010 0110
D23.6	D7	110 10111	111010 0110	000101 0110
D24.6	D8	110 11000	110011 0110	001100 0110
D25.6	D9	110 11001	100110 0110	100110 0110
D26.6	DA	110 11010	010110 0110	010110 0110
D27.6	DB	110 11011	110110 0110	001001 0110
D28.6	DC	110 11100	001110 0110	001110 0110
D29.6	DD	110 11101	101110 0110	010001 0110
D30.6	DE	110 11110	011110 0110	100001 0110
D31.6	DF	110 11111	101011 0110	010100 0110
D0.7	E0	111 00000	100111 0001	011000 1110
D1.7	E1	111 00001	011101 0001	100010 1110
D2.7	E2	111 00010	101101 0001	010010 1110
D3.7	E3	111 00011	110001 1110	110001 0001
D4.7	E4	111 00100	110101 0001	001010 1110
D5.7	E5	111 00101	101001 1110	101001 0001
D6.7	E6	111 00110	011001 1110	011001 0001
D7.7	E7	111 00111	111000 1110	000111 0001
D8.7	E8	111 01000	111001 0001	000110 1110
D9.7	E9	111 01001	100101 1110	100101 0001
D10.7	EA	111 01010	010101 1110	010101 0001
D11.7	EB	111 01011	110100 1110	110100 1000
D12.7	EC	111 01100	001101 1110	001101 0001

Table 4-1. Data Character Encodings

Character Name	Character Value (hex)	Character Bits HGF EDCBA	Current RD –	Current RD +
			abcdei fghj	abcdei fghj
D13.7	ED	111 01101	101100 1110	101100 1000
D14.7	EE	111 01110	011100 1110	011100 1000
D15.7	EF	111 01111	010111 0001	101000 1110
D16.7	F0	111 10000	011011 0001	100100 1110
D17.7	F1	111 10001	100011 0111	100011 0001
D18.7	F2	111 10010	010011 0111	010011 0001
D19.7	F3	111 10011	110010 1110	110010 0001
D20.7	F4	111 10100	001011 0111	001011 0001
D21.7	F5	111 10101	101010 1110	101010 0001
D22.7	F6	111 10110	011010 1110	011010 0001
D23.7	F7	111 10111	111010 0001	000101 1110
D24.7	F8	111 11000	110011 0001	001100 1110
D25.7	F9	111 11001	100110 1110	100110 0001
D26.7	FA	111 11010	010110 1110	010110 0001
D27.7	FB	111 11011	110110 0001	001001 1110
D28.7	FC	111 11100	001110 1110	001110 0001
D29.7	FD	111 11101	101110 0001	010001 1110
D30.7	FE	111 11110	011110 0001	100001 1110
D31.7	FF	111 11111	101011 0001	010100 1110

Table 4-2. Special Character Encodings

Character Name	Character Value (hex)	Character Bits HGF EDCBA	Current RD –	Current RD +	Notes
			abcdei fghj	abcdei fghj	
K28.0	1C	000 11100	001111 0100	110000 1011	
K28.1	3C	001 11100	001111 1001	110000 0110	1,2
K28.2	5C	010 11100	001111 0101	110000 1010	1
K28.3	7C	011 11100	001111 0011	110000 1100	
K28.4	9C	100 11100	001111 0010	110000 1101	1
K28.5	BC	101 11100	001111 1010	110000 0101	2
K28.6	DC	110 11100	001111 0110	110000 1001	1
K28.7	FC	111 11100	001111 1000	110000 0111	1,2
K23.7	F7	111 10111	111010 1000	000101 0111	1
K27.7	FB	111 11011	110110 1000	001001 0111	

Table 4-2. Special Character Encodings

Character Name	Character Value (hex)	Character Bits HGF EDCBA	Current RD –	Current RD +	Notes
			abcdei fghj	abcdei fghj	
K29.7	FD	111 11101	101110 1000	010001 0111	
K30.7	FE	111 11110	011110 1000	100001 0111	1

1 - Reserved code-groups.

2 - The code-groups /K28.5/, /K28.7/, and /K28.1/ contain a comma.

4.5.7 Special Characters and Columns

Table 4-3 defines the special characters and columns of special characters used by 1x and 4x LP-Serial links. Special characters are used for the following functions:

1. Alignment to code-group (10-bit) boundaries on lane-by-lane basis.
2. Alignment of the receive data stream across four lanes.
3. Clock rate compensation between receiver and transmitter.
4. Control symbol delimiting.

Table 4-3. Special Characters and Columns

Code-Group/Column Designation	Code-Group/Column Use	Number of Code-groups	Encoding
/PD/	Packet_Delimiter Control Symbol	1	/K28.3/
/SC/	Start_of_Control_Symbol	1	/K28.0/
/**I**/	Idle		
/K/	1x Sync	1	/K28.5/
/R/	1x Skip	1	/K29.7/
/A/	1x Align	1	/K27.7/
||**I**||	Idle column		
||K||	4x Sync column	4	/K28.5/K28.5/K28.5/K28.5/
||R||	4x Skip column	4	/K29.7/K29.7/K29.7/K29.7/
||A||	4x Align column	4	/K27.7/K27.7/K27.7/K27.7/

4.5.7.1 Packet Delimiter Control Symbol (/PD/)

PD and /PD/ are aliases for respectively the K28.3 character and the /K28.3/ code-group which are used to delimit the beginning of a control symbol that contains a packet delimiter.

4.5.7.2 Start of Control Symbol (/SC/)

SC and /SC/ are aliases for respectively the K28.0 character and the /K28.0/ code-group which are used to delimit the beginning of a control symbol that does not contain a packet delimiter.

4.5.7.3 Idle (/I/)

I and /I/ are aliases for respectively any of the idle sequence characters (A, K, or R) and idle sequence code-groups (/A/, /K/, or /R/).

4.5.7.4 Sync (/K/)

K and /K/ are aliases for respectively the K28.5 character and the /K28.5/ code-group which is used in the idle sequence to provide the receiver with the information it requires to achieve and maintain bit and 10-bit code-group boundary synchronization. The /K28.5/ code-group was selected as the Sync character for the following reasons:

1. It contains the comma pattern in bits **abcdeif** which can be easily found in the code-group bit stream and marks the code-group boundary.
2. The bits **ghj** provide the maximum number of transitions (i.e. 101 or 010).

A comma is a 7-bit string defined as either b'0011111' (comma+) or b'1100000' (comma-). Within the code-group set it is a singular bit pattern, which, in the absence of transmission errors, cannot appear in any other location of a code-group and cannot be generated across the boundaries of any two adjacent code-groups with the following exception:

> The /K28.7/ special code-group when followed by any of the data code-groups /D3.y/, /D11.y/, /D12.y/, /D19.y/, /D20.y/, /D28.y/, or /K28.y/, where y is an integer in the range 0 through 7, may (depending on the value of running disparity) cause a comma to be generated across the boundary of the two code-groups. A comma that is generated across the boundary between two adjacent code-groups may cause the receiver to change the 10-bit code-group alignment. As a result, the /K28.7/ special code-group may be used for test and diagnostic purposes only.

4.5.7.5 Skip (/R/)

R and /R/ are aliases for respectively the K29.7 character and the /29.7/ code-group which are used in the idle sequence and are also used in the clock compensation sequence.

4.5.7.6 Align (/A/)

A and /A/ are aliases for respectively the K27.7 character and the /27.7/ code-group which are used in the idle sequence and are used for lane alignment on 4x links.

4.5.8 Effect of Single Bit Code-Group Errors

Single bit code-group errors will be the dominant code-group error by many orders of magnitude. It is therefore useful to know the variety of code-group corruptions that can be caused by a single bit error.

Table 4-4 lists all possible code-group corruptions that can be caused by a single-bit error. The notation /X/ => /Y/ means that the code-group for the character X has been corrupted by a single-bit error into the code-group for the character Y. If the corruption results in a code-group that is invalid for the current receiver running disparity, the notation /X/ => /INVALID/ is used. The table provides the information required to deterministically detect all isolated single bit transmission errors.

Table 4-4. Code-Group Corruption Caused by Single Bit Errors

Corruption	Detection
/SC/ => /INVALID/	Detectable as an error when decoding the code-group. When this error occurs within a packet, it is indistinguishable from a /Dx.y/ => /INVALID/. When this error occurs outside of a packet, the type of error can be inferred from whether the /INVALID/ is followed by the three /Dx.y/ that comprise the control symbol data.
/PD/ => /INVALID/	Detectable as an error when decoding the code-group. When this error occurs within a packet, it is indistinguishable from a /Dx.y/ => /INVALID/. When this error occurs outside of a packet, the type of error can be inferred from whether the /INVALID/ is followed by the three /Dx.y/ that comprise the control symbol data.
/A/, /K/ or /R/ => /Dx.y/	Detectable as an error as /Dx.y/ is illegal outside of a packet or control symbol and /A/, /K/ and /R/ are illegal within a packet or control symbol.
/A/, /K/ or /R/ => /INVALID/	Detectable as an error when decoding the code-group.
/Dx.y/ => /A/, /K/ or /R/	Detectable as an error as /A/, /K/ and /R/ are illegal within a packet or control symbol and /Dx.y/ is illegal outside of a packet or control symbol.
/Dx.y/ => /INVALID/	Detectable as an error when decoding the code-group.
/Dx.y/ => /Du.v/	Detectable as an error by the packet or control symbol CRC. The error will also result in a subsequent unerrored code-group being decoded as INVALID, but that resulting INVALID code-group may occur an arbitrary number of code-groups after the errored code-group.

4.5.9 Idle Sequence

The idle sequence is a sequence of code-groups that shall be transmitted continuously over each lane of an LP-Serial link whenever packets or control symbols are not being transmitted. An idle sequence may not be inserted in a packet or delimited control symbol. The idle sequence is transmitted over each lane as part of the port initialization process as required in Section 4.7.3.5 and Section 4.7.3.6. An idle sequence containing a special clock compensation sequence shall be transmitted at least once every 5000 code-groups even when there are packets or

control symbols available to transmit to allow clock rate compensation.

The 1x idle sequence consists of a sequence of the code-groups /K/, /A/, and /R/ (the idle code-groups) and shall be used by ports in operating is 1x mode. The 4x idle sequence consists of a sequence of the columns ||K||, ||A||, ||R|| (the idle columns) and shall be used by ports operating in 4x mode. Both sequences shall comply with the following requirements:

1. The first code-group (column) of an idle sequence generated by a port operating in 1x mode (4x mode) shall be a /K/ (||K||). The first code-group (column) shall be transmitted immediately following the last code-group (column) of a packet or delimited control symbol.
2. At least once every 5000 code-groups (columns) transmitted by a port operating in 1x mode (4x mode), an idle sequence containing the /K/R/R/R/ code-group sequence (||K||R||R||R|| column sequence) shall be transmitted by the port. This sequence is referred to as the “compensation sequence”.
3. When not transmitting the compensation sequence, all code-groups (columns) following the first code-group (column) of an idle sequence generated by a port operating in 1x mode (4x mode) shall be a pseudo-randomly selected sequence of /A/, /K/, and /R/ (||A||, ||K||, and ||R||) based on a pseudo-random sequence generator of 7th order or greater and subject to the minimum and maximum /A/ (||A||) spacing requirements.
4. The number of non /A/ code-groups (non ||A|| columns) between /A/ code-groups (||A|| columns) in the idle sequence of a port operating in 1x mode (4x mode) shall be no less than 16 and no more than 32. The number shall be pseudo-randomly selected, uniformly distributed across the range and based on a pseudo-random sequence generator of 7th order or greater.

There are no requirements on the length of an idle sequence. An idle sequence may be of any length.

The idle sequence is transmitted on each lane of a link when neither packets nor delimited control symbols are being transmitted to allow each lane receiver to maintain bit and 10-bit code-group boundary synchronization.

The pseudo-random selection of code-groups in the idle sequence results in an idle sequence whose spectrum has no discrete lines which minimizes the EMI of long idle sequences.

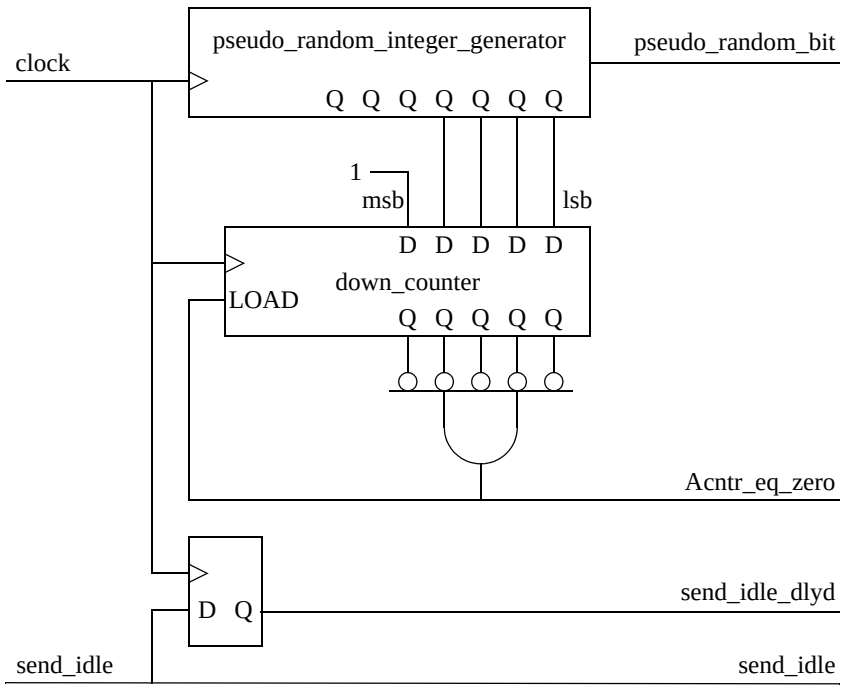
The compensation sequence allows retiming repeaters (discussed in Section 4.6) to compensate for up to a +/- 200 ppm difference between input bit rate and output bit rate, each of which have a +/-100 ppm tolerance. It may also be used to allow the input side of a port to compensate for up to a +/-200 ppm difference between the input bit rate and the bit rate of the device core which may be running off a different clock. This is done by dropping or adding an /R/ or ||R|| as needed to avoid overrun/underrun. Since a packet or delimited control symbol may not be interrupted by an idle sequence, designers must be careful to guarantee that no more

than 5000 code-groups are transmitted between compensation sequences.

4.5.9.1 Idle Sequence Generation

A primitive 7th order polynomial is recommended as the generating polynomial for the pseudo-random sequence that is used in the generation of the idle sequence. The polynomials $x^7 + x^6 + 1$ and $x^7 + x^3 + 1$ are examples of primitive 7th order polynomials which could be used as generator polynomials. The pseudo-random sequence generator is clocked (generates a new pseudo-random sequence value) once per idle sequence code-group (column). Four of the pseudo-random sequence generator state bits may be selected as the pseudo-random value for /A/ (||A||) spacing and any other state bit or logical function of state bits may be selected as the /K/ vs. /R/ selector.

Figure 4-4 shows an example circuit illustrating how this may be done. The clock ticks whenever a code-group or column is transmitted. Send_idle is asserted whenever an idle sequence begins. The equations indicate the states in which to transmit the indicated idle code-group, except when the compensation sequence is being transmitted. Any equivalent method is acceptable.



send_K = send_idle & (!send_idle_dlyd | send_idle_dlyd & !Acntr_eq_zero & pseudo_random_bit)
send_A = send_idle & send_idle_dlyd & Acntr_eq_zero
send_R = send_idle & send_idle_dlyd & !Acntr_eq_zero & !pseudo_random_bit

Figure 4-4. Example of a Pseudo-Random Idle Code-Group Generator

4.5.10 1x Link Transmission Rules

A 1x LP-Serial link has a single lane (differential pair) in each direction. A 1x LP-Serial port shall be encoded and transmit the character stream of delimited control symbols and packets received from the upper layers over the differential pair in the order the characters were received from the upper layers. When neither control symbols nor packets are available from the upper layers for transmission, the 1x idle sequence shall be fed to the input of the encoder for encoding and transmission. On reception, the code-group stream is decoded and passed to the upper layers.

When a 4x port is operating in 1x mode, the character stream from the upper layers is not striped across the lanes before encoding as is done when operating in 4x mode. The entire character stream from the upper layers is fed in parallel to both lanes 0 and 2. A 4x LP-Serial port operating in 1x mode shall encoded and transmit the character stream of delimited control symbols and packets received from the upper layers over both lane 0 and lane 2, with the following exception. A 4x port operating in 1x mode may elect to disable the output driver of the lane which was not selected by the initialization state machine. It is recommended that the mechanism for disabling the output driver be under software control.

When neither delimited control symbols nor packets are available from the upper layers for transmission, the 1x idle sequence shall be fed in parallel to the input of the lane 0 and lane 2 encoders for encoding and transmission on lanes 0 and 2. On reception, the code-group stream from either lane 0 or 2 is selected according to the state of the 1x/4x_Initialization state machine (Section 4.7.3.6, “1x/4x Mode Initialization State Machine), decoded and passed to the upper layers.

Figure 4-5 shows the encoding and transmission order for a control symbol transmitted over a 1x LP-Serial link.

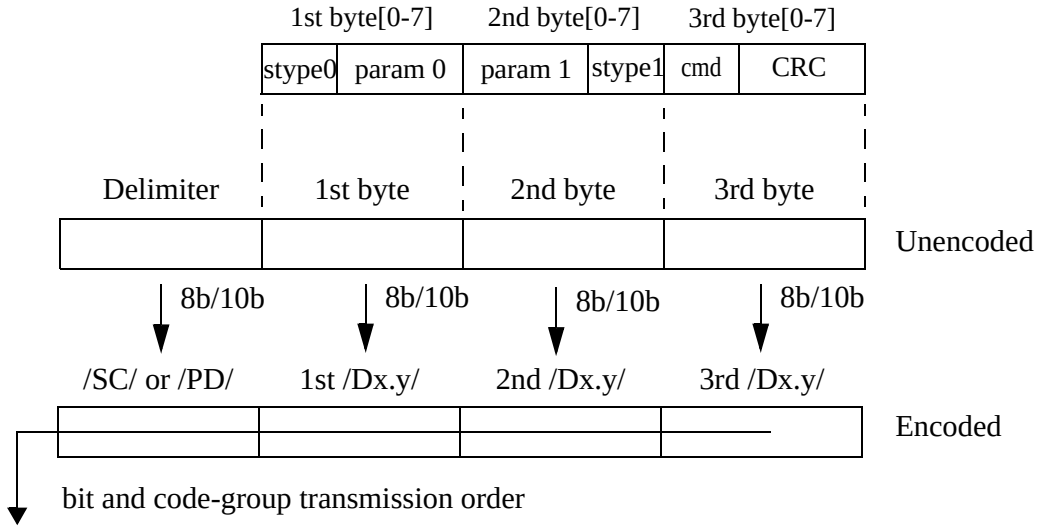


Figure 4-5. 1x Mode Control Symbol Encoding and Transmission Order

Figure 4-6 shows the encoding and transmission order for a packet transmitted over a 1x LP-Serial link.

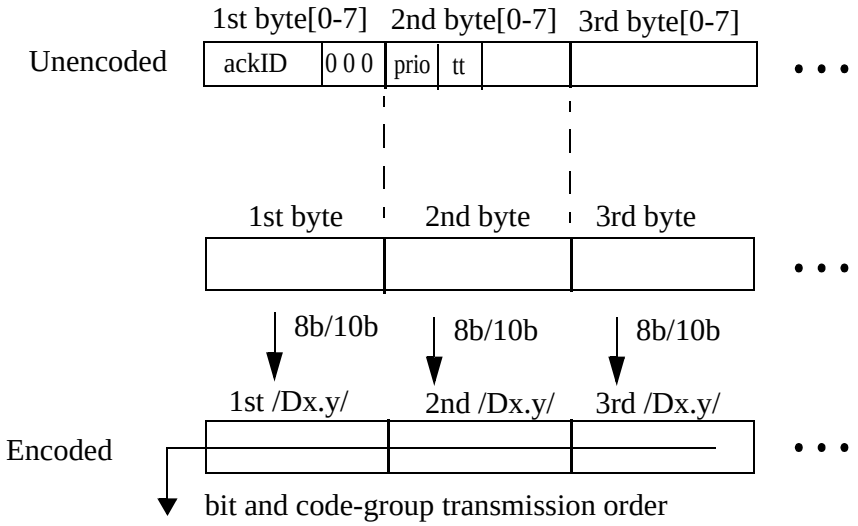


Figure 4-6. 1x Mode Packet Encoding and Transmission Order

Figure 4-7 shows an example of control symbol, packet, and idle sequence transmission on a 1x LP-Serial link.

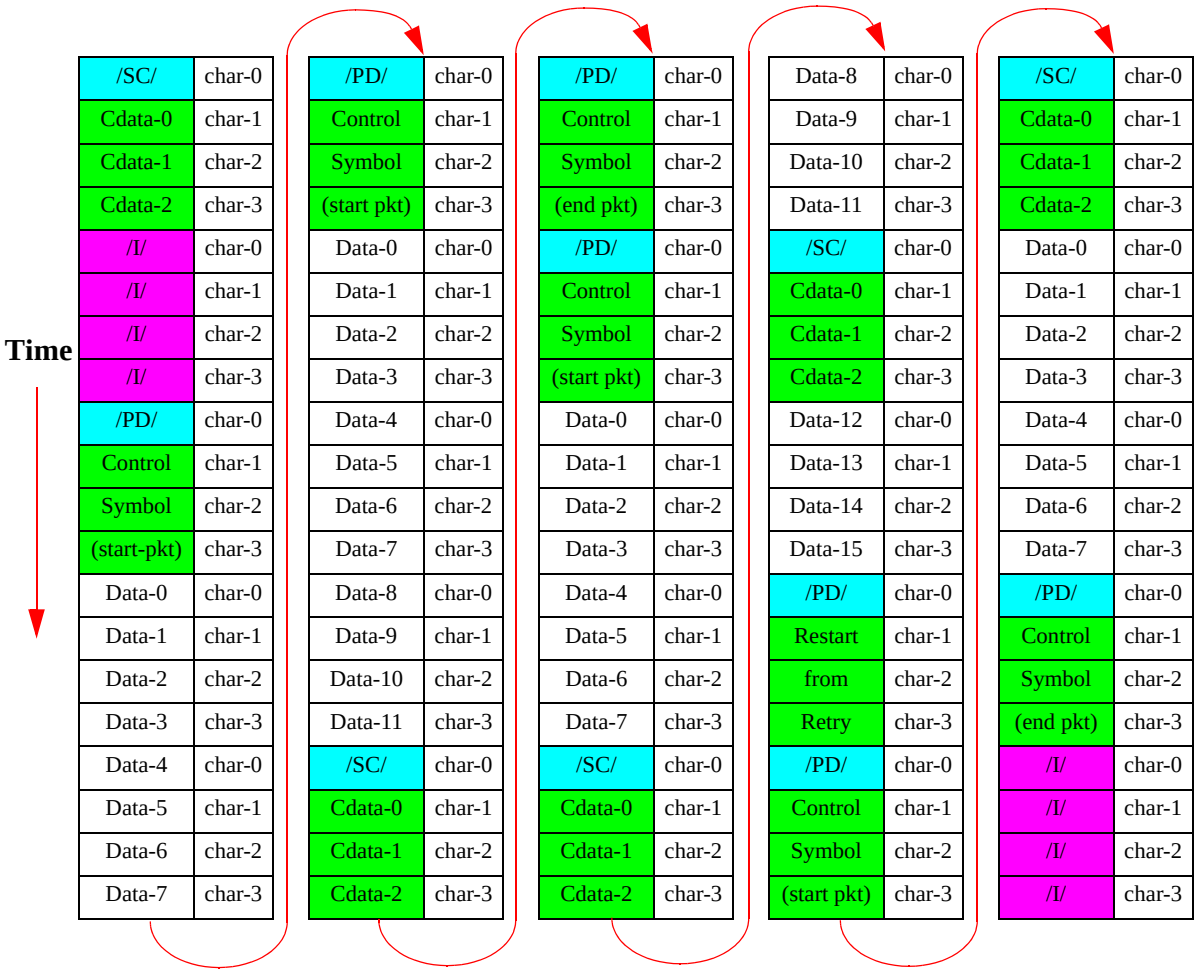


Figure 4-7. 1x Typical Data Flow

4.5.11 4x Link Striping and Transmission Rules

A LP-Serial port operating in 4x mode shall stripe the character stream of delimited control symbols and packets that it receives from the upper layers across the four lanes before 8B/10B encoding as follows:

Packets and delimited control symbols shall be striped across the four lanes beginning with lane 0. The first character of each packet, or delimited control symbol, is placed in lane 0, the second character is placed in lane1, the third character is placed in lane 2, and the fourth character is placed in lane 3. The fifth character and subsequent characters wrap around and continue beginning in lane 0.

As a result of their defined lengths, delimited control symbols will form a column after striping and a packet will form an integer number of contiguous columns.

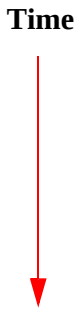
After striping, each of the 4 streams of characters shall be independently 8B/10B encoded and transmitted.

When neither delimited control symbols nor packets are available from the upper layers for transmission, the 4x idle sequence shall be transmitted. This can be achieved by feeding the 1x idle sequence in parallel to the inputs of the encoders for all four lanes for encoding and transmission on the four lanes. Feeding the 1x idle sequence in parallel to the input of all four lane encoders converts each 1x idle sequence character in to a 4x idle sequence column. The 1x sequence is not striped across the four lanes.

On reception, each lane shall be 8B/10B decoded. After decoding, the four lanes shall be aligned. The ||A|| columns transmitted as part of the 4x idle sequence provide the information needed to perform alignment. After alignment, the columns are destriped into a single character stream and passed to the upper layers.

The lane alignment process eliminates the skew between lanes so that after destriping, the ordering of characters in the received character stream is the same as the ordering of characters before striping and transmission. Since the minimum number of non ||A|| columns between ||A|| columns is 16, the maximum lane skew that can be unambiguously corrected is the time it takes for to transmit 7 code-groups on a lane.

Figure 4-8 shows an example of idle sequence, packet, and delimited control symbol transmission on a 4x link.



Lane-0	Lane-1	**Lane-2**	**Lane-3**
/SC/	Cdata-0	Cdata-1	Cdata-2
/I/	/I/	/I/	/I/
/PD/	Control Symbol (Start-of-packet)		
Data-0	Data-1	Data-2	Data-3
Data-4	Data-5	Data-6	Data-7
/PD/	Control Symbol (Start-of-packet)		
Data-0	Data-1	Data-2	Data-3
Data-4	Data-5	Data-6	Data-7
Data-8	Data-9	Data-10	Data-11
/SC/	Cdata-0	Cdata-1	Cdata-2
/PD/	Control Symbol (End-of-packet)		
/PD/	Control Symbol (Start-of-packet)		
Data-0	Data-1	Data-2	Data-3
Data-4	Data-5	Data-6	Data-7
/SC/	Cdata-0	Cdata-1	Cdata-2
Data-8	Data-9	Data-10	Data-11
/SC/	Cdata-0	Cdata-1	Cdata-2
Data-12	Data-13	Data-14	Data-15
/PD/	Control Symbol (Restart-from-retry)		
/PD/	Control Symbol (Start-of-packet)		
/SC/	Cdata-0	Cdata-1	Cdata-2
Data-0	Data-1	Data-2	Data-3
Data-4	Data-5	Data-6	Data-7
/PD/	Control Symbol (End-of-packet)		
/I/	/I/	/I/	/I/

Figure 4-8. Typical 4x Data Flow

4.6 Retimers and Repeaters

The LP-Serial Specification allows “retimers” and “repeaters”. Retimers amplify a weakened signal, but do not transfer jitter to the next segment. Repeaters also amplify a weakened signal, but transfer jitter to the next segment. Retimers allow greater distances between end points at the cost of additional latency. Repeaters support less distance between end points than retimers and only add a small amount of latency.

4.6.1 Retimers

A retimer shall comply with all AC specifications found in Chapter 8, “Electrical Specifications”. This includes resetting the jitter budget thus extending the transmission distance for the link. The retimer repeats the received code-groups after performing code-group synchronization and serializes the bitstream again on transmission, based on a local clock reference. Up to two retimers are allowed between two end nodes.

A retimer is not RapidIO protocol-aware or addressable in any way. The only awareness a retimer has is to the synchronization on the /K/ code-group and the function of /R/ insertion and removal. A retimer may insert up to one /R/ code-group immediately following a /K/ code-group sequence, or remove one /R/ code-group that immediately follows a /K/ code-group sequence. Note that the /R/ code-group is disparity neutral and therefore its insertion or deletion does not affect the running disparity.

A 4-lane retimer must perform lane synchronization and deskew, in exactly the same way a RapidIO device implementing this physical layer does when synchronizing inputs during initialization and startup. A 4-lane retimer will synchronize and align all lanes that are driven to it. Therefore, such a retimer allows for the degradation of an input 4x link to a 1x link on either lane 0 or 2. If any link drops out, the retimer must merely continue to pass the active links, monitoring for the compensation sequence and otherwise passing through whatever code-groups appear on its inputs. A retimer may optionally not drive any outputs whose corresponding inputs are not active.

Any insertion or removal of a /R/ code-groups in a 4-lane retimer must be done on a full column. A retimer may retime links operating at the same width only (i.e. cannot connect a link operating at 1x to a link operating at 4x). A retimer may connect a 1x link to a 4x link that is operating at 1x. Retimers perform clock tolerance compensation between the receive and transmit clock. The transmit clock is usually derived from a local reference.

Retimers do not check for code violations. Code-groups received on one port are transmitted on the other regardless of code violations or running disparity errors.

4.6.2 Repeaters

A repeater is used to amplify the signal, but does not retime the signal, and therefore can add additional jitter to the signal. It does not compensate for clock rate variation. The repeater repeats the received code-groups as the bits are received by sampling the incoming bits with a clock derived from the bit stream, and then retransmitting them based on that clock. Repeaters may be used with 4x links but lane-to-lane skew may be amplified. Repeaters do not interpret or alter the bit stream in any way.

4.7 Port Initialization

This section specifies the port initialization process. The process includes detecting the presence of a partner at the other end of the link (a link partner), establishing bit synchronization and code-group boundary alignment and if the port is capable of supporting both 1x and 4x modes (a 1x/4x port), discovering whether the link partner is capable of 4x mode (4-lane) operation, selecting 1x or 4x mode operation and if 1x mode is selected, selecting lane 0 or lane 2 for link reception.

The initialization process is controlled by several state machines. The number and type of state machines depends on whether the port supports only 1x mode (a 1x port) or supports both 1x and 4x modes (a 1x/4x port). In either case, there is a primary state machine and one or more secondary state machines. The use of multiple state machines results in a simpler overall design. As might be expected, the initialization process for a 1x port is simpler than and a subset of the initialization process for a 1x/4x port.

The initialization process for 1x and 1x/4x ports is both described in text and specified with state machine diagrams. **In the case of conflict between the text and a state machine diagram, the state machine diagram takes precedence.**

4.7.1 1x Mode Initialization

The initialization process for ports that support only 1x mode shall be controlled by two state machines, 1x_Initialization and Lane_Synchronization. 1x_Initialization is the primary state machine and Lane_Synchronization is the secondary state machine. The operation of these state machines is described and specified in Section 4.7.3.5 and Section 4.7.3.3 respectively.

After a 1x port has been initialized, the port is required to receive seven error-free status control symbols without intervening detected errors to verify link operation before transmitting any packets.

4.7.2 1x/4x Mode Initialization

The initialization process for ports that support both 1x and 4x modes shall be controlled by six state machines, 1x/4x_Initialization, Lane_Synchronization[0-3] (one for each of the four lanes) and Lane_Alignment. 1x/4x_Initialization is the primary state machine. Lane_Synchronization[0-3] and Lane_Alignment are the secondary state machines. The operation of these state machines is described and specified in Section 4.7.3.6, Section 4.7.3.3 and Section 4.7.3.4 respectively.

After a 4x port has been initialized, the port is required to receive seven error-free status control symbols without intervening detected errors to verify link operation before transmitting any packets.

4.7.3 State Machines

4.7.3.1 State Machine Conventions

A state machine state is persistent until an exit condition occurs.

A state machine variable that is listed in the body of a state but is not part of an assignment statement is asserted for the duration of that state only.

A state machine variable that is assigned a value in the body of a state retains that value until assigned a new value in another state.

A state machine function that is listed in the body of a state is executed once during the state.

4.7.3.2 State Machine Variables and Functions

A state machine variable is asserted when its value is 1 and deasserted when it value is 0.

The functions used in the state machines are defined as follows.

change( )

Asserted when the variable on which it operates changes state.

next_code_group( )

Gets the next 10 bit code-group for the lane when it becomes available.

next_column( )

Gets the next column of 4 code-groups when it becomes available.

The variables used in the state machines are defined as follows.

4x-mode

Asserted when the port is operating in 4x mode

||A||

Asserted when the current column contains all /A/s.

Acounter

A counter used in the Lane Alignment state machine to count received alignment columns (||A||s).

align_error

Asserted when the current column contains at least one /A/, but not all /A/s.

all_lane_sync

Asserted when all four lanes of a 4x mode receiver are in bit synchronization and

code-group boundary alignment. (all_lane_sync = lane_sync[0] & lane_sync[1] & lane_sync[2] & lane_sync[3])

discovery_timer_done

Asserted when discovery_timer_en has been continuously asserted for 12 +/- 4 msec and the state machine is in the DISCOVERY state. The assertion of discovery_timer_done causes discovery_timer_en to be deasserted. When the state machine is not in the DISCOVERY state, discovery_timer_done is deasserted.

discovery_timer_en

When asserted, the discovery_timer runs. When deasserted, the discovery_timer is reset to and maintains its initial value.

force_1x_mode

When asserted, forces the 1x/4x Initialization state machine to use 1x mode.

force_lane2

When asserted in 1x mode, forces the 1x/4x Initialization state machine to use lane 2 for reception.

force_reinit

When asserted, forces a 1x or 1x/4x Initialization state machine to reinitialize. The signal is set under software control and is cleared by the Initialization state machine.

Icounter

Counter used in the Lane_Synchronization state machine to count INVALID received code-groups. There is one Icounter for each lane in a 4x mode receiver.

/K28.5/

Asserted when the current code-group is /K28.5/

Kcounter

Counter used in the Lane_Synchronization state machine to count received /K.28.5/ code-groups. There is one Kcounter for each lane in a 4x mode receiver.

lane02_drvr_oe

The output enable for the lane 0 and the lane 2 output drivers of a 1x/4x mode port.

lane13_drvr_oe

The output enable for the lane 1 and the lane 3output drivers of a 1x/4x mode port.

lanes_aligned

Asserted by the Lane_Alignment state machine when it determines that all four

lanes are in sync and aligned.

lane_sync

Asserted by the Lane_Synchronization state machine when it determines that the lane it is monitoring is in bit synchronization and code-group boundary alignment. The is a lane_sync signal for each lane in a 4x mode receiver.

link_drvr_oe

When asserted, the output link driver of a 1x port is enabled.

Mcounter

Mcounter is used in the Lane_Alignment state machine to count columns received that contain at least one /A/, but not all /A/s.

Vcounter

Vcounter is used in the Lane_Synchronization state machine to count VALID received code-groups. There is one Vcounter for each lane in a 4x mode receiver.

port_initialized

When asserted, port_initialized indicates that the link is initialized and is available for transmitting control symbols and packets. When deasserted, the link is not initialized and is not available for transmitting control symbols and packets.

receive_lane2

In a 1x/4x capable port that is initialized and is operating in 1x mode (4x_mode deasserted), receive_lane2 indicates which lane the port has selected for input. When asserted, the port input is from lane 2. When deasserted the port input is from lane 0. When the port is operating in 4x mode (4x_mode asserted), receive_lane2 is undefined and shall be ignored.

signal_detect

Asserted when a lane receiver is enabled and a signal meeting an implementation defined criteria is present at the input of the receiver. The use of signal_detect is implementation dependent. It may be continuously asserted or it may be used to require that some implementation defined additional condition be met before the Lane_Synchronization state machine is allowed to exit the NO_SYNC state. Signal_detect might for example be used to ensure that the input signal to a lane receiver meet some minimum AC input power requirement so that the receiver can not lock up on crosstalk from an output of the same port.

silence_timer_done

Asserted when silence_timer_en has been continuously asserted for 120 +/- 40 µs and the state machine is in the SILENT state. The assertion of silence_timer_done causes silence_timer_en to be deasserted. When the state machine is not in the SILENT state, silence_timer_done is deasserted.

silence_timer_en

When asserted, the silence_timer runs. When deasserted, the silence_timer is reset to and maintains its initial value.

/VALID/

When asserted, /VALID/ indicates that the current code-group is a valid code-group given the current running disparity.

/INVALID/

When asserted, /INVALID/ indicates that the current code-group is an invalid code-group given the current running disparity.

4.7.3.3 Lane Synchronization State Machine

The Lane_Synchronization state machine monitors the bit synchronization and code-group boundary alignment for a lane receiver. A port that supports only 1x mode has one Lane_Synchronization state machine. A port that supports both 1x and 4x modes has four Lane_Synchronization state machines, one for each lane (Lane_Synchronization[0] through Lane_Synchronization[3]).

The state machine determines the bit synchronization and code-group boundary alignment state of a lane receiver by monitoring the received code-groups and looking for /K28.5/s, other valid code-groups and invalid code-groups. The code-group /K28.5/ contains the “comma” bit sequence that is used to establish code-group boundary alignment. When a lane is error free, the “comma” pattern occurs only in the /K28.5/ code-group. Several counters are used to provide hysteresis so that occasional bit errors do not cause spurious lane_sync state changes.

The state machine does not specify how bit synchronization and code-group boundary alignment is to be achieved. The methods used by a lane receiver to achieve bit synchronization and code-group boundary alignment are implementation dependent. However, isolated single bit errors shall not cause the code-group boundary alignment mechanism to change alignment. For example, a isolated single bit error that results in a “comma” pattern across a code-group boundary may not cause the code-group boundary alignment mechanism to change alignment.

The state machine starts in the NO_SYNC state and sets the variables Kcounter[0] and lane_sync[n] to 0 (lane n is out of code-group boundary sync). It then looks for a /K28.5/ code-group. When it finds one and the signal signal_detect[n] is asserted, the machine moves to the NO_SYNC_1 state.

The NO_SYNC_1 state in combination with the NO_SYNC_2 state looks for 127 /K28.5/ code-groups without any intervening /INVALID/ code-groups. When this condition is achieved, machine goes to state SYNC. If an intervening /INVALID/ code-group is detected, the machine goes back to the NO_SYNC state. The number

127 was selected because it is large enough that it would be highly unlikely that SYNC would be falsely achieved and also because it fits in a 7-bit counter. Since the /K28.5/ code-group comprises slightly less than half of the code-groups in the idle sequence, something more than 256 code-groups must be received after the first /K28.5/ to achieve the 128 /K28.5/ code-group criteria to transition to the SYNC state.

In the SYNC state, the machine sets the variable lane_sync[n] to 1 (lane n is in code-group boundary sync), set the variable Icounter[n] to 0 and begins looking for /INVALID/ code-groups. If an /INVALID/ code-group is detected, the machine goes to state SYNC_1.

The SYNC_1 state in combination with the SYNC_2, SYNC_3, and SYNC_4 states looks for 255 consecutive /VALID/ code-groups without any /INVALID/ code-groups. When 255 /VALID/ symbols are received, the Icounter[n] value is decremented in the transition through the SYNC_4 state. If it does not, it increments Icounter[n]. If Icounter[n] is decremented back to 0, the machine returns to the SYNC state. If Icounter[n] is incremented to 3, the machine goes to the NO_SYNC state and starts over. This algorithm tolerates isolated single bit errors in that an isolated single bit error will not cause the machine to change the variable lane_sync[n] from 1 to 0 (in sync to out of sync). Three errors (sometimes two errors when one of them causes a disparity error on the following code group) within 256 code-groups will result in an out-of-sync indication.

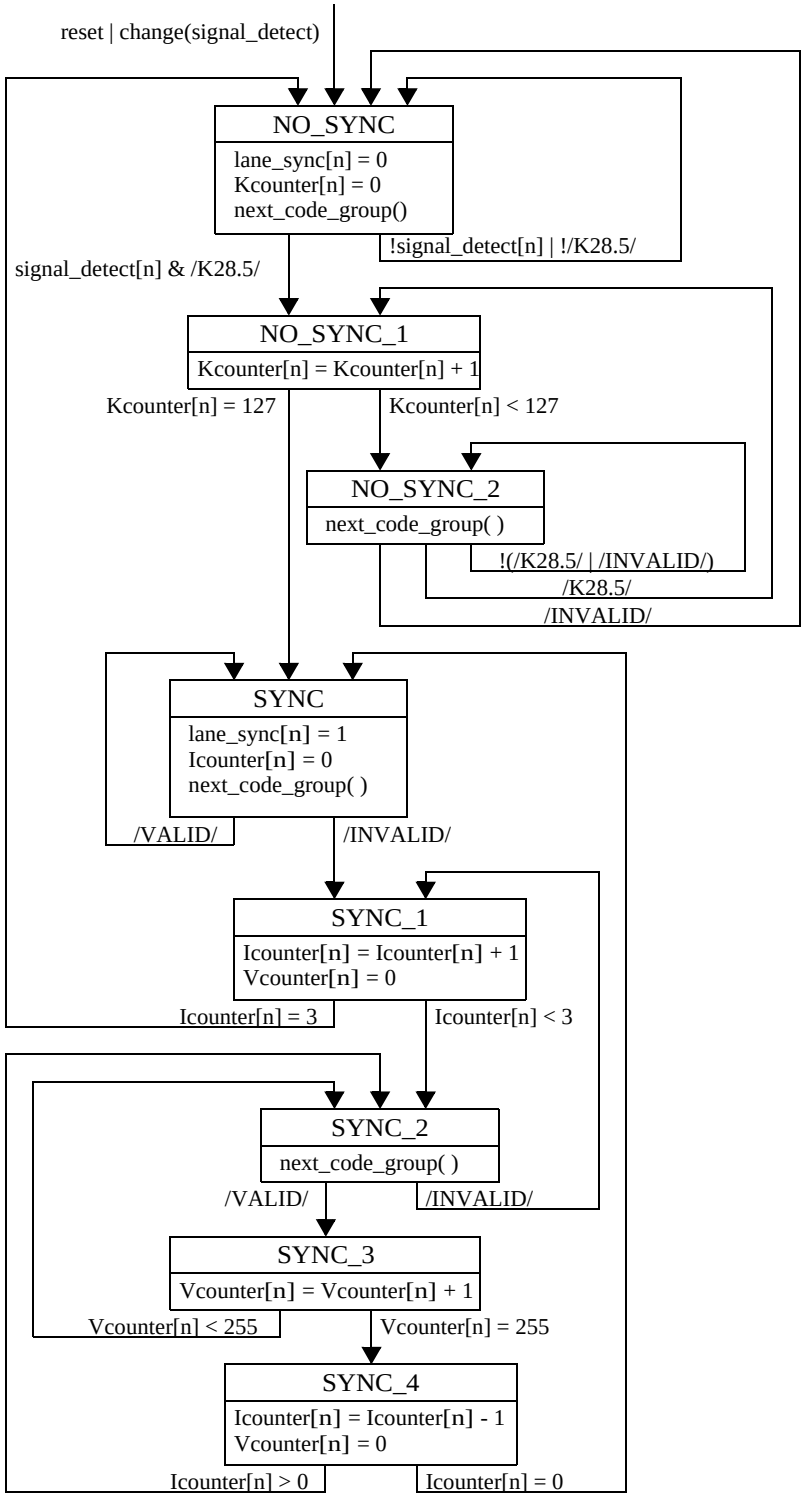


Figure 4-9. Lane_Synchronization State

4.7.3.4 Lane Alignment State Machine

The Lane_Alignment state machine monitors the alignment of the output of the four lane receivers in a port operating in 4x mode. A port supporting 4x mode has one Lane_alignment state machine, a port supporting only 1x mode does not have a Lane_Alignment state machine. (Lane alignment is required in a 4x port receiver to compensate for unequal propagation delays through the four lanes.)

The state machine determines the alignment state by monitoring the fours lanes for columns containing all /A/s (||A||), columns containing at least one but not all /A/s and columns containing no /A/s. Several counters are used to provide hysteresis so that occasional bit errors do not cause spurious lanes_aligned state changes.

The state machine does not specify how lane alignment is to be achieved. The methods used by a 4x port receiver to achieve lane alignment are implementation dependent. However, isolated single bit errors shall not cause the lane alignment mechanism to change lane alignment. For example, a isolated single bit error that results in a column that contains at least one /A/ but not all /A/s may not cause the lane alignment mechanism to change the lane alignment.

The state machine starts in the NOT_ALIGNED state where the variables Acounter and lanes_aligned are set to 0 (all lanes are not aligned). The machine then waits for all (four) lanes to achieve code-group boundary alignment (all_lanes_sync asserted) and the reception of an ||A|| (a column of all /A/s). When this obtains, the machine goes to NOT_ALIGNED_1 state.

The NOT_ALIGNED_1 state in combination with the NOT_ALIGNED_2 state looks for the reception of four ||A||s without the intervening reception of a misaligned column (a column with at least one /A/ but not all /A/s which causes the signal align_error to be asserted). When this obtains, the machine goes to the ALIGNED state. If an intervening misaligned column is received, the machine goes back to the NOT_ALIGNED state.

In the ALIGNED state, the machine sets the variable lanes_aligned to 1 (all lanes are aligned) and the variable Mcounter to 0 and looks for a misaligned column (align_error asserted). If a misaligned column is detected, the machine goes to the ALIGNED_1 state.

The ALIGNED_1 state in combination with the ALIGNED_2 and ALIGNED_3 states look for the reception of four ||A||s without the intervening reception of a misaligned column. If this condition obtains, the machine returns to the ALIGNED state. If an intervening misaligned column occurs, the machine goes to the NOT_ALIGNED state and starts over. This algorithm tolerates isolated single bit errors in that an isolated single bit error will not cause the machine to change the variable lanes_aligned from 1 to 0 (in lane alignment to out of lane alignment). The Lane_Alignment state machine is specified in Figure 4-10.

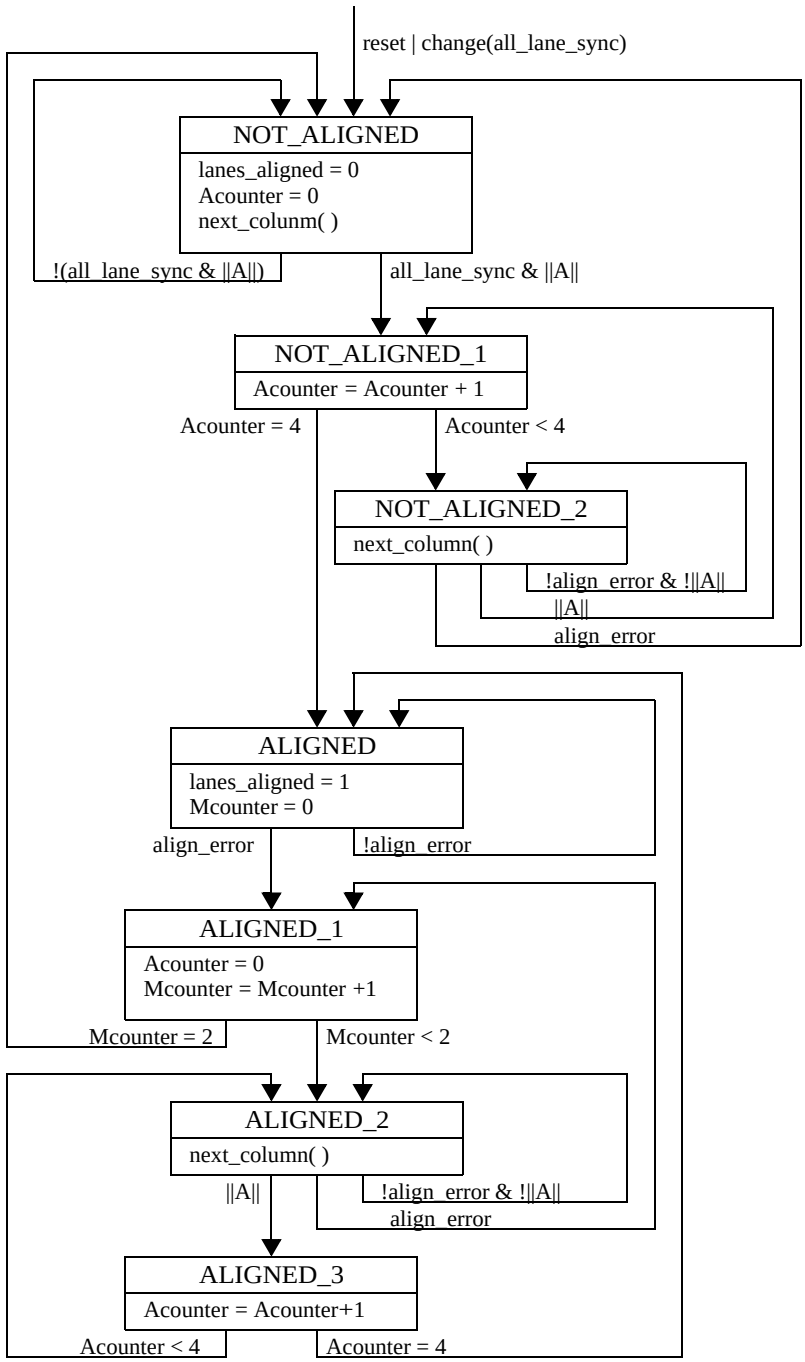


Figure 4-10. Lane_Alignment State Machine

4.7.3.5 1x Mode Initialization State Machine

The 1x_Initialization state machine shall be used by ports that support only 1x mode (1x ports).

The machine starts in the SILENT state. The link output driver is disabled to force the link partner to initialize regardless of its current state. The duration of the SILENT state is controlled by the silence_timer. The duration must be long enough to ensure that the link partner detects the silence (as a loss of lane_sync) and is forced to initialize but short enough that it is readily distinguished from a link break. When the silent interval is complete, the SEEK state is entered.

In the SEEK state, the link output driver is enabled, the idle sequence is transmitted, and the port waits for lane_sync to be asserted indicating the presence of a link partner. While lane_sync as defined indicates the bit and code-group boundary alignment synchronization state of the link receiver, it may also be thought of as indicating the presence of a link partner. When lane-sync is asserted, the 1X_MODE state is entered.

The input signal force_reinit allows the port to force link initialization at any time.

The variable port_initialized is asserted only in the 1X_MODE state. When port_initialized is deasserted, the port shall transmit a continuous idle sequence uninterrupted by control symbols or packets. To maintain receiver bit synchronization and code-group alignment, the port shall transmit the idle sequence when no control symbol or packet is being transmitted.

The 1x_Initialization state machine is specified in Figure 4-11.

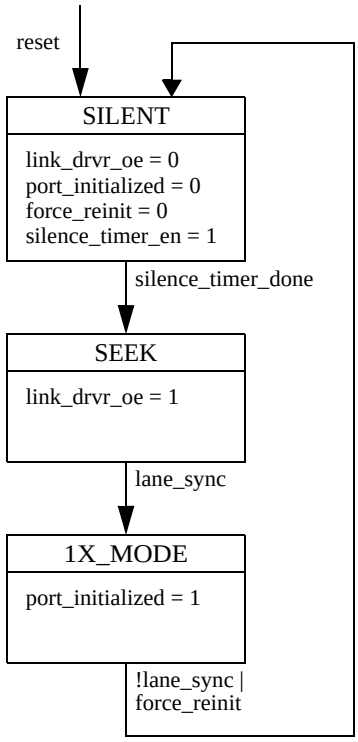


Figure 4-11. 1x_Initialization State Machine

4.7.3.6 1x/4x Mode Initialization State Machine

The 1x/4x_Initialization state machine shall be used by ports that support both 1x and 4x mode (1x/4x ports). In addition to determining when the link is initialized, the state machine controls whether the port receiver operates in 1x or 4x mode and in 1x mode whether lane 0 or lane 2 is selected as the inbound lane.

The machine starts in SILENT state. All four lane output drivers are disabled to force the link partner to initialize regardless of its current state. The duration of the SILENT state is controlled by the silence_timer. The duration must be long enough to ensure that the link partner detects the silence (as a loss of lane_sync) and is forced to initialize but short enough that it is readily distinguished from a link break. When the silent interval is complete, the SEEK state is entered.

In the SEEK state, a 1x/4x port transmits the idle sequence on lanes 0 and 2 (the lane 1 and lane 3 output drivers remain disabled to save power) and waits for an indication that a link partner is present. While lane_sync as defined indicates the bit and code-group boundary alignment synchronization state of a lane receiver, it is also used to indicate the presence of a link partner. A link partner is declared to be present when either lane_sync[0] or lane_sync[2] is asserted. If force_1x_mode is not asserted, the assertion of either lane_sync[0] or lane_sync[2] causes the state machine to enter the DISCOVERY state. If force_1x_mode is asserted, the state machine enters either the 1X_MODE_LANE0 or 1X_MODE_LANE2 state depending on whether lane_sync[0] or lane_sync[2] is asserted first and whether

force_lane2 is asserted.

In the DISCOVERY state, the port enables the output drivers for lanes 1 and 3 and transmits the idle sequence on all four lanes. The discovery_timer is also started. The discovery_timer allows time for the link partner to enter its DISCOVERY state and if the link partner is supporting 4x mode, for all four local lane receivers to acquire bit synchronization and code-group boundary alignment and for all four lanes to be aligned.

If lane alignment is achieved (lanes_aligned asserted) while in the DISCOVERY state, the machine enters the 4X_MODE state. It remains in this state until lane alignment or at least one lane_sync is lost (lanes_aligned deasserted) or reinitialization is forced (force_reinit is asserted).

At the end of the discovery period (discovery_timer_done is asserted), if lane alignment has not been achieved (lanes_aligned not asserted), the machine enters one of the 1x mode states. If code-group alignment has been achieved on lane 0 (lane_sync[0] asserted), the machine enters 1X_MODE_LANE0 and remains in that state until code-group boundary alignment is lost (lane0_sync deasserted) or reinitialization is forced (force_reinit is asserted). If code-group alignment has been achieved on lane 2 (lane_sync[2] asserted) but not on lane 0 (lane_sync[0] not asserted), the machine enters 1X_MODE_LANE2 and remains in that state until code-group boundary alignment is lost (lane_sync[2] deasserted) or reinitialization is forced (force_reinit is asserted).

If lane synchronization for both lane 0 and lane 2 is lost during the DISCOVERY state, it is not possible to successfully complete initialization and it is necessary to re-start the initialization process. Re-starting the initialization process is done by transitioning to the SILENT state. This results in the link partner also losing lane syncronization on both lanes 0 and 2, resulting in both ends of the link re-entering the SEEK state after the end of the respective silent periods (silence_timer_done asserted).

When in the 4X_MODE state, if lane alignment or at least one lane_sync is lost (lanes_aligned deasserted), the state machine transitions to either the SILENT state if both lane_sync[0] and land_sync[2] are deasserted or the DISCOVERY state if either lane_sync[0] or lane_sync[2] is asserted. This allows a 1x/4x port in the 4X_MODE state to return to 4X_MODE if lanes_aligned is deasserted due to multi-bit reception error, but also allows the port to switch to 1x mode if the connected 1x/4x port is not able to receive in 4x mode and has switched to 1x mode.

The input signals force_1x_mode and force_lane2 allow the state of the machine to be forced during initialization into 1x mode, and in 1x mode to be forced to receive on lane 2. The input signal force_reinit allows the port to force link initialization at any time.

The variable port_initialized is asserted only in the 1X_MODE_LANE0, 1X_MODE_LANE2 and 4X_MODE states. When port_initialized is deasserted, the

port shall transmit a continuous idle sequence uninterrupted by control symbols or packets. To maintain receiver bit synchronization and code-group alignment, the port shall transmit the idle sequence when no control symbol or packet is being transmitted.

The 1x/4x_Initialization state machine is specified in Figure 4-12.

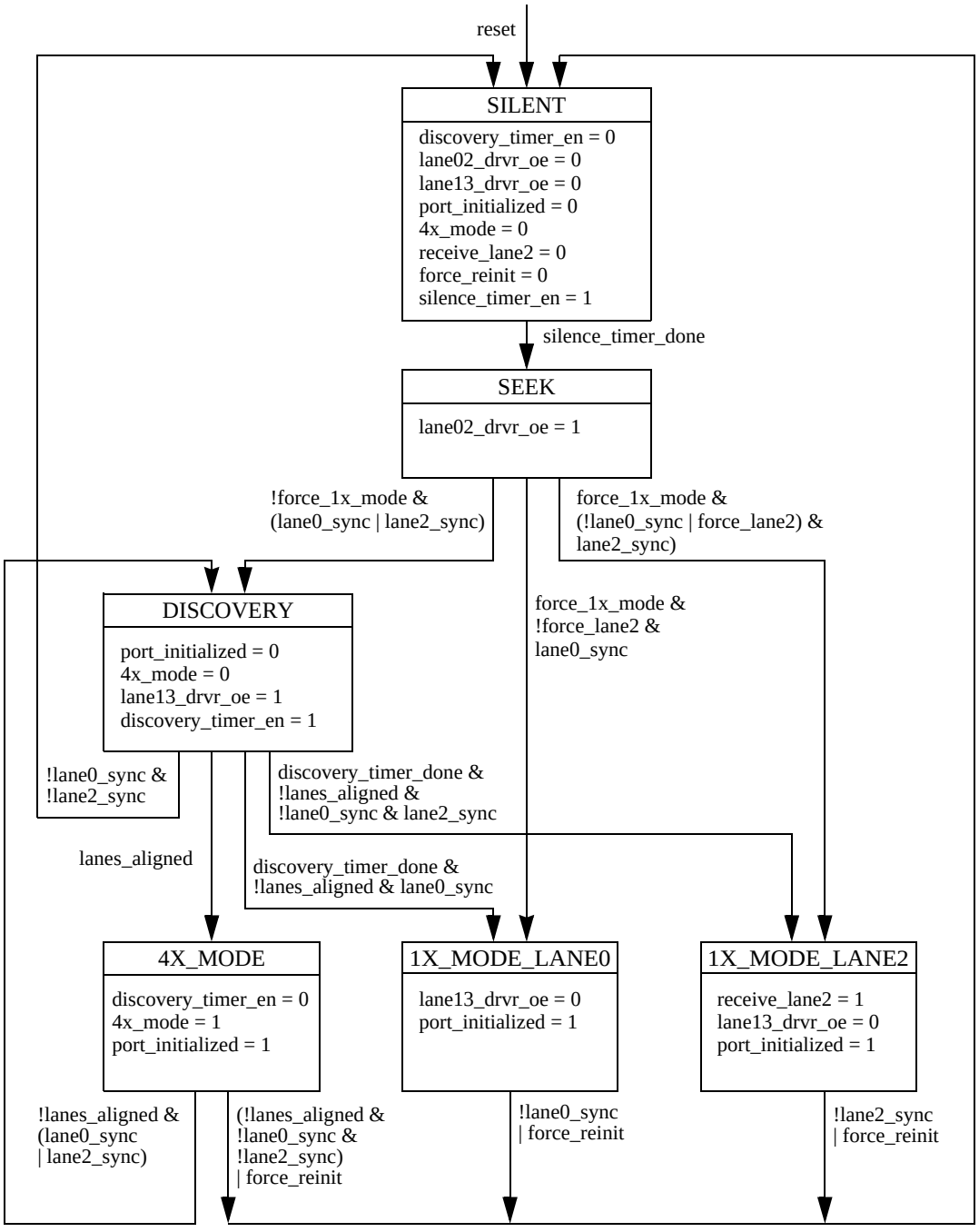


Figure 4-12. 1x/4x_Initialization State Machine

Chapter 5 LP-Serial Protocol

5.1 Introduction

This chapter specifies the LP-serial protocol. The protocol provides the reliable delivery of packets between two RapidIO devices that are connected by an LP-Serial link.

Packet priority, the mapping of transaction request flows onto packet priority, buffer management, and the use of control symbols in managing the delivery of packets between two devices is explained in this chapter.

5.2 Packet Exchange Protocol

This physical layer LP-Serial specification defines a protocol for devices connected by a LP-Serial link in which each packet transmitted by one device is acknowledged by control symbols transmitted by the other device. If a packet cannot be accepted for any reason, an acknowledgment control symbol indicates the reason and that the original packet and any transmitted subsequent packets must be resent. This behavior provides a flow control and error control mechanism between connected processing elements.

Figure 5-1 shows an example of transporting a request and response packet pair across an interconnect fabric with acknowledgments between the link transmitter/receiver pairs along the way. This allows flow control and error handling to be managed between each electrically connected device pair rather than between the original source and final target of the packet. An end point device shall transmit an acknowledgment control symbol for a request packet before transmitting the response packet corresponding to that request.

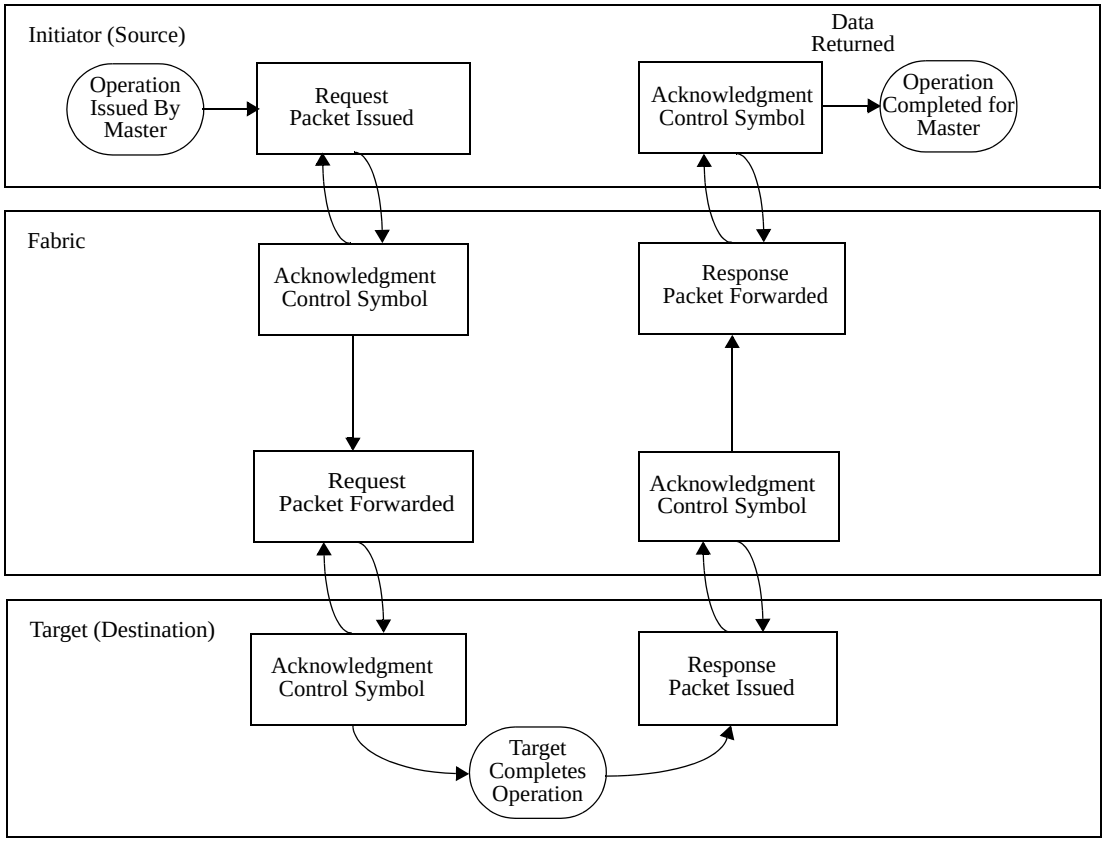


Figure 5-1. Example Transaction with Acknowledgment

5.3 Control Symbols

Control Symbols are the message elements used by ports connected by an LP-Serial link to manage all aspects of LP-Serial link operation. They are used for link maintenance, packet delimiting, packet acknowledgment, error reporting, and error recovery.

5.3.1 Control Symbol Delimiting

LP-Serial control symbols are delimited for transmission by a single 8B/10B special (control) character. The control character marks the beginning of the control symbol and immediately precedes the first bit of the control symbol. The control symbol delimiting special character is added to the control symbol before the control symbol is passed to the PCS sublayer for lane striping (if applicable) and 8B/10B encoding. Since control symbol length is constant and known, control symbols do not need end delimiters. The combined delimiter and control symbol is referred to as a delimited control symbol.

One of two special characters is used to delimit a control symbol. If the control

symbol contains a packet delimiter, the special character PD (K28.3) is used. If the control symbol does not contain a packet delimiter, the special character SC (K28.0) is used. This use of special characters provides the receiver with an “early warning” of the content of the control symbol.

5.3.2 Control Symbol Transmission

After an LP-Serial port is initialized, it shall transmit the idle sequence interrupted by one status control symbol every 1024 transmitted code-groups until the port has received an error free status control symbol from the connected port. The transmission of status control symbols indicates to the connected port that the port has completed initialization. The transmission of the idle sequence is required for the connected port to complete initialization.

After an initialized LP-Serial port has received an error free status control symbol from the connected port, the port shall transmit the idle sequence and a minimum of 15 status control symbols and shall receive an additional 6 error free status control symbols with no intervening detected errors before entering the normal operational state. Once in the normal operational state, the port may begin the transmission of packets and other (non-status) control symbols. This group of 15 status control symbols may be sent more rapidly than the minimum rate of one every 1024 code-groups. The requirement that a total of seven status control symbols be received provides a degree of link verification before packets and other control symbols are transmitted. The requirement that at least 15 status control symbols be transmitted is to minimize the probability that the connected port will not receive 7 control symbols even if a single bit error occurs.

When an LP-Serial port is in the normal operational state, it shall transmit a control symbol containing the buf_status field at least once every 1024 transmitted code-groups. To comply with this requirement, the port shall transmit a status control symbol if no other control symbol containing the buf_status field is available for transmission.

In each of the above cases, the time required to transmit 1024 code-groups shall be computed based on the aggregate unidirectional baud rate of the link. For example, a 4x link with each lane operating at 3.125 GBaud has an aggregate unidirectional baud rate of (4*3.125 =) 12.5 GBaud. In this example, the time required to transmit 1024 code-groups is (1024*10/12.5*10^9 =) 819.2 ns.

5.3.3 Embedded Control Symbols

Any control symbol that does not contain a packet delimiter may be embedded in a packet. An embedded control symbol may contain any defined encoding of stype0 and an stype1 encoding of “multicast-event” or “NOP”. Control symbols with stype1 encodings of start-of-packet, end-of-packet, stomp, restart-from-retry, or link-request cannot be embedded as they would terminate the packet.

When a control symbol is embedded in a packet, the control symbol delimiting special character shall begin on a 4-character boundary of the packet. That is, the number of packet characters between the start of the first packet character and the start of the delimited control symbol shall be a non-negative integer multiple of 4.

The manner and degree to which control symbol embedding is used on a link impacts both link and system performance. For example, embedding multicast-event control symbols allows their propagation delay and delay variation through switch processing elements to be minimized and is highly desirable for some multicast-event applications. On the other hand, embedding all packet acknowledgment control symbols rather than combining as many of them as possible with packet delimiter control symbols reduces the link bandwidth available for packet transmission and may be undesirable.

5.3.4 Multicast-Event Control Symbols

The Multicast-Event control symbol provides a mechanism through which end points are notified that some system defined event has occurred. This event can be selectively multicast through the system. Refer to Section 3.5.6 for the format of the multicast-event control symbol.

When a switch processing element receives a Multicast-Event control symbol, the switch shall forward the Multicast-Event by issuing a Multicast-Event control symbol from each port that is designated in the port's CSR as a Multicast-Event output port. A switch port shall never forward a Multicast-Event control symbol back to the device from which it received a Multicast-Event control symbol regardless of whether the port is designated a Multicast-Event output or not.

It is intended that at any given time, Multicast-Event control symbols will be sourced by a single device. However, the source device can change (in case of failover, for example). In the event that two or more Multicast-Event control symbols are received by a switch processing element close enough in time that more than one is present in the switch at the same time, at least one of the Multicast-Event control symbols shall be forwarded. The others may be forwarded or discarded (device dependent).

The system defined event whose occurrence Multicast-Event gives notice of has no required temporal characteristics. It may occur randomly, periodically, or anything in between. For instance, Multicast-Event may be used for a heartbeat function or for a clock synchronization function in a multiprocessor system.

In an application such as clock synchronization in a multiprocessor system, both the propagation time of the notification through the system and the variation in propagation time from Multicast-Event to Multicast-Event are of concern. For these reasons and the need to multicast, control symbols are used to convey Multicast-Events as control symbols have the highest priority for transmission on a link and can be embedded in packets.

While this specification places no limits on Multicast-Event forwarding delay or forwarding delay variation, switch functions should be designed to minimize these characteristics. In addition, switch functions shall include in their specifications the maximum value of Multicast-Event forwarding delay (the maximum value of Multicast-Event forwarding delay through the switch) and the maximum value of Multicast-Event forwarding delay variation (the maximum value of Multicast-Event forwarding delay through the switch minus the minimum value of Multicast-Event forwarding delay through the switch).

5.4 Packets

5.4.1 Packet Delimiting

LP-Serial packets are delimited for transmission by control symbols. Since packet length is variable, both start-of-packet and end-of-packet delimiters are required. The control symbol marking the end of a packet (packet termination) follows the end of the packet or the end of an embedded control symbol.

The following control symbols are used to delimit packets.

- Start-of-packet
- End-of-packet
- Stomp
- Restart-from-retry
- Any link-request

5.4.1.1 Packet Start

The beginning of a packet (packet start) shall be marked by a start-of-packet control symbol.

5.4.1.2 Packet Termination

A packet shall be terminated in one of the following three ways:

- The end of a packet is marked with an end-of-packet control symbol.
- The end of a packet is marked with a start-of-packet control symbol that also marks the beginning of a new packet.
- The packet is canceled by a restart-from-retry, stomp, or any link-request control symbol

5.4.2 Acknowledgment Identifier

Each packet requires an identifier to uniquely identify its acknowledgment control symbol. This identifier, the acknowledge ID (ackID), is five bits long, allowing for a range of one to thirty two outstanding unacknowledged request or

response packets between adjacent processing elements, however only up to thirty one outstanding unacknowledged packets are allowed at any one time. The first value of ackID assigned after a reset shall be 0b00000. Subsequent values of ackID shall be assigned sequentially (in increasing numerical order, wrapping back to 0 on overflow) to indicate the order of the packet transmission. The acknowledgments themselves are a number of control symbols defined in Chapter 3, "Control Symbols.

5.4.3 Packet Priority and Transaction Request Flows

Each packet has a priority, and optionally a critical request flow, that is assigned by the end point processing element that is the source of (initiates) the packet. The priority is carried in the prio field of the packet and has four possible values: 0, 1, 2, or 3. Packet priority increases with the priority value with 0 being the lowest priority and 3 being the highest. Packet priority is used in RapidIO for several purposes which include transaction ordering and deadlock prevention. The critical request flow is carried in the CRF bit. It allows a flow to be designated as a critical or preferred flow with respect to other flows of the same priority. Support for critical request flows is strongly encouraged.

When a transaction is encapsulated in a packet for transmission, the transaction request flow indicator (flowID) of the transaction is mapped into the prio field (and optionally the CRF bit) of the packet. If the CRF bit is not supported, transaction request flows A and B are mapped to priorities 0 and 1 respectively and transaction request flows C and above are mapped to priority 2 as specified in Table 5-1.

Table 5-1. Transaction Request Flow to Priority Mapping

Flow	System Priority	Request Packet Priority	Response Packet Priority
C or higher	Highest	2	3
B	Next	1	2 or 3
A	Lowest	0	1, 2, or 3

If the CRF bit is supported, the transaction request flows are mapped similarly as specified in Table 5-2. Devices that do not support the CRF bit treat it as reserved, setting it to logic 0 on transmit and ignoring it on receive.

Table 5-2. Transaction Request Flow to Priority and Critical Request Flow Mapping

Flow	System Priority	CRF Bit Setting	Request Packet Priority	Response Packet Priority
F or higher	Highest	1	2	3
E	Higher than A, B, C, D	0	2	3
D	Higher than A, B, C	1	1	2 or 3
C	Higher than A, B	0	1	2 or 3

Flow	System Priority	CRF Bit Setting	Request Packet Priority	Response Packet Priority
B	Higher than A	1	0	1, 2, or 3
A	Lowest	0	0	1, 2, or 3

The mapping of transaction request flows allows a RapidIO transport fabric to maintain transaction request flow ordering without the fabric having any knowledge of transaction types or their interdependencies. This allows a RapidIO fabric to be forward compatible as the types and functions of transactions evolve. A fabric can maintain transaction request flow ordering by simply maintaining the order of packets with the same priority and critical request flow for each path through the fabric and can maintain transaction request flow priority by never allowing a lower priority packet to pass a higher priority packet taking the same path through the fabric. In the case of congestion or some other restriction, a set CRF bit indicates that a flow of a priority can pass a flow of the same priority without the CRF bit set.

5.5 Link Maintenance Protocol

The link maintenance protocol involves a request and response pair between ports connected by an LP-Serial link. For software management, the request is generated through ports in the configuration space of the sending device. An external host write of a command to the link-request register with an I/O logical specification maintenance write transaction causes a link-request control symbol to be issued onto the output port of the device, but only one link-request can be outstanding on a link at a time.

The device that is linked to the sending device shall respond with an link-response control symbol if the link-request command required it to do so. The external host retrieves the link-response by polling the link-response register with I/O logical maintenance read transactions. A device with multiple RapidIO interfaces has a link-request and a link-response register pair for each corresponding RapidIO interface.

The automatic error recovery mechanism relies on the hardware generating link-request/input-status control symbols under the transmission error conditions described in Section 5.11.2.1, “Recoverable Errors and using the corresponding link-response information to attempt recovery.

Due to the undefined reliability of system designs, it is necessary to put a safety lockout on the reset function of the link-request/reset-device control symbol. A device receiving a link-request/reset-device control symbol shall not perform the reset function unless it has received four link-request/reset-device control symbols in a row without any intervening packets or other control symbols, except status control symbols. This will prevent spurious reset-device commands inadvertently resetting a device. The link-request/reset-device control symbol does not require a response.

The input-status command of the link-request/input-status control symbol is used by the hardware to recover from transmission errors. If the input port had stopped due to a transmission error that generated a packet-not-accepted control symbol back to the sender, the link-request/input-status control symbol acts as a link-request/restart-from-error control symbol, and the receiver is re-enabled to receive new packets after generating the link-response control symbol. The link-request/input-status control symbol may also be used to restart the receiving device if it is waiting for a restart-from-retry control symbol after retrying a packet. This situation can occur if transmission errors are encountered while trying to resynchronize the sending and receiving devices after the retry.

The link-request/input-status control symbol requires a response. A port receiving a link-request/input-status control symbol returns a link-response control symbol containing two pieces of information:

- port_status
- ackID_status

These status indicators are described in Table 3-5.

The retry-stopped state indicates that the port has retried a packet and is waiting to be restarted. This state is cleared when a restart-from-retry (or a link-request/input-status) control symbol is received. The error-stopped state indicates that the port has encountered a transmission error and is waiting to be restarted. This state is cleared when a link-request/input-status control symbol is received.

5.6 Packet Transmission Protocol

The LP-Serial protocol for packet transmission provides link level flow and error detection and recovery.

The LP-Serial link protocol uses control symbols to delimit packets when they are transmitted across an LP-Serial link as specified in Section 5.4.1, “Packet Delimiting.

The LP-Serial link protocol uses acknowledgment to monitor packet transmission. Each packet transmitted across an LP-Serial link shall be acknowledged by the receiving port with a packet acknowledgment control symbol. To associate packet acknowledgment control symbols with transmitted packets, each packet shall be assigned an ackID value that is carried in the ackID field of the packet and the packet_ackID field of the associated acknowledgment control symbol. AckID values are assigned to packets sequentially in increasing numerical order wrapping to 0 on overflow. The ackID value carried by packets indicates their order of transmission.

The LP-Serial link protocol uses retransmission to recover from packet transmission errors. To enable packet retransmission, a copy of each packet transmitted across an

LP-Serial link shall kept by the sending port until either a packet-accepted packet acknowledgment control symbol is received for the packet from the receiving port indicating that the port has received the packet without detected error and has accepted responsibility for the packet or the port determines that the packet has an encountered an unrecoverable error condition.

The LP-Serial link protocol uses the ackID value carried in each packet to ensure that no packets are lost due to transmission errors. A port shall accept packets from an LP-Serial link only in sequential ackID order, i.e. if the ackID value of the last packet accepted was N, the ackID value of the next packet that is accepted must be (N+1) modulo32.

An LP-Serial port accepts or rejects each error-free packet it receives depending on whether the port has input buffer space available at the priority level of the packet. The use of the packet-accepted, packet-retry, and restart-from-retry control symbols and the buf_status field in packet acknowledgment control symbols to control the flow of packets across an LP-Serial link is cover in Section 5.7, “Flow Control.

The LP-Serial link protocol allows a packet that is being transmitted to be canceled at any point during its transmission. Packet cancelation is covered in Section 5.8, “Canceling Packets.

The LP-Serial link protocol provides detection and recovery processes for both transmission errors and protocol violations. The enumeration of detectable errors, the detection of errors and the associated error recovery processes are covered in Section 5.11, “Error Detection and Recovery.

In order to prevent internal switch processing element internal errors, such as SRAM soft bit errors, from silently corrupting a packet and the system, switch processing elements shall maintain packet error detection coverage while a packet is passing through the switch. The simplest method for maintain packet error detection coverage is pass the packet CRC through the switch as part of the packet. This works well for all non-maintenance packets whose CRC does not change as the packets are transported from source to destination through the fabric. Maintaining error detection coverage is more complicated for maintenance packets as their hop_count and CRC change every time they pass through a switch.

In order to support transaction ordering requirements of the I/O Logical Layer specification, the LP-Serial protocol imposes packet delivery ordering requirements within the physical layer and transaction delivery ordering requirements between the physical layer and the transport layer in end point processing elements. These requirements are covered in Section 5.9, “Transaction and Packet Delivery Ordering Rules.

In order to prevent deadlock, the LP-Serial protocol imposes a set of deadlock prevention rules. These rules are covered in Section 5.10, “Deadlock Avoidance.

The LP-Serial specification does not require the use of fair bandwidth allocation

mechanisms within the transport fabric, therefore, it is possible that traffic associated with higher flow levels can starve traffic associated with lower flow levels. Any sort of starvation prevention, flow level bandwidth allocation, or fairness mechanisms are device and system dependent and are beyond the scope of this specification.

5.7 Flow Control

This section defines RapidIO LP-Serial link level flow control. The flow control operates between each pair of ports connected by an LP-Serial link. The purpose of link level flow control is to prevent the loss of packets due to a lack of buffer space in a link receiver.

The LP-Serial protocol defines two methods or modes of flow control. These are named receiver-controlled flow control and transmitter-controlled flow control. Every RapidIO LP-Serial port shall support receiver-controlled flow control. LP-Serial ports may optionally support transmitter-controlled flow control.

5.7.1 Receiver-Controlled Flow Control

Receiver-controlled flow control is the simplest and basic method of flow control. In this method, the input side of a port controls the flow of packets from its link partner by accepting or rejecting (retrying) packets on a packet by packet basis. The receiving port provides no information to its link partner about the amount of buffer space it has available for packet reception.

As a result, its link partner transmits packets with no *a priori* expectation as to whether a given packet will be accepted or rejected. A port signals its link partner that it is operating in receiver-controlled flow control mode by setting the buf_status field to all 1's in every control symbol containing the field that the port transmits. This method is named receiver-controlled flow control because the receiver makes all of the decisions about how buffers in the receiver are allocated for packet reception.

A port operating in receiver-controlled flow control mode accepts or rejects each inbound error-free packet based on whether the receiving port has enough buffer space available at the priority level of the packet. If there is enough buffer space available, the port accepts the packet and transmits a packet-accepted control symbol to its link partner that contains the ackID of the accepted packet in its packet_ackID field. This informs the port's link partner that the packet has been received without detected errors and that it has been accepted by the port. On receiving the packet-accepted control symbol, the link partner discards its copy of the accepted packet freeing buffer space in the partner.

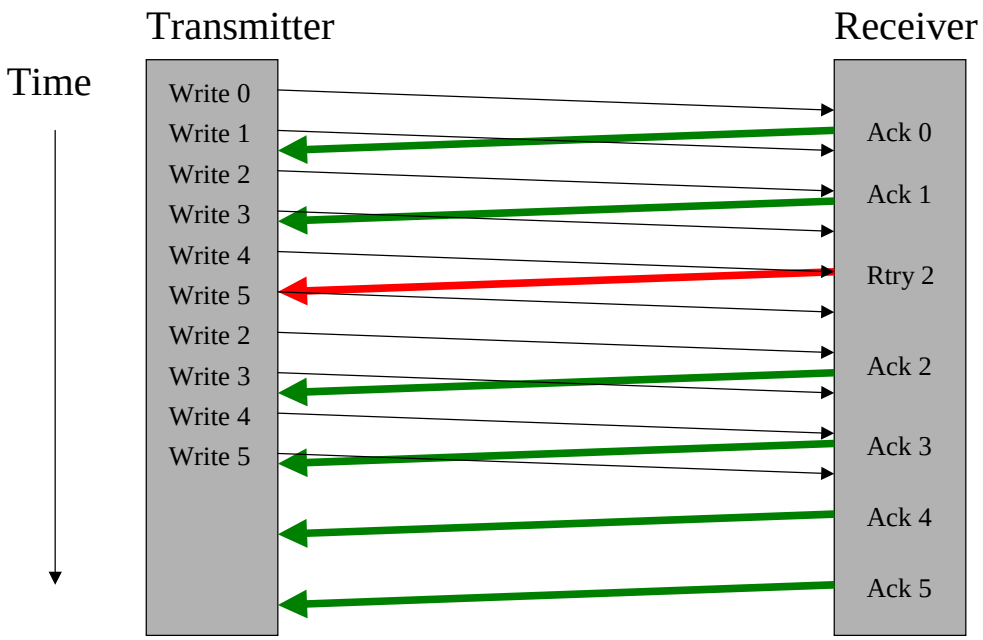
If buffer space is not available, the port rejects the packet. When a port rejects (retries) an error-free packet, it immediately enters the Input Retry-stopped state and follows the Input Retry-stopped recovery process specified in Section 5.7.2.1,

"Input Retry-Stopped Recovery Process. As part of the Input Retry-stopped recovery process, the port sends a packet-retry control symbol to its link partner indicating that the packet whose ackID is in the packet_ackID field of the control symbol and all packets subsequently transmitted by the port have been discarded by the link partner and must all be retransmitted. The control symbol also indicates that the link partner is temporarily out of buffers for packets of priority less than or equal to the priority of the retried packet.

A port that receives a packet-retry control symbol immediately enters the Output Retry-stopped state and follows the Output Retry-stopped recovery process specified in Section 5.7.2.2, "Output Retry-Stopped Recovery Process. As part of the Output Retry-stopped recovery process, the port receiving the packet-retry control symbol sends a restart-from-retry control symbol which causes its link partner to exit the Input Retry-stopped state and resume packet reception. The ackID assigned to that first packet transmitted after the restart-from-retry control symbol is the ackID of the packet that was retried.

Figure 5-2 shows an example of receiver-controlled flow control operation. In this example the transmitter is capable of sending packets faster than the receiver is able to absorb them. Once the transmitter has received a retry for a packet, the transmitter may elect to cancel any packet that is presently being transmitted since it will be discarded anyway. This makes bandwidth available for any higher priority packets that may be pending transmission.

Figure 5-2. Receiver-Controlled Flow Control



5.7.2 Transmitter-Controlled Flow Control

In transmitter-controlled flow control, the receiving port provides information to its link partner about the amount of buffer space it has available for packet reception. With this information, the sending port can allocate the use of the receiving port's receive buffers according to the number and priority of packets that the sending port has waiting for transmission without concern that one or more of the packets shall be forced to retry.

A port signals its link partner that it is operating in transmitter-controlled flow control mode by setting the buf_status field to a value different from all 1's in every control symbol containing the field that the port transmits. This method is named transmitter-controlled flow control because the transmitter makes almost all of the decisions about how the buffers in the receiver are allocated for packet reception.

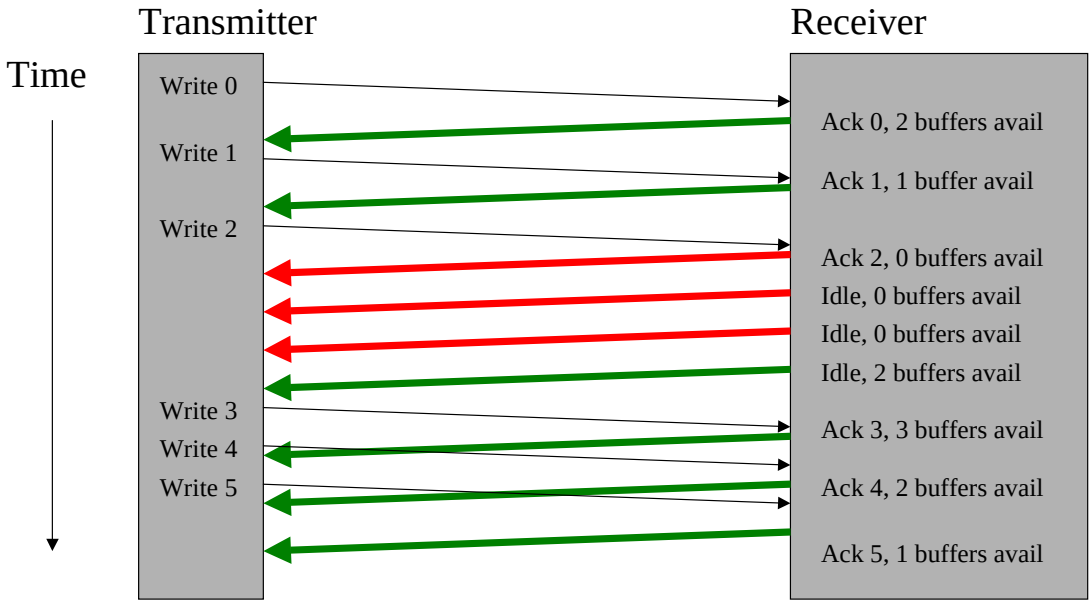
The number of free buffers that a port has available for packet reception is conveyed to its link partner by the value of the buf_status field in the control symbols that the port transmits. The value conveyed by the buf_status field is the number of maximum length packet buffers currently available for packet reception up to the limit that can be reported in the field. If a port has more buffers available than the maximum value that can be reported in the buf_status field, the port sets the field to that maximum value. A port may report a smaller number of buffers than it actually has available, but it shall not report a greater number.

A port informs its link partner when the number of free buffers available for packet reception changes. The new value of buf_status is conveyed in the buf_status field of a packet-accepted, packet-retry, or status control symbol. Each change in the number of free buffers a port has available for packet reception need not be conveyed to the link partner. However, a port shall send a control symbol containing the buf_status field to its link partner no less often than the minimum rate specified in Section 5.3.2, "Control Symbol Transmission.

A port whose link partner is operating in transmitter-control flow control mode should never receive a packet-retry control symbol from its link partner unless the port has transmitted more packets than its link partner has receive buffers, violated the rules that all input buffer may not be filled with low priority packets or there is some fault condition. If a port whose link partner is operating in transmitter-control flow control mode receives a packet-retry control symbol, the output side of the port immediately enters the Output Retry-stopped state and follows the Output Retry-stopped recovery process specified in Section 5.7.2.2, "Output Retry-Stopped Recovery Process.

A simple example of transmitter-controlled flow control is shown in Figure 5-3.

Figure 5-3. Transmitter-Controlled Flow Control



5.7.2.1 Input Retry-Stopped Recovery Process

When the input side of a port retries a packet, it immediately enters the Input Retry-stopped state. To recover from this state, the input side of the port takes the following actions.

- Discards the rejected or canceled packet without reporting a packet error and ignores all subsequently received packets while the port is in the Input Retry-stopped state.
- Causes the output side of the port to issue a packet-retry control symbol containing the ackID value of the retried packet in the packet_ackID field of the control symbol. (The packet-retry control symbol causes the output side of the link partner to enter the Output Retry-stopped state and send a restart-from-retry control symbol.)
- When a restart-from-retry control symbol is received, exit the Input Retry-stopped state and resume packet reception.

An example state machine with the behavior described in this section is included in Section A.2, “Packet Retry Mechanism.”

5.7.2.2 Output Retry-Stopped Recovery Process

To recover from the Output Retry-stopped state, the output side of a port takes the following actions.

- Immediately stops transmitting new packets. Resets the link packet acknowledgment timers for all transmitted but unacknowledged packets. (This prevents the generation of spurious time-out errors.)
- Transmits a restart-from-retry control symbol.
- Backs up to the first unaccepted packet (the retried packet) which is the packet whose ackID value is specified by the packet_ackID value contained in the packet-retry control symbol. (The packet_ackID value is also the value of ackID field the port retrying the packet expects in the first packet it receives after receiving the restart-from-retry control symbol.)
- Exits the Output Retry-stopped state and resumes transmission with either the retried packet or a higher priority packet which is assigned the ackID value contained in the packet_ackID field of the packet-retry control symbol.

An example state machine with the behavior described in this section is included in Section A.2, “Packet Retry Mechanism.”

5.7.2.3 Receive Buffer Management

In transmitter-controlled flow control, the transmitter manages the packet receive buffers in the receiver. This may be done in a number of ways, but the selected method shall not violate the rules in Section 5.10, “Deadlock Avoidance concerning the acceptance of packets by ports

One possible implementation to organize the buffers is establish watermarks and use them to progressively limit the packet priorities that can be transmitted as the effective number of free buffers in the receiver decreases. For example, RapidIO LP-Serial has four priority levels. Three non-zero watermarks are needed to progressively limit the packet priorities that may be transmitted as the effective number of free buffers decreases. Designate the three watermarks as WM0, WM1, and WM2 where WM0 > WM1 > WM2 > 0 and employ the following rules.

> If free_buffer_count >= WM0, all priority packets may be transmitted.
>
> If WM0 > free_buffer_count >= WM1, only priority 1, 2, and 3 packets may be transmitted.
>
> If WM1 > free_buffer_count >= WM2, only priority 2 and 3 packets may be transmitted.
>
> If WM2 > free_buffer_count, only priority 3 packets may be transmitted.

If this method is implemented, the initial values of the watermarks may be set by the hardware at reset as follows.

WM0 = 4

WM1 = 3

WM2 = 2

These initial values may be modified by hardware or software. The modified watermark values shall be based on the number of free buffers reported in the buf_status field of status control symbols received by the port following link initialization and before the start of packet transmission.

The three watermark values and the number of free buffers reported in the buf_status field of status control symbols received by the port following link initialization and before the start of packet transmission may be stored in a CSR. Since the maximum value of each of these four items is 30, each will fit in an 8-bit field and all four will fit in a single 32-bit CSR. If the watermarks are software setable, the three watermark fields in the CSR should be writable. For the greatest flexibility, a watermark register should be provided for each port on a device.

5.7.2.4 Effective Number of Free Receive Buffers

The number of buffers available in a port's link partner for packet reception is typically less than the value of the buf_status field most recently received from the link partner. The value in the buf_status field does not account for packets that have been transmitted by the port but not acknowledged by its link partner. The variable free_buffer_count is defined to be the effective number of free buffers available in the link partner for packet reception. The value of free_buffer_count shall be determined according to the following rules.

The port shall maintain a count of the packets that it has transmitted but that have not been acknowledged by its link partner. This count is named the outstanding_packet_count.

After link initialization and before the start of packet transmission,

```
If (received_buf_status < 31) {
        flow_control_mode = transmitter;
        free_buffer_count = received_buf_status;
        outstanding_packet_count = 0;
}
else
        flow_control_mode = receiver;
```

When a packet is transmitted by the port,

```
outstanding_packet_count =
                        outstanding_packet_count + 1;
```

When a status control symbol is received by the port,

```
free_buffer_count = received_buf_status -
```

```
                                        outstanding_packet_count;
```

When a packet-accepted control symbol is received by the port indicating that a packet has been accepted by the link partner,

```
Outstanding_packet_count =
                                    Outstanding_packet_count - 1;
free_buffer_count = received_buf_status -
                                        outstanding_packet_count;
```

When a packet-retry control symbol is received by the port indicating that a packet has been forced by the link partner to retry,

```
Outstanding_packet_count = 0;
free_buffer_count = received_buf_status;
```

When a packet-not-accepted control symbol is received by the port indicating that a packet has been rejected by the link partner because of one or more detected errors,

```
Outstanding_packet_count = 0;
free_buffer_count = 0;
```

The port then transmits a link-request/input-status (for input-status) control symbol and waits for the link partner to respond with a link-response control symbol. When the link-response control symbol is received,

```
free_buffer_count = received_buf_status;
```

5.7.2.5 Speculative Packet Transmission

A port whose link partner is operating in transmitter-controlled flow control mode may send more packets than the number of free buffers indicated by the link partner. Packets transmitted in excess of the free_buffer_count are transmitted on a speculative basis and are subject to retry by the link partner. The link partner accepts or rejects these packets on a packet by packet basis in exactly the same way it would if operating in receiver-controlled flow control mode. A port may use such speculative transmission in an attempt to maximize the utilization of the link. However, speculative transmission that results in a significant number of retries and discarded packets can reduce the effective bandwidth of the link.

5.7.3 Flow Control Mode Negotiation

Immediately following the initialization of a link, each port begins sending status control symbols to its link partner. The value of the buf_status field in these control symbols indicates to the link partner the flow control mode supported by the sending port.

The flow control mode negotiation rule is as follows:

> If the port and its link partner both support transmitter-controlled flow control, then both ports shall use transmitter-controlled flow control. Otherwise, both ports shall use receiver-controlled flow control.

5.8 Canceling Packets

When a port becomes aware of some condition that will require the packet it is currently transmitting to be retransmitted, the port may cancel the packet. This allows the port to avoid wasting bandwidth by not completing the transmission of a packet that the port knows must be retransmitted. Alternatively, the sending port may choose to complete transmission of the packet normally.

A port may cancel a packet if the port detects a problem with the packet as it is being transmitted or if the port receives a packet-retry or packet-not-accepted control symbol for a packet that is still being transmitted or that was previously transmitted. A packet-retry or packet-not-accepted control symbol can be transmitted by a port for a packet at any time after the port begins receiving the packet.

The sending device shall use the stomp control symbol, the restart-from-retry control symbol (in response to a packet-retry control symbol), or link-request/input-status control symbol (in response to a packet-not-accepted control symbol) or any link request control symbol to cancel a packet.

A port receiving a canceled packet shall drop the packet. The cancelation of a packet shall not result in the generation of any errors. If the packet was canceled because the sender received a packet-not-accepted control symbol, the error that caused the packet-not-accepted to be sent shall be reported in the normal manner.

The behavior of a port that receives a canceled packet depends on the control symbol that canceled the packet. A port that is not in an input stopped state (Retry-stopped or Error-stopped) while receiving the canceled packet and has not previously acknowledged the packet shall have the following behavior. If the packet is canceled by a link-request/input-status control symbol, the port shall drop the packet without reporting a packet error. If the packet is canceled by a restart-from-retry control symbol a protocol error has occurred and the port enters the Input Error-stopped state and follows the Input Error-stopped recovery process specified in Section 5.11.2.6, “Input Error-Stopped Recovery Process. If the packet was canceled by other than a restart-from-retry or link-request/input-status control symbol, the port shall immediately enter the Input Retry-Stopped state and follow the Input Retry-Stopped recovery process specified in Section 5.7.2.1, “Input Retry-Stopped Recovery Process. The Input Retry-Stopped recovery process includes the dropping of the canceled packet without reporting a packet error and the transmission of a packet-retry control symbol. In either case, if the packet was canceled before the packet ackID field was received by the port, the packet_ackID field of the associated retry control symbol will be set to the ackID the port expected

in the canceled packet. The packet sent following a canceled packet has the same ackID value as the canceled packet.

5.9 Transaction and Packet Delivery Ordering Rules

The rules specified in this section are required for the physical layer to support the transaction ordering rules specified in the logical layer specifications.

Transaction Delivery Ordering Rules:

1. **The physical layer of an end point processing element port shall encapsulate in packets and forwarded to the RapidIO fabric transactions comprising a given transaction request flow in the same order that the transactions were received from the transport layer of the processing element.**
2. **The physical layer of an end point processing element port shall ensure that a higher priority request transaction that it receives from the transport layer of the processing element before a lower priority request transaction with the same sourceID and the same destinationID is forwarded to the fabric before the lower priority transaction.**
3. **The physical layer of an end point processing element port shall deliver transactions to the transport layer of the processing element in the same order that the packetized transactions were received by the port.**

Packet Delivery Ordering Rules:

1. **A packet initiated by a processing element shall not be considered committed to the RapidIO fabric and does not participate in the packet delivery ordering rules until the packet has been accepted by the device at the other end of the link. (RapidIO does not have the concept of delayed or deferred transactions. Once a packet is accepted into the fabric, it is committed.)**
2. **A switch shall not alter the priority or critical request flow of a packet.**
3. **Packet forwarding decisions made by a switch processing element shall provide a consistent output port selection which is based solely on the value of the destinationID field carried in the packet.**
4. **A switch processing element shall not change the order of packets comprising a transaction request flow (packets with the same sourceID, the same destinationID, the same priority, same critical request flow and ftype != 8) as the packets pass through the switch.**
5. **A switch processing element shall not allow lower priority non-maintenance packets (ftype != 8) to pass higher priority**

non-maintenance packets with the same sourceID and destinationID as the packets pass through the switch.

6. **A switch processing element shall not allow a priority N maintenance packet (ftype = 8) to pass another maintenance packet of priority N or greater that takes the same path through the switch (same switch input port and same switch output port).**

5.10 Deadlock Avoidance

To allow a RapidIO protocol to evolve without changing the switching fabric, switch processing elements are not required, with the sole exception of ftype 8 maintenance transactions, to discern between packet types, their functions or their interdependencies. Switches, for instance, are not required to discern between packets carrying request transactions and packets carrying response transactions. As a result, it is possible for two end points, A and B to each fill all of their output buffers, the fabric connecting them and the other end point's input buffers with read requests. This would result in an input to output dependency loop in each end point in which there would be no buffer space to hold the responses necessary to complete any of the outstanding read requests.

To break input to output dependencies, end point processing elements must have the ability to issue outbound response packets even if outbound request packets awaiting transmission are congestion blocked by the connected device. Two techniques are provided to break input to output dependencies. First, a response packet (a packet carrying a response transaction) is always assigned an initial priority one priority level greater than the priority of the associated request packet (the packet carrying the associated request transaction).

This requirement is specified in Table 5-1. It breaks the dependency cycle at the request flow level. Second, the end point processing element that is the source of the response packet may additionally raise the priority of the response packet to a priority higher than the minimum required by Table 5-1 if necessary for the packet to be accepted by the connected device. This additional increase in response packet priority above the minimum required by Table 5-1 is called promotion. An end point processing element may promote a response packet only to the degree necessary for the packet to be accepted by the connected device.

The following rules define the deadlock prevention mechanism:

Deadlock Prevention Rules:

1. **A RapidIO fabric shall be dependency cycle free for all operations that do not require a response. (This rule is necessary as there are no mechanisms provided in the fabric to break dependency cycles for operations not requiring responses.)**

2. **A packet carrying a request transaction that requires a response shall not be issued at the highest priority. (This rule ensures that an end point processing element can issue a response packet at a priority higher then the priority of the associated request. This rule in combination with rule 3 are basis for the priority assignments in Table 5-1.)**
3. **A packet carrying a response shall have a priority at least one priority level higher than the priority of the associated request. (This rule in combination with rule 2 are basis for the priority assignments in Table 5-1.)**
4. **A switch processing element port shall accept an error-free packet of priority N if there is no packet of priority greater than or equal to N that was previously received by the port and is still waiting in the switch to be forwarded. (This rule has multiple implications which include but are not limited to the following. First, a switch processing element port must have at least as many maximum length packet input buffers as there are priority levels. Second, a minimum of one maximum length packet input buffer must be reserved for each priority level. A input buffer reserved for priority N might be restricted to only priority N packets or might be allowed to hold packets of priority greater than or equal to N, either approach complies with the rule.)**
5. **A switch processing element port that transmits a priority N packet that is forced to retry by the connected device shall select a packet of priority greater than N, if one is available, for transmission. (This guarantees that packets of a given priority will not block higher priority packets.)**
6. **An end point processing element port shall accept an error-free packet of priority N if the port has enough space for the packet in the input buffer space of the port allocated for packets of priority N. (Lack of input buffer space is the only reason an end point may retry a packet.)**
7. **The decision of an end point processing element to accept or retry an error-free packet of priority N shall not dependent on the ability of the end point to issue request packets of priority less than or equal to N from any of its ports. (This rule works in conjunction with rule 6. It prohibits a device's inability to issue packets of priority less than or equal to N, due to congestion in the connected device, from resulting in a lack of buffers to receive inbound packets of priority greater than or equal to N which in turn would result in packets of priority greater than or equal to N being forced to retry. The implications and some ways of complying with this rule are presented in the following paragraphs.)**

One implication of Rule 7 is that a port may not fill all of its buffers that can be used to hold packets awaiting transmission with packets carrying request transactions. If this situation was allowed to occur and the output was blocked due to congestion in

the connected device, read transactions could not be processed (no place to put the response packet), input buffer space would become filled and all subsequent inbound request packets would be forced to retry violating Rule 7.

Another implication is that a port must have a way of preventing output blockage at priority less than or equal to N, due to congestion in the connected device, from resulting in a lack of input buffer space for inbound packets of priority greater than or equal to N. There are multiple ways of doing this.

One way is to provide a port with input buffer space for at least four maximum length packets and reserve input buffer space for higher priority packets in a manner similar to that required by Rule 4 for switches. In this case, output port blockage at priority less than or equal to N will not result is blocking inbound packets of priority greater than or equal to N as any responses packets they generate will be of priority greater than N which is not congestion blocked. The port must however have the ability to select packets of priority greater than N for transmission from the packets awaiting transmission. This approach does not require the use of response packet priority promotion.

Alternatively, a port that does not have enough input buffer space for at least four maximum length packets or that does not reserve space for higher priority packets can use the promotion mechanism to increase the priority of response packets until they are accepted by the connected device. This allows output buffer space containing response packets to be freed even though all request packets awaiting transmission are congestion blocked.

As an example, suppose an end point processing element has a blocked input port because all available resources are being used for a response packet that the processing element is trying to send. If the response packet is retried by the downstream processing element, raising the priority of the response packet until it is accepted allows the processing element’s input port to unblock so the system can make forward progress.

5.11 Error Detection and Recovery

Error detection and recovery is becoming a more important issue for many systems. The LP-Serial specification provides extensive error detection and recovery by combining retry protocols with cyclic redundancy codes, the selection of delimiter control characters and response timers.

One feature of the error protection strategy is that with the sole exception of maintenance packets, the CRC value carried in a packet remains unchanged as the packet moves through the fabric. The CRC carried in a maintenance packet must be regenerated at each switch as the hop count changes.

5.11.1 Lost Packet Detection

Some types of errors, such as a lost request or response packet or a lost acknowledgment, result in a system with hung resources. To detect this type of error there shall be time-out counters that expire when sufficient time has elapsed without receiving the expected response from the system. Because the expiration of one of these timers should indicate to the system that there is a problem, this time interval should be set long enough so that a false time-out is not signaled. The response to this error condition is implementation dependent.

The RapidIO specifications require time-out counters for the physical layer, the port link time-out counters, and counters for the logical layer, the port response time-out counters. The interpretation of the counter values is implementation dependent, based on a number of factors including link clock rate, the internal clock rate of the device, and the desired system behavior.

The physical layer time-out occurs between the transmission of a packet and the receipt of an acknowledgment control symbol. This time-out interval is likely to be comparatively short because the packet and acknowledgment pair must only traverse a single link.

The logical layer time-out occurs between the issuance of a request packet that requires a response packet and the receipt of that response packet. This time-out is counted from the time that the logical layer issues the packet to the physical layer to the time that the associated response packet is delivered from the physical layer to the logical layer. Should the physical layer fail to complete the delivery of the packet, the logical layer time-out will occur. This time-out interval is likely to be comparatively long because the packet and response pair have to traverse the fabric at least twice and be processed by the target. Error handling for a response time-out is implementation dependent.

Certain GSM operations may require two response transactions, and both must be received for the operation to be considered complete. In the case of a device implementation with multiple links, one response packet may be returned on the same link where the operation was initiated and the other response packet may be returned on a different link. If this behavior is supported by the issuing processing element, the port response time-out implementation must look for both responses, regardless on which links they are returned.

5.11.2 Link Behavior Under Error

The LP-Serial link uses an error detection and retransmission protocol to protect against and recover from transmission errors. Transmission error detection is done at the input port, and all transmission error recovery is also initiated at the input port.

The protocol requires that each packet transmitted be acknowledged by the receiving port and that a copy of each transmitted packet be retained by the sender

until the sender receives a packet-accepted control symbol acknowledgment for the packet or the sending port determines that the packet has encountered an unrecoverable error. If the receiving port detects a transmission error in a packet, the port sends a packet-not-accepted control symbol acknowledgment back to the sender indicating that the packet was corrupted as received. After a link-request/input-status and link-response control symbol exchange, the sender begins retransmission with either the packet that was corrupted during transmission or a higher priority packet if one is awaiting transmission.

All packets corrupted in transmission are retransmitted. The number of times a packet may be retransmitted before the sending port determines that the packet has encountered an unrecoverable condition is implementation dependent.

5.11.2.1 Recoverable Errors

The following four basic types of errors are detected by an LP-Serial port:

- An idle sequence error
- A control symbol error
- A packet error
- A time-out waiting for an acknowledgment control symbol

5.11.2.2 Idle Sequence Errors

The idle sequence is comprised of A, K, and R (8B/10B special) characters. If an input port detects an invalid character or any valid character other then A, K, or R in an idle sequence, it shall enter the Input Error-stopped state and follow the Input Error-stopped recovery process specified in Section 5.11.2.6, “Input Error-Stopped Recovery Process.

To limit input port complexity, the port is no required to determine the specific error that resulted in an idle sequence error. Following are several examples of idle sequence errors.

- A single bit transmission error can change an /A/, /K/, or /R/ code-group into a /Dx.y/ (data) code-group which is illegal in an idle sequence.
- A single bit transmission error can change an /A/, /K/, or /R/ code-group into an invalid code-group.
- A single bit transmission error can change an /SP/ or /PD/ (control symbol delimiters) into an invalid code-group.

5.11.2.3 Control Symbol Errors

There are two types of detectable control symbol errors

- An uncorrupted control symbol that violates the link protocol
- A corrupted control symbol

5.11.2.3.1 Link Protocol Violations

The reception of a control symbol with no detected corruption that violates the link protocol shall cause the receiving port to immediately enter the appropriate Error-stopped state. Stype1 control symbol protocol errors shall cause the receiving port to immediately enter the Input Error-stopped state if not already in the Input Error-stopped state and follow the Error-stopped recovery process specified in Section 5.11.2.6, “Input Error-Stopped Recovery Process”. Stype0 control symbol protocol errors shall cause the receiving port to immediately enter the Output Error-stopped state if not already in the Output Error-stopped state and follow the Output Error-stopped recovery process specified in Section 5.11.2.7, “Output Error-Stopped Recovery Process”. If both stype0 and stype1 control symbols contain protocol errors, then the receiving port shall enter both Error-stopped states and follow both error recovery processes.

Link protocol violations include the following:

- Unexpected packet-accepted, packet-retry, or packet-not-accepted control symbol
- Packet acknowledgment control symbol with an unexpected packet_ackID value
- Link time-out while waiting for an acknowledgment control symbol

The following is an example of a link protocol violation and recovery. A sender transmits packets labeled ackID 2, 3, 4, and 5. It receives acknowledgments for packets 2, 4, and 5, indicating a probable error associated with ackID 3. The sender then stops transmitting new packets and sends a link-request/input-status (restart-from-error) control symbol to the receiver. The receiver then returns a link-response control symbol indicating which packets it has received properly. These are the possible responses and the sender’s resulting behavior:

- expecting ackID = 3 - sender must retransmit packets 3, 4, and 5
- expecting ackID = 4 - sender must retransmit packets 4 and 5
- expecting ackID = 5 - sender must retransmit packet 5
- expecting ackID = 6 - receiver got all packets, resume operation
- expecting ackID = anything else - fatal (non-recoverable) error

5.11.2.3.2 Corrupted Control symbols

The reception of a control symbol with detected corruption shall cause the receiving port to immediately enter the Input Error-stopped state and follow the Input Error-stopped recovery process specified in Section 5.11.2.6, “Input Error-Stopped Recovery Process. For this type of error, the packet-not-accepted control symbol sent by the output side of the port as part of the recovery process shall have an undefined packet_ackID value.

Input ports detect the following types of control symbol corruption.

- A control symbol containing invalid characters or valid but non-data characters
- A control symbol with an incorrect CRC value

5.11.2.4 Packet Errors

The reception of a packet with detected corruption shall cause the receiving port to immediately enter the Input Error-stopped state and follow the Input Error-stopped recovery process specified in Section 5.11.2.6, “Input Error-Stopped Recovery Process.

Input ports detect the following types of packet corruption

- Packet with an unexpected ackID value
- Packet with an incorrect CRC value
- Packet containing invalid characters or valid non-data characters
- Packet that overruns some defined boundary such as the maximum data payload.

An optional alternative behavior, error recovery suppression, can be enabled with the Re-transmit Suppression Mask field in the Port *n* Control CSR in systems that can withstand “lossy” transaction request flows (transaction request flows that are not required to have guaranteed data delivery). If this behavior is supported and enabled, when a processing element receives a corrupted packet as indicated by a bad CRC value, the processing element sends a packet-accepted control symbol, discards the corrupted packet, does not transition the “Input Error-stopped” state, and accepts new packets normally. The reporting mechanism for such an event is implementation dependent. Note that care must be taken when using this feature as it is possible that the CRF and/or the priority bits themselves could be corrupt.

5.11.2.5 Link Time-Out

A link time-out while waiting for an acknowledgment control symbol is handled as a link protocol violation as described in Section 5.11.2.3.1, “Link Protocol Violations

5.11.2.6 Input Error-Stopped Recovery Process

When the input side of a port detects a transmission error, it immediately enters the Input Error-stopped state. To recover from this state, the input side of the port takes the following actions.

- Record the error(s) that caused the port to enter the Input Error-stopped state.
- If the detected error(s) occurred in a control symbol or packet, discard the control symbol or packet.
- Ignore all subsequently received packets while the port is in the Input Error-stopped state.

- Cause the output side of the port to issue a packet-not-accepted control symbol. The packet_ackID field of the control symbol contains an undefined value. (The packet-not-accepted control symbol causes the output side of the receiving port to enter the Output Error-stopped state and send a link-request/input-status control symbol.)
- When an link-request/input-status control symbol is received, cause the output side of the port to issue a link-response control symbol, exit the Input Error-stopped state and resume packet reception.

An example state machine with the behavior described in this section is included in Section A.3, “Error Recovery”.

5.11.2.7 Output Error-Stopped Recovery Process

To recover from the Output Error-stopped state, the output side of a port takes the following actions.

- Immediately stops transmitting new packets.
- Resets the link packet acknowledgment timers for all transmitted but unacknowledged packets. (This prevents the generation of spurious time-out errors.)
- Transmits an input-status link-request/input-status (restart-from-error) control symbol. (The input status link-request/input-status control symbol causes the receiving port to transmit a link-response control symbol that contains the input_status and ackID_status of the input side of the port. The ackID_status is the ackID value that is expected in the next packet that the port receives.)
- When the link-response is received, the port backs up the first unaccepted packet, exits the Output Error-stopped state and resumes transmission with either the first unaccepted packet or a higher priority packet.

An example state machine with the behavior described in this section is included in Section A.3, “Error Recovery”.

5.12 Power Management

Power management is currently beyond the scope of this specification and is implementation dependent. A device that supports power management features can make these features accessible to the rest of the system using the device’s local configuration registers.

Chapter 6 LP-Serial Registers

6.1 Introduction

This chapter describes the visible register set that allows an external processing element to determine the capabilities, configuration, and status of a processing element using this physical layer specification. This chapter only describes registers or register bits defined by this specification. Refer to the other RapidIO logical, transport, and physical specifications of interest to determine a complete list of registers and bit definitions. All registers are 32-bits and aligned to a 32-bit boundary.

There are four types of 1x/4x LP-Serial devices, an end point device, an end point device with additional software recovery registers, an end point free (or switch) device, and an end point free device with additional software recovery registers. Each has a different set of CSRs, specified in Section 6.5, Section 6.6, Section 6.7, and Section 6.8, respectively. All four device types have the same CARs, specified in Section 6.4.

6.2 Register Map

These registers utilize the Extended Features blocks and can be accessed using *RapidIO Part 1: Input/Output Logical Specification* maintenance operations. Any register offsets not defined are considered reserved for this specification unless otherwise stated. Other registers required for a processing element are defined in other applicable RapidIO specifications and by the requirements of the specific device and are beyond the scope of this specification. Read and write accesses to reserved register offsets shall terminate normally and not cause an error condition in the target device.

The Extended Features pointer (EF_PTR) defined in the RapidIO logical specifications contains the offset of the first Extended Features block in the Extended Features data structure for a device. The 1x/4x LP-Serial physical features block shall exist in any position in the Extended Features data structure and shall exist in any portion of the Extended Features Space in the register address map for the device.

Register bits defined as reserved are considered reserved for this specification only. Bits that are reserved in this specification may be defined in another RapidIO specification.

Table 6-1. 1x/4x LP-Serial Register Map

Configuration Space Byte Offset	Register Name
0x0-C	Reserved
0x10	Processing Element Features CAR
0x14–FC	Reserved
0x100–FFFC	Extended Features Space
0x10000–FFFFFC	Implementation-defined Space

6.3 Reserved Register and Bit Behavior

Table 6-2 describes the required behavior for accesses to reserved register bits and reserved registers for the RapidIO register space,

Table 6-2. Configuration Space Reserved Access Behavior

Byte Offset	Space Name	Item	Initiator behavior	Target behavior
0x0–3C	Capability Register Space (CAR Space - this space is read-only)	Reserved bit	read - ignore returned value[1]	read - return logic 0
			write -	write - ignored
		Implementation-defined bit	read - ignore returned value unless implementation-defined function understood	read - return implementation-defined value
			write -	write - ignored
		Reserved register	read - ignore returned value	read - return logic 0s
			write -	write - ignored
0x40–FC	Command and Status Register Space (CSR Space)	Reserved bit	read - ignore returned value	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

Table 6-2. Configuration Space Reserved Access Behavior (Continued)

Byte Offset	Space Name	Item	Initiator behavior	Target behavior
0x100–FFFC	Extended Features Space	Reserved bit	read - ignore returned value	read - return logic 0
			write - preserve current value	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
0x10000–FFFFFC	Implementation-defined Space	Reserved bit and register	All behavior implementation-defined	

[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.

6.4 Capability Registers (CARs)

Every processing element shall contain a set of registers that allows an external processing element to determine its capabilities using the I/O logical maintenance read operation. All registers are 32 bits wide and are organized and accessed in 32-bit (4 byte) quantities, although some processing elements may optionally allow larger accesses. CARs are read-only. Refer to Table 6-2 for the required behavior for accesses to reserved registers and register bits.

CARs are big-endian with bit 0 the most significant bit.

6.4.1 Processing Element Features CAR (Configuration Space Offset 0x10)

The processing element features CAR identifies the major functionality provided by the processing element. The bit settings are shown in Table 6-3.

Table 6-3. Bit Settings for Processing Element Features CAR

Bits	Name	Description
0–24	—	Reserved
25	Re-transmit Suppression Support	PE supports suppression of error recovery on packet CRC errors 0b0 - The error recovery suppression option is not supported by the PE 0b1 - The error recovery suppression option is supported by the PE
26	CRF Support	PE supports the Critical Request Flow (CRF) indicator 0b0 - Critical Request Flow is not supported 0b1 - Critical Request Flow is supported
27–31	—	Reserved

6.5 Generic End Point Devices

This section describes the 1x/4x LP-Serial registers for a general end point device. This Extended Features register block is assigned Extended Features block ID=0x0001.

6.5.1 Register Map

Table 6-4 shows the register map for generic RapidIO 1x/4x LP-Serial end point devices. The Block Offset is the offset relative to the 16-bit Extended Features Pointer (EF_PTR) that points to the beginning of the block.

The address of a byte in the block is calculated by adding the block byte offset to EF_PTR that points to the beginning of the block. This is denoted as [EF_PTR+xx] where xx is the block byte offset in hexadecimal.

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 + 0xBC]. Register map offset [EF_PTR + 0xC0] can be used for another Extended Features block.

Table 6-4. LP-Serial Register Map - Generic End Point Devices

	Block Byte Offset	Register Name
General	0x0	1x/4x LP-Serial Register Block Header
	0x4–1C	Reserved
	0x20	Port Link Time-Out Control CSR
	0x24	Port Response Time-Out Control CSR
	0x28-38	Reserved
	0x3C	Port General Control CSR
Port 0	0x40-54	Reserved
	0x58	Port 0 Error and Status CSR
	0x5C	Port 0 Control CSR
Port 1	0x60-74	Reserved
	0x78	Port 1 Error and Status CSR
	0x7C	Port 1 Control CSR
Ports 2-14	0x80–218	Assigned to Port 2-14 CSRs

Table 6-4. LP-Serial Register Map (Continued) - Generic End Point Devices

	Block Byte Offset	Register Name
Port 15	0x220-234	Reserved
	0x238	Port 15 Error and Status CSR
	0x23C	Port 15 Control CSR

6.5.2 Command and Status Registers (CSRs)

Refer to Table 6-2 for the required behavior for accesses to reserved registers and register bits.

6.5.2.1 1x/4x LP-Serial Register Block Header (Block Offset 0x0)

The 1x/4x LP-Serial register block header register contains the EF_PTR to the next extended features block and the EF_ID that identifies this as the generic end point 1x/4x LP-Serial register block header.

Table 6-5. Bit Settings for 1x/4x LP-Serial Register 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	0x0001	Hard wired Extended Features ID

6.5.2.2 Port Link Time-out Control CSR (Block Offset 0x20)

The port link time-out control register contains the time-out timer value for all ports on a device. This time-out is for link events such as sending a packet to receiving the corresponding acknowledge, and sending a link-request to receiving the corresponding link-response. The reset value is the maximum time-out interval, and represents between 3 and 6 seconds.

Table 6-6. Bit Settings for Port Link Time-out Control CSR

Bit	Name	Reset Value	Description
0–23	time-out value	All 1s	time-out interval value
24-31	—		Reserved

6.5.2.3 Port Response Time-out Control CSR (Block Offset 0x24)

The port response time-out control register contains the time-out timer count for all ports on a device. This time-out is for sending a request packet to receiving the corresponding response packet.The reset value is the maximum time-out interval, and represents between 3 and 6 seconds.

Table 6-7. Bit Settings for Port Response Time-out Control CSR

Bit	Name	Reset Value	Description
0–23	time-out value	All 1s	time-out interval value
24-31	—		Reserved

6.5.2.4 Port General Control CSR (Block Offset 0x3C)

The bits accessible through the Port General Control CSR are bits that apply to all ports on a device. There is a single copy of each such bit per device. These bits are also accessible through the Port General Control CSR of any other physical layers implemented on a device.

Table 6-8. Bit Settings for Port General Control CSRs

Bit	Name	Reset Value	Description
0	Host	see footnote[1]	A Host device is a device that is responsible for system exploration, initialization, and maintenance. Agent or slave devices are typically initialized by Host devices. 0b0 - agent or slave device 0b1 - host device
1	Master Enable	see footnote[2]	The Master Enable bit controls whether or not a device is allowed to issue requests into the system. If the Master Enable is not set, the device may only respond to requests. 0b0 - processing element cannot issue requests 0b1 - processing element can issue requests
2	Discovered	see footnote[3]	This device has been located by the processing element responsible for system configuration 0b0 - The device has not been previously discovered 0b1 - The device has been discovered by another processing element
3-31	—		Reserved

[1]The Host reset value is implementation dependent

[2]The Master Enable reset value is implementation dependent

[3]The Discovered reset value is implementation dependent

6.5.2.5 Port *n* Error and Status CSRs (Block Offsets 0x58, 78, ..., 238)

These registers are accessed when a local processor or an external device wishes to examine the port error and status information.

Table 6-9. Bit Settings for Port *n* Error and Status CSRs

Bit	Name	Reset Value	Description
0-10	—		Reserved
11	Output Retry-encountered	0b0	Output port has encountered a retry condition.This bit is set when bit 13 is set. Once set, remains set until written with a logic 1 to clear.
12	Output Retried	0b0	Output port has received a packet-retry control symbol and can not make forward progress. This bit is set when bit 13 is set and is cleared when a packet-accepted or a packet-not-accepted control symbol is received (read-only).
13	Output Retry-stopped	0b0	Output port has received a packet-retry control symbol and is in the “output retry-stopped” state (read-only).
14	Output Error-encountered	0b0	Output port has encountered (and possibly recovered from) a transmission error. This bit is set when bit 15 is set. Once set, remains set until written with a logic 1 to clear.
15	Output Error-stopped	0b0	Output is in the “output error-stopped” state (read-only).
16-20	—		Reserved
21	Input Retry-stopped	0b0	Input port is in the “input retry-stopped” state (read-only).
22	Input Error-encountered	0b0	Input port has encountered (and possibly recovered from) a transmission error. This bit is set when bit 23 is set. Once set, remains set until written with a logic 1 to clear.
23	Input Error-stopped	0b0	Input port is in the “input error-stopped” state (read-only).
24-26	—		Reserved
27	Port-write Pending	0b0	Port has encountered a condition which required it to initiate a Maintenance Port-write operation This bit is only valid if the device is capable of issuing a maintenance port-write transaction. Once set remains set until written with a logic 1 to clear.
28	—		Reserved
29	Port Error	0b0	Input or output port has encountered an error from which hardware was unable to recover. Once set, remains set until written with a logic 1 to clear.
30	Port OK	0b0	The input and output ports are initialized and the port is exchanging error-free control symbols with the attached device (read-only).
31	Port Uninitialized	0b1	Input and output ports are not initialized. This bit and bit 30 are mutually exclusive (read-only).

6.5.2.6 Port *n* Control CSR (Block Offsets 0x5C, 7C, ..., 23C)

The port *n* control registers contain control register bits for individual ports on a processing element.

Table 6-10. Bit Settings for Port *n* Control CSRs

Bit	Name	Reset Value	Description
0-1	Port Width	see footnote[1]	Hardware width of the port (read-only): 0b00 - Single-lane port 0b01 - Four-lane port 0b10 - 0b11 - Reserved
2-4	Initialized Port Width	see footnote[2]	Width of the ports after initialized (read only): 0b000 - Single-lane port, lane 0 0b001 - Single-lane port, lane 2 0b010 - Four-lane port 0b011 - 0b111 - Reserved
5-7	Port Width Override	0b000	Soft port configuration to override the hardware size: 0b000 - No override 0b001 - Reserved 0b010 - Force single lane, lane 0 0b011 - Force single lane, lane 2 0b100 - 0b111 - Reserved If the port size is overridden, the port will re-initialize to change to the requested size.
8	Port Disable	0b0	Port disable: 0b0 - port receivers/drivers are enabled 0b1 - port receivers/drivers are disabled and are unable to receive/transmit any packets or control symbols
9	Output Port Enable	see footnote[3]	Output port transmit enable: 0b0 - port is stopped and not enabled to issue any packets except to route or respond to I/O logical MAINTENANCE packets. Control symbols are not affected and are sent normally. This is the recommended state after device reset. 0b1 - port is enabled to issue any packets
10	Input Port Enable	see footnote[4]	Input port receive enable: 0b0 - port is stopped and only enabled to route or respond I/O logical MAINTENANCE packets. Other packets generate packet-not-accepted control symbols to force an error condition to be signaled by the sending device. Control symbols are not affected and are received and handled normally. This is the recommended state after device reset. 0b1 - port is enabled to respond to any packet
11	Error Checking Disable	0b0	This bit disables all RapidIO transmission error checking 0b0 - Error checking and recovery is enabled 0b1 - Error checking and recovery is disabled Device behavior when error checking and recovery is disabled and an error condition occurs is undefined
12	Multicast-event Participant	see footnote[5]	Send incoming Multicast-event control symbols to this port (multiple port devices only)
13	—		Reserved

Table 6-10. Bit Settings for Port *n* Control CSRs

Bit	Name	Reset Value	Description
14	Enumeration Boundary	see footnote[6]	An enumeration boundary aware system enumeration algorithm shall honor this flag. The algorithm, on either the ingress or the egress port, shall not enumerate past a port with this bit set. This provides for software enforced enumeration domains within the RapidIO fabric.
15-19	—		Reserved
20-27	Re-transmit Suppression Mask	0x00	Suppress packet re-transmission on CRC error. For devices that support CRF: 0b0000_0000 - Error recovery suppression disabled 0bxxxx_xxx1 - Suppress CRF=0, priority 0 re-transmission 0bxxxx_xx1x - Suppress CRF=0, priority 1 re-transmission 0bxxxx_x1xx - Suppress CRF=0, priority 2 re-transmission 0bxxxx_1xxx - Suppress CRF=0, priority 3 re-transmission 0bxxx1_xxxx - Suppress CRF=1, priority 0 re-transmission 0bxx1x_xxxx - Suppress CRF=1, priority 1 re-transmission 0bx1xx_xxxx - Suppress CRF=1, priority 2 re-transmission 0b1xxx_xxxx - Suppress CRF=1, priority 3 re-transmission For devices that do not support CRF: 0b0000_0000 - Error recovery suppression disabled 0b0000_xxx1 - Suppress priority 0 re-transmission 0b0000_xx1x - Suppress priority 1 re-transmission 0b0000_x1xx - Suppress priority 2 re-transmission 0b0000_1xxx - Suppress priority 3 re-transmission 0b0001_0000 - reserved ... 0b1111_1111 - reserved This field is only valid if bit 25 of the Processing Element Features CAR is set.
28-30	—		Reserved
31	Port Type		This indicates the port type (read only) 0b0 - Reserved 0b1 - Serial port

[1]The Port Width reset value is implementation dependent

[2]The Initialized Port Width reset value is implementation dependent

[3]The Output Port Enable reset value is implementation dependent

[4]The Input Port Enable reset value is implementation dependent

[5]The Multicast-event Participant reset value is implementation dependent

[6]The Enumeration Boundary reset value is implementation dependent. Provision shall be made to allow the reset value of this bit to be configurable on a per system basis if this feature is supported.

6.6 Generic End Point Devices, software assisted error recovery option

This section describes the 1x/4x LP-Serial registers for a general end point device that supports software assisted error recovery. This is most useful for devices that for whatever reason do not want to implement error recovery in hardware and to allow software to generate link-request control symbols and see the results of the responses. This Extended Features register block is assigned Extended Features block ID=0x0002.

6.6.1 Register Map

Table 6-11 shows the register map for generic RapidIO 1x/4x LP-Serial end point devices with software assisted error recovery. The Block Offset is the offset based on the Extended Features pointer (EF_PTR) to this block. 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 + 0xBC]. Register map offset [EF_PTR + 0xC0] can be used for another Extended Features block.

Table 6-11. LP-Serial Register Map - Generic End Point Devices (SW assisted)

	Block Byte Offset	Register Name
General	0x0	1x/4x LP-Serial Register Block Header
	0x4–1C	Reserved
	0x20	Port Link Time-Out Control CSR
	0x24	Port Response Time-Out Control CSR
	0x28-38	Reserved
	0x3C	Port General Control CSR
Port 0	0x40	Port 0 Link Maintenance Request CSR
	0x44	Port 0 Link Maintenance Response CSR
	0x48	Port 0 Local ackID Status CSR
	0x4C-54	Reserved
	0x58	Port 0 Error and Status CSR
	0x5C	Port 0 Control CSR
Port 1	0x60	Port 1 Link Maintenance Request CSR
	0x64	Port 1 Link Maintenance Response CSR
	0x68	Port 1 Local ackID Status CSR
	0x6C-74	Reserved
	0x78	Port 1 Error and Status CSR
	0x7C	Port 1 Control CSR

Table 6-11. LP-Serial Register Map (Continued)- Generic End Point Devices (SW assisted)

	Block Byte Offset	Register Name
Ports 2-14	0x80–218	Assigned to Port 2-14 CSRs
Port 15	0x220	Port 15 Link Maintenance Request CSR
	0x224	Port 15 Link Maintenance Response CSR
	0x228	Port 15 Local ackID Status CSR
	0x22C-234	Reserved
	0x238	Port 15 Error and Status CSR
	0x23C	Port 15 Control CSR

6.6.2 Command and Status Registers (CSRs)

Refer to Table 6-2 for the required behavior for accesses to reserved registers and register bits.

6.6.2.1 1x/4x LP-Serial Register Block Header (Block Offset 0x0)

The 1x/4x LP-Serial register block header register contains the EF_PTR to the next extended features block and the EF_ID that identifies this as the generic end point 1x/4x LP-Serial register block header.

Table 6-12. Bit Settings for 1x/4x LP-Serial Register 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	0x0002	Hard wired Extended Features ID

6.6.2.2 Port Link Time-out Control CSR (Block Offset 0x20)

The port link time-out control register contains the time-out timer value for all ports on a device. This time-out is for link events such as sending a packet to receiving the corresponding acknowledge and sending a link-request to receiving the corresponding link-response. The reset value is the maximum time-out interval, and represents between 3 and 6 seconds.

Table 6-13. Bit Settings for Port Link Time-out Control CSR

Bit	Name	Reset Value	Description
0–23	time-out value	All 1s	time-out interval value
24-31	—		Reserved

6.6.2.3 Port Response Time-out Control CSR (Block Offset 0x24)

The port response time-out control register contains the time-out timer count for all ports on a device. This time-out is for sending a request packet to receiving the corresponding response packet.The reset value is the maximum time-out interval, and represents between 3 and 6 seconds.

Table 6-14. Bit Settings for Port Response Time-out Control CSR

Bit	Name	Reset Value	Description
0–23	time-out value	All 1s	time-out interval value
24-31	—		Reserved

6.6.2.4 Port General Control CSR (Block Offset 0x3C)

The port general control register contains control register bits applicable to all ports on a processing element.

Table 6-15. Bit Settings for Port General Control CSRs

Bit	Name	Reset Value	Description
0	Host	see footnote[1]	A Host device is a device that is responsible for system exploration, initialization, and maintenance. Agent or slave devices are initialized by Host devices. 0b0 - agent or slave device 0b1 - host device
1	Master Enable	see footnote[2]	The Master Enable bit controls whether or not a device is allowed to issue requests into the system. If the Master Enable is not set, the device may only respond to requests. 0b0 - processing element cannot issue requests 0b1 - processing element can issue requests
2	Discovered	see footnote[3]	This device has been located by the processing element responsible for system configuration 0b0 - The device has not been previously discovered 0b1 - The device has been discovered by another processing element
3-31	—		Reserved

[1]The Host reset value is implementation dependent

[2]The Master Enable reset value is implementation dependent

[3]The Discovered reset value is implementation dependent

6.6.2.5 Port *n* Link Maintenance Request CSRs (Block Offsets 0x40, 60, ..., 220)

The port link maintenance request registers are accessible both by a local processor and an external device. A write to one of these registers generates a link-request control symbol on the corresponding RapidIO port interface.

Table 6-16. Bit Settings for Port *n* Link Maintenance Request CSRs

Bit	Name	Reset Value	Description
0–28	—		Reserved
29-31	Command	0b000	Command to be sent in the link-request control symbol. If read, this field returns the last written value.

6.6.2.6 Port *n* Link Maintenance Response CSRs (Block Offsets 0x44, 64, ..., 224)

The port link maintenance response registers are accessible both by a local processor and an external device. A read to this register returns the status received in a link-response control symbol. The ackID_status and port_status fields are defined in Table 3-3 and Table 3-5. This register is read-only.

Table 6-17. Bit Settings for Port *n* Link Maintenance Response CSRs

Bit	Name	Reset Value	Description
0	response_valid	0b0	If the link-request causes a link-response, this bit indicates that the link-response has been received and the status fields are valid. If the link-request does not cause a link-response, this bit indicates that the link-request has been transmitted. This bit automatically clears on read.
1-21	—		Reserved
22-26	ackID_status	0b00000	ackID status field from the link-response control symbol
27-31	link_status	0b00000	link status field from the link-response control symbol

6.6.2.7 Port *n* Local ackID CSRs (Block Offsets 0x48, 68, ..., 228)

The port link local ackID status registers are accessible both by a local processor and an external device. A read to this register returns the local ackID status for both the out and input ports of the device.

Table 6-18. Bit Settings for Port *n* Local ackID Status CSRs

Bit	Name	Reset Value	Description
0	Clr_outstanding_ackIDs	0b0	Writing 0b1 to this bit causes all outstanding unacknowleged packets to be discarded. This bit should only be written when trying to recover a failed link. This bit is always logic 0 when read.
1-2	—		Reserved
3-7	Inbound_ackID	0b00000	Input port next expected ackID value

Table 6-18. Bit Settings for Port *n* Local ackID Status CSRs (Continued)

Bit	Name	Reset Value	Description
8-18	—		Reserved
19-23	Outstanding_ackID	0x00000	Output port unacknowledged ackID status. Next expected acknowledge control symbol ackID field that indicates the ackID value expected in the next received acknowledge control symbol.
24-26	—		Reserved
27-31	Outbound_ackID	0b00000	Output port next transmitted ackID value. Software writing this value can force retransmission of outstanding unacknowledged packets in order to manually implement error recovery.

6.6.2.8 Port *n* Error and Status CSRs (Block Offset 0x58, 78, ..., 238)

These registers are accessed when a local processor or an external device wishes to examine the port error and status information.

Table 6-19. Bit Settings for Port *n* Error and Status CSRs

Bit	Name	Reset Value	Description
0-10	—		Reserved
11	Output Retry-encountered	0b0	Output port has encountered a retry condition.This bit is set when bit 13 is set. Once set, remains set until written with a logic 1 to clear.
12	Output Retried	0b0	Output port has received a packet-retry control symbol and can not make forward progress. This bit is set when bit 13 is set and is cleared when a packet-accepted or a packet-not-accepted control symbol is received (read-only).
13	Output Retry-stopped	0b0	Output port has received a packet-retry control symbol and is in the “output retry-stopped” state (read-only).
14	Output Error-encountered	0b0	Output port has encountered (and possibly recovered from) a transmission error. This bit is set when bit 15 is set. Once set, remains set until written with a logic 1 to clear.
15	Output Error-stopped	0b0	Output is in the “output error-stopped” state (read-only).
16-20	—		Reserved
21	Input Retry-stopped	0b0	Input port is in the “input retry-stopped” state (read-only).
22	Input Error-encountered	0b0	Input port has encountered (and possibly recovered from) a transmission error. This bit is set when bit 23 is set. Once set, remains set until written with a logic 1 to clear.
23	Input Error-stopped	0b0	Input port is in the “input error-stopped” state (read-only).
24-26	—		Reserved
27	Port-write Pending	0b0	Port has encountered a condition which required it to initiate a Maintenance Port-write operation This bit is only valid if the device is capable of issuing a maintenance port-write transaction. Once set remains set until written with a logic 1 to clear.
28	—		Reserved

Table 6-19. Bit Settings for Port *n* Error and Status CSRs

Bit	Name	Reset Value	Description
29	Port Error	0b0	Input or output port has encountered an error from which hardware was unable to recover. Once set, remains set until written with a logic 1 to clear.
30	Port OK	0b0	The input and output ports are initialized and the port is exchanging error-free control symbols with the attached device (read-only).
31	Port Uninitialized	0b1	Input and output ports are not initialized. This bit and bit 30 are mutually exclusive (read-only).

6.6.2.9 Port *n* Control CSR (Block Offsets 0x5C, 7C, ..., 23C)

The port *n* control registers contain control register bits for individual ports on a processing element.

Table 6-20. Bit Settings for Port *n* Control CSRs

Bit	Name	Reset Value	Description
0-1	Port Width	see footnote[1]	Hardware width of the port (read-only): 0b00 - Single-lane port 0b01 - Four-lane port 0b10 - 0b11 - Reserved
2-4	Initialized Port Width	see footnote[2]	Width of the ports after initialized (read only): 0b000 - Single-lane port, lane 0 0b001 - Single-lane port, lane 2 0b010 - Four-lane port 0b011 - 0b111 - Reserved
5-7	Port Width Override	0b000	Soft port configuration to override the hardware size: 0b000 - No override 0b001 - Reserved 0b010 - Force single lane, lane 0 0b011 - Force single lane, lane 2 0b100 - 0b111 - Reserved If the port size is overridden, the port will re-initialize to change to the requested size.
8	Port Disable	0b0	Port disable: 0b0 - port receivers/drivers are enabled 0b1 - port receivers/drivers are disabled and are unable to receive/transmit any packets or control symbols
9	Output Port Enable	see footnote[3]	Output port transmit enable: 0b0 - port is stopped and not enabled to issue any packets except to route or respond to I/O logical MAINTENANCE packets. Control symbols are not affected and are sent normally. This is the recommended state after device reset. 0b1 - port is enabled to issue any packets

Bit	Name	Reset Value	Description
10	Input Port Enable	see footnote[4]	Input port receive enable: 0b0 - port is stopped and only enabled to route or respond I/O logical MAINTENANCE packets. Other packets generate packet-not-accepted control symbols to force an error condition to be signaled by the sending device. Control symbols are not affected and are received and handled normally. This is the recommended state after device reset. 0b1 - port is enabled to respond to any packet
11	Error Checking Disable	0b0	This bit disables all RapidIO transmission error checking 0b0 - Error checking and recovery is enabled 0b1 - Error checking and recovery is disabled Device behavior when error checking and recovery is disabled and an error condition occurs is undefined
12	Multicast-event Participant	see footnote[5]	Send incoming Multicast-event control symbols to this port (multiple port devices only)
13	—		Reserved
14	Enumeration Boundary	see footnote[6]	An enumeration boundary aware system enumeration algorithm shall honor this flag. The algorithm, on either the ingress or the egress port, shall not enumerate past a port with this bit set. This provides for software enforced enumeration domains within the RapidIO fabric.
15-19	—		Reserved
20-27	Re-transmit Suppression Mask	0x00	Suppress packet re-transmission on CRC error. For devices that support CRF: 0b0000_0000 - Error recovery suppression disabled 0bxxxx_xxx1 - Suppress CRF=0, priority 0 re-transmission 0bxxxx_xx1x - Suppress CRF=0, priority 1 re-transmission 0bxxxx_x1xx - Suppress CRF=0, priority 2 re-transmission 0bxxxx_1xxx - Suppress CRF=0, priority 3 re-transmission 0bxxx1_xxxx - Suppress CRF=1, priority 0 re-transmission 0bxx1x_xxxx - Suppress CRF=1, priority 1 re-transmission 0bx1xx_xxxx - Suppress CRF=1, priority 2 re-transmission 0b1xxx_xxxx - Suppress CRF=1, priority 3 re-transmission For devices that do not support CRF: 0b0000_0000 - Error recovery suppression disabled 0b0000_xxx1 - Suppress priority 0 re-transmission 0b0000_xx1x - Suppress priority 1 re-transmission 0b0000_x1xx - Suppress priority 2 re-transmission 0b0000_1xxx - Suppress priority 3 re-transmission 0b0001_0000 - reserved ... 0b1111_1111 - reserved This field is only valid if bit 25 of the Processing Element Features CAR is set.
28-30	—		Reserved
31	Port Type		This indicates the port type (read only) 0b0 - Reserved 0b1 - Serial port

[1]The Port Width reset value is implementation dependent

[2]The Initialized Port Width reset value is implementation dependent

[3]The Output Port Enable reset value is implementation dependent

[4]The Input Port Enable reset value is implementation dependent

[5]The Multicast-Event Participant reset value is implementation dependent

[6]The Enumeration Boundary reset value is implementation dependent. Provision shall be made to allow the reset value of this bit to be configurable on a per system basis if this feature is supported.

6.7 Generic End Point Free Devices

This section describes the 1x/4x LP-Serial registers for a general devices that do not contain end point functionality (i.e. switches). This Extended Features register block uses extended features block ID=0x0003.

6.7.1 Register Map

Table 6-21 shows the register map for generic RapidIO 1x/4x LP-Serial end point-free devices. The Block Offset is the offset based on the Extended Features pointer (EF_PTR) to this block. 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 + 0xBC]. Register map offset [EF_PTR + 0xC0] can be used for another Extended Features block.

Table 6-21. LP-Serial Register Map - Generic End Point Free Devices

	Block Byte Offset	Register Name
General	0x0	1x/4x LP-Serial Register Block Header
	0x4–1C	Reserved
	0x20	Port Link Time-Out Control CSR
	0x24-38	Reserved
	0x3C	Port General Control CSR
Port 0	0x40-54	Reserved
	0x58	Port 0 Error and Status CSR
	0x5C	Port 0 Control CSR
Port 1	0x60-74	Reserved
	0x78	Port 1 Error and Status CSR
	0x7C	Port 1 Control CSR
Ports 2-14	0x80–218	Assigned to Port 2-14 CSRs
Port 15	0x220-234	Reserved
	0x238	Port 15 Error and Status CSR
	0x23C	Port 15 Control CSR

6.7.2 Command and Status Registers (CSRs)

Refer to Table 6-2 for the required behavior for accesses to reserved registers and register bits.

6.7.2.1 1x/4x LP-Serial Register Block Header (Block Offset 0x0)

The 1x/4x LP-Serial register block header register contains the EF_PTR to the next extended features block and the EF_ID that identifies this as the generic end point 1x/4x LP-Serial register block header.

Table 6-22. Bit Settings for 1x/4x LP-Serial Register 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	0x0003	Hard wired Extended Features ID

6.7.2.2 Port Link Time-out Control CSR (Block Offset 0x20)

The port link time-out control register contains the time-out timer value for all ports on a device. This time-out is for link events such as sending a packet to receiving the corresponding acknowledge and sending a link-request to receiving the corresponding link-response. The reset value is the maximum time-out interval, and represents between 3 and 6 seconds.

Table 6-23. Bit Settings for Port Link Time-out Control CSR

Bit	Name	Reset Value	Description
0–23	time-out value	All 1s	time-out interval value
24-31	—		Reserved

6.7.2.3 Port General Control CSR (Block Offset 0x3C)

The port general control register contains control register bits applicable to all ports on a processing element.

Table 6-24. Bit Settings for Port General Control CSRs

Bit	Name	Reset Value	Description
0-1	—		Reserved
2	Discovered	0b0	This device has been located by the processing element responsible for system configuration 0b0 - The device has not been previously discovered 0b1 - The device has been discovered by another processing element
3-31	—		Reserved

6.7.2.4 Port *n* Error and Status CSRs (Block Offsets 0x58, 78, .., 238)

These registers are accessed when a local processor or an external device wishes to examine the port error and status information.

Table 6-25. Bit Settings for Port *n* Error and Status CSRs

Bit	Name	Reset Value	Description
0-10	—		Reserved
11	Output Retry-encountered	0b0	Output port has encountered a retry condition.This bit is set when bit 13 is set. Once set, remains set until written with a logic 1 to clear.
12	Output Retried	0b0	Output port has received a packet-retry control symbol and can not make forward progress. This bit is set when bit 13 is set and is cleared when a packet-accepted or a packet-not-accepted control symbol is received (read-only).
13	Output Retry-stopped	0b0	Output port has received a packet-retry control symbol and is in the “output retry-stopped” state (read-only).
14	Output Error-encountered	0b0	Output port has encountered (and possibly recovered from) a transmission error. This bit is set when bit 15 is set. Once set, remains set until written with a logic 1 to clear.
15	Output Error-stopped	0b0	Output is in the “output error-stopped” state (read-only).
16-20	—		Reserved
21	Input Retry-stopped	0b0	Input port is in the “input retry-stopped” state (read-only).
22	Input Error-encountered	0b0	Input port has encountered (and possibly recovered from) a transmission error. This bit is set when bit 23 is set. Once set, remains set until written with a logic 1 to clear.
23	Input Error-stopped	0b0	Input port is in the “input error-stopped” state (read-only).
24-26	—		Reserved

Table 6-25. Bit Settings for Port *n* Error and Status CSRs

Bit	Name	Reset Value	Description
27	Port-write Pending	0b0	Port has encountered a condition which required it to initiate a Maintenance Port-write operation This bit is only valid if the device is capable of issuing a maintenance port-write transaction. Once set remains set until written with a logic 1 to clear.
28	—		Reserved
29	Port Error	0b0	Input or output port has encountered an error from which hardware was unable to recover. Once set, remains set until written with a logic 1 to clear.
30	Port OK	0b0	The input and output ports are initialized and the port is exchanging error-free control symbols with the attached device (read-only).
31	Port Uninitialized	0b1	Input and output ports are not initialized. This bit and bit 30 are mutually exclusive (read-only).

6.7.2.5 Port *n* Control CSR (Block Offsets 0x5C, 7C, ..., 23C)

The port *n* control registers contain control register bits for individual ports on a processing element.

Table 6-26. Bit Settings for Port *n* Control CSRs

Bit	Name	Reset Value	Description
0-1	Port Width	see footnote[1]	Hardware width of the port (read-only): 0b00 - Single-lane port 0b01 - Four-lane port 0b10 - 0b11 - Reserved
2-4	Initialized Port Width	see footnote[2]	Width of the ports after initialized (read only): 0b000 - Single-lane port, lane 0 0b001 - Single-lane port, lane 2 0b010 - Four-lane port 0b011 - 0b111 - Reserved
5-7	Port Width Override	0b000	Soft port configuration to override the hardware size: 0b000 - No override 0b001 - Reserved 0b010 - Force single lane, lane 0 0b011 - Force single lane, lane 2 0b100 - 0b111 - Reserved If the port size is overridden, the port will re-initialize to change to the requested size.
8	Port Disable	0b0	Port disable: 0b0 - port receivers/drivers are enabled 0b1 - port receivers/drivers are disabled and are unable to receive/transmit any packets or control symbols
9	Output Port Enable	see footnote[3]	Output port transmit enable: 0b0 - port is stopped and not enabled to issue any packets except to route or respond to I/O logical MAINTENANCE packets. Control symbols are not affected and are sent normally. This is the recommended state after device reset. 0b1 - port is enabled to issue any packets

Bit	Name	Reset Value	Description
10	Input Port Enable	see footnote[4]	Input port receive enable: 0b0 - port is stopped and only enabled to route or respond I/O logical MAINTENANCE packets. Other packets generate packet-not-accepted control symbols to force an error condition to be signaled by the sending device. Control symbols are not affected and are received and handled normally. This is the recommended state after device reset. 0b1 - port is enabled to respond to any packet
11	Error Checking Disable	0b0	This bit disables all RapidIO transmission error checking 0b0 - Error checking and recovery is enabled 0b1 - Error checking and recovery is disabled Device behavior when error checking and recovery is disabled and an error condition occurs is undefined
12	Multicast Event Participant	see footnote[5]	Send incoming Multicast-event control symbols to this port (multiple port devices only)
13	—		Reserved
14	Enumeration Boundary	see footnote[6]	An enumeration boundary aware system enumeration algorithm shall honor this flag. The algorithm, on either the ingress or the egress port, shall not enumerate past a port with this bit set. This provides for software enforced enumeration domains within the RapidIO fabric.
15-19	—		Reserved
20-27	Re-transmit Suppression Mask	0x00	Suppress packet re-transmission on CRC error. For devices that support CRF: 0b0000_0000 - Error recovery suppression disabled 0bxxxx_xxx1 - Suppress CRF=0, priority 0 re-transmission 0bxxxx_xx1x - Suppress CRF=0, priority 1 re-transmission 0bxxxx_x1xx - Suppress CRF=0, priority 2 re-transmission 0bxxxx_1xxx - Suppress CRF=0, priority 3 re-transmission 0bxxx1_xxxx - Suppress CRF=1, priority 0 re-transmission 0bxx1x_xxxx - Suppress CRF=1, priority 1 re-transmission 0bx1xx_xxxx - Suppress CRF=1, priority 2 re-transmission 0b1xxx_xxxx - Suppress CRF=1, priority 3 re-transmission For devices that do not support CRF: 0b0000_0000 - Error recovery suppression disabled 0b0000_xxx1 - Suppress priority 0 re-transmission 0b0000_xx1x - Suppress priority 1 re-transmission 0b0000_x1xx - Suppress priority 2 re-transmission 0b0000_1xxx - Suppress priority 3 re-transmission 0b0001_0000 - reserved ... 0b1111_1111 - reserved This field is only valid if bit 25 of the Processing Element Features CAR is set.
28-30	—		Reserved
31	Port Type		This indicates the port type (read only) 0b0 - Reserved 0b1 - Serial port

[1]The Port Width reset value is implementation dependent

[2]The Initialized Port Width reset value is implementation dependent

[3]The Output Port Enable reset value is implementation dependent

[4]The Input Port Enable reset value is implementation dependent

[5]The Multicast-event Participant reset value is implementation dependent

[6]The Enumeration Boundary reset value is implementation dependent. Provision shall be made to allow the reset value of this bit to be configurable on a per system basis if this feature is supported.

6.8 Generic End Point Free Devices, software assisted error recovery option

This section describes the 1x/4x LP-Serial registers for a general device that does not contain end point functionality with software assisted error recovery.Typically these devices are switches. This is most useful for devices that for whatever reason do not want to implement error recovery in hardware and to allow software to generate link-request control symbols and see the results of the responses. This Extended Features register block is assigned Extended Features block ID=0x0009.

6.8.1 Register Map

Table 6-27 shows the register map for generic RapidIO 1x/4x LP-Serial end point-free devices with software assisted error recovery. The Block Offset is the offset based on the Extended Features pointer (EF_PTR) to this block. 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 + 0xBC]. Register map offset [EF_PTR + 0xC0] can be used for another Extended Features block.

Table 6-27. LP-Serial Register Map - Generic End Point-free Devices (SW assisted)

	Block Byte Offset	Register Name
General	0x0	1x/4x LP-Serial Register Block Header
	0x4–1C	Reserved
	0x20	Port Link Time-Out Control CSR
	0x24-38	Reserved
	0x3C	Port General Control CSR
Port 0	0x40	Port 0 Link Maintenance Request CSR
	0x44	Port 0 Link Maintenance Response CSR
	0x48	Port 0 Local ackID Status CSR
	0x4C-54	Reserved
	0x58	Port 0 Error and Status CSR
	0x5C	Port 0 Control CSR
Port 1	0x60	Port 1 Link Maintenance Request CSR
	0x64	Port 1 Link Maintenance Response CSR
	0x68	Port 1 Local ackID Status CSR
	0x6C-74	Reserved
	0x78	Port 1 Error and Status CSR
	0x7C	Port 1 Control CSR

Table 6-27. LP-Serial Register Map (Continued)- Generic End Point-free Devices (SW assisted)

	Block Byte Offset	Register Name
Ports 2-14	0x80–218	Assigned to Port 2-14 CSRs
Port 15	0x220	Port 15 Link Maintenance Request CSR
	0x224	Port 15 Link Maintenance Response CSR
	0x228	Port 15 Local ackID Status CSR
	0x22C-234	Reserved
	0x238	Port 15 Error and Status CSR
	0x23C	Port 15 Control CSR

6.8.2 Command and Status Registers (CSRs)

Refer to Table 6-2 for the required behavior for accesses to reserved registers and register bits.

6.8.2.1 1x/4x LP-Serial Register Block Header (Block Offset 0x0)

The 1x/4x LP-Serial register block header register contains the EF_PTR to the next extended features block and the EF_ID that identifies this as the generic end point 1x/4x LP-Serial register block header.

Table 6-28. Bit Settings for 1x/4x LP-Serial Register 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	0x0009	Hard wired Extended Features ID

6.8.2.2 Port Link Time-out Control CSR (Block Offset 0x20)

The port link time-out control register contains the time-out timer value for all ports on a device. This time-out is for link events such as sending a packet to receiving the corresponding acknowledge and sending a link-request to receiving the corresponding link-response. The reset value is the maximum time-out interval, and represents between 3 and 6 seconds.

Table 6-29. Bit Settings for Port Link Time-out Control CSR

Bit	Name	Reset Value	Description
0–23	time-out value	All 1s	time-out interval value
24-31	—		Reserved

6.8.2.3 Port General Control CSR (Block Offset 0x3C)

The port general control register contains control register bits applicable to all ports on a processing element.

Table 6-30. Bit Settings for Port General Control CSRs

Bit	Name	Reset Value	Description
0-1	—		Reserved
2	Discovered	0b0	This device has been located by the processing element responsible for system configuration 0b0 - The device has not been previously discovered 0b1 - The device has been discovered by another processing element
3-31	—		Reserved

6.8.2.4 Port *n* Link Maintenance Request CSRs (Block Offsets 0x40, 60, ..., 220)

The port link maintenance request registers are accessible both by a local processor and an external device. A write to one of these registers generates a link-request control symbol on the corresponding RapidIO port interface.

Table 6-31. Bit Settings for Port *n* Link Maintenance Request CSRs

Bit	Name	Reset Value	Description
0–28	—		Reserved
29-31	Command	0b000	Command to be sent in the link-request control symbol. If read, this field returns the last written value.

6.8.2.5 Port *n* Link Maintenance Response CSRs (Block Offsets 0x44, 64, ..., 224)

The port link maintenance response registers are accessible both by a local processor and an external device. A read to this register returns the status received in a link-response control symbol. The ackID_status and port_status fields are defined in Table 3-3 and Table 3-5. This register is read-only.

Table 6-32. Bit Settings for Port *n* Link Maintenance Response CSRs

Bit	Name	Reset Value	Description
0	response_valid	0b0	If the link-request causes a link-response, this bit indicates that the link-response has been received and the status fields are valid. If the link-request does not cause a link-response, this bit indicates that the link-request has been transmitted. This bit automatically clears on read.
1-21	—		Reserved
22-26	ackID_status	0b00000	ackID status field from the link-response control symbol
27-31	link_status	0b00000	link status field from the link-response control symbol

6.8.2.6 Port *n* Local ackID CSRs (Block Offsets 0x48, 68, ..., 228)

The port link local ackID status registers are accessible both by a local processor and an external device. A read to this register returns the local ackID status for both the out and input ports of the device.

Table 6-33. Bit Settings for Port *n* Local ackID Status CSRs

Bit	Name	Reset Value	Description
0	Clr_outstanding_ackIDs	0b0	Writing 0b1 to this bit causes all outstanding unacknowleged packets to be discarded. This bit should only be written when trying to recover a failed link. This bit is always logic 0 when read.
1-2	—		Reserved
3-7	Inbound_ackID	0b00000	Input port next expected ackID value
8-18	—		Reserved
19-23	Outstanding_ackID	0x00000	Output port unacknowledged ackID status. Next expected acknowledge control symbol ackID field that indicates the ackID value expected in the next received acknowledge control symbol.
24-26	—		Reserved
27-31	Outbound_ackID	0b00000	Output port next transmitted ackID value. Software writing this value can force retransmission of outstanding unacknowledged packets in order to manually implement error recovery.

6.8.2.7 Port *n* Error and Status CSRs (Block Offset 0x58, 78, ..., 238)

These registers are accessed when a local processor or an external device wishes to examine the port error and status information.

Table 6-34. Bit Settings for Port *n* Error and Status CSRs

Bit	Name	Reset Value	Description
0-10	—		Reserved
11	Output Retry-encountered	0b0	Output port has encountered a retry condition.This bit is set when bit 13 is set. Once set, remains set until written with a logic 1 to clear.
12	Output Retried	0b0	Output port has received a packet-retry control symbol and can not make forward progress. This bit is set when bit 13 is set and is cleared when a packet-accepted or a packet-not-accepted control symbol is received (read-only).
13	Output Retry-stopped	0b0	Output port has received a packet-retry control symbol and is in the “output retry-stopped” state (read-only).
14	Output Error-encountered	0b0	Output port has encountered (and possibly recovered from) a transmission error. This bit is set when bit 15 is set. Once set, remains set until written with a logic 1 to clear.
15	Output Error-stopped	0b0	Output is in the “output error-stopped” state (read-only).
16-20	—		Reserved
21	Input Retry-stopped	0b0	Input port is in the “input retry-stopped” state (read-only).

Table 6-34. Bit Settings for Port *n* Error and Status CSRs

Bit	Name	Reset Value	Description
22	Input Error-encountered	0b0	Input port has encountered (and possibly recovered from) a transmission error. This bit is set when bit 23 is set. Once set, remains set until written with a logic 1 to clear.
23	Input Error-stopped	0b0	Input port is in the “input error-stopped” state (read-only).
24-26	—		Reserved
27	Port-write Pending	0b0	Port has encountered a condition which required it to initiate a Maintenance Port-write operation This bit is only valid if the device is capable of issuing a maintenance port-write transaction. Once set remains set until written with a logic 1 to clear.
28	—		Reserved
29	Port Error	0b0	Input or output port has encountered an error from which hardware was unable to recover. Once set, remains set until written with a logic 1 to clear.
30	Port OK	0b0	The input and output ports are initialized and the port is exchanging error-free control symbols with the attached device (read-only).
31	Port Uninitialized	0b1	Input and output ports are not initialized. This bit and bit 30 are mutually exclusive (read-only).

6.8.2.8 Port *n* Control CSR (Block Offsets 0x5C, 7C, ..., 23C)

The port *n* control registers contain control register bits for individual ports on a processing element.

Table 6-35. Bit Settings for Port *n* Control CSRs

Bit	Name	Reset Value	Description
0-1	Port Width	see footnote[1]	Hardware width of the port (read-only): 0b00 - Single-lane port 0b01 - Four-lane port 0b10 - 0b11 - Reserved
2-4	Initialized Port Width	see footnote[2]	Width of the ports after initialized (read only): 0b000 - Single-lane port, lane 0 0b001 - Single-lane port, lane 2 0b010 - Four-lane port 0b011 - 0b111 - Reserved
5-7	Port Width Override	0b000	Soft port configuration to override the hardware size: 0b000 - No override 0b001 - Reserved 0b010 - Force single lane, lane 0 0b011 - Force single lane, lane 2 0b100 - 0b111 - Reserved If the port size is overridden, the port will re-initialize to change to the requested size.
8	Port Disable	0b0	Port disable: 0b0 - port receivers/drivers are enabled 0b1 - port receivers/drivers are disabled and are unable to receive/transmit any packets or control symbols

Bit	Name	Reset Value	Description
9	Output Port Enable	see footnote[3]	Output port transmit enable: 0b0 - port is stopped and not enabled to issue any packets except to route or respond to I/O logical MAINTENANCE packets. Control symbols are not affected and are sent normally. This is the recommended state after device reset. 0b1 - port is enabled to issue any packets
10	Input Port Enable	see footnote[4]	Input port receive enable: 0b0 - port is stopped and only enabled to route or respond I/O logical MAINTENANCE packets. Other packets generate packet-not-accepted control symbols to force an error condition to be signaled by the sending device. Control symbols are not affected and are received and handled normally. This is the recommended state after device reset. 0b1 - port is enabled to respond to any packet
11	Error Checking Disable	0b0	This bit disables all RapidIO transmission error checking 0b0 - Error checking and recovery is enabled 0b1 - Error checking and recovery is disabled Device behavior when error checking and recovery is disabled and an error condition occurs is undefined
12	Multicast-event Participant	see footnote[5]	Send incoming Multicast-event control symbols to this port (multiple port devices only)
13	—		Reserved
14	Enumeration Boundary	see footnote[6]	An enumeration boundary aware system enumeration algorithm shall honor this flag. The algorithm, on either the ingress or the egress port, shall not enumerate past a port with this bit set. This provides for software enforced enumeration domains within the RapidIO fabric.
15-19	—		Reserved
20-27	Re-transmit Suppression Mask	0x00	Suppress packet re-transmission on CRC error. For devices that support CRF: 0b0000_0000 - Error recovery suppression disabled 0bxxxx_xxx1 - Suppress CRF=0, priority 0 re-transmission 0bxxxx_xx1x - Suppress CRF=0, priority 1 re-transmission 0bxxxx_x1xx - Suppress CRF=0, priority 2 re-transmission 0bxxxx_1xxx - Suppress CRF=0, priority 3 re-transmission 0bxxx1_xxxx - Suppress CRF=1, priority 0 re-transmission 0bxx1x_xxxx - Suppress CRF=1, priority 1 re-transmission 0bx1xx_xxxx - Suppress CRF=1, priority 2 re-transmission 0b1xxx_xxxx - Suppress CRF=1, priority 3 re-transmission For devices that do not support CRF: 0b0000_0000 - Error recovery suppression disabled 0b0000_xxx1 - Suppress priority 0 re-transmission 0b0000_xx1x - Suppress priority 1 re-transmission 0b0000_x1xx - Suppress priority 2 re-transmission 0b0000_1xxx - Suppress priority 3 re-transmission 0b0001_0000 - reserved ... 0b1111_1111 - reserved This field is only valid if bit 25 of the Processing Element Features CAR is set.

Bit	Name	Reset Value	Description
28-30	—		Reserved
31	Port Type		This indicates the port type (read only) 0b0 - Reserved 0b1 - Serial port

[1]The Port Width reset value is implementation dependent

[2]The Initialized Port Width reset value is implementation dependent

[3]The Output Port Enable reset value is implementation dependent

[4]The Input Port Enable reset value is implementation dependent

[5]The Multicast-Event Participant reset value is implementation dependent

[6]The Enumeration Boundary reset value is implementation dependent. Provision shall be made to allow the reset value of this bit to be configurable on a per system basis if this feature is supported.

Chapter 7 Signal Descriptions

7.1 Introduction

This chapter contains the signal pin descriptions for a RapidIO 1x/4x LP-Serial port. The interface is defined either as a single- or four-lane, full duplex, point-to-point interface using differential signaling. A single-lane implementation consists of 4 wires and a four-lane implementation consists of 16 wires. The electrical details are described in Chapter 8, “Electrical Specifications.”

7.2 Signal Definitions

Table 7-1 provides a summary of the RapidIO 1x/4x LP-Serial signal pins as well as a short description of their functionality.

Table 7-1. 1x/4x LP-Serial Signal Description

Signal Name	I/O	Signal Meaning	Timing Comments
TD[0-3]	O	Transmit Data - The transmit data is a unidirectional point to point bus designed to transmit the packet information. The TD bus of one device is connected to the RD bus of the receiving device. TD[0] is used in 1x mode.	Clocking is embedded in data using 8B/10B encoding.
$\overline{\text{TD[0-3]}}$	O	Transmit Data complement—These signals are the differential pairs of the TD signals.	
RD[0-3]	I	Receive Data - The receive data is a unidirectional point to point bus designed to receive the packet information. The RD bus of one device is connected to the TD bus of the receiving device. RD[0] is used in 1x mode.	
$\overline{\text{RD[0-3]}}$	I	Receive Data complement—These signals are the differential pairs of the RD signals.	

7.3 Serial RapidIO Interface Diagrams

Figure 7-1 shows the signal interface diagram connecting two 1x devices together with the RapidIO 1x/4x LP-Serial interconnect.

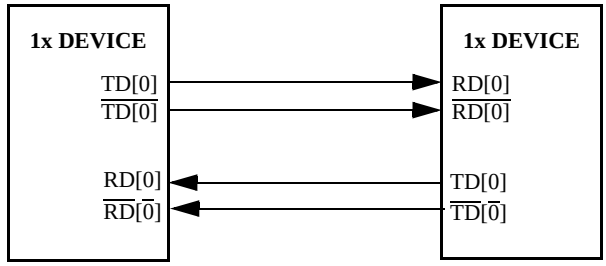


Figure 7-1. RapidIO 1x Device to 1x Device Interface Diagram

Figure 7-2 shows the signal interface diagram connecting two 4x devices together with the RapidIO 1x/4x LP-Serial interconnect.

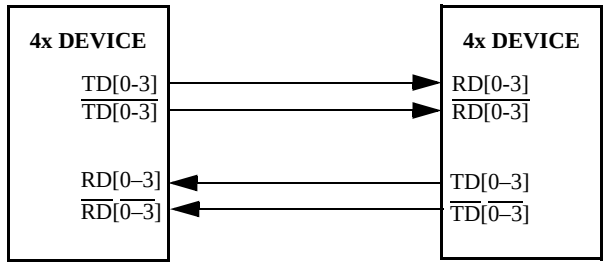


Figure 7-2. RapidIO 4x Device to 4x Device Interface Diagram

Figure 7-3 shows the connections between a 4x LP-Serial device and a 1x LP-Serial device.

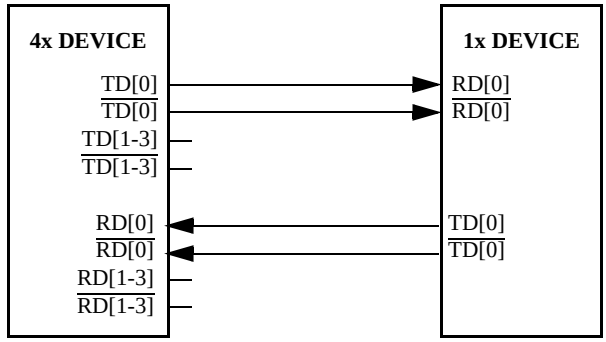


Figure 7-3. RapidIO 4x Device to 1x Device Interface Diagram

Chapter 8 Electrical Specifications

8.1 Introduction

The chapter defines the electrical specifications for the LP-Serial physical layer. The electrical specifications covers both single and multiple-lane links. Two transmitters (short run and long run) and a single receiver are specified for each of three baud rates, 1.25, 2.50, and 3.125 GBaud.

Two transmitter specifications allow for solutions ranging from simple board-to-board interconnect to driving two connectors across a backplane. A single receiver specification is given that will accept signals from both the short run and long run transmitter specifications.

The short run transmitter should be used mainly for chip-to-chip connections on either the same printed circuit board or across a single connector. This covers the case where connections are made to a mezzanine (daughter) card. The minimum swings of the short run specification reduce the overall power used by the transceivers.

The long run transmitter specifications use larger voltage swings that are capable of driving signals across backplanes. This allows a user to drive signals across two connectors and a backplane. The specifications allow a distance of at least 50 cm at all baud rates.

All unit intervals are specified with a tolerance of +/- 100 ppm. The worst case frequency difference between any transmit and receive clock will be 200 ppm.

To ensure interoperability between drivers and receivers of different vendors and technologies, AC coupling at the receiver input must be used.

8.2 Signal Definitions

LP-Serial links use differential signaling. This section defines terms used in the description and specification of differential signals. Figure 8-1 shows how the signals are defined. The figures shows waveforms for either a transmitter output (TD and $\overline{TD}$) or a receiver input (RD and $\overline{RD}$). Each signal swings between A Volts and B Volts where A > B. Using these waveforms, the definitions are as follows:

5. The transmitter output signals and the receiver input signals TD, $\overline{TD}$, RD and $\overline{RD}$ each have a peak-to-peak swing of A - B Volts
6. The differential output signal of the transmitter, V_{OD}, is defined as V_{TD}-$V_{\overline{TD}}$.
7. The differential input signal of the receiver, V_{ID}, is defined as V_{RD}-$V_{\overline{RD}}$.
8. The differential output signal of the transmitter and the differential input signal of the receiver each range from A - B to -(A - B) Volts
9. The peak value of the differential transmitter output signal and the differential receiver input signal is A - B Volts
10. The peak-to-peak value of the differential transmitter output signal and the differential receiver input signal is 2 * (A - B) Volts

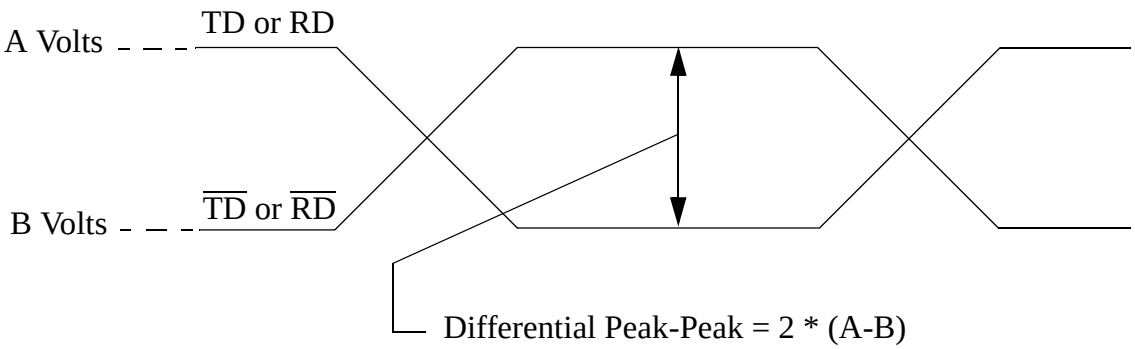


Figure 8-1. Differential Peak-Peak Voltage of Transmitter or Receiver

To illustrate these definitions using real values, consider the case of a CML (Current Mode Logic) transmitter that has a common mode voltage of 2.25 V and each of its outputs, TD and $\overline{TD}$, has a swing that goes between 2.5V and 2.0V. Using these values, the peak-to-peak voltage swing of the signals TD and $\overline{TD}$ is 500 mV p-p. The differential output signal ranges between 500 mV and -500 mV. The peak differential voltage is 500 mV. The peak-to-peak differential voltage is 1000 mV p-p.

8.3 Equalization

With the use of high speed serial links, the interconnect media will cause degradation of the signal at the receiver. Effects such as Inter-Symbol Interference (ISI) or data dependent jitter are produced. This loss can be large enough to degrade the eye opening at the receiver beyond what is allowed in the specification. To negate a portion of these effects, equalization can be used. The most common equalization techniques that can be used are:

- Pre-emphasis on the transmitter
- A passive high pass filter network placed at the receiver. This is often referred to as passive equalization.
- The use of active circuits in the receiver. This is often referred to as adaptive equalization.

8.4 Explanatory Note on Transmitter and Receiver Specifications

AC electrical specifications are given for transmitter and receiver. Long run and short run interfaces at three baud rates (a total of six cases) are described.

The parameters for the AC electrical specifications are guided by the XAUI electrical interface specified in Clause 47 of IEEE 802.3ae-2002.

XAUI has similar application goals to serial RapidIO, as described in Section 8.1. The goal of this standard is that electrical designs for serial RapidIO can reuse electrical designs for XAUI, suitably modified for applications at the baud intervals and reaches described herein.

8.5 Transmitter Specifications

LP-Serial transmitter electrical and timing specifications are stated in the text and tables of this section.

The differential return loss, S11, of the transmitter in each case shall be better than

-10 dB for (Baud Frequency)/10 < Freq(f) < 625 MHz, and

-10 dB + 10log(f/625 MHz) dB for 625 MHz <= Freq(f) <= Baud Frequency

The reference impedance for the differential return loss measurements is 100 Ohm resistive. Differential return loss includes contributions from on-chip circuitry, chip packaging and any off-chip components related to the driver. The output impedance requirement applies to all valid output levels.

It is recommended that the 20%-80% rise/fall time of the transmitter, as measured at the transmitter output, in each case have a minimum value 60 ps.

It is recommended that the timing skew at the output of an LP-Serial transmitter between the two signals that comprise a differential pair not exceed 25 ps at 1.25 GB, 20 ps at 2.50 GB and 15 ps at 3.125 GB.

Table 8-1. Short Run Transmitter AC Timing Specifications - 1.25 GBaud

Characteristic	Symbol	Range		Unit	Notes
		Min	Max		
Output Voltage,	V_O	-0.40	2.30	Volts	Voltage relative to COMMON of either signal comprising a differential pair
Differential Output Voltage	V_{DIFFPP}	500	1000	mV p-p	
Deterministic Jitter	J_D		0.17	UI p-p	
Total Jitter	J_T		0.35	UI p-p	
Multiple output skew	S_{MO}		1000	ps	Skew at the transmitter output between lanes of a multilane link
Unit Interval	UI	800	800	ps	+/- 100 ppm

Table 8-2. Short Run Transmitter AC Timing Specifications - 2.5 GBaud

Characteristic	Symbol	Range		Unit	Notes
		Min	Max		
Output Voltage,	V_O	-0.40	2.30	Volts	Voltage relative to COMMON of either signal comprising a differential pair
Differential Output Voltage	V_{DIFFPP}	500	1000	mV p-p	

Table 8-2. Short Run Transmitter AC Timing Specifications - 2.5 GBaud

Characteristic	Symbol	Range		Unit	Notes
		Min	Max		
Deterministic Jitter	J_D		0.17	UI p-p	
Total Jitter	J_T		0.35	UI p-p	
Multiple Output skew	S_{MO}		1000	ps	Skew at the transmitter output between lanes of a multilane link
Unit Interval	UI	400	400	ps	+/- 100 ppm

Table 8-3. Short Run Transmitter AC Timing Specifications - 3.125 GBaud

Characteristic	Symbol	Range		Unit	Notes
		Min	Max		
Output Voltage,	V_O	-0.40	2.30	Volts	Voltage relative to COMMON of either signal comprising a differential pair
Differential Output Voltage	V_{DIFFPP}	500	1000	mV p-p	
Deterministic Jitter	J_D		0.17	UI p-p	
Total Jitter	J_T		0.35	UI p-p	
Multiple output skew	S_{MO}		1000	ps	Skew at the transmitter output between lanes of a multilane link
Unit Interval	UI	320	320	ps	+/- 100 ppm

Table 8-4. Long Run Transmitter AC Timing Specifications - 1.25 GBaud

Characteristic	Symbol	Range		Unit	Notes
		Min	Max		
Output Voltage,	V_O	-0.40	2.30	Volts	Voltage relative to COMMON of either signal comprising a differential pair
Differential Output Voltage	V_{DIFFPP}	800	1600	mV p-p	
Deterministic Jitter	J_D		0.17	UI p-p	
Total Jitter	J_T		0.35	UI p-p	
Multiple output skew	S_{MO}		1000	ps	Skew at the transmitter output between lanes of a multilane link
Unit Interval	UI	800	800	ps	+/- 100 ppm

Table 8-5. Long Run Transmitter AC Timing Specifications - 2.5 GBaud

Characteristic	Symbol	Range		Unit	Notes
		Min	Max		
Output Voltage,	V_O	-0.40	2.30	Volts	Voltage relative to COMMON of either signal comprising a differential pair
Differential Output Voltage	V_{DIFFPP}	800	1600	mV p-p	
Deterministic Jitter	J_D		0.17	UI p-p	
Total Jitter	J_T		0.35	UI p-p	
Multiple output skew	S_{MO}		1000	ps	Skew at the transmitter output between lanes of a multilane link
Unit Interval	UI	400	400	ps	+/- 100 ppm

Table 8-6. Long Run Transmitter AC Timing Specifications - 3.125 GBaud

Characteristic	Symbol	Range		Unit	Notes
		Min	Max		
Output Voltage,	V_O	-0.40	2.30	Volts	Voltage relative to COMMON of either signal comprising a differential pair
Differential Output Voltage	V_{DIFFPP}	800	1600	mV p-p	
Deterministic Jitter	J_D		0.17	UI p-p	
Total Jitter	J_T		0.35	UI p-p	
Multiple output skew	S_{MO}		1000	ps	Skew at the transmitter output between lanes of a multilane link
Unit Interval	UI	320	320	ps	+/- 100 ppm

For each baud rate at which an LP-Serial transmitter is specified to operate, the output eye pattern of the transmitter shall fall entirely within the unshaded portion of the Transmitter Output Compliance Mask shown in Figure 8-2 with the parameters specified in Table 8-7 when measured at the output pins of the device and the device is driving a 100 Ohm +/- 5% differential resistive load. The output eye pattern of a LP-Serial transmitter that implements pre-emphasis (to equalize the link and reduce inter-symbol interference) need only comply with the Transmitter Output Compliance Mask when pre-emphasis is disabled or minimized.

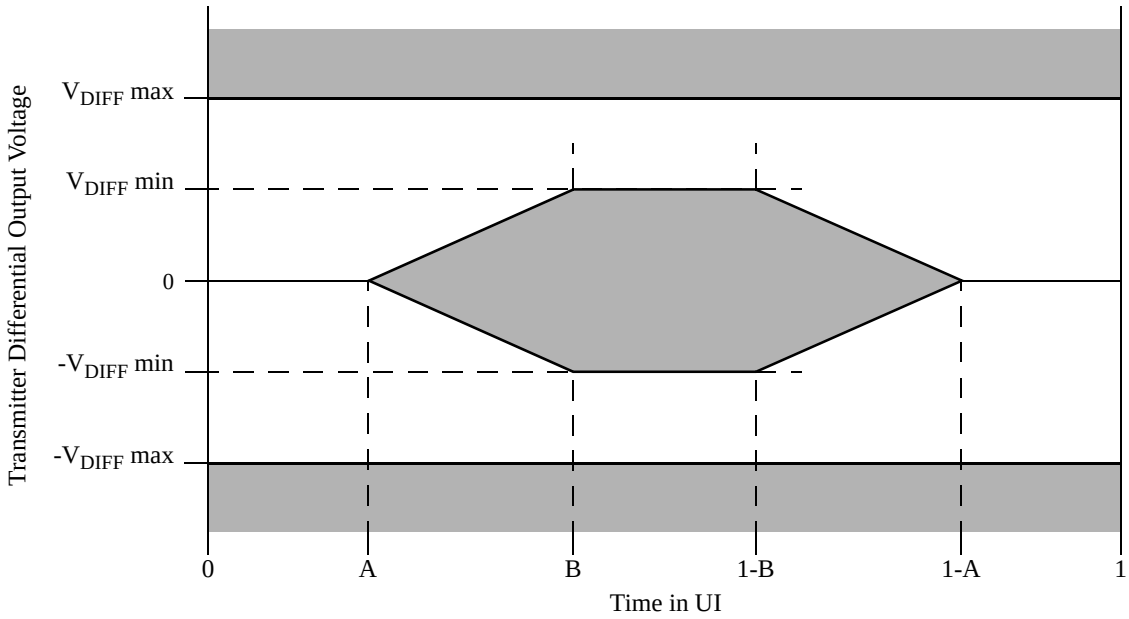


Figure 8-2. Transmitter Output Compliance Mask

Table 8-7. Transmitter Differential Output Eye Diagram Parameters

Transmitter Type	V_{DIFF}min (mV)	V_{DIFF}max (mV)	A (UI)	B (UI)
1.25 GBaud short range	250	500	0.175	0.39
1.25 GBaud long range	400	800	0.175	0.39
2.5 GBaud short range	250	500	0.175	0.39
2.5 GBaud long range	400	800	0.175	0.39
3.125 GBaud short range	250	500	0.175	0.39
3.125 GBaud long range	400	800	0.175	0.39

8.6 Receiver Specifications

LP-Serial receiver electrical and timing specifications are stated in the text and tables of this section.

Receiver input impedance shall result in a differential return loss better that 10 dB and a common mode return loss better than 6 dB from 100 MHz to (0.8)*(Baud Frequency). This includes contributions from on-chip circuitry, the chip package and any off-chip components related to the receiver. AC coupling components are included in this requirement. The reference impedance for return loss measurements is 100 Ohm resistive for differential return loss and 25 Ohm resistive for common mode.

Table 8-8. Receiver AC Timing Specifications - 1.25 GBaud

Characteristic	Symbol	Range		Unit	Notes
		Min	Max		
Differential Input Voltage	V_{IN}	200	1600	mV p-p	Measured at receiver
Deterministic Jitter Tolerance	J_D	0.37		UI p-p	Measured at receiver
Combined Deterministic and Random Jitter Tolerance	J_{DR}	0.55		UI p-p	Measured at receiver
Total Jitter Tolerance[1]	J_T	0.65		UI p-p	Measured at receiver
Multiple Input Skew	S_{MI}		24	ns	Skew at the receiver input between lanes of a multi-lane link
Bit Error Ratio	BER		10^{-12}		
Unit Interval	UI	800	800	ps	+/- 100 ppm

Notes
1. Total jitter is composed of three components, deterministic jitter, random jitter and single frequency sinusoidal jitter. The sinusoidal jitter may have any amplitude and frequency in the unshaded region of Figure 8-3. The sinusoidal jitter component is included to ensure margin for low frequency jitter, wander, noise, crosstalk and other variable system effects.

Table 8-9. Receiver AC Timing Specifications - 2.5 GBaud

Characteristic	Symbol	Range		Unit	Notes
		Min	Max		
Differential Input Voltage	V_{IN}	200	1600	mV p-p	Measured at receiver
Deterministic Jitter Tolerance	J_D	0.37		UI p-p	Measured at receiver
Combined Deterministic and Random Jitter Tolerance	J_{DR}	0.55		UI p-p	Measured at receiver
Total Jitter Tolerance[1]	J_T	0.65		UI p-p	Measured at receiver
Multiple Input Skew	S_{MI}		24	ns	Skew at the receiver input between lanes of a multi-lane link

Table 8-9. Receiver AC Timing Specifications - 2.5 GBaud

Characteristic	Symbol	Range		Unit	Notes
		Min	Max		
Bit Error Ratio	BER		10^{-12}		
Unit Interval	UI	400	400	ps	+/- 100 ppm

Notes
1. Total jitter is composed of three components, deterministic jitter, random jitter and single frequency sinusoidal jitter. The sinusoidal jitter may have any amplitude and frequency in the unshaded region of Figure 8-3. The sinusoidal jitter component is included to ensure margin for low frequency jitter, wander, noise, crosstalk and other variable system effects.

Table 8-10. Receiver AC Timing Specifications - 3.125 GBaud

Characteristic	Symbol	Range		Unit	Notes
		Min	Max		
Differential Input Voltage	V_{IN}	200	1600	mV p-p	Measured at receiver
Deterministic Jitter Tolerance	J_D	0.37		UI p-p	Measured at receiver
Combined Deterministic and Random Jitter Tolerance	J_{DR}	0.55		UI p-p	Measured at receiver
Total Jitter Tolerance[1]	J_T	0.65		UI p-p	Measured at receiver
Multiple Input Skew	S_{MI}		22	ns	Skew at the receiver input between lanes of a multi-lane link
Bit Error Ratio	BER		10^{-12}		
Unit Interval	UI	320	320	ps	+/- 100 ppm

Notes
1. Total jitter is composed of three components, deterministic jitter, random jitter and single frequency sinusoidal jitter. The sinusoidal jitter may have any amplitude and frequency in the unshaded region of Figure 8-3. The sinusoidal jitter component is included to ensure margin for low frequency jitter, wander, noise, crosstalk and other variable system effects.

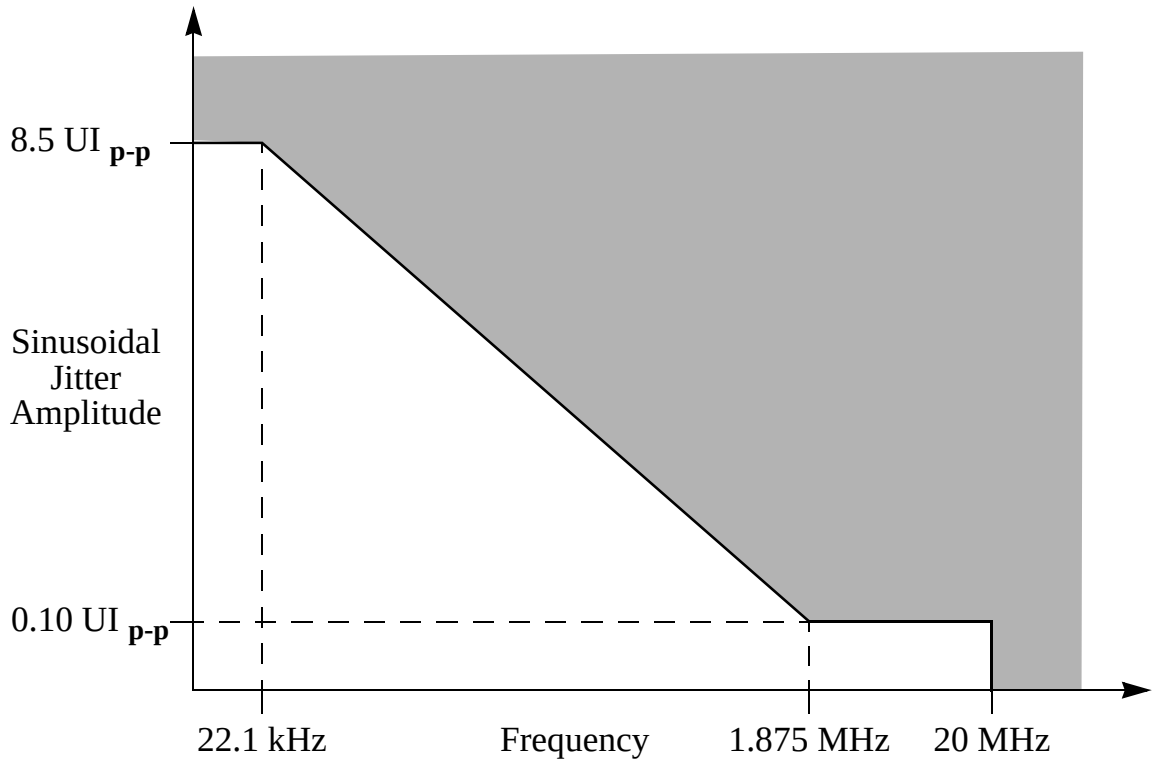


Figure 8-3. Single Frequency Sinusoidal Jitter Limits

8.7 Receiver Eye Diagrams

For each baud rate at which an LP-Serial receiver is specified to operate, the receiver shall meet the corresponding Bit Error Ratio specification (Table 8-8, Table 8-9, Table 8-10) when the eye pattern of the receiver test signal (exclusive of sinusoidal jitter) falls entirely within the unshaded portion of the Receiver Input Compliance Mask shown in Figure 8-4 with the parameters specified in Table 8-11. The eye pattern of the receiver test signal is measured at the input pins of the receiving device with the device replaced with a 100 Ohm +/- 5% differential resistive load.

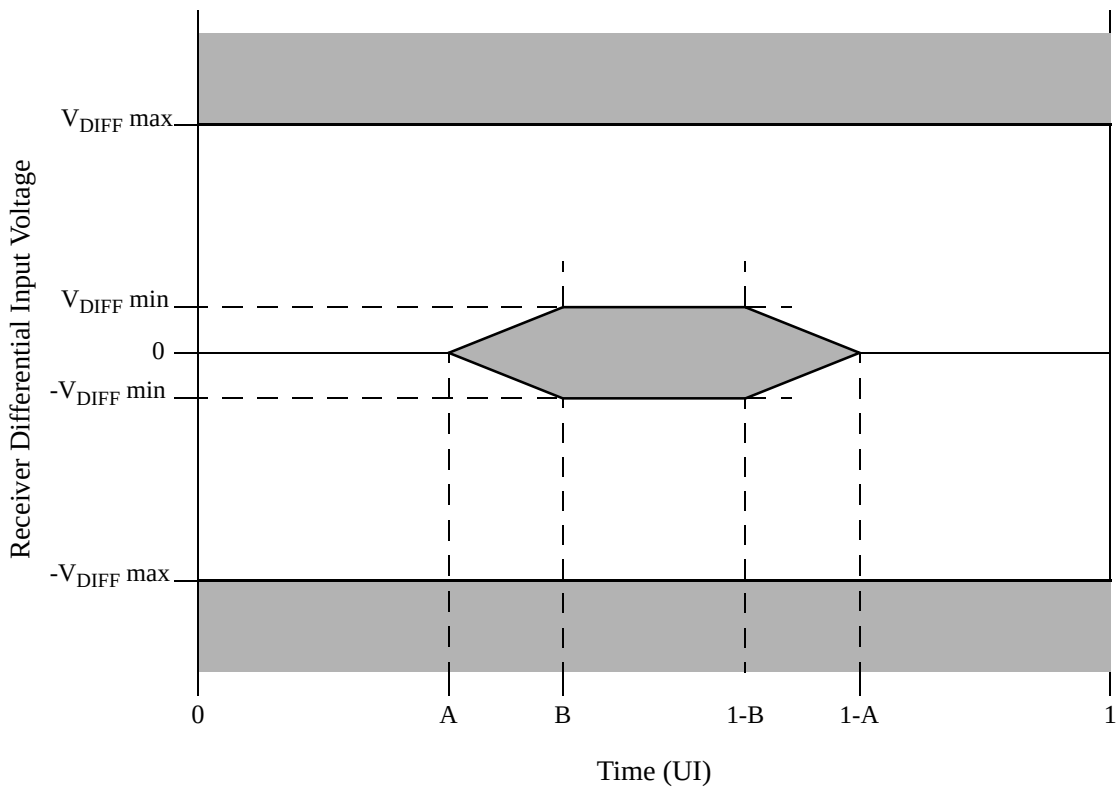


Figure 8-4. Receiver Input Compliance Mask

Table 8-11. Receiver Input Compliance Mask Parameters exclusive of Sinusoidal Jitter

Receiver Type	V_{DIFF}min (mV)	V_{DIFF}max (mV)	A (UI)	B (UI)
1.25 GBaud	100	800	0.275	0.400
2.5 GBaud	100	800	0.275	0.400
3.125 GBaud	100	800	0.275	0.400

8.8 Measurement and Test Requirements

Since the LP-Serial electrical specification are guided by the XAUI electrical interface specified in Clause 47 of IEEE 802.3ae-2002, the measurement and test requirements defined here are similarly guided by Clause 47. In addition, the CJPAT test pattern defined in Annex 48A of IEEE802.3ae-2002 is specified as the test pattern for use in eye pattern and jitter measurements. Annex 48B of IEEE802.3ae-2002 is recommended as a reference for additional information on jitter test methods.

8.8.1 Eye template measurements

For the purpose of eye template measurements, the effects of a single-pole high pass filter with a 3 dB point at (Baud Frequency)/1667 is applied to the jitter. The data pattern for template measurements is the Continuous Jitter Test Pattern (CJPAT) defined in Annex 48A of IEEE802.3ae. All lanes of the LP-Serial link shall be active in both the transmit and receive directions, and opposite ends of the links shall use asynchronous clocks. Four lane implementations shall use CJPAT as defined in Annex 48A. Single lane implementations shall use the CJPAT sequence specified in Annex 48A for transmission on lane 0. The amount of data represented in the eye shall be adequate to ensure that the bit error ratio is less than 10^{-12}. The eye pattern shall be measured with AC coupling and the compliance template centered at 0 Volts differential. The left and right edges of the template shall be aligned with the mean zero crossing points of the measured data eye. The load for this test shall be 100 Ohms resistive +/- 5% differential to 2.5 GHz.

8.8.2 Jitter test measurements

For the purpose of jitter measurement, the effects of a single-pole high pass filter with a 3 dB point at (Baud Frequency)/1667 is applied to the jitter. The data pattern for jitter measurements is the Continuous Jitter Test Pattern (CJPAT) pattern defined in Annex 48A of IEEE802.3ae. All lanes of the LP-Serial link shall be active in both the transmit and receive directions, and opposite ends of the links shall use asynchronous clocks. Four lane implementations shall use CJPAT as defined in Annex 48A. Single lane implementations shall use the CJPAT sequence specified in Annex 48A for transmission on lane 0. Jitter shall be measured with AC coupling and at 0 Volts differential. Jitter measurement for the transmitter (or for calibration of a jitter tolerance setup) shall be performed with a test procedure resulting in a BER curve such as that described in Annex 48B of IEEE802.3ae.

8.8.3 Transmit jitter

Transmit jitter is measured at the driver output when terminated into a load of 100 Ohms resistive +/- 5% differential to 2.5 GHz.

8.8.4 Jitter tolerance

Jitter tolerance is measured at the receiver using a jitter tolerance test signal. This signal is obtained by first producing the sum of deterministic and random jitter defined in Section 8.6 and then adjusting the signal amplitude until the data eye contacts the 6 points of the minimum eye opening of the receive template shown in Figure 8-4 and Table 8-11. Note that for this to occur, the test signal must have vertical waveform symmetry about the average value and have horizontal symmetry (including jitter) about the mean zero crossing. Eye template measurement requirements are as defined above. Random jitter is calibrated using a high pass filter with a low frequency corner at 20 MHz and a 20 dB/decade rolloff below this. The required sinusoidal jitter specified in Section 8.6 is then added to the signal and the test load is replaced by the receiver being tested.

Blank page

Annex A Interface Management (Informative)

A.1 Introduction

This appendix contains state machine descriptions that illustrate a number of behaviors that are described in the *RapidIO Part 6: 1x/4x LP-Serial Physical Layer Specification*. They are included as examples and are believed to be correct, however, actual implementations should not use the examples directly.

A.2 Packet Retry Mechanism

This section contains the example packet retry mechanism state machine referred to in Section 5.6, “Packet Transmission Protocol”.

Packet retry recovery actually requires two inter-dependent state machines in order to operate, one associated with the input port and the other with the output port on the two connected devices. The two state machines work together to attempt recovery from a retry condition.

A.2.1 Input port retry recovery state machine

If a packet cannot be accepted by a receiver for reasons other than error conditions, such as a full input buffer, the receiver follows the state sequence shown in Figure A-1.

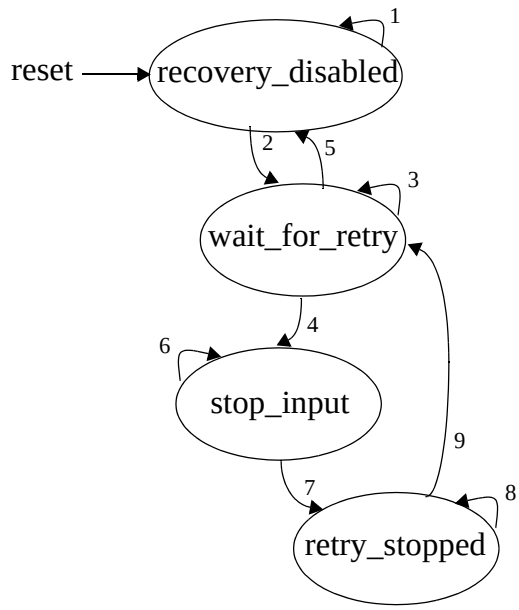


Figure A-1. Input Port Retry Recovery State Machine

Table A-1 describes the state transition arcs for Figure A-1. The states referenced in the comments in quotes are the RapidIO 1x/4x LP-Serial defined status states, not states in this state machine.

Table A-1. Input Port Retry Recovery State Machine Transition Table

Arc	Current State	Next state	cause	Comments
1	recovery_disabled	recovery_disabled	Remain in this state until the input port is enabled to receive packets.	This is the initial state after reset. The input port can't be enabled before the initialization sequence has been completed, and may be controlled through other mechanisms as well, such as a software enable bit.
2	recovery_disabled	wait_for_retry	Input port is enabled.	
3	wait_for_retry	wait_for_retry	Remain in this state until a packet retry situation has been detected.	
4	wait_for_retry	stop_input	A packet retry situation has been detected.	Usually this is due to an internal resource problem such as not having packet buffers available for low priority packets.
5	wait_for_retry	recovery_disabled	Input port is disabled.	
6	stop_input	stop_input	Remain in this state until described input port stop activity is completed.	Send a packet-retry control symbol with the expected ackID, discard the packet, and don't change the expected ackID. This will force the attached device to initiate recovery starting at the expected ackID. Clear the "Port Normal" state and set the "Input Retry-stopped" state.
7	stop_input	retry_stopped	Input port stop activity is complete.	

Table A-1. Input Port Retry Recovery State Machine Transition Table (Continued)

Arc	Current State	Next state	cause	Comments
8	retry_stopped	retry_stopped	Remain in this state until a restart-from-retry or link request (restart-from-error) control symbol is received or an input port error is encountered.	The "Input Retry-stopped" state causes the input port to silently discard all incoming packets and not change the expected ackID value.
9	retry_stopped	wait_for_retry	Received a restart-from-retry or a link request (restart-from-error) control symbol or an input port error is encountered.	Clear the "Input Retry-stopped" state and set the "Port Normal" state. An input port error shall cause a clean transition between the retry recovery state machine and the error recovery state machine.

A.2.2 Output port retry recovery state machine

On receipt of an error-free packet-retry control symbol, the attached output port follows the behavior shown in Figure A-2. The states referenced in the comments in quotes are the RapidIO 8/16 LP-LVDS defined status states, not states in this state machine.

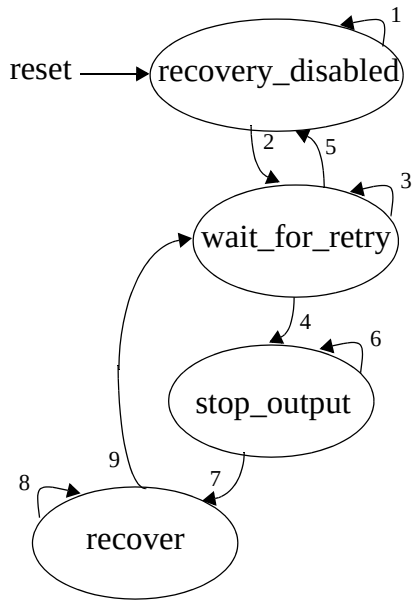


Figure A-2. Output Port Retry Recovery State Machine

Table A-2 describes the state transition arcs for Figure A-2.

Table A-2. Output Port Retry Recovery State Machine Transition Table

Arc	Current State	Next state	cause	Comments
1	recovery_disabled	recovery_disabled	Remain in this state until the output port is enabled to receive packets.	This is the initial state after reset. The output port can't be enabled before the initialization sequence has been completed, and may be controlled through other mechanisms as well, such as a software enable bit.
2	recovery_disabled	wait_for_retry	Output port is enabled.	
3	wait_for_retry	wait_for_retry	Remain in this state until a packet-retry control symbol is received.	The packet-retry control symbol shall be error free.
4	wait_for_retry	stop_output	A packet-retry control symbol has been received.	Start the output port stop procedure.
5	wait_for_retry	recovery_disabled	Output port is disabled.	
6	stop_output	stop_output	Remain in this state until the output port stop procedure is completed.	Clear the "Port Normal" state, set the "Output Retry-stopped" state, and stop transmitting new packets.
7	stop_output	recover	Output port stop procedure is complete.	
8	recover	recover	Remain in this state until the internal recovery procedure is completed.	The packet sent with the ackID value returned in the packet-retry control symbol and all subsequent packets shall be retransmitted. Output port state machines and the outstanding ackID scoreboard shall be updated with this information, then clear the "Output Retry-stopped" state and set the "Port Normal" state to restart the output port. Receipt of a packet-not-accepted control symbol or other output port error during this procedure shall cause a clean transition between the retry recovery state machine and the error recovery state machine. Send restart-from-retry control symbol.
9	recover	wait_for_retry	Internal recovery procedure is complete.	Retransmission has started, so return to the wait_for_retry state to wait for the next packet-retry control symbol.

A.3 Error Recovery

This section contains the error recovery state machine referred to in Section 5.11.2, “Link Behavior Under Error.”

Error recovery actually requires two inter-dependent state machines in order to operate, one associated with the input port and the other with the output port on the two connected devices. The two state machines work together to attempt recovery.

A.3.1 Input port error recovery state machine

There are a variety of recoverable error types described in detail in Section 5.11.2, “Link Behavior Under Error”. The first group of errors are associated with the input port, and consists mostly of corrupt packet and control symbols. An example of a corrupt packet is a packet with an incorrect CRC. An example of a corrupt control symbol is a control symbol with error on the 5-bit CRC control symbol. The recovery state machine for the input port of a RapidIO link is shown in Figure A-3.

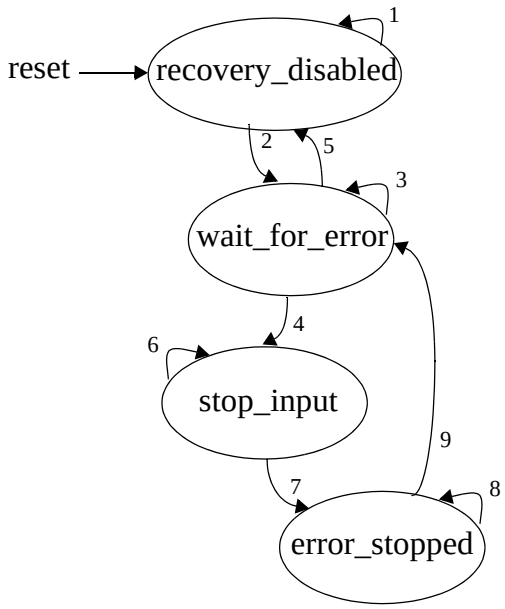


Figure A-3. Input Port Error Recovery State Machine

Table A-3 describes the state transition arcs for Figure A-3. The states referenced in the comments in quotes are the RapidIO 1x/4x LP-Serial defined status states, not states in this state machine.

Table A-3. Input Port Error Recovery State Machine Transition Table

Arc	Current State	Next state	cause	Comments
1	recovery_disabled	recovery_disabled	Remain in this state until error recovery is enabled.	This is the initial state after reset. Error recovery can't be enabled before the initialization sequence has been completed, and may be controlled through other mechanisms as well, such as a software enable bit.
2	recovery_disabled	wait_for_error	Error recovery is enabled.	
3	wait_for_error	wait_for_error	Remain in this state until a recoverable error is detected.	Detected errors and the level of coverage is implementation dependent.
4	wait_for_error	stop_input	A recoverable error has been detected.	An output port associated error will not cause this transition, only an input port associated error.
5	wait_for_error	recovery_disabled	Error recovery is disabled.	
6	stop_input	stop_input	Remain in this state until described input port stop activity is completed.	Send a packet-not-accepted control symbol and, if the error was on a packet, discard the packet and don't change the expected ackID value. This will force the attached device to initiate recovery. Clear the "Port Normal" state and set the "Input Error-stopped" state.
7	stop_input	error_stopped	Input port stop activity is complete.	
8	error_stopped	error_stopped	Remain in this state until a link request (restart-from-error) control symbol is received.	The "Input Error-stopped" state causes the input port to silently discard all subsequent incoming packets and ignore all subsequent input port errors.
9	error_stopped	wait_for_error	Received a link request (restart-from-error) control symbol.	Clear the "Input Error-stopped" state and set the "Port Normal" state, which will put the input port back in normal operation.

A.3.2 Output port error recovery state machine

The second recoverable group of errors described in Section 5.11.2, "Link Behavior Under Error" is associated with the output port, and is comprised of control symbols that are error-free and indicate that the attached input port has detected a transmission error or some other unusual situation has occurred. An example of this situation is indicated by the receipt of a packet-not-accepted control symbol. The state machine for the output port is shown in Figure A-4.

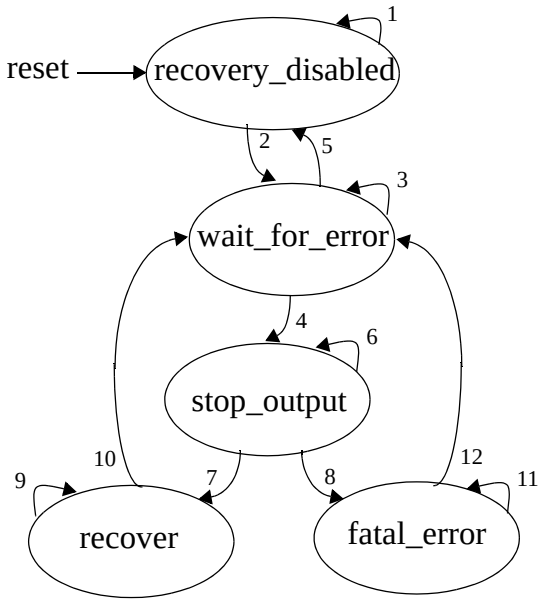


Figure A-4. Output Port Error Recovery State Machine

Table A-4 describes the state transition arcs for Figure A-4. The states referenced in the comments in quotes are the RapidIO 8/16 LP-LVDS defined status states, not states in this state machine.

Table A-4. Output Port Error Recovery State Machine Transition Table

Arc	Current State	Next state	cause	Comments
1	recovery_disabled	recovery_disabled	Remain in this state until error recovery is enabled.	This is the initial state after reset. Error recovery can't be enabled before the initialization sequence has been completed, and may be controlled through other mechanisms as well, such as a software enable bit.
2	recovery_disabled	wait_for_error	Error recovery is enabled.	
3	wait_for_error	wait_for_error	Remain in this state until a recoverable error is detected.	Detected errors and the level of coverage is implementation dependent.
4	wait_for_error	stop_output	A recoverable error has been detected.	An input port associated error will not cause this transition, only an output port associated error.
5	wait_for_error	recovery_disabled	Error recovery is disabled.	

Table A-4. Output Port Error Recovery State Machine Transition Table (Continued)

Arc	Current State	Next state	cause	Comments
6	stop_output	stop_output	Remain in this state until an exit condition occurs.	Clear the "Port Normal" state, set the "Output Error-stopped" state, stop transmitting new packets, and send a link-request/input-status control symbol. Ignore all subsequent output port errors. The input on the attached device is in the "Input Error-stopped" state and is waiting for a link-request/input-status in order to be re-enabled to receive packets. An implementation may wish to time-out several times before regarding a time-out as fatal using a threshold counter or some other mechanism.
7	stop_output	recover	The link-response is received and returned an outstanding ackID value	An outstanding ackID is a value sent out on a packet that has not been acknowledged yet. In the case where no ackID is outstanding the returned ackID value shall match the next expected/next assigned ackID value, indicating that the devices are synchronized. Recovery is possible, so follow recovery procedure.
8	stop_output	fatal_error	The link-response is received and returned an ackID value that is not outstanding, or timed out waiting for the link-response.	Recovery is not possible, so start error shutdown procedure.
9	recover	recover	Remain in this state until the internal recovery procedure is completed.	The packet sent with the ackID value returned in the link-response and all subsequent packets shall be retransmitted. All packets transmitted with ackID values preceding the returned value were received by the attached device, so they are treated as if packet-accepted control symbols have been received for them. Output port state machines and the outstanding ackID scoreboard shall be updated with this information, then clear the "Output Error-stopped" state and set the 'Port Normal" state to restart the output port.
10	recover	wait_for_error	The internal recovery procedure is complete.	retransmission (if any was necessary) has started, so return to the wait_for_error state to wait for the next error.

Table A-4. Output Port Error Recovery State Machine Transition Table (Continued)

Arc	Current State	Next state	cause	Comments
11	fatal_error	fatal_error	Remain in this state until error shutdown procedure is completed.	Clear the "Output Error-stopped" state, set the "Port Error" state, and signal a system error.
12	fatal_error	wait_for_error	Error shutdown procedure is complete.	Return to the wait_for_error state.

Blank page

Annex B Critical Resource Performance Limits (Informative)

The RapidIO LP-Serial layer is intended for use over links whose length ranges from centimeters to tens of meters. The shortest length links will almost certainly use copper printed circuit board traces. The longer lengths will require the use of fiber optics (optical fiber and electro-optical converters) to overcome the high frequency losses of long copper printed circuit board traces or cable. The longer lengths will also have significant propagation delay which can degrade the usable bandwidth of a link.

The serial protocol is a handshake protocol. Each packet transmitted by a port is assigned an ID (the ackID) and a copy of the packet is retained by the port in a holding buffer until the packet is accepted by the port's link partner. The number of packets that a port can transmit without acknowledgment is limited to the lesser of the number of distinct ackIDs and the number of buffers available to hold unacknowledged packets. Which ever is the limiting resource, ackIDs or holding buffers, will be called the "critical resource".

The concern is the time between the assignment of a critical resource to a packet and the release of that resource as a consequence of the packet being accepted by the link partner. Call this time the resource_release_delay. When the resource_release_delay is less than the time it takes to transmit a number of packets equal to the number of distinct critical resource elements, there is no degradation of link performance. When the resource_release_delay is greater than the time it takes to transmit a number of packets equal to the number of distinct critical resource elements, the transmitter may have to stall from time to time waiting for a free critical resource. This will degraded the usable link bandwidth. The onset of degradation will depend on the average length of transmitted packets and the physical length of the link as reflected in the resource_release_delay.

The following example provides some idea of the impact on link performance of the interaction between link length and a critical resource. For purposes of this example, the following assumptions are made.

1. The link is a 4 lane (4x) link.
2. The link uses optical fiber and electro-optical transceivers to allow link lengths of tens of meters. The propagation delay of the optical fiber is 0.45c.
3. The width of the data path within the port is 4 bytes.

4. The data path and logic within the port run at a clock rate equal to the aggregate unidirectional data rate of the link divided by 32. This is referred to as the logic clock. One cycle of this clock is referred to a one logic clock cycle. (If the aggregate unidirectional baud rate of the link was used to compute the logic clock, the baud rate would be divided by 40. With 8B/10B encoding, the baud rate is 1.25 times the data rate.)
5. The minimum length packet header is used. Write request packets have a length of 12 bytes plus a payload containing an integer multiple of 8 bytes. Read request packets have a length of 12 bytes. Read response packets have a length of 8 bytes plus a payload containing an integer multiple of 8 bytes.
6. The beginning and end of each packet is delimited by a control symbol. A single control symbol may delimit both the end of one packet and the beginning of the next packet.
7. Packet acknowledgments are carried in packet delimiter control symbols when ever possible to achieve the efficiency provided by the dual stype control symbol. This implies that a packet acknowledgment must wait for an end-of-packet control symbol if packet transmission is in progress when the packet acknowledgment becomes available.
8. The logic and propagation delay in the packet transmission direction is comprised of the following components.

Table B-12. Packet Transmission Delay Components

Item	Time required
Generate start-of-packet control symbol (critical resource is available)	1 logic clock cycle
Generate start-of-packet control symbol CRC	1 logic clock cycle
8B/10B encode delimiter and start-of-packet control symbol	1 logic clock cycle
Serialize and transmit delimiter and start-of-packet control symbol	1 logic clock cycle
PCB copper and electro-optical transmitter delay	2 ns
Optical fiber delay	fiber_length/0.45c
Electro-optical receiver and pcb copper delay	2 ns
Receive and deserialize delimiter and start-of-packet control symbol	0.5 logic clock cycles
Receive and deserialize packet	depends on packet
Receive and deserialize delimiter and end-of-packet control symbol	1 logic clock cycle
8B/10B decode delimiter and end-of-packet control symbol	1 logic clock cycle
Check CRC of end-of-packet control symbol	1 logic clock cycle
Make packet acceptance decision	1 logic clock cycle

9. The logic and propagation delay in the packet acknowledgment direction is comprised of the following.

Table B-13. Packet Acknowledgment Delay Components

Item	Time required
Wait for end-of-packet if packet transmission is in progress, generate packet-acknowledgment control symbol and control symbol CRC	depends on packet >= 2 logic clock cycles
8B/10B encode delimiter and packet-acknowledgment control symbol	1 logic clock cycle
Serialize and transmit delimiter and packet-acknowledgment control symbol	1 logic clock cycle
PCB copper and electro-optical transmitter delay	2 ns
Optical fiber delay	fiber_length/0.45c
Electro-optical receiver and pcb copper delay	2 ns
Receive and deserialize delimiter and packet-acknowledgment control symbol	0.5 logic clock cycles
8B/10B decode delimiter and packet-acknowledgment control symbol	1 logic clock cycle
Check CRC of packet-acknowledgment control symbol	1 logic clock cycle
Make decision to free critical resource	1 logic clock cycle

The packet times in the above tables depend on packet length which in turn depends on packet type and payload size. Since packet traffic will typically involve a mixture of packet types and payload sizes, the traffic in each direction will be assumed to contain an equal number of read, write and response packets and average payloads of 8, 32, and 64 bytes.

The number of logic clock cycles required to transmit or receive a packet is given in the following table as a function of packet type and payload size.

Table B-14. Packet Delays

Packet Type	Packet Header bytes	Data Payload bytes	Transmit/Receive Time logic clock cycles
Read	12	0	3
Response	8	8	4
		32	10
		64	18
Write	12	8	5
		32	11
		64	19

Using the above table and the assumed equal number of read, write and response packets, the average number of logic clock cycles to transmit or received a packet is 4, 8, and 13.3 respectively for packet payloads of 8, 32, and 64 bytes. The average wait for the completion of a packet being transmitted is assumed to be 1/2 the transmit time.

The following table gives the maximum length of the optical fiber before the packet transmission rate becomes limited by the critical resource for a 4x link operating at unidirectional data rates of 4.0, 8.0 and 10.0 Gb/s.

Table B-15. Maximum Transmission Distances

Number of Critical Resources Available	Data Payload (Bytes)	Maximum Fiber Length Before Critical Resource Limited (Meters)		
		4.0 Gb/s link	8.0 Gb/s link	10.0 Gb/s link
4	8	-	-	-
	32	4.3	1.9	1.4
	64	11.4	5.5	4.3
8	8	9.7	4.6	3.5
	32	23.6	11.5	9.1
	64	42.2	20.8	16.6
16	8	31.1	15.3	12.1
	32	62.2	30.8	24.6
	64	103.7	51.6	41.1
24	8	52.5	26.0	20.7
	32	100.8	50.2	40.0
	64	165.2	82.3	65.7
32	8	74.0	36.7	29.3
	32	139.5	69.5	55.5
	64	226.7	113.1	90.3

Annex C Manufacturability and Testability (Informative)

It is not possible in many cases for assembly vendors to verify the integrity of soldered connections between components and the printed circuit boards to which they are attached. Alternative methods to direct probing are needed to insure high yields for printed circuit assemblies which include LP-Serial RapidIO devices.

It is recommended that component vendors support IEEE Std. 1149.6 (commonly known as “AC-JTAG”) on all connections to LP-Serial RapidIO links. (Note: IEEE Std. 1149.6 is needed, in addition to IEEE Std. 1149.1, due the fact that RapidIO LP-Serial lanes are AC-coupled.) This provides boundary scan capability on all TD, TDN, RD, and RDN pins on a component which supports one or more LP-Serial RapidIO ports.

The IEEE Std. 1149.6 is available from the IEEE.

Blank page

Glossary of Terms and Abbreviations

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

A

AC Coupling. A method of connecting two devices together that does not pass DC.

Agent. A processing element that provides services to a processor.

ANSI. American National Standards Institute.

B

Big-endian. A byte-ordering method in memory where the address n of a word corresponds to the most significant byte. In an addressed memory word, the bytes are ordered (left to right) 0, 1, 2, 3, with 0 being the most significant byte.

Bridge. A processing element that connects one computer bus to another, allowing a processing element on one bus to access an processing element on the other.

C

Capability registers (CARs). A set of read-only registers that allow a processing element to determine another processing element's capabilities.

Code-group. A 10-bit entity produced by the 8B/10B encoding process and the input to the 8B/10B decoding process.

Command and status registers (CSRs). A set of registers that allow a processing element to control and determine the status of another processing element's internal hardware.

Control symbol. A quantum of information transmitted between two linked devices to manage packet flow between the devices.

CRC. Cyclic redundancy code

D

Deadlock. A situation in which two processing elements that are sharing resources prevent each other from accessing the resources, resulting in a halt of system operation.

Deferred or delayed transaction. The process of the target of a transaction capturing the transaction and completing it after responding to the the source with a retry.

Destination. The termination point of a packet on the RapidIO interconnect, also referred to as a target.

Device. A generic participant on the RapidIO interconnect that sends or receives RapidIO transactions, also called a processing element.

Device ID. The identifier of a processing element connected to the RapidIO interconnect.

Direct Memory Access (**DMA**). A process element that can independently read and write system memory.

Distributed memory. System memory that is distributed throughout the system, as opposed to being centrally located.

Double word. An eight byte quantity, aligned on eight byte boundaries.

E

EMI. Electromagnetic Interference.

End point. A processing element which is the source or destination of transactions through a RapidIO fabric.

End point device. A processing element which contains end point functionality.

End point free device. A processing element which does not contain end point functionality.

Ethernet. A common local area network (LAN) technology.

External processing element. A processing element other than the processing element in question.

F

Fabric. A series of interconnected switch devices, typically used in reference to a switch fabric.

Field or Field name. A sub-unit of a register, where bits in the register are named and defined.

FIFO. First in, first out.

Full-duplex. Data can be transmitted in both directions between connected processing elements at the same time.

G

Globally shared memory (GSM). Cache coherent system memory that can be shared between multiple processors in a system.

H

Half-word. A two byte or 16-bit quantity, aligned on two byte boundaries.

Header. Typically the first few bytes of a packet, containing control information.

I

Initiator. The origin of a packet on the RapidIO interconnect, also referred to as a source.

I/O. Input-output.

IP. Intellectual Property

ITU. International Telecommunication Union.

L

Little-endian. A byte-ordering method in memory where the address n of a word corresponds to the least significant byte. In an addressed memory word, the bytes are ordered (left to right) 3, 2, 1, 0, with 3 being the most significant byte.

Local memory. Memory associated with the processing element in question.

LP. Link Protocol

LSB. Least significant byte.

LVDS. Low voltage differential signaling.

M

Message passing. An application programming model that allows processing elements to communicate through special hardware instead of through memory as with the globally shared memory programming model.

MSB. Most significant byte.

N

Non-coherent. A transaction that does not participate in any system globally shared memory cache coherence mechanism.

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.

P

Packet. A set of information transmitted between devices in a RapidIO system.

Payload. The user data embedded in the RapidIO packet.

PCB. Printed circuit board.

PCS. Physical Coding Sublayer.

PMA. Physical Media Attachment.

Port-write. An address-less write operation.

Priority. The relative importance of a transaction or packet; in most systems a higher priority transaction or packet will be serviced or transmitted before one of lower priority.

Processing Element (**PE**). A generic participant on the RapidIO interconnect that sends or receives RapidIO transactions, also called a device.

Processor. The logic circuitry that responds to and processes the basic instructions that drive a computer.

R

Receiver. The RapidIO interface input port on a processing element.

S

Sender. The RapidIO interface output port on a processing element.

Semaphore. A technique for coordinating activities in which multiple processing elements compete for the same resource.

Serializer. A device which converts parallel data (such as 8-bit data) to a single bit-wide datastream.

Source. The origin of a packet on the RapidIO interconnect, also referred to as an initiator.

SRAM. Static random access memory.

Switch. A multiple port processing element that directs a packet received on one of its input ports to one of its output ports.

T

Target. The termination point of a packet on the RapidIO interconnect, also referred to as a destination.

Transaction. A specific request or response packet transmitted between end point devices in a RapidIO system.

Transaction request flow. A sequence of transactions between two processing elements that have a required completion order at the destination processing element. There are no ordering requirements between transaction request flows.

W

Word. A four byte or 32 bit quantity, aligned on four byte boundaries.

Write port. Hardware within a processing element that is the target of a port-write operation.

Blank page

Blank page

Blank page