RapidIO™ Interconnect Specification
Part 4: Physical Layer 8/16 LP-LVDS Specification

3.0, 10/2013

RapidIO.org

Revision History

Revision	Description	Date
1.1	First public release	03/08/2001
1.2	Technical changes: incorporate Rev. 1.1 errata rev. 1.1.1, errata 3, showing 02-02-00009	06/26/2002
1.3	Technical changes: incorporate Rev 1.2 errata 1 as applicable, the following errata showings: 03-07-00000.002, 03-12-00000.002, 03-12-00002.004 and the following new features showings: 02-06-00001.004, 02-03-00003.004, 04-08-00013.002, 04-09-00022.002 Convert to ISO-friendly templates; re-formatted	02/23/2005
2.0	Technical changes: errata showing 06-04-00000.003 new features showing 05-04-00001.005	06/14/2007
2.1	No technical changes	07/09/2009
2.2	Technical changes: errata showing 10-08-00001.005, Consolidated Comments on 11-01-00000.000	05/05/2011
3.0	Changed RTA contact information. No technical changes. This part of the specification is now deprecated.	10/11/2013

Questions regarding RapidIO.org, specifications, or membership should be forwarded to:

RapidIO.org
8650 Spicewood Springs #145-515
Austin, TX 78759
512-827-7680 Tel.

Table of Contents

Chapter 1 Overview

Chapter 2 Physical Layer Protocol

Table of Contents

Table of Contents

Table of Contents

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

This chapter provides an overview of the *RapidIO Part 4: 8/16 LP-LVDS Physical Layer Specification*, including a description of the relationship between this specification and the other specifications of the RapidIO interconnect.

1.2 Overview

The *RapidIO Part 4: 8/16 LP-LVDS Physical Layer Specification* is one of the RapidIO physical layer specifications that define the device to device communications protocol and packet formats. Other RapidIO physical layer specifications include the *RapidIO Part 6: 1x/4x LP-Serial Physical Layer Specification*.

The physical layer defines the signal definitions, flow control and error management for RapidIO. An 8-bit and 16-bit parallel (8/16 LP-LVDS), point-to-point interface is defined in this specification. An 8/16 LP-LVDS device interface contains a dedicated 8- or 16-bit input port with clock and frame signals, and a 8- or 16-bit output port with clock and frame signals. A source-synchronous-clock signal clocks packet data on the rising and falling edges. A frame signal provides a control reference. Differential signaling is used to reduce interface complexity, provide robust signal quality, and promote good frequency scalability across printed circuit boards and connectors.

1.3 Features of the Input/Output Specification

The following are features of the RapidIO I/O specification designed to satisfy the needs of various applications and systems:

1.3.1 Functional features

- RapidIO provides a flow control mechanism between devices that communicate on the RapidIO interconnect fabric, because infinite data buffering is not available in a device.

1.3.2 Physical Features

- Connections are point-to-point unidirectional, one in and one out, with 8-bit or 16-bit ports
- Physical layer protocols and packet formats are to some degree independent of the topology of the physical interconnect; however; the physical structure is assumed to be link-based.
- There is no dependency in RapidIO on the bandwidth or latency of the physical fabric.
- Physical layer protocols handle out-of-order and in-order packet transmission and reception.
- Physical layer protocols are tolerant of transient errors caused by high frequency operation of the interface or excessive noise in the system environment.

1.3.3 Performance Features

- Physical protocols and packet formats allow for the smallest to the largest data payload sizes
- Packet headers are as small as possible to minimize the control overhead and are organized for fast, efficient assembly and disassembly.
- Multiple transactions are allowed concurrently in the system, preventing much potential system input from being wasted.
- The electrical specification allows for the fastest possible speed of operation for future devices.

1.4 Contents

RapidIO Part 4: 8/16 LP-LVDS Physical Layer Specification contains nine chapters and an appendix:

- Chapter 1, “Overview” (this chapter) provides an overview of the specification
- Chapter 2, “Physical Layer Protocol,” describes the physical layer protocol for packet delivery to the RapidIO fabric, including packet transmission, flow control, error management, and link maintenance protocols.
- Chapter 3, “Packet and Control Symbol Transmission,” defines packet and control symbol delineation and alignment on the physical port and mechanisms to control the pacing of a packet.
- Chapter 4, “Control Symbol Formats,” explains the physical layer control formats that manage the packet delivery protocols mentioned in Chapter 2.
- Chapter 5, “8/16 LP-LVDS Registers,” describes the register set that allows an external processing element to determine the physical capabilities and status of an 8/16 LP-LVDS RapidIO implementation.

- Chapter 6, “System Clocking Considerations,” discusses the RapidIO synchronous clock and how it is distributed in a typical switch configuration.
- Chapter 7, “Board Routing Guidelines,” explains board layout guidelines and application environment considerations for the RapidIO architecture.
- Chapter 8, “Signal Descriptions,” contains the signal pin descriptions for a RapidIO end point device.
- Chapter 9, “Electrical Specifications,” describes the low voltage differential signaling (LVDS) electrical specifications of the RapidIO 8/16 LP-LVDS device.
- Annex A, “Interface Management (Informative),” contains information pertinent to interface management in a RapidIO system, including SECDED error tables, error recovery, link initialization, and packet retry state machines.

1.5 Terminology

Refer to the Glossary at the back of this document.

1.6 Conventions

||	Concatenation, used to indicate that two fields are physically associated as consecutive bits
ACTIVE_HIGH	Names of active high signals are shown in uppercase text with no overbar. Active-high signals are asserted when high and not asserted when low.
$\overline{\text{ACTIVE\_LOW}}$	Names of active low signals are shown in uppercase text with an overbar. Active low signals are asserted when low and not asserted when high.
italics	Book titles in text are set in italics.
REG[FIELD]	Abbreviations or acronyms for registers are shown in uppercase text. Specific bits, fields, or ranges appear in brackets.
TRANSACTION	Transaction types are expressed in all caps.
operation	Device operation types are expressed in plain text.
n	A decimal value.
[n-m]	Used to express a numerical range from n to m.
0bnn	A binary value, the number of bits is determined by the number of digits.
0x*nn*	A hexadecimal value, the number of bits is determined by the number of digits or from the surrounding context; for

	example, 0x*nn* may be a 5, 6, 7, or 8 bit value.
x	This value is a don’t care

Chapter 2 Physical Layer Protocol

2.1 Introduction

This chapter describes the *RapidIO Part 4: 8/16 LP-LVDS Physical Layer Specification* physical layer protocol for packet delivery to the interconnect fabric including packet transmission, flow control, error management, and other system functions. See the user's manual or implementation specification for specific implementation details of a device.

2.2 Packet Exchange Protocol

The *RapidIO Part 4: 8/16 LP-LVDS Physical Layer Specification* defines an exchange of packet and acknowledgment control symbols in which a destination or intermediate processing element (such as a switch) acknowledges receipt of a request or response packet from a source.

If a packet cannot be accepted for any reason, an acknowledgment control symbol indicates that the original packet and any already transmitted subsequent packets should be resent. This behavior provides a flow control and transaction ordering mechanism between processing elements. Figure 2-1 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 transaction. An end point device shall transmit an acknowledge control symbol for a request before the response transaction corresponding to that request.

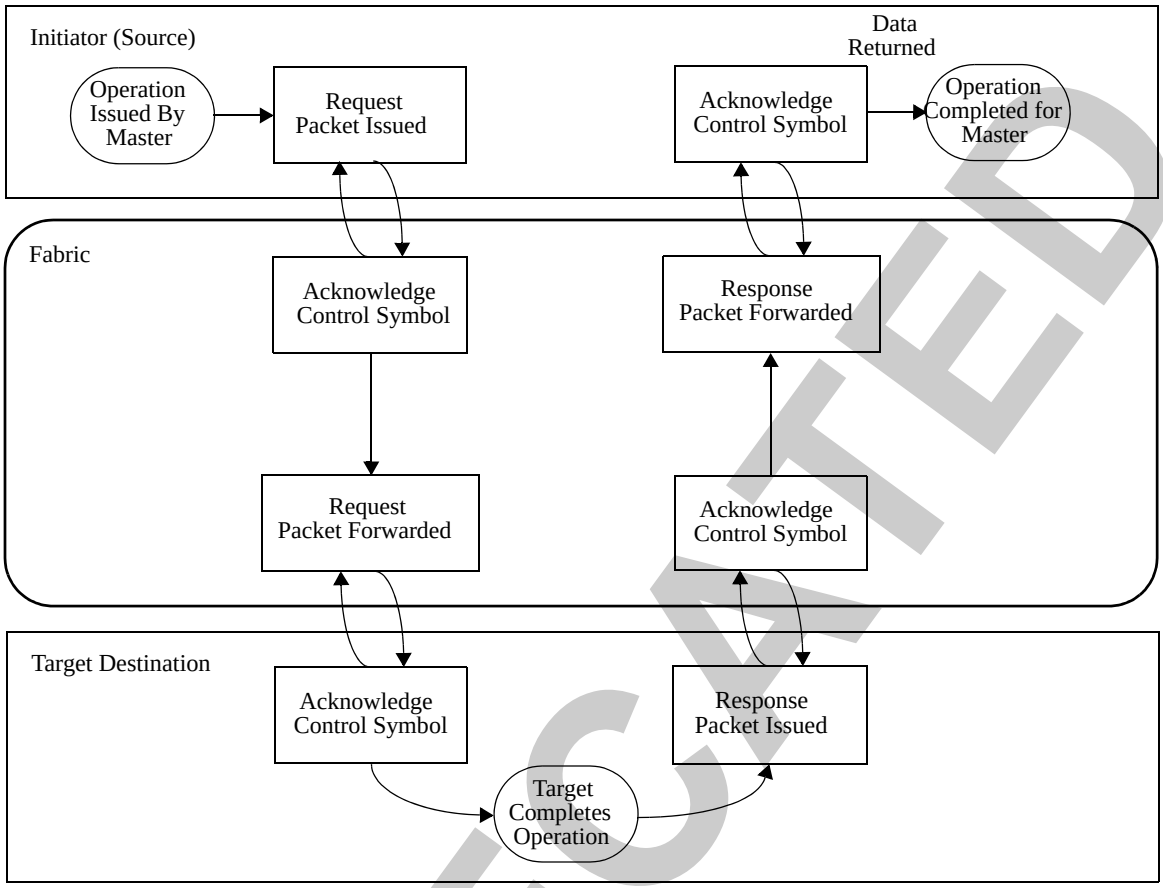


Figure 2-1. Example Transaction with Acknowledge

2.2.1 Packet and Control Alignment

All packets defined by the combination of this specification and the appropriate logical and transport specifications are aligned to 16-bit boundaries, however, all packets and control symbols sent over the 8-bit and 16-bit ports are aligned to 32-bit boundaries. This alignment allows devices to work on packets using a larger internal width thus requiring lower core operating frequencies. Packets that are not naturally aligned to a 32-bit boundary are padded. See Figure 2-11 and Figure 2-12 for examples of padded packets. Control symbols are nominally 16-bit quantities, but are defined as a 16-bit control symbol followed by a bit-wise inverted copy of itself to align it to the 32-bit boundary. This, in turn, adds error detection capability to the interface. These 32-bit quantities are referred to as aligned control symbols.

The 16-bit wide port is compatible with an 8-bit wide port. If an 8-bit wide port is properly connected to a 16-bit wide port, the port will function as an 8-bit interface between the devices. Port width connections are described in Chapter 8, “Signal Descriptions”.

2.2.2 Acknowledge Identification

A packet requires an identifier to uniquely identify its acknowledgment. This identifier, known as the acknowledge ID (or ackID), is three bits, allowing for a range of one to eight outstanding unacknowledged request or response packets between adjacent processing elements, however only up to seven outstanding unacknowledged packets are allowed at any one time. The ackIDs are assigned sequentially (in increasing order, wrapping back to 0 on overflow) to indicate the order of the packet transmission. The acknowledgments themselves are a number of aligned control symbols defined in Chapter 4, “Control Symbol Formats.”

2.3 Field Placement and Definition

This section contains the 8/16 LP-LVDS specification for the additional physical layer bit fields and control symbols required to implement the flow control, error management, and other specified system functions.

2.3.1 Flow Control Fields Format

The fields used to control packet flow in the system are described in Table 2-1.

Table 2-1. Fields that Control Packet Flow

Field	Description
S	0b0 - RapidIO request or response packet 0b1 - Physical layer control symbol
$\overline{S}$	Inverse of S-bit for redundancy (odd parity bit)
ackID	Acknowledge ID is the packet identifier for acknowledgments back to the packet sender—see Section 2.2.2
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 2.3.2 for an explanation of prioritizing packets
prio	Sets packet priority: 0b00 - lowest priority 0b01 - medium priority 0b10 - high priority 0b11 - highest priority See Section 2.3.2 for an explanation of prioritizing packets
buf_status	Specifies the number of available packet buffers in the receiving device. See Section 2.3.4 and Table 2-2.
stype	Control symbol type—see Chapter 4, “Control Symbol Formats” for definition.
rsrv	Reserved

Table 2-2. buf_status Field Definition

buf_status Encoding Value	Description
0b0000	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 control symbol is the number of maximum packets that can be accepted, inclusive of the effect of the packet being accepted or retried. **Value 0-13**: 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 14**: The value 14 signifies that the downstream device can receive 14 or more new maximum sized packets. **Value 15**: The downstream device can receive an undefined number of maximum sized packets, and relies on the retry protocol for flow control.
0b0001	
0b0010	
0b0011	
0b0100	
0b0101	
0b0110	
0b0111	
0b1000	
0b1001	
0b1010	
0b1011	
0b1100	
0b1101	
0b1110	
0b1111	

Figure 2-2 shows the format for the physical layer fields for packets. In order to pad packets to the 16-bit boundary there are two reserved bits in a packet's physical layer fields. These bits are assigned to logic 0 when generated and ignored when received.

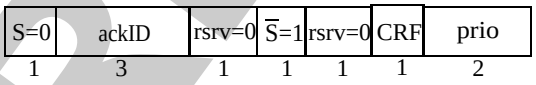


Figure 2-2. Packet Physical Layer Fields Format

Figure 2-3 shows the basic format for the physical layer fields for control symbols. In order to pad the control symbol to the 16-bit boundary there are four reserved bits in the control symbol. These bits are assigned to logic 0 when generated and ignored when received. The field formats for all control symbols are defined in Chapter 4, "Control Symbol Formats."

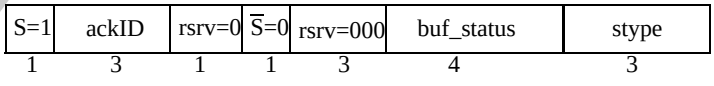


Figure 2-3. Control Symbol Physical Layer Fields Format

Figure 2-4 shows how the physical layer fields are prefixed to the combined

transport and logical layer packet.

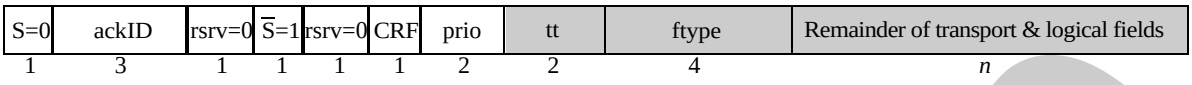


Figure 2-4. Flow Control Fields Bit Stream

The unshaded fields are the physical layer fields defined by this physical specification. The shaded fields are the bits associated with the combined transport and logical transaction definitions. The first transport and logical field shown is the two bit tt field specified in the *RapidIO Part 3: Common Transport Specification*. The second field is the four bit format type (ftype) defined in the logical specifications. The third combined field is the remainder of the transport and logical packet of a size determined by those specifications.

2.3.2 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 2-3.

Table 2-3. 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 similarily as specified in Table 2-4. 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 2-4. 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
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.

2.3.3 Transaction and Packet Delivery

Certain physical layer fields and a number of control symbols are used for handling flow control. One physical layer field contains the ackID field (Table 2-1), which is assigned by the sending processing element, and expected by the receiving processing element, in a sequential fashion.

Packets shall be accepted by the receiving processing element only when ackID values of successive packets occur in the specified sequence. The receiving processing element signals the acceptance of a packet by returning a packet-accepted control symbol to the sender. This order allows a device to detect when a packet has been lost and also provides a mechanism to maintain ordering.

A device that retries a packet (by returning a packet-retry control symbol to the sender) due to some temporary internal condition shall silently discard all new incoming packets until it receives a restart-from-retry control symbol from the sender. The sender then retransmits all packets starting from the retried ackID, reestablishing the proper ordering between the devices. The packet sent with the retried ackID may be the original retried packet or a higher priority packet, if one is available, allowing higher priority packets to bypass lower priority packets across the link. This behavior is shown in an example state machine in Section A.3, “Packet Retry Mechanism.”

Similarly, if a receiving processing element encounters an error condition, it shall

return a packet-not-accepted control symbol, indicating an error condition, to the sender. It shall also silently discard all new incoming packets. If the error condition is due to a transmission error the sender may able to recover from the effects of the error condition. The error recovery mechanism is described in Section 2.4.5.

A retried transaction shall eventually be retransmitted by the sending device.

2.3.3.1 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, the 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).**

2.3.3.2 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 carry 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). The CRF bit setting is assigned the same value as was received in the associated request packet if the CRF bit is supported by the receiving end point.

This requirement is specified in Table 1-3 and Table 2-4. 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 1-3 and Table 2-4 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 1-3 and Table 2-4 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 1-3 and Table 2-4.)
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 1-3 and Table 2-4.)
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 be 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 in 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.

2.3.4 Resource Allocation

This section defines RapidIO LP-LVDS link level flow control. The flow control operates between each pair of ports connected by an LP-LVDS 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-LVDS protocol defines two methods or modes of flow control. These are named receiver-controlled flow control and transmitter-controlled flow control. Every RapidIO LP-LVDS port shall support receiver-controlled flow control. LP-LVDS ports may optionally support transmitter-controlled flow control.

2.3.4.1 Receiver-Controlled Flow Control

Receiver-controlled flow control is the simplest and most 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 1s 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.

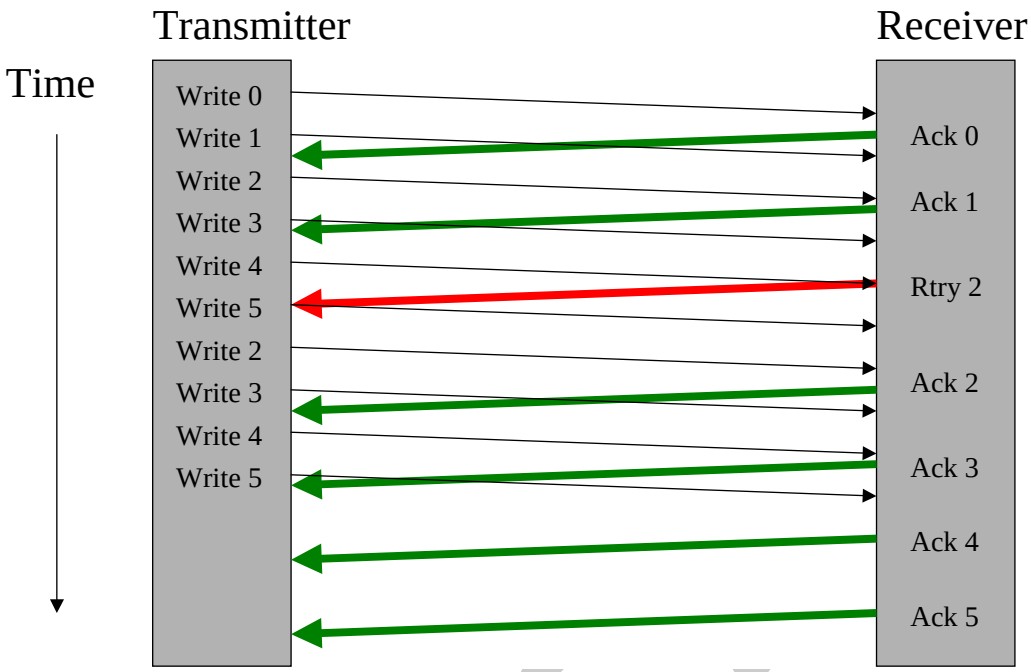
A port operating in receiver-controlled flow control mode accepts or rejects each inbound packet based on whether the receiving port has enough buffer space available at the priority level of the packet. If there is enough buffer space available, the port accepts the packet and transmits a packet-accepted 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) a packet, it behaves as described in Section 2.3.3, "Transaction and Packet Delivery". As part of the 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 also behaves as described in Section 2.3.3. As part of the recovery process, the port receiving the packet-retry control symbol sends a restart-from-retry control symbol which causes its link partner to 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 2-5 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 2-5. Receiver-Controlled Flow Control



2.3.4.2 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 1s 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 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.

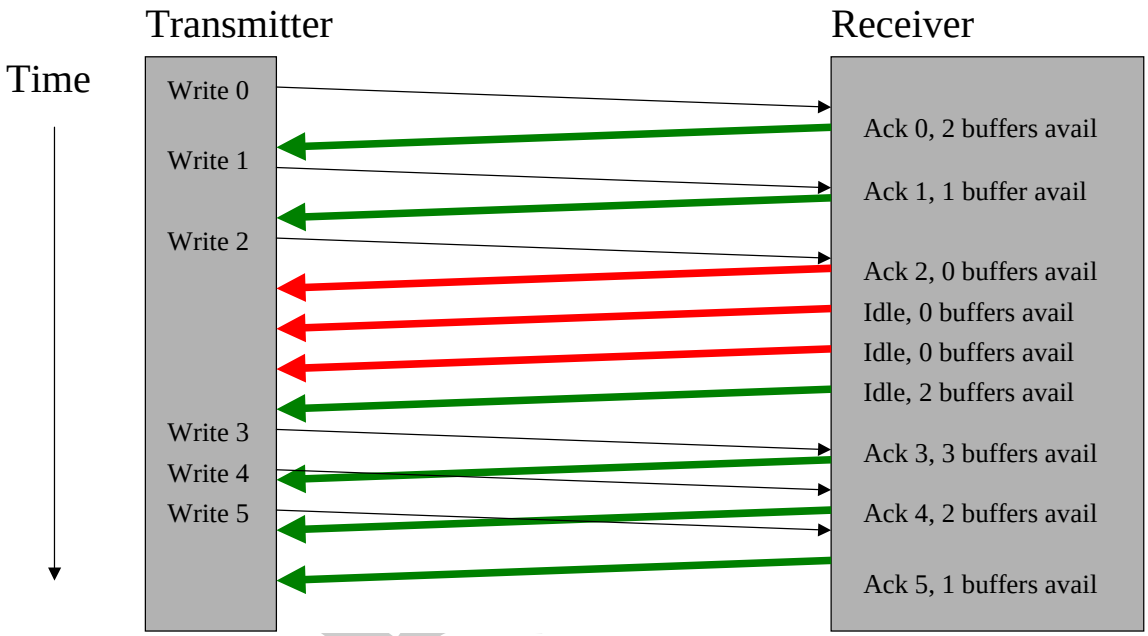
A port informs its link partner when the number of free buffers available for packet reception changes. The new value of buf_status is conveyed in the buf_status field in every control symbol containing the field that the port transmits. Each change in

the number of free buffers a port has available for packet reception need not be conveyed to the link partner.

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 behaves as described in Section 2.3.3.

A simple example of transmitter-controlled flow control is shown in Figure 2-6.

Figure 2-6. Transmitter-Controlled Flow Control



2.3.4.3 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 2.3.2, “Packet Priority and Transaction Request Flows” 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-LVDS 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 idle 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 idle 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 14, 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.

2.3.4.4 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 < 15) {
        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 control symbol containing a buf_status field 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 = received_buf_status;
```

2.3.4.5 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.

2.3.5 Flow Control Mode Negotiation

Immediately following the initialization of a link, each port begins sending idle 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.

2.4 Error Detection and Recovery

Error detection and recovery is becoming a more important issue for many systems as operational frequencies increase and system electrical margins are reduced. The 8/16 LP-LVDS specification provides extensive error detection and recovery by combining retry protocols, cyclic redundancy codes, and single and multiple error detect capabilities, thereby tolerating all single-bit errors and many multiple bit errors. One goal of the error protection strategy is to keep the interconnect fabric from having to regenerate a CRC value as the packet moves through the fabric. All RapidIO ports require error checking.

2.4.1 Control Symbol Protection

The control symbols defined in this specification are protected in two ways:

- The S bit, distinguishing a control symbol from a packet header, has an odd parity bit to protect a control symbol from being interpreted as a packet.
- The entire aligned control symbol is protected by the bit-wise inversion of the control symbol used to align it to the 32-bit boundary described in Section 2.2.1. This allows extensive error detection.

A transmission error in the buf_status field, regardless of the control symbol type, may optionally not be treated as an error condition because it is always a reserved or an information only field that is not critical for proper system behavior. For example, if a corrupt value of buf_status is used, a low value may temporarily prevent a packet from being issued, or a high value may result in a packet being issued when it should not have been, causing a retry. In either case the problems are temporary and will properly resolve themselves through the existing protocol.

2.4.2 Packet Protection

The packets specified in the *RapidIO Common Transport Specification* and the *RapidIO Logical Specification* are protected with a CRC code that also covers the two bit priority field of this specification. The S bit is duplicated as in the control symbols to protect the packet from being interpreted as a control symbol, and the packet is also protected by protocol as described below.

Figure 2-7 shows the error coverage for the first 16 bits of a packet header. CRC protects the prio, tt, and ftype fields and one of the reserved bits as well as the remainder of the transport and logical fields. Since a new packet has an expected value for the ackID field at the receiver, bit errors on this field are easily detected and the packet is not accepted due to the unexpected value. An error on the S bit is detected with the redundant inverted S parity bit.

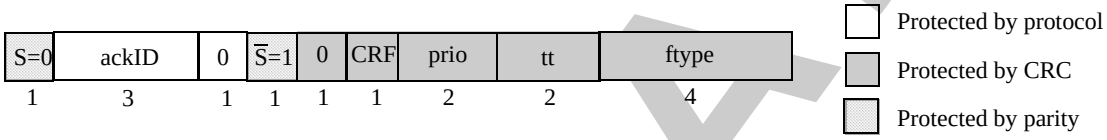


Figure 2-7. Error Coverage of First 16 Bits of Packet Header

This structure does not require that a packet's CRC value be regenerated when the uncovered physical fields are assigned in the fabric.

NOTE:

All packets defined in the combination of this specification and the RapidIO interconnect logical and common transport specifications are now evenly divisible by 16 bits, or the complete packets are now naturally 16-bit aligned. This is illustrated in Figure 2-8. The leading 16 bits of the packet are referred to as the *first symbol* of the packet. The first symbol of a packet shall always land on the most significant half of the 32-bit boundary. Other aligned 16-bit packet quantities are also referred to as symbols.

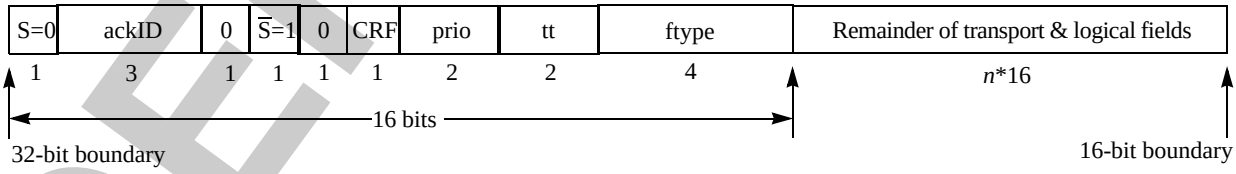


Figure 2-8. Naturally Aligned Packet Bit Stream

2.4.3 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 timeout 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 timeout is not signaled. The response to this error condition is implementation dependent.

The RapidIO specifications assume an implementation has timeout counters for the physical layer, the port link timeout counters, and counters for the logical layer, the port response timeout counters. The logical layer timers are discussed here in the physical layer specification because the packet delivery mechanism is an artifact of the physical layer. The values for these counters are specified in the physical layer registers in Chapter 5, “8/16 LP-LVDS Registers,” on page 73. The interpretation of the 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 timeout occurs between the transmission of a packet and the receipt of an acknowledgment control symbol. This timeout interval is likely to be comparatively short because the packet and acknowledgment pair must only traverse a single link. For the purpose of error recovery, a port link timeout should be treated as a packet acknowledge error.

The logical layer timeout occurs between the issuance of a request packet that requires a response packet and the receipt of that response packet. This timeout 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 timeout will occur. This timeout 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 timeout 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 is behavior is supported by the issuing processing element, the port response timeout implementation must look for both responses, regardless of which links they are returned on.

2.4.4 Implementation Note: Transactional Boundaries

A system address map usually contains memory boundaries that separate one type of memory space from another. Memory spaces are typically allocated with a preset minimum granularity. These spaces are often called page boundaries. Page boundaries allow the operating system to manage the entire address space through a standard mechanism. These boundaries are often used to mark the start and end of read-only space, peripheral register space, data space, and so forth.

RapidIO allows DMA streaming of data between two processing elements. Typically, in system interfaces that allow streaming, the targeted device of the transaction has a way to disconnect from the master once a transactional boundary has been crossed. The RapidIO specifications do not define a page boundary, nor a mechanism by which a target can disconnect part way through a transaction. Therefore, it is up to the system software and/or hardware implementation to guarantee that a transaction can complete gracefully to the address space requested.

As an example, a RapidIO write transaction does not necessarily have a size associated with it. Given a bus error condition whereby a packet delimiting control symbol is missed, the target hardware could continue writing data beyond the intended address space, thus possibly corrupting memory. Hardware implementations should set up page boundaries so this condition does not occur. In such an implementation, should a transaction cross the boundary, an error should be indicated and the transaction discarded.

2.4.5 Link Behavior Under Error

Transmission error detection is done at the input port, and all transmission error recovery is also initiated at the input port. Error detection can be done in a number of ways and at differing levels of complexity depending upon the requirements and implementation of a device.

2.4.5.1 Recoverable Errors

Four basic types of errors are detected by a port: an error on a packet, an error on a control symbol, an indeterminate error (an S bit parity failure), and a timeout waiting for a control symbol. A detailed state machine description of the behavior described in the sections below is included in Section A.2, "Error Recovery". The error recovery mechanism requires that a copy of each transmitted data packet be retained by the sending port so that the packet can be retransmitted if it is not accepted by the receiving port. The copy is retained until the sending port either receives a packet-accepted control symbol for the packet or determines that the packet has encountered an unrecoverable error condition.

When a sending port detects that the receiving port has not accepted a packet because one or more of the errors listed above has occurred (or the port has received a retry control symbol), the sending port resets the link timeout counters for the

affected packet and all subsequently transmitted data packets. This prevents the generation of spurious timeout errors.

Any awaiting higher priority data packets are transmitted and all unaccepted data packets are retransmitted by the sending port. The number of times a data packet is retransmitted due to a recoverable error before the sending port declares an unrecoverable error condition exists is implementation dependent.

2.4.5.1.1 Packet Errors

Three types of packet errors exist: a packet with an unexpected ackID value, a corrupted packet indicated by a bad CRC value, and a packet that overruns some defined boundary such as the maximum data payload or a transactional boundary as described in Section 2.4.4. A processing element that detects a packet error immediately transitions into an “Input Error-stopped” state and silently discards all new packets until it receives a restart-from-error control symbol from the sender. The device also sends a packet-not-accepted control symbol with an undefined ackID value back to the sender. The sender then initiates recovery as described in Section 2.4.5.1.2 for a packet acknowledge error.

2.4.5.1.2 Control Symbol Errors

There are three types of detectable control symbol errors: a packet acknowledge error, a corrupt control symbol error, and an uncorrupted protocol violating control symbol.

A packet acknowledge error is a packet-accepted or packet-retry control symbol which has an unexpected ackID value or an unexpected packet-not-accepted control symbol. This error shall cause the receiving device to enter an “Output Error-stopped” state, immediately stop transmitting new packets, and issue a restart-from-error control symbol. The restart-from-error control symbol receives a response containing receiver internal state, including the expected ackID. This expected ackID indicates to the sender where to begin re-transmission because the interface may have gotten out of sequence. The sender shall then back up to the appropriate unaccepted packet and begin re-transmission.

The following is an example of a packet acknowledge error and recovery from that error. The 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 restart-from-error control symbol to the receiver. The receiver then returns a 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 re-transmit packets 3, 4, and 5
- expecting ackID = 4 - sender must re-transmit packets 4 and 5
- expecting ackID = 5 - sender must re-transmit packet 5
- expecting ackID = 6 - receiver got all packets, resume operation

• expecting ackID = anything else - fatal (non-recoverable) error

A corrupt control symbol is detected as a mismatch between the true and complement 16-bit halves of the aligned control symbol. A corrupt control symbol shall cause the receiver to enter the “Input Error-stopped” state and send a packet-not-accepted control symbol. This informs the sending device that a transmission error has occurred and the sender shall enter the recovery process described above, sending a restart-from-error control symbol.

An uncorrupted protocol violating control symbol is a control symbol that is received unexpectedly. Some examples of this type of error are:

- an unsolicited packet-accepted or packet-retry control symbol
- a restart-from-retry control symbol received while in the Input OK state,
- a 2nd link-request received before returning a link-response, or
- a packet-accepted received before packet transmission has completed

Such errors are an indication of either an otherwise undetectable multi-bit error on the link or a blatant protocol violation by the sender. Such errors may cause unpredictable behavior and the link may not be recoverable. Upon detecting such an error, the receiver shall enter Input Error-stopped state and/or Output Error-stopped state if attempting to recover from these types of errors.

2.4.5.1.3 Indeterminate errors

An indeterminate error is an S bit parity error in which it is unclear whether the information being received is for a packet or a control symbol. These errors shall be handled as a corrupt control symbols.

2.4.5.1.4 Timeout Error

A link timeout on an acknowledge control symbol for a packet is treated like an acknowledge control symbol with an unexpected ackID value.

2.4.6 CRC Operation

A 16-bit CRC is selected as the method of error detection for the 8/16 LP-LVDS physical layer. This CRC is generated over all of a packet header, and all of the data payload except the first 6 bits of the added physical layer fields as shown in Figure 2-7. This checksum is appended to a packet in one of two ways. For a packet that has up to 80 bytes of header (including all logical, transport, and 8/16 LP-LVDS fields) and logical data payload, a single CRC value is appended to the packet. For packets with greater than 80 bytes of header and logical data payload, a CRC value is inserted after the first 80 bytes, aligning it to the first half of the 32-bit alignment boundary, and a second CRC value is appended at the end of the packet. The second CRC value is a continuation of the first and included in the running calculation, meaning that the running CRC value is not re-initialized after it is inserted after the first 80 bytes of the packet. This allows intervening devices to regard the embedded

CRC value as 2 bytes of packet payload for CRC checking purposes.

NOTE:

The embedded CRC value is itself used in the running CRC. As a result, from the CRC generator's point of view the running CRC value is guaranteed to be all logic 0s because the running CRC is XORed with itself. This fact may be useful in an 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. If the final appended CRC value does not cause the total packet to align to the 32-bit boundary, a 2 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.

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 1.3.5, 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.

Switch devices shall maintain the packet error coverage internally in order to preserve the integrity of the packets though the fabric. This will prevent undetected device internal errors such as SRAM bit errors from silently corrupting the system. The simplest method for preserving error coverage is to pass the CRC values 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 thought 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.

Figure 2-9 is an example of a naturally 32-bit aligned packet of less than or equal to 80 bytes.

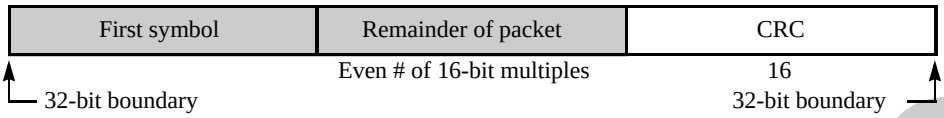


Figure 2-9. Naturally Aligned Packet Bit Stream Example 1

Figure 2-10 is an example of a naturally 32-bit aligned packet of greater than 80 bytes.

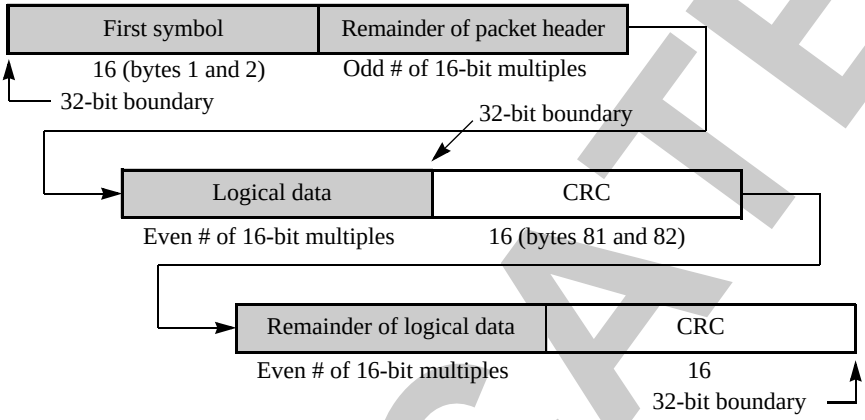


Figure 2-10. Naturally Aligned Packet Bit Stream Example 2

Figure 2-11 is an example of a padded 32-bit aligned packet of less than or equal to 80 bytes.

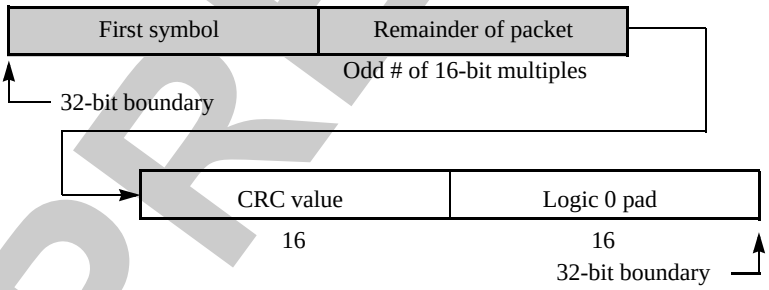


Figure 2-11. Padded Aligned Packet Bit Stream Example 1

Figure 2-12 is an example of a padded 32-bit aligned packet of greater than 80 bytes.

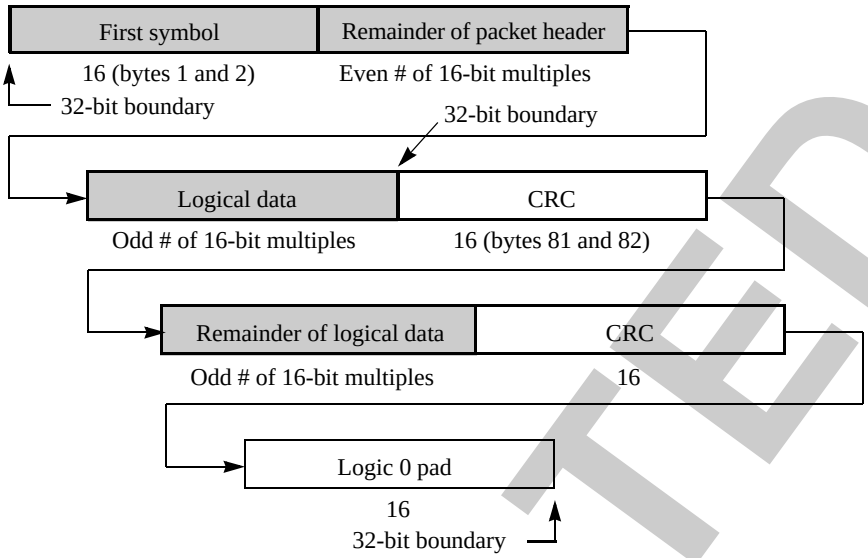


Figure 2-12. Padded Aligned Packet Bit Stream Example 2

2.4.7 CRC Code

The CCITT polynomial $X^{16}+X^{12}+X^5+1$ is a popular CRC code. The initial value of the CRC is 0xFFFF (all logic 1s). For the CRC calculation, the uncovered 6 bits are treated as logic 0s. As an example, a 16-bit wide parallel calculation is described in the equations in Table 2-5. Equivalent implementations of other widths can be employed.

Table 2-5. 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			
C13		x	x			x				x				x		

Table 2-5. Parallel CRC Intermediate Value Equations (Continued)

Check Bit	e00	e01	e02	e03	e04	e05	e06	e07	e08	e09	e10	e11	e12	e13	e14	e15
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 2-13

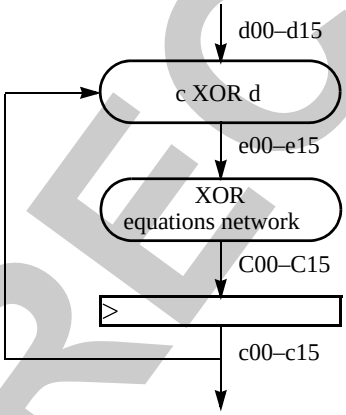


Figure 2-13. CRC Generation Pipeline

2.5 Maximum Packet Size

The maximum packet size permitted by the 8/16 LP-LVDS 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 2-6. 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**		

2.6 Link Maintenance Protocol

To initialize, explore, and recover from errors it is necessary to have a secondary mechanism to communicate between connected system devices. This mechanism is used to establish communications between connected devices (described in Section 3.7.1, “Port and Link Initialization”), attempt automatic error recovery as described above in Section 2.4.5, “Link Behavior Under Error,” and allows software-managed link maintenance operations.

This protocol involves a request and response pair between electrically connected (linked) devices in the system. For software management, the request is generated through ports in the configuration space of the sending device. An external processing element write of a command to the link-request register with a *RapidIO Part 1: Input/Output Logical Specification* maintenance write transaction causes an aligned 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 aligned link-response control symbol if the link-request command required it to do so. The external processing element 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 recovery mechanism relies on the hardware generating link-request control symbols under the transmission error conditions described in Section 2.4.5.1 and using the corresponding link-response information to attempt recovery.

Automatic link initialization also depends upon hardware generation of the appropriate link-requests and link-responses.

2.6.1 Command Descriptions

Table 2-7 contains a summary of the link maintenance commands that use the link maintenance protocol described above. Three link request commands are defined currently. The input-status command generates a paired link-response control symbol; the reset and send-training commands do not.

Table 2-7. Secondary Link Maintenance Command Summary

Command	Description
Reset	Resets the device
Input-status	Returns input port status; functions as a restart-from-error control symbol under error conditions. Generates a paired link-response control symbol.
Send-training	Stops normal operation and transmits 256 training pattern iterations

2.6.1.1 Reset and Safety Lockouts

The reset command causes the receiving device to go through its hard reset or power up sequence. All state machines and the configuration registers reset to the original power on states. The reset 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 command in a link-request control symbol shall not perform the reset function unless it has received four reset commands in a row without any other intervening packets or control symbols, except idle control symbols. This will prevent spurious reset commands 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.

2.6.1.2 Input-status

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. The receiver then responds with a link-response control symbol.

The input-status command is the command 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, this input-status command acts as a restart-from-error control symbol, and the receiver is re-enabled to receive new packets after generating the link-response control symbol. This restart-from-error 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 re-synchronize the sending and receiving devices after the retry.

2.6.1.3 Send-training

The send-training command causes the recipient device to suspend normal operation and begin transmitting a special training pattern. The receiving device transmits a total of 256 iterations of the training pattern followed by at least one idle control symbol and then resumes operation. The usage of this command is described in Section 3.7.1.1, “Sampling Window Alignment.” The send-training command does not generate a link-response control symbol.

2.6.2 Status Descriptions

The input-status request generates two pieces of information that are returned in the link-response:

- link status
- ackID usage

The first type of data is the current operational status of the interface. These status indicators are described in Table 2-8.

Table 2-8. Link Status Indicators

Status Indicator	Description
OK	The port is working properly.
Error	The port has encountered an unrecoverable error and has shut down.
Retry-stopped[1]	The port has been stopped due to a retry.
Error-stopped[1]	The port has been stopped due to a transmission error.

[1]Valid only with the Stopped indicator

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.

The second field returned in the link-response control symbol is state information about the acknowledge identifier usage. The input port returns a value indicating the next ackID expected to be received by the port. The automatic error recovery mechanism uses this information to determine where to begin packet re-transmission after a transmission error condition has been encountered.

Blank page

Chapter 3 Packet and Control Symbol Transmission

3.1 Introduction

This RapidIO chapter defines the *RapidIO Part 4: 8/16 LP-LVDS Physical Layer Specification* packet and control symbol delineation and alignment on the physical port and mechanisms to control the pacing of a packet. Each input and output port is either one or two bytes wide. All 8/16 LP-LVDS defined protocols are used identically for both the 8- and 16-bit wide versions of the physical interface. The only difference is the number of pins used to transmit the packets and aligned control symbols.

3.2 Packet Start and Control Symbol Delineation

The control framing signal used to delineate the start of a packet or a control symbol on the physical port is a no-return-to-zero, or NRZ signal. This frame signal is toggled for the first symbol (see the Note in Section 2.4.2, “Packet Protection”) of each packet and for the first control symbol of each aligned control symbol. Therefore, if a 16-bit symbol contains a RapidIO logical packet format type (the ftype field in the RapidIO logical specifications) or a control symbol (stype) field, the frame signal shall toggle. In order for the receiving processing element to sample the data and frame signals, a data reference signal is supplied that toggles on all possible transitions of the interface pins. This type of data reference signal is also known as a double-data-rate clock. These received clocks on devices with multiple RapidIO ports have no required frequency or phase relationship.

The framing signal is not toggled for other symbols such as those containing remaining packet header and data bytes. However, it is toggled for all idle control symbols between packets. This means that the maximum toggle rate of the control framing signal is every 4 bytes, and the framing signal is only allowed to toggle on every fourth byte. Therefore, the framing signal is aligned to a 32-bit boundary as are all of the packets and aligned control symbols. Additionally, the data reference signal shall transition from low to high on this same boundary. Examples of these constraints are shown in Figure 3-1 and Figure 3-3 for an 8-bit port and Figure 3-2 and Figure 3-4 for a 16-bit port.

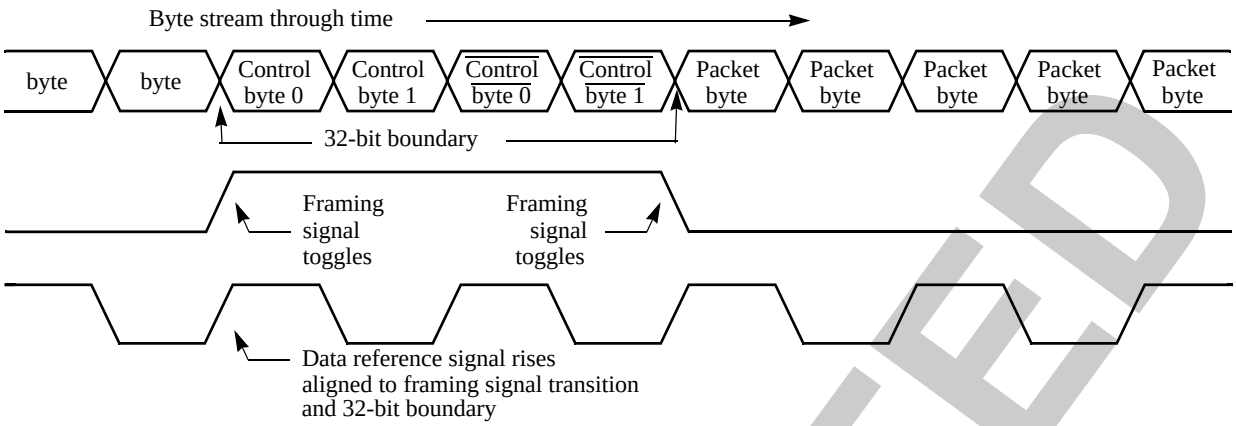


Figure 3-1. Framing Signal Maximum Toggle Rate for 8-bit Port

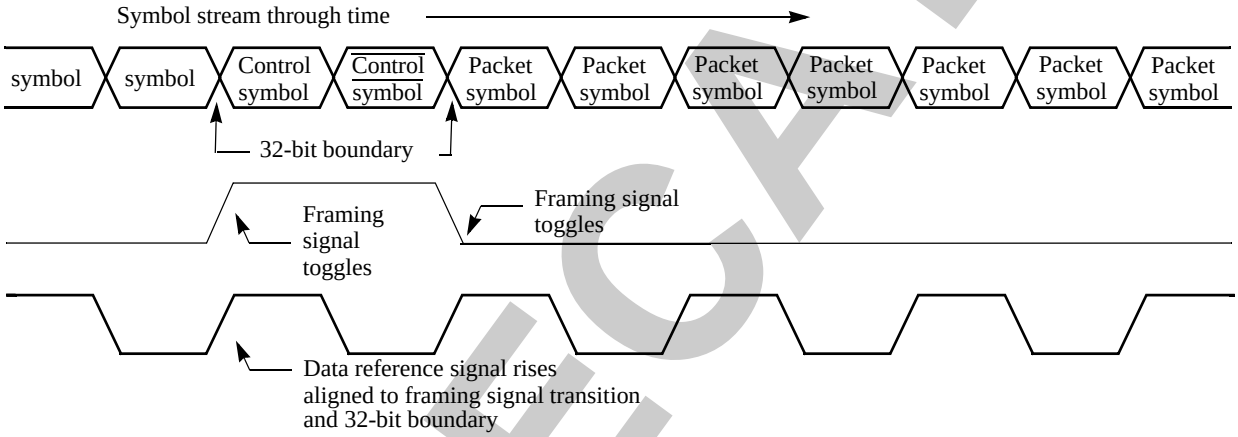


Figure 3-2. Framing Signal Maximum Toggle Rate for 16-bit Port

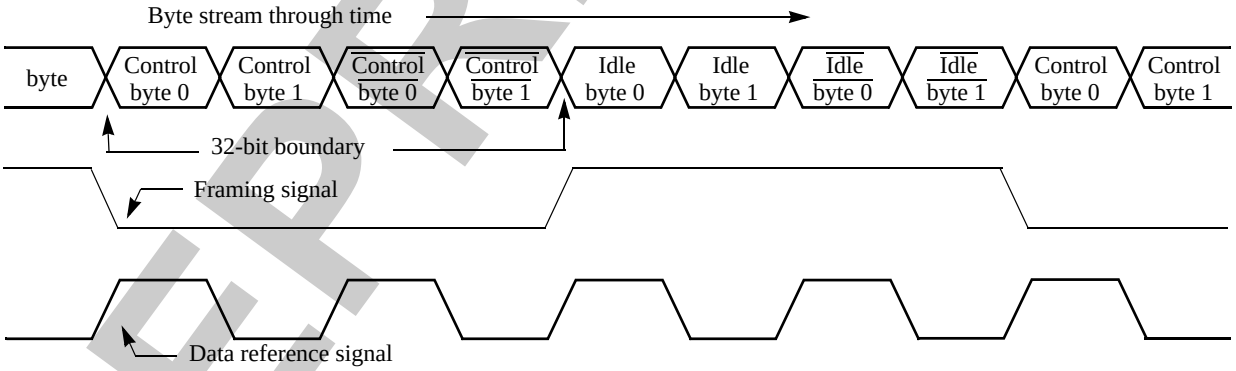


Figure 3-3. Control Symbol Delineation Example for 8-bit Port

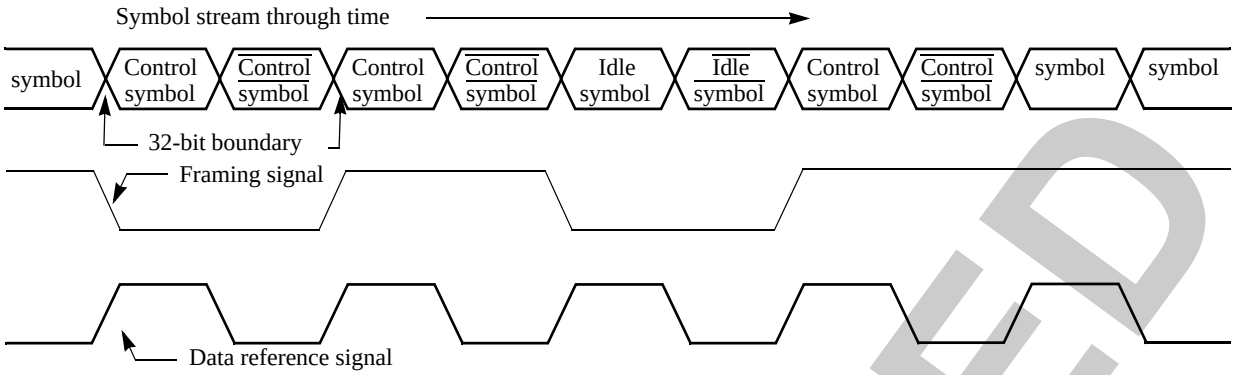


Figure 3-4. Control Symbol Delineation Example for 16-bit Port

Errors on the framing and data reference signals can be detected either directly by verifying that the signals transition only when they are allowed and expected to transition, or indirectly by depending upon detection of packet header or CRC or control symbol corruption, etc. if these signals behave improperly. Either method of error detection on the framing and data reference signals allows error recovery by following the mechanisms described in Section 2.4.5.1, “Recoverable Errors” and Section A.4, “Error Recovery.”

For simplicity, the data reference signal will not be included in any additional figures in this document. It is always rising on the 32-bit boundary when it is legal for the frame signal to toggle as shown in Figure 3-1 through Figure 3-4.

3.3 Packet Termination

A packet is terminated in one of two ways:

- The beginning of a new packet marks the end of a previous packet.
- The end of a packet may be marked with one of the following: an aligned end-of-packet (eop), restart-from-retry, link-request, or stomp control symbol.

The stomp control symbol is used if a transmitting processing element detects a problem with the transmission of a packet. It may choose to cancel the packet by sending the stomp control symbol instead of terminating it in a different, possibly system fatal, fashion like corrupting the CRC value.

The restart-from-retry control symbol can cancel the current packet as well as be transmitted on an idle link. This control symbol is used to enable the receiver to start accepting packets after the receiver has retried a packet.

The link-request control symbol can cancel the current packet as well as be transmitted on an idle link and has several applications. It can be used by software for system observation and maintenance, and it can be used by software or hardware to enable the receiver to start accepting packets after the receiver has refused a packet due to a transmission error as described in Section 2.4, “Error Detection and

Recovery."

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.

If a port receiving a canceled packet has not previously acknowledged the packet and is not in an "Input Stopped" stopped state (Retry-Stopped or Error-Stopped), the port shall immediately enter the Input Retry-stopped state and follow the Packet Retry mechanism specified in Section 2.3.3, "Transaction and Packet Delivery", if the packet was canceled with a control symbol other than a restart-from-retry or a link-request/input-status control symbol. As part of the Packet Retry mechanism, the port sends a packet-retry control symbol to the sending port indicating that the canceled packet was not accepted.

Figure 3-5 is an example of a new packet marking the end of a packet.

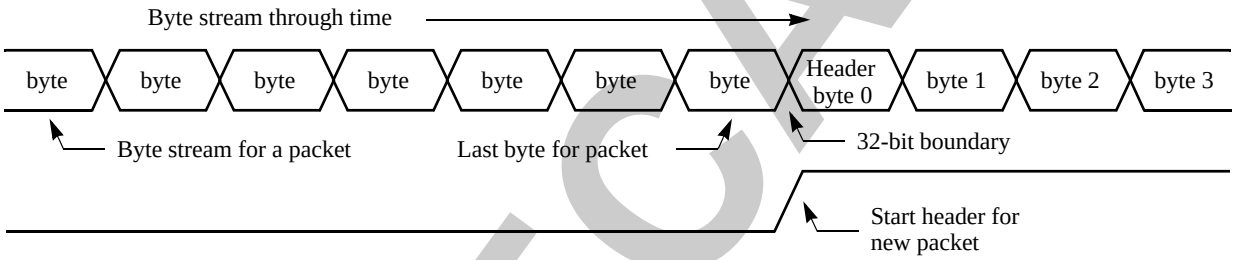


Figure 3-5. Header Marked End of Packet (8-bit Port)

Figure 3-6 is an example of an aligned end-of-packet control symbol marking the end of a packet. The stomp, link-request, and restart-from-retry control symbol cases look similar.

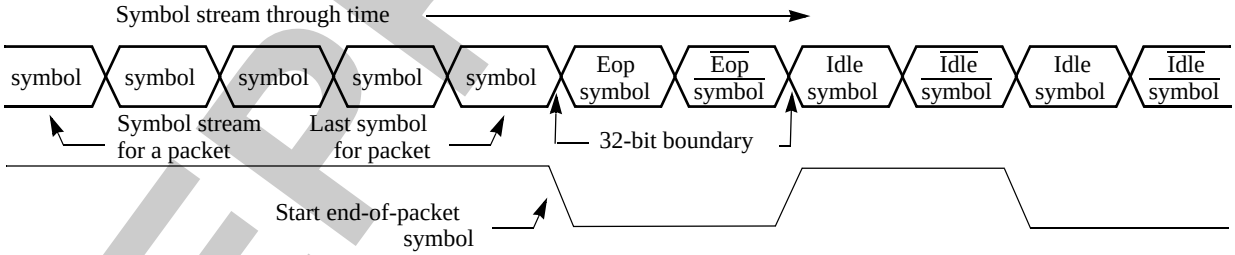


Figure 3-6. End-Of-Packet Control Symbol Marked End of Packet (16-bit Port)

3.4 Packet Pacing

If a device cannot transmit a packet as a contiguous stream of control symbols, it may force wait states by inserting idle control symbols called pacing idles. As with the other control symbols, the pacing idle control symbols are always followed by a bit-wise inverted copy and are then called aligned pacing idle control symbols. Any number of aligned pacing idle control symbols can be inserted, up to some

implementation defined limit, at which point the sender should instead send a stomp control symbol and cancel the packet in order to attempt to transmit a different packet. Figure 3-7 shows an example of packet pacing. These idle control symbols are ignored by the receiving device, and more data is sent when it becomes available. Pacing idle control symbols can be embedded anywhere in a packet where they can be legally delineated.

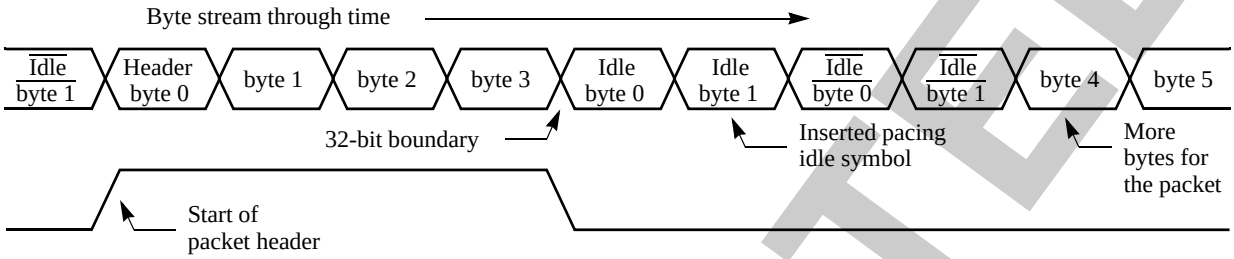


Figure 3-7. Pacing Idle Insertion in Packet (8-bit Port)

The receiver of a packet may request that the sender insert pacing idle control symbols on its behalf by sending a throttle control symbol specifying the number of aligned pacing idle control symbols to delay. The packet sender then inserts that number of aligned pacing idles into the packet stream. If additional delay is needed, the receiver can send another throttle control symbol.

If the receiver requests too many aligned pacing idles indicating an excessive delay, determined by some implementation defined limit, it should terminate the packet transmission by issuing a packet-retry acknowledge control symbol. Alternatively, the sender may issue a stomp control symbol to cancel the packet if too many aligned pacing idle control symbols are requested by the receiver. The throttle control symbol shall be honored because it is used to force insertion of idle control symbols for clock re-synchronization in the receiver as described in Chapter 6, “System Clocking Considerations.”

The maximum allowed response time from the receipt of the last byte of an aligned throttle control symbol at the input pins to the appearance of the first byte of an aligned pacing idle control symbol on the output pins is 40 interface clocks (80 data ticks).

Note that for CRC values for a packet, the aligned pacing idle control symbols are not included in the calculation.

3.5 Embedded Control Symbols

Control symbols can be embedded anywhere in a packet in the same fashion as pacing idle control symbols, as long as all delineation and alignment rules are followed.

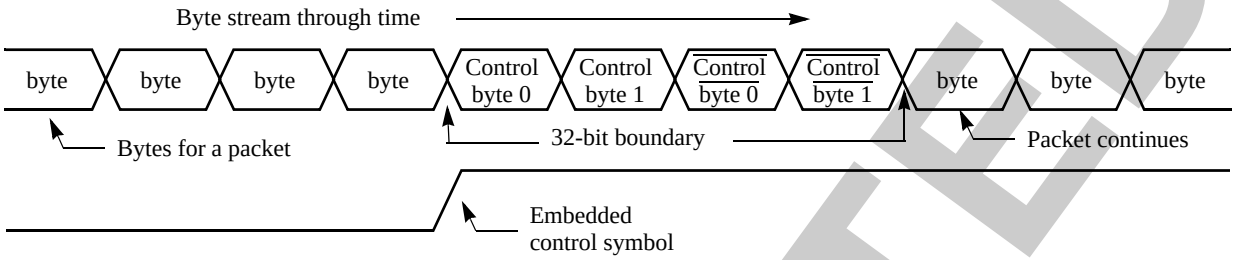


Figure 3-8. Embedded Control Symbols for 8-bit Port

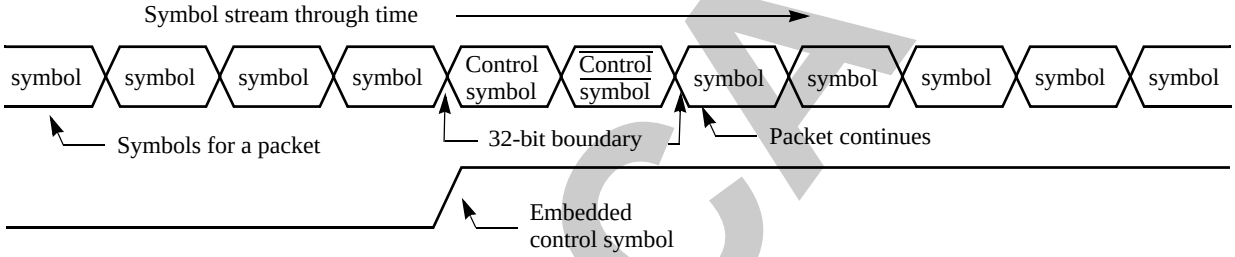


Figure 3-9. Embedded Control Symbols for 16-bit Port

As with the pacing idle control symbols, the embedded aligned control symbols are not included in the CRC value calculation for the packet.

A special error case exists when a corrupt embedded control symbol is detected. In this case a packet-not-accepted control symbol shall be generated and the embedding packet is discarded.

3.6 Packet to Port Alignment

This section shows examples of packet transmission over the 8-bit and 16-bit interfaces. The corresponding control symbol alignment is shown in Section 4.7, “Control Symbol to Port Alignment.”

Figure 3-10 shows the byte transmission ordering on a port through time using a small transport format ftype 2 packet from the *RapidIO Part 1: Input/Output Logical Specification* and *RapidIO Part 3: Common Transport Specification*. Note that for this example the two bytes following the CRC would indicate some form of packet termination such as a new packet or an eop.

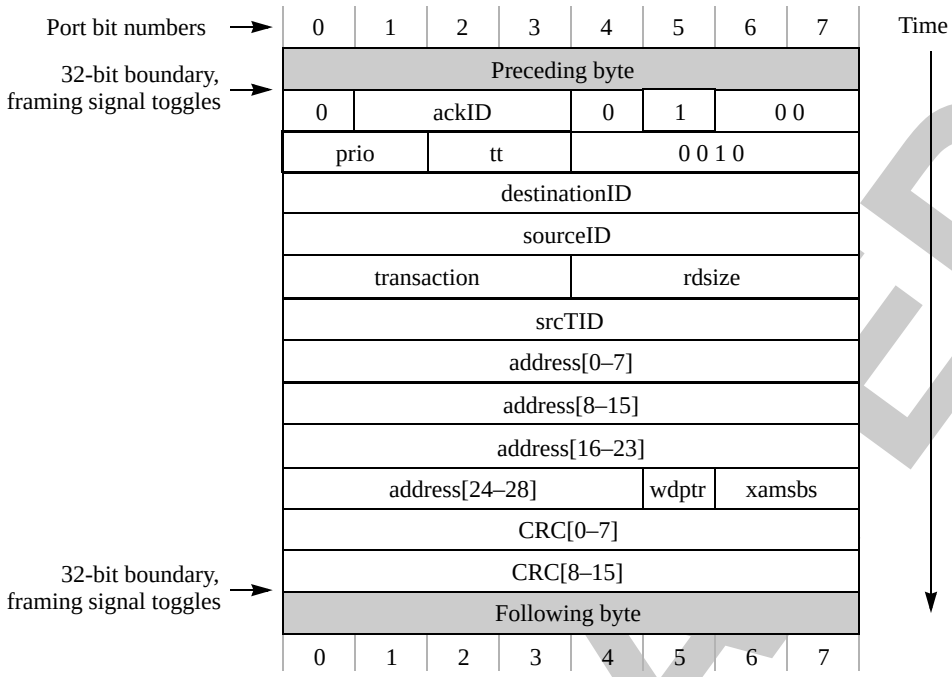


Figure 3-10. Request Packet Transmission Example 1

Figure 3-11 shows the same packet transmitted over a 16-bit port.

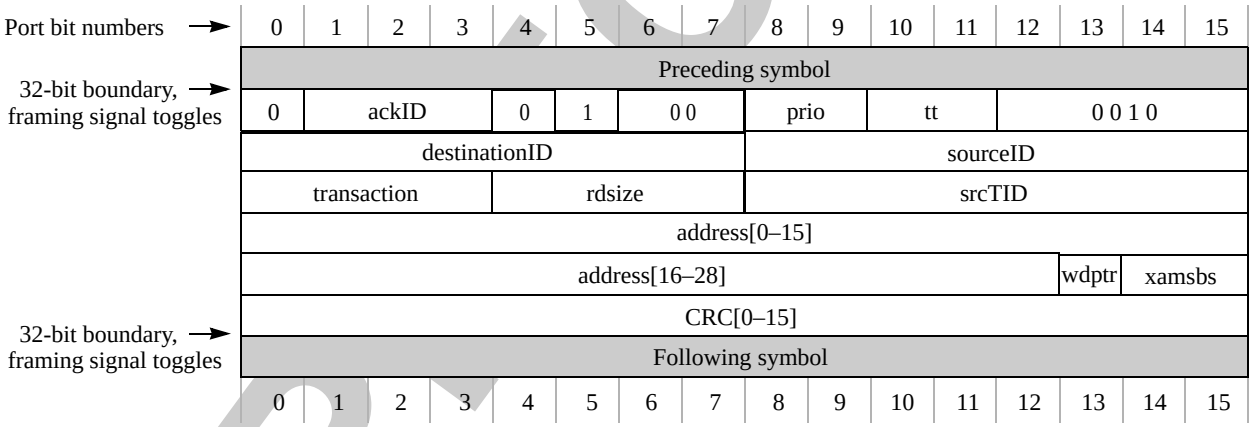


Figure 3-11. Request Packet Transmission Example 2

Figure 3-12 shows the same example again but with the large transport format over the 8-bit port. Note that for this example the two bytes following the CRC of the packet are all logic 0 pads.

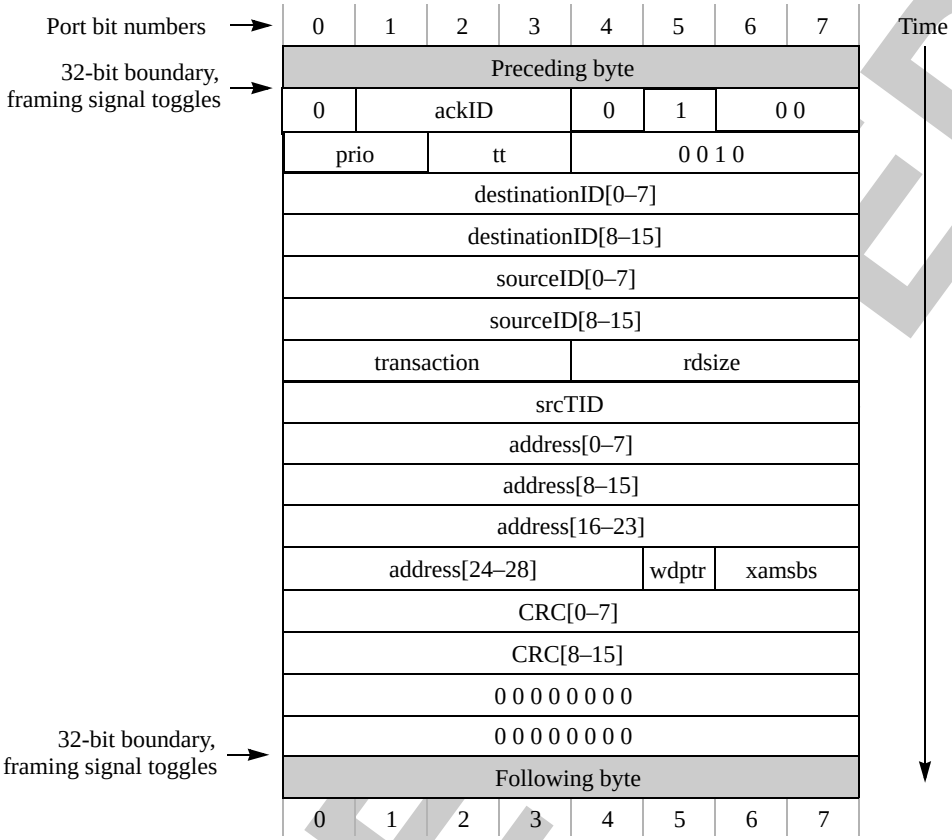


Figure 3-12. Request Packet Transmission Example 3

Figure 3-13 is the same packet as for Figure 3-12 but over the 16-bit port.

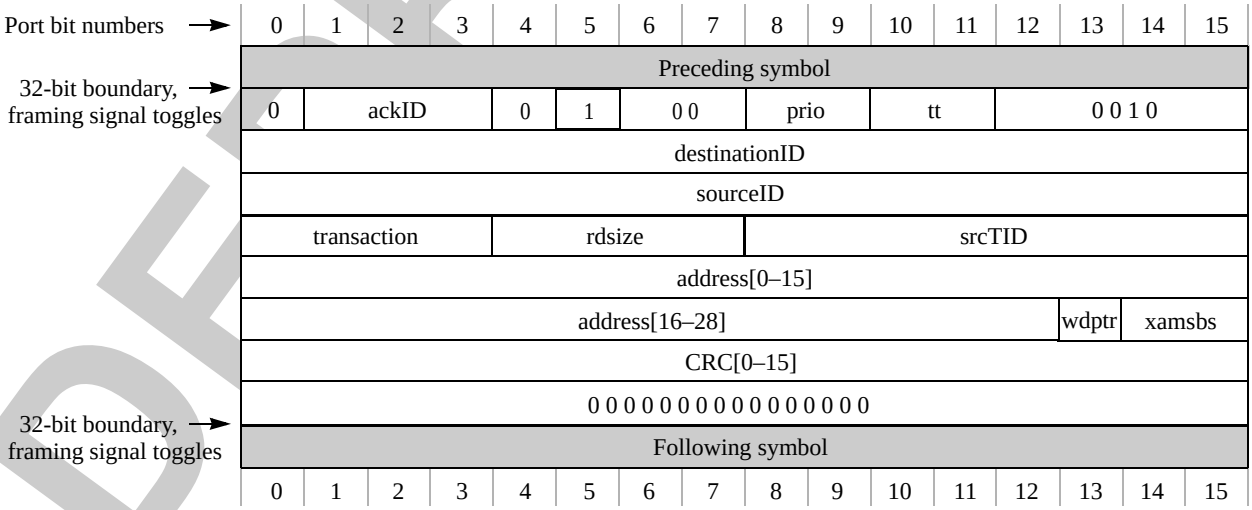


Figure 3-13. Request Packet Transmission Example 4

Figure 3-14 and Figure 3-15 show the ftype 13 response packet for request example—the small transport format packet. Note that the two bytes following the packet CRC may be logic 0 pads depending on the size of the packet.

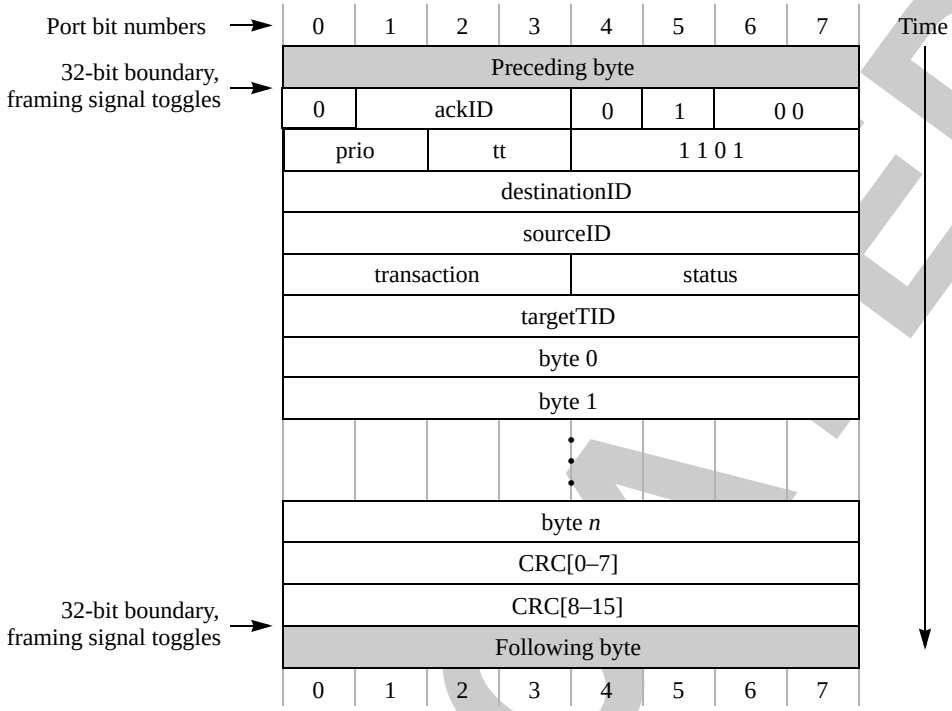


Figure 3-14. Response Packet Transmission Example 1

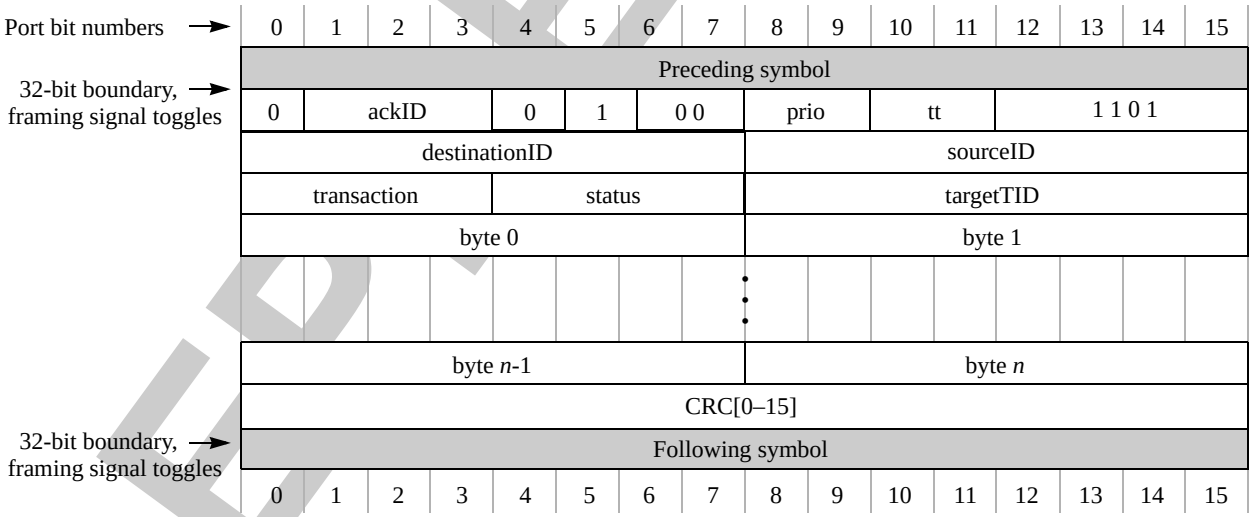


Figure 3-15. Response Packet Transmission Example 2

3.7 System Maintenance

A necessary part of any system are methods for initializing, configuring, and maintaining the system during operation.

3.7.1 Port and Link Initialization

Before an LP-LVDS port can exchange packets with the port to which it is connected, the port may require initialization and the link connecting the two ports must be initialized.

Many ports, especially those operating at higher data rates, must adjust the timing of when they sample input data from the link in order to achieve an acceptable or optimal received bit error rate. This process is called training and is executed while the port is receiving a special training pattern. In addition, each 16-bit wide port must decide whether to operate in 8-bit or 16-bit unless it has been statically configured for 8-bit or 16-mode.

Link initialization requires that each port must receive at least one idle control symbol from the connected port before beginning normal operation.

The state of a port is indicated by bits 30 (Port OK) and 31 (Port Uninitialized) of the associated Port *n* Error and Status Register. Bit 31 is set when the port is in the Uninitialized state and cleared when the port initialization process is completed and the port is initialized. Bit 30 is set when the port has received an idle control symbol from the connected port and the port is in the normal operation mode and is cleared if the port or the connected port is uninitialized or if there is no connected port (the port is not receiving valid input signals).

3.7.1.1 Sampling Window Alignment

Any port whose link receiver input sample timing is not fixed and any 16-bit port whose width mode is not statically configured require port initialization.

3.7.1.1.1 Port Width Mode Selection

All 16-bit LP-LVDS ports shall be capable of operating in both 8-bit and 16-bit modes.The width mode in which a 16-bit port operates may be either statically or dynamically configured. If the width mode of a port is statically configured, the port will operate correctly only if connected to a port operating in the same width mode. If the width mode of a 16-bit port is not statically configured, the width mode is determined as part of the port initialization process.

When operating in 8-bit mode, only the signal pairs CLK0/$\overline{\text{CLK0}}$, FRAME/$\overline{\text{FRAME}}$ and D[0-7]/$\overline{\text{D[0-7]}}$ shall be used. The 16-bit mode output signal pairs TCLK1/$\overline{\text{TCLK1}}$, and TD[8-15]/$\overline{\text{TD[8-15]}}$ may be driven as outputs, but the input signal pairs RCLK1/$\overline{\text{RCLK1}}$ and RD[8-15]/$\overline{\text{RD[8-16]}}$ shall be ignored as inputs.

Dynamic port width selection shall be based on the presence of valid signals at the

inputs of the CLK0, CLK1, FRAME D[0-7] and D[8-15] receivers. If valid signals are present at the inputs of the CLK0, CLK1, FRAME D[0-7] and D[8-15] receivers, the port shall operate in 16-bit mode. If valid signals are present at the inputs of the CLK0, FRAME and D[0-7] receivers, but not at the inputs of the CLK1 and D[8-15] receivers, the port shall operate in 8-bit mode. If valid signals are not present at the inputs of the CLK0, FRAME and D[0-7] receivers, the width mode is undefined and the port shall not exit Uninitialized state.

3.7.1.1.2 Input Sampling Window Alignment

Input sampling window alignment is the process in which a port adjusts the timing of when input data is sample by the link receiver. The timing is adjusted to achieve an acceptable or optimal received bit error rate. The process is also called “training”. When the process is successfully completed, the port is said to be “aligned” or “trained”. The process or algorithm used by a port to align the input sampling window is not specified.

Sampling window alignment is done while the port is receiving a special data pattern called the training pattern. A special data pattern is required to ensure that enough transition timing information is available to the receiver to correctly adjust the input sample timing and to ensure that bytes transmitted by a port operating in 8-bit or that half-words transmitted by a port operating in 16-bit are correctly recovered by the link receiver.

There are two types of training, initialization training and maintenance training. Initialization training is used when a device powers up or is reset or when a port loses input sampling window alignment due to events such as excessive system noise or power fluctuations. Maintenance training is used when a port is nominally input sampling window aligned, but in need of some “fine-tuning” of the input sampling window timing to maintain an acceptable or optimum received bit error rate.

3.7.1.1.3 Training Pattern

The training pattern shall be the bit sequence 0b11110000. The training pattern shall be transmitted left to right with the left most bit transmitted first and the right most bit transmitted last.

When transmitted, the training pattern shall be transmitted simultaneously on all of the data signals, D[0-7] for an 8-bit port or a 16-bit port statically configured to operate in 8-bit mode, D[0-15] for a 16-port not statically configured to operate in 8-bit mode, and the training pattern or is complement shall be transmitted on the FRAME signal. The training pattern or its complement is selected for transmission on the FRAME signal such that the FRAME signal shall toggle at the beginning of training pattern transmission. The training pattern shall never be transmitted on a CLK signal. The training pattern shall never be transmitted on signals D[8-15] of a 16-bit port if the port is statically configured to operate in 8-bit mode.

The result of these rules is that during training pattern transmission, FRAME and

data signals transmitted by the port have the following properties.

- The FRAME signal toggles at the beginning of training pattern transmission. (Individual data bits may or may not toggle at the beginning of training pattern transmission depending on their value during the bit time immediately preceding the training pattern.)
- After the first bit of the training pattern, FRAME and all data bits all toggle at the same nominal time.
- Each byte transmitted by a port transmitting in 8-bit mode is either all ones, 0xFF, or all zeros, 0x00.
- Each half-word transmitted by a port transmitting in 16-bit mode is either all ones, 0xFFFF, or all zeros, 0x0000.

The reception of the training pattern by an initialized port is readily identified by looking at RD[0-7] when FRAME toggles. If the received value of RD[0-7] is either all ones, 0xFF, or all zeros, 0x00, the training pattern is being received.

3.7.1.1.4 Training Pattern Transmission

When transmitted, the training pattern shall be transmitted in bursts. Each burst shall contain 256 repetitions of the training pattern. Each burst shall be followed by either a link-request/send-training or an idle control symbol.

The training pattern shall be transmitted by an initialized port only at request of the connected port. The link-request/send-training control symbol is used to request that the connected port transmit the training pattern. A port that is not initialized and therefore unable to reliably receive control symbols assumes that the connected port is sending link-request/send-training control symbols and therefore continuously transmits training sequence bursts with each burst followed by a link-request/send-training control symbol as specified by the Port Initialization Process.

The training pattern shall neither be embedded in a packet nor used to terminate a packet.

3.7.1.1.5 Ports Not Requiring Port Initialization

Similarly, a 16-bit port with fixed input sampling window timing and whose width mode is statically configured does not require port initialization. On device power-up and on device reset, such ports shall enter and remain in the Initialized state and shall never be in the Uninitialized state. Such ports shall transmit idle control symbols until an idle control symbol is received from the connected port. Upon the reception of an idle control symbol from the connected port, the port shall transmit an idle control symbol, set the “port OK” bit in its Port *n* Control and Status Register, enter the normal operation state and may then begin the transmission of packets and non-idle control symbols.

If while waiting to receive an idle control symbol from the connected port, the

reception by the port of a link-request/send-training control symbol from the connected port immediately followed by the training pattern indicates that the connected port in not initialized. When this occurs, the port shall stop sending idle control symbols and repeatedly send training pattern bursts, each burst followed by an idle control symbol, until an idle control symbol is received from the connected port indicating that the connected port is now initialized. Upon receiving an idle control symbol from the connected port, the port shall complete transmission of the current training pattern burst, transmit an idle control symbol, set the "port OK" bit in its Port *n* Control and Status Register and enter the normal operation state. The port may then transmit packets and non-idle control symbols.

3.7.1.1.6 Ports Requiring Port Initialization

Ports that do not have fixed input sampling window timing and 16-bit ports whose width mode is not statically configured require port initialization. Such ports shall enter the Uninitialized state on device power-up and device reset. Such a port shall also enter the Uninitialized state if the port loses correct input sampling window timing due to events such as excessive system noise or power fluctuations. The algorithm used to determine when a port has lost input sample window alignment is not specified. A port in the Uninitialized state shall execute the Port Initialization Process to exit the Uninitialized state.

The output signals of a LP-LVDS port may be erratic when the device containing the port is powering up or being reset. For example, the output drivers may be temporarily disabled, the signals may have erratic HIGH or LOW times and/or the clock signals may stop toggling. A LP-LVDS port must be tolerant of such behavior and shall properly initialize after the signals from the connected port return to normal and comply with the LP-LVDS electrical specifications.

3.7.1.1.7 Port Initialization Process

Upon entering the Uninitialized state, a port shall execute the following Port Initialization Process.

- The port sets the "Port Uninitialized" bit and clears the "Port OK" bit in its Port *n* Control and Status Register.
- The port transmits a link-request/send-training control symbol followed by one or more bursts of the training sequence. The port continuously transmits training pattern bursts, each followed by a link-request/send-training or idle control symbol, until the port has achieved input sample timing alignment and has received an idle control symbol from the connected port.
- The port attempts to detect a valid clock signal on its CLK0 input and, if present, on its CLK1 input and to detect the training pattern on its FRAME and D[0-7] inputs and, if present, on its D[8-15] inputs.
- Once valid input signals are detected, a 16-bit ports whose width mode is not statically configured determines the width of the connected port and selects the matching width mode.

- Once the width mode of the port is established, either statically or dynamically, the port attempts to achieve input sampling window timing alignment. While attempting to achieve input sampling window timing alignment, the port shall transmit a link-request/send-training control symbol after each training pattern burst.
- When the port achieves input sampling window timing alignment, it clears the "Port Uninitialized" bit in the Port *n* Control and Status Register and transmits an idle control symbol after each training pattern burst instead of a link-request/send-training control symbol. This indicates to the connected port that the port has completed input sampling window alignment.
- Upon receiving an idle control symbol from the connected port, indicating that the connected port has completed input sampling window alignment, the port completes transmitting the current training pattern burst, sends an idle control symbol, sets the "Port OK" bit in the Port *n* Control and Status Register and enters normal operation.

3.7.1.2 Link Initialization

After a port is in the Initialized state, the port shall not begin transmission of packets and control symbols other than the idle control until it has received an idle control symbol from the connected port. The reception of an idle control symbol indicates that the connected port is in the Initialized state and is ready to receive packets and non-idle control symbols. When both ports connected by a link have received an idle control symbol from the connected port, the link is initialized.

3.7.1.3 Maintenance Training

Depending upon their implementation, some ports may require occasional adjustment of their input sampling window timing while in the Initialized state to maintain an optimal received bit error rate. Such adjustment is called maintenance training. A port requiring maintenance training shall do the following.

- The port shall transmit a single link-request/send-training control symbol and then resume normal transmit operation.
- If the port is not able to complete maintenance training with one burst of the training pattern, the port may transmit additional link-request/send-training control symbols and shall resume normal transmit operation after transmitting each link-request/send-training control symbol.

A port requiring maintenance training shall not transmit the training pattern after transmitting a link-request/send-training control symbol. (The transmission by a port of a link-request/send-training control symbol followed by the training pattern indicates that the port has become uninitialized.)

A port receiving a link-request/send-training control symbol that is not followed by the training pattern shall end the transmission of packets and control symbols as quickly as possible without violating the link protocol, transmit one burst of the

training pattern followed by an idle control symbol and then resume normal operation.

3.7.1.4 Unexpected Training Pattern Reception

At any time, the reception by an initialized port of unsolicited training pattern, whether or not preceded by a link-request/sent-training control symbol, indicates that the connected port is in the Uninitialized state. When this occurs, the port receiving the unsolicited training pattern shall repeatedly transmit training pattern bursts, each burst followed by an idle control symbol, until an idle control symbol is received from the connected port indicating that the connected port is now initialized. Upon receiving an idle control symbol from the connected port, the port shall complete transmission of the current training pattern burst, transmit an idle control symbol, set the "port OK" bit in its Port *n* Control and Status Register and enter the normal operation state. The port may then transmit packets and non-idle control symbols.

Once a link has been initialized, the reception of unsolicited training pattern is a protocol violation. It indicates that the sending port has lost input sampling window alignment and has most likely not received some previously sent packets and control symbols. Once the link has been initialized, a port receiving an unsolicited training pattern shall enter the output Error-stopped state. The port shall execute the Output Error-stopped recovery process specified in Section 2.4.5.1.2, "Control Symbol Errors" once communication with the connected port has been re-established.

3.7.2 Multicast-Event

The Multicast-Event control symbol provides a mechanism through which notice that some system defined event has occurred, can be selectively multicast throughout the system. Refer to Section 4.3 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).

3.8 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 in the device's local configuration registers.

Chapter 4 Control Symbol Formats

4.1 Introduction

This chapter defines the *RapidIO Part 4: 8/16 LP-LVDS Physical Layer Specification* control symbols described in Chapter 2, "Physical Layer Protocol." Note that the S bit defined in Section 2.3.1 is always set to logic 1 and the $\overline{S}$ bit (also defined in Section 2.3.1) is always set to logic 0 for the physical layer control symbols. All control symbols are aligned to 32 bits with the last 16 bits as a bit-wise inverse of the first 16. 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.

4.2 Acknowledgment Control Symbol Formats

An acknowledgment control symbol is a transmission status indicator issued by a processing element when it has received a packet from another processing element to which it is electrically connected. Acknowledgment control symbols are used for flow control and resource de-allocation between adjacent devices. The following are the different acknowledgment control symbols that can be transmitted back to sending elements from receiving elements:

- Packet-accepted
- Packet-retry
- Packet-not-accepted

Because receipt of an acknowledgment control symbol does not imply the end of a packet, a control symbol can be embedded in a packet, as well as sent when the interconnect is idle. Embedded control symbols are discussed in Section 3.5, "Embedded Control Symbols."

Field definitions for the acknowledgment control symbols are shown in Table 4-1.

Table 4-1. Field Definitions for Acknowledgment Control Symbols

Field	Definition
packet_ackID	Acknowledgment ID is the packet identifier for acknowledgments back to the request or response packet sender.
buf_status	buf_status field indicates the number of maximally sized packets that can be received, described in Section 2.3.1
cause	cause field indicates the type of error encountered by an input port, defined in Table 4-2

4.2.1 Packet-Accepted Control Symbol

The packet-accepted acknowledgment control symbol indicates that the adjacent device in the interconnect fabric 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. This control symbol format is displayed in Figure 4-1.

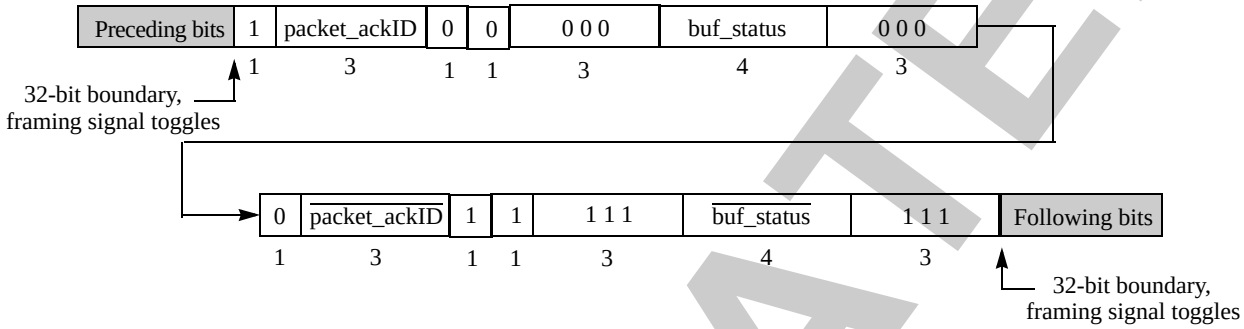


Figure 4-1. Type 0 Packet-Accepted Control Symbol Format

4.2.2 Packet-Retry Control Symbol

A packet-retry acknowledgment control symbol indicates that the adjacent device in the interconnect fabric was not able to accept the packet due to some temporary resource conflict such as insufficient buffering and the source should retransmit the packet. This control symbol can be generated at any time after the start of a packet, which allows the sender to cancel the packet and try sending a packet with a different priority or destination. This will avoid wasting bandwidth by transmitting all of the rejected packet. This control symbol format is displayed in Figure 4-2.

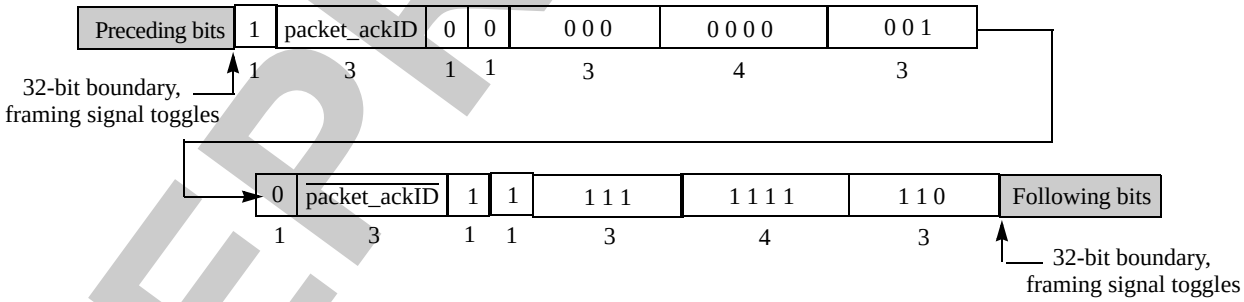


Figure 4-2. Type 1 Packet-Retry Control Symbol Format

4.2.3 Packet-Not-Accepted Control Symbol

A packet-not-accepted acknowledgment control symbol means that the receiving device could not accept the packet due to an error condition, and that the source should retransmit the packet. This control symbol can be generated at any time after the start of a packet, which allows the sender to cancel the packet and try sending a

packet with a different priority or destination. Generating this control symbol at any point in packet transmission avoids wasting bandwidth by transmitting all of the rejected packet. The packet-not-accepted control symbol contains a field describing the cause of the error condition, shown in Table 4-2. If the receiving device is not able to specify the cause for some reason, or the cause is not one of defined options, the general error encoding shall be used. This control symbol format is displayed in Figure 4-3.

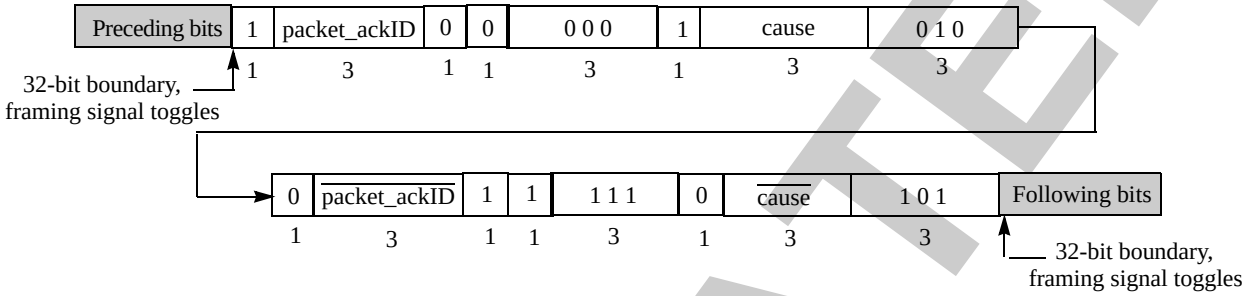


Figure 4-3. Type 2 Packet-Not-Accepted Control Symbol Format

The cause field shall be used to display informational fields useful for debug. Table 4-2 displays the reasons a packet may not be accepted, indicated by the cause field.

Table 4-2. cause Field Definition

Encoding	Definition
0b000	Encountered internal error
0b001	Received unexpected ackID on packet
0b010	Received error on control symbol
0b011	Non-maintenance packet reception is stopped
0b100	Received bad CRC on packet
0b101	Received S bit parity error on packet/control symbol
0b110	Reserved
0b111	General error

4.2.4 Canceling Packets

A packet-retry or packet-not-accepted acknowledgment control symbol that is received for a packet that is still being transmitted may result with the sender canceling the packet.

The sending device can use the stomp (see Chapter 3, “Packet and Control Symbol Transmission”), restart-from-retry (in response to a packet-retry control symbol), or link-request (in response to a packet-not-accepted control symbol) control symbol to cancel the packet. Because the receiver has already rejected the packet, it will not detect any induced error. Alternatively, the sending device can choose to complete transmission of the packet normally.

4.3 Packet Control Symbol Formats

Packet control symbols are used for packet delineation, transmission, pacing, and other link interface control functions as described in Chapter 3, “Packet and Control Symbol Transmission.”

The packet control symbols are the throttle, stomp, restart-from-retry control symbols, idle, end-of-packet (eop), and multicast-event control symbols, which are specified in the sub_type field of the type 4 control symbol format. The packet control symbols also have a contents field, which has a different meaning depending upon the particular control symbol. Of these control symbols, all control symbols that are not defined as terminating a packet may be embedded within a packet.

This control symbol format is displayed in Figure 4-4.

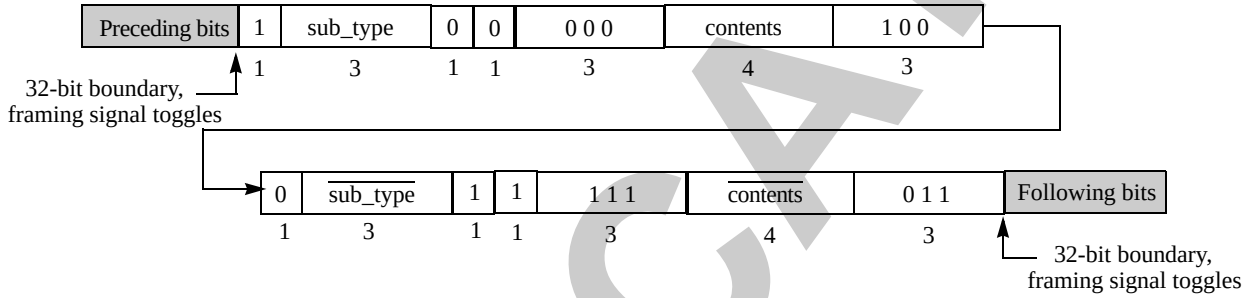


Figure 4-4. Type 4 Packet Control Symbol Format

Table 4-3 shows how sub_type values function with values of the contents field. For the idle, eop, and multicast-event control symbols the contents field is used as the buf_status field described in Section 2.3.1, whose encodings are specified in Table 2-2. For a throttle control symbol, the contents field specifies the number of aligned pacing idle control symbols that the sender should insert in the packet. One of the specified encodings indicates to the sender that it can immediately begin to resume packet transmission, as can be seen in Table 4-4. For the stomp and restart-from-retry control symbols, the contents field is unused and shall be tied to all logic 0s and ignored by the receiving device.

Table 4-3. sub_type and contents Field Definitions

sub_type Field Definition	sub_type Encoding	contents Field Definition
idle	0b000	Used as a buf_status field that indicates the number of maximum-sized packets that can be received. Described in Section 2.3.1, encodings are defined in Table 2-2.
stomp	0b001	Unused, contents=0b0000
eop	0b010	Used as a buf_status field that indicates the number of maximum-sized packets that can be received. Described in Section 2.3.1, encodings are defined in Table 2-2.
restart-from-retry	0b011	Unused, contents=0b0000

Table 4-3. sub_type and contents Field Definitions (Continued)

sub_type Field Definition	sub_type Encoding	contents Field Definition
throttle	0b100	Specifies the number of aligned pacing idles that the sender inserts in a packet. The encodings are defined in Table 4-4.
Multicast-event	0b101	Used as a buf_status field that indicates the number of maximally sized packets that can be received. Described in Section 2.3.1, encodings are defined in Table 2-2.
Reserved	0b110-111	

The pacing idle count content field for a throttle control symbol is defined in Table 4-4.

Table 4-4. Throttle Control Symbol contents Field Definition

Encoding	Definition
0b0000	1 aligned pacing idle control symbol
0b0001	2 aligned pacing idle control symbols
0b0010	4 aligned pacing idle control symbols
0b0011	8 aligned pacing idle control symbols
0b0100	16 aligned pacing idle control symbols
0b0101	32 aligned pacing idle control symbols
0b0110	64 aligned pacing idle control symbols
0b0111	128 aligned pacing idle control symbols
0b1000	256 aligned pacing idle control symbols
0b1001	512 aligned pacing idle control symbols
0b1010	1024 aligned pacing idle control symbols
0b1011-1101	Reserved
0b1110	1 aligned pacing idle control symbol for oscillator drift compensation
0b1111	Stop transmitting pacing idles, can immediately resume packet transmission

4.4 Link Maintenance Control Symbol Formats

Maintenance of a link is controlled by link-request/link-response control symbol pairs as described in the link maintenance protocol of Section 2.6. Each of the control symbols is described below:

- A link-request control symbol issues a command to or requests status from the device that is electrically connected, or linked, to the issuing device. The link-request control symbol is followed by a complemented version of itself as with the other control symbols. 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 restart-from-error control symbol as described in Section 2.4.5.1, “Recoverable Errors.” This control symbol format is displayed in Figure 4-5.

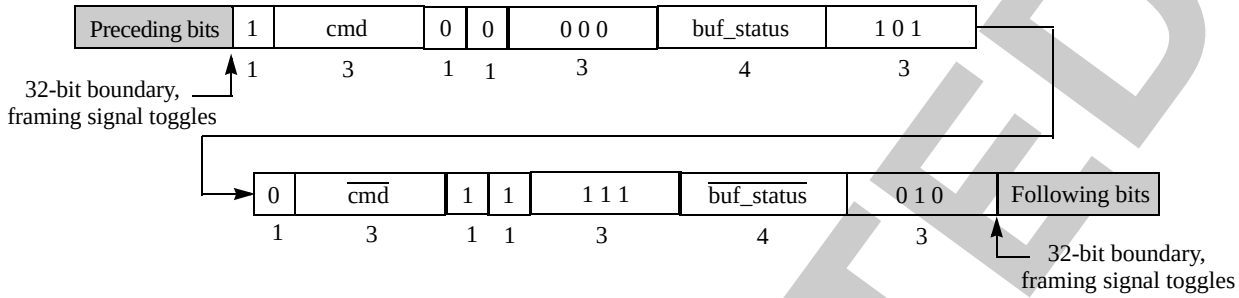


Figure 4-5. Type 5 Link-Request Control Symbol Format

The cmd, or command, field of the link-request control symbol format is defined in Table 4-5.

Table 4-5. cmd Field Definition

cmd Encoding	Command Name	Description
0b000	Send-training	Send 256 iterations of the training pattern
0b001-010		Reserved
0b011	Reset	Reset the receiving device
0b100	Input-status	Return input port status; functions as a restart-from-error control symbol under error conditions
0b101-111		Reserved

- 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 2.6. The link-response control symbol is the same as all other control symbols in that the second 16 bits are a bit-wise inversion of the first 16 bits. A link-response control symbol can be embedded in a packet. This control symbol format is displayed in Figure 4-6.

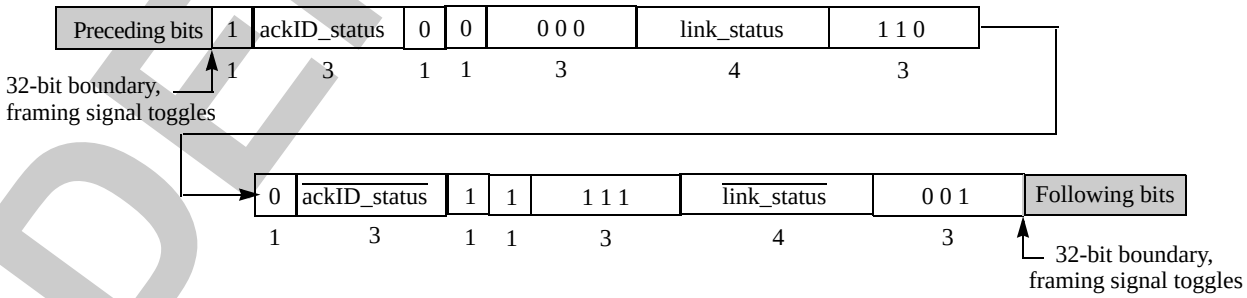


Figure 4-6. Type 6 Link-Response Control Symbol Format

The ackID_status field of the link-response format is defined in Table 4-6.

Table 4-6. ackID_status Field Definition

Encoding	Description
0b000	Expecting ackID 0
0b001	Expecting ackID 1
0b010	Expecting ackID 2
0b011	Expecting ackID 3
0b100	Expecting ackID 4
0b101	Expecting ackID 5
0b110	Expecting ackID 6
0b111	Expecting ackID 7

The link_status field is defined in Table 4-7. Note that the ackID information is included in both fields for additional error coverage if the receiver is working properly (encodings 8-15).

Table 4-7. link_status Field Definition

link_status Encoding	Port Status	Description
0b0000 - 0b0001	Reserved	
0b0010	Error	Unrecoverable error encountered.
0b0011	Reserved	
0b0100	Retry-stopped	The port has been stopped due to a retry.
0b0101	Error-stopped	The port has been stopped due to a transmission error; this state is cleared after the link-request/input-status command is completed.
0b0110 - 0b0111	Reserved	
0b1000	OK, ackID0	Working properly, expecting ackID 0.
0b1001	OK, ackID1	Working properly, expecting ackID 1.
0b1010	OK, ackID2	Working properly, expecting ackID 2.
0b1011	OK, ackID3	Working properly, expecting ackID 3.
0b1100	OK, ackID4	Working properly, expecting ackID 4.
0b1101	OK, ackID5	Working properly, expecting ackID 5.
0b1110	OK, ackID6	Working properly, expecting ackID 6.
0b1111	OK, ackID7	Working properly, expecting ackID 7.

4.5 Reserved Symbol Formats

The control symbol corresponding to stype 0b011 is reserved.

4.6 Implementation-defined Symbol Formats

The control symbol corresponding to stype 0b111 is implementation defined. In general, implementation-defined control symbols will result in inter-operability problems with devices that are not designed to handle them. Inter-operability problems can include undefined and/or inconsistant behavior, data corruption, or system failure.

4.7 Control Symbol to Port Alignment

This section shows examples of control symbol transmission over the 8-bit and 16-bit interfaces. The corresponding packet transmission alignment is shown in Section 3.6, “Packet to Port Alignment.”

Figure 4-7 shows the byte transmission ordering on an 8-bit port through time using an aligned packet-accepted control symbol as an example.

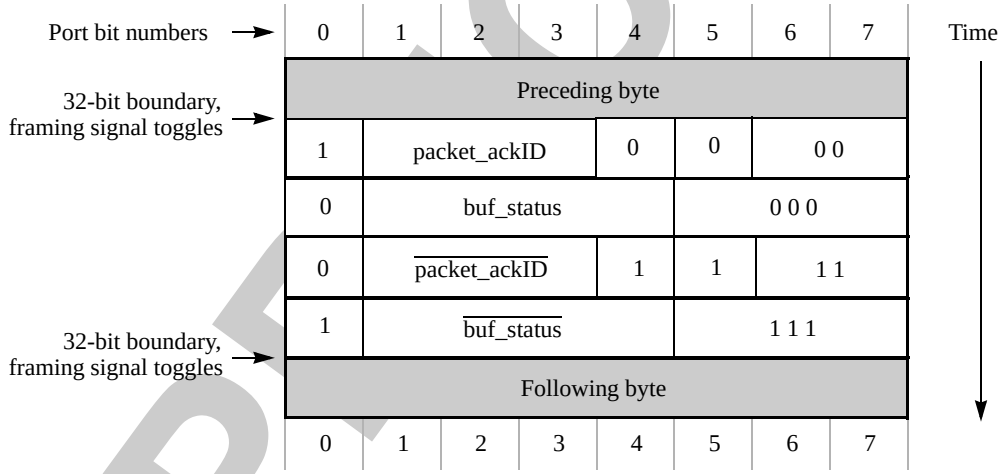


Figure 4-7. Control Symbol Transmission Example 1

Figure 4-8 shows the same control symbol over the 16-bit interface.

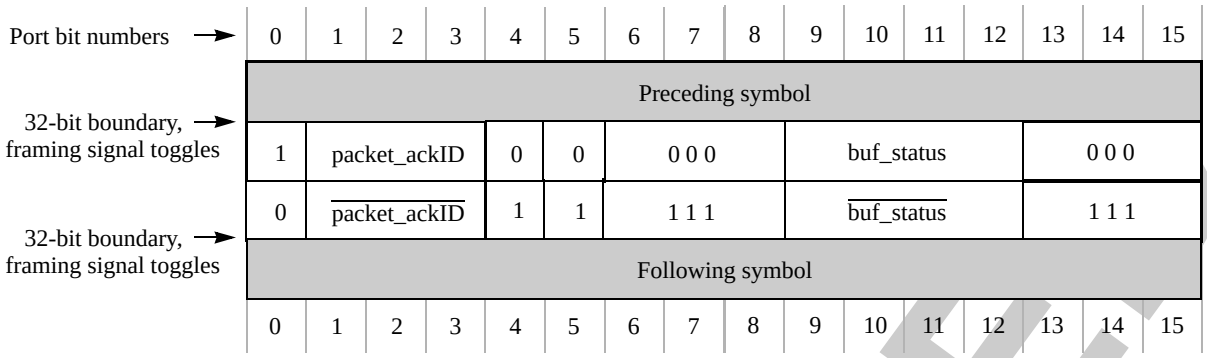


Figure 4-8. Control Symbol Transmission Example 2

Blank page

Chapter 5 8/16 LP-LVDS Registers

5.1 Introduction

This chapter describes the *RapidIO Part 4: 8/16 LP-LVDS Physical Layer Specification* 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 8/16 LP-LVDS 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 5.5, Section 5.6, Section 5.7, and Section 5.8, respectively. All four device types have the same CARs, specified in Section 5.4.

5.2 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 8/16 LP-LVDS 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 5-1. 8/16 LP-LVDS 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

5.3 Reserved Register, Bit and Bit Field Value Behavior

Table 5-2 describes the required behavior for accesses to reserved register bits and reserved registers for the RapidIO register space,

Table 5-2. 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 5-2. 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.

When a writable bit field is set to a reserved value, device behavior is implementation specific.

5.4 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 5-2 for the required behavior for accesses to reserved registers and register bits.

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

5.4.1 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 5-3.

Table 5-3. Bit Settings for Processing Element Features CAR

Bits	Name	Description
0–24	—	Reserved
25	Implementation-defined	Implementation-defined
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

5.5 Generic End Point Devices

This section describes the 8/16 LP-LVDS registers for a general end point device. This Extended Features register block is assigned Extended Features block ID=0x0001.

5.5.1 Register Map

Table 5-4 shows the register map for generic RapidIO 8/16 LP-LVDS end point 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 5-4. 8/16 LP-LVDS Register Map - Generic End Point Devices

	Block Byte Offset	Register Name
General	0x0	8/16 LP-LVDS Register Block Header
	0x4–1C	Reserved
	0x20	Port Link Timeout Control CSR
	0x24	Port Response Timeout 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
Port 15	0x220-234	Reserved
	0x238	Port 15 Error and Status CSR
	0x23C	Port 15 Control CSR

5.5.2 Command and Status Registers (CSRs)

Refer to Table 5-2 for the required behavior for accesses to reserved registers and register bits.

5.5.2.1 8/16 LP-LVDS Register Block Header (Block Offset 0x0)

The 8/16 LP-LVDS register block header register contains the EF_PTR to the next EF_BLK and the EF_ID that identifies this as the generic end point 8/16 LP-LVDS register block header.

Table 5-5. Bit Settings for 8/16 LP-LVDS 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

5.5.2.2 Port Link Timeout Control CSR (Block Offset 0x20)

The port link timeout control register contains the timeout timer value for all ports on a device. This timeout 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 timeout interval, and represents between 3 and 5 seconds.

Table 5-6. Bit Settings for Port Link Timeout Control CSR

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

5.5.2.3 Port Response Timeout Control CSR (Block Offset 0x24)

The port response timeout control register contains the timeout timer count for all ports on a device. This timeout is for sending a request packet to receiving the corresponding response packet.The reset value is the maximum timeout interval, and represents between 3 and 5 seconds.

Table 5-7. Bit Settings for Port Response Timeout Control CSR

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

5.5.2.4 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 5-8. 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

5.5.2.5 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 5-9. 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 port 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	Port Present	0b0	The port is receiving the free-running clock on the input port.
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	Input and output ports are initialized and can communicate with the adjacent device. This bit and bit 31 are mutually exclusive (read-only).
31	Port Uninitialized	0b1	Input and output ports are not initialized and is in training mode. This bit and bit 30 are mutually exclusive (read-only).

5.5.2.6 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 5-10. Bit Settings for Port *n* Control CSRs

Bit	Name	Reset Value	Description
0	Output Port Width	see footnote[1]	Operating width of the port (read-only): 0b0 - 8-bit port 0b1 - 16-bit port
1	Output Port Enable	see footnote[2]	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
2	Output Port Driver Disable	0b0	Output port driver disable: 0b0 - output port drivers are turned on and will drive the pins normally 0b1 - output port drivers are turned off and will not drive the pins This is useful for power management.
3	—		Reserved
4	Input Port Width	see footnote[3]	Operating width of the port (read-only): 0b0 - 8-bit port 0b1 - 16-bit port
5	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
6	Input Port Receiver Disable	0b0	Input port receiver enable: 0b0 - input port receivers are enabled 0b1 - input port receivers are disabled and are unable to receive to any packets or control symbols
7	—		Reserved
8	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
9	Multicast-event Participant	see footnote[5]	Send incoming multicast-event control symbols to this port (multiple port devices only)
10-13	—		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

Table 5-10. Bit Settings for Port *n* Control CSRs (Continued)

Bit	Name	Reset Value	Description
20-27	Implementation-defined		Implementation-defined
28-30	—		Reserved
31	Port Type		This indicates the port type (read only) 0b0 - Parallel port 0b1 - Reserved

[1]The output port width reset value is implementation dependent

[2]The output port enable reset value is implementation dependent

[3]The input port width 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.

5.6 Generic End Point Devices, software assisted error recovery option

This section describes the 8/16 LP-LVDS 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.

5.6.1 Register Map

Table 5-11 shows the register map for generic RapidIO 8/16 LP-LVDS 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 5-11. 8/16 LP-LVDS Register Map - Generic End Point Devices (SW assisted)

	Block Byte Offset	Register Name
General	0x0	8/16 LP-LVDS Register Block Header
	0x4–1C	Reserved
	0x20	Port Link Timeout Control CSR
	0x24	Port Response Timeout 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 5-11. 8/16 LP-LVDS Register Map - 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

5.6.2 Command and Status Registers (CSRs)

Refer to Table 5-2 for the required behavior for accesses to reserved registers and register bits.

5.6.2.1 8/16 LP-LVDS Register Block Header (Block Offset 0x0)

The 8/16 LP-LVDS register block header register contains the EF_PTR to the next EF_BLK and the EF_ID that identifies this as the generic end point 8/16 LP-LVDS register block header.

Table 5-12. Bit Settings for 8/16 LP-LVDS 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

5.6.2.2 Port Link Timeout Control CSR (Block Offset 0x20)

The port link timeout control register contains the timeout timer value for all ports on a device. This timeout 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 timeout interval, and represents between 3 and 5 seconds.

Table 5-13. Bit Settings for Port Link Timeout Control CSR

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

5.6.2.3 Port Response Timeout Control CSR (Block Offset 0x24)

The port response timeout control register contains the timeout timer count for all ports on a device. This timeout is for sending a request packet to receiving the corresponding response packet.The reset value is the maximum timeout interval, and represents between 3 and 5 seconds.

Table 5-14. Bit Settings for Port Response Timeout Control CSR

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

5.6.2.4 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 5-15. 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

5.6.2.5 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 5-16. 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.

5.6.2.6 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 link_status and ackID_status fields are defined in Section 4.4, “Link Maintenance Control Symbol Formats.” This register is read-only.

Table 5-17. 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.
1-24	—		Reserved
25-27	ackID_status	0b000	ackID status field from the link-response control symbol
28-31	link_status	0b0000	link status field from the link-response control symbol

5.6.2.7 Port *n* Local ackID Status 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 5-18. 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 unacknowledged 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.
1-4	—		Reserved
5-7	Inbound_ackID	0b000	Input port next expected ackID value
8-15	—		Reserved
16-23	Outstanding_ackID	0x00	Output port unacknowledged ackID status. A set bit indicates that the corresponding ackID value has been used to send a packet to an attached device but a corresponding acknowledge control symbol has not been received. 0b1xxx_xxxx indicates ackID 0, 0bx1xx_xxxx indicates ackID 1, 0bxx1x_xxxx indicates ackID 2, etc.
24-28	—		Reserved
29-31	Outbound_ackID	0b000	Output port next transmitted ackID value. Software writing this value can force re-transmission of outstanding unacknowledged packets in order to manually implement error recovery.

5.6.2.8 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 5-19. 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 port 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	Port Present	0b0	The port is receiving the free-running clock on the input port.
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	Input and output ports are initialized and can communicate with the adjacent device. This bit and bit 31 are mutually exclusive (read-only).
31	Port Uninitialized	0b1	Input and output ports are not initialized and is in training mode. This bit and bit 30 are mutually exclusive (read-only).

5.6.2.9 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 5-20. Bit Settings for Port *n* Control CSRs

Bit	Name	Reset Value	Description
0	Output Port Width	see footnote[1]	Operating width of the port (read-only): 0b0 - 8-bit port 0b1 - 16-bit port
1	Output Port Enable	see footnote[2]	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
2	Output Port Driver Disable	0b0	Output port driver disable: 0b0 - output port drivers are turned on and will drive the pins normally 0b1 - output port drivers are turned off and will not drive the pins This is useful for power management.
3	—		Reserved
4	Input Port Width	see footnote[3]	Operating width of the port (read-only): 0b0 - 8-bit port 0b1 - 16-bit port
5	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
6	Input Port Receiver Disable	0b0	Input port receiver enable: 0b0 - input port receivers are enabled 0b1 - input port receivers are disabled and are unable to receive to any packets or control symbols
7	—		Reserved
8	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
9	Multicast-event Participant	see footnote[5]	Send incoming multicast-event control symbols to this port (multiple port devices only)
10-13	—		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

Table 5-20. Bit Settings for Port *n* Control CSRs (Continued)

Bit	Name	Reset Value	Description
20-27	Implementation-defined		Implementation-defined
28-30	—		Reserved
31	Port Type		This indicates the port type (read only) 0b0 - Parallel port 0b1 - Reserved

[1]The output port width reset value is implementation dependent

[2]The output port enable reset value is implementation dependent

[3]The input port width 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.

5.7 Generic End Point Free Devices

This section describes the 8/16 LP-LVDS registers for a general devices that do not contain end point functionality. Typically these devices are switches. This Extended Features register block uses extended features block ID=0x0003.

5.7.1 Register Map

Table 5-21 shows the register map for generic RapidIO 8/16 LP-LVDS 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 5-21. 8/16 LP-LVDS Register Map - Generic End Point Free Devices

	Block Byte Offset	Register Name
General	0x0	8/16 LP-LVDS Register Block Header
	0x4–1C	Reserved
	0x20	Port Link Timeout 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

5.7.2 Command and Status Registers (CSRs)

Refer to Table 5-2 for the required behavior for accesses to reserved registers and register bits.

5.7.2.1 8/16 LP-LVDS Register Block Header (Block Offset 0x0)

The 8/16 LP-LVDS register block header register contains the EF_PTR to the next EF_BLK and the EF_ID that identifies this as the generic end point 8/16 LP-LVDS register block header.

Table 5-22. Bit Settings for 8/16 LP-LVDS 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

5.7.2.2 Port Link Timeout Control CSR (Block Offset 0x20)

The port link timeout control register contains the timeout timer value for all ports on a device. This timeout 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 timeout interval, and represents between 3 and 5 seconds.

Table 5-23. Bit Settings for Port Link Timeout Control CSR

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

5.7.2.3 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 5-24. 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

5.7.2.4 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 5-25. 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 port 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	Port Present	0b0	The port is receiving the free-running clock on the input port.
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	Input and output ports are initialized and can communicate with the adjacent device. This bit and bit 31 are mutually exclusive (read-only).
31	Port Uninitialized	0b1	Input and output ports are not initialized and is in training mode. This bit and bit 30 are mutually exclusive (read-only).

5.7.2.5 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 5-26. Bit Settings for Port *n* Control CSRs

Bit	Name	Reset Value	Description
0	Output Port Width	see footnote[1]	Operating width of the port (read-only): 0b0 - 8-bit port 0b1 - 16-bit port
1	Output Port Enable	see footnote[2]	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
2	Output Port Driver Disable	0b0	Output port driver disable: 0b0 - output port drivers are turned on and will drive the pins normally 0b1 - output port drivers are turned off and will not drive the pins This is useful for power management.
3	—		Reserved
4	Input Port Width	see footnote[3]	Operating width of the port (read-only): 0b0 - 8-bit port 0b1 - 16-bit port
5	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
6	Input Port Receiver Disable	0b0	Input port receiver enable: 0b0 - input port receivers are enabled 0b1 - input port receivers are disabled and are unable to receive to any packets or control symbols
7	—		Reserved
8	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
9	Multicast-event Participant	see footnote[5]	Send incoming multicast-event control symbols to this output port (multiple port devices only)
10-13	—		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

Table 5-26. Bit Settings for Port *n* Control CSRs (Continued)

Bit	Name	Reset Value	Description
20-27	Implementation-defined		Implementation-defined
28-30	—		Reserved
31	Port Type		This indicates the port type (read only) 0b0 - Parallel port 0b1 - Reserved

[1]The output port width reset value is implementation dependent

[2]The output port enable reset value is implementation dependent

[3]The input port width 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.

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

This section describes the 8/16 LP-LVDS registers for a general device that does not contain end point device functionality that supports 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.

5.8.1 Register Map

Table 5-11 shows the register map for generic RapidIO 8/16 LP-LVDS 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 5-27. 8/16 LP-LVDS Register Map - Generic End Point-free Devices (SW assisted)

	Block Byte Offset	Register Name
General	0x0	8/16 LP-LVDS Register Block Header
	0x4–1C	Reserved
	0x20	Port Link Timeout 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 5-27. 8/16 LP-LVDS Register Map - 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

5.8.2 Command and Status Registers (CSRs)

Refer to Table 5-2 for the required behavior for accesses to reserved registers and register bits.

5.8.2.1 8/16 LP-LVDS Register Block Header (Block Offset 0x0)

The 8/16 LP-LVDS register block header register contains the EF_PTR to the next EF_BLK and the EF_ID that identifies this as the generic end point 8/16 LP-LVDS register block header.

Table 5-28. Bit Settings for 8/16 LP-LVDS 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

5.8.2.2 Port Link Timeout Control CSR (Block Offset 0x20)

The port link timeout control register contains the timeout timer value for all ports on a device. This timeout 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 timeout interval, and represents between 3 and 5 seconds.

Table 5-29. Bit Settings for Port Link Timeout Control CSR

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

5.8.2.3 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 5-30. 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

5.8.2.4 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 5-31. 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.

5.8.2.5 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 link_status and ackID_status fields are defined in Section 4.4, “Link Maintenance Control Symbol Formats.” This register is read-only.

Table 5-32. 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.
1-24	—		Reserved
25-27	ackID_status	0b000	ackID status field from the link-response control symbol
28-31	link_status	0b0000	link status field from the link-response control symbol

5.8.2.6 Port *n* Local ackID Status 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 5-33. 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 unacknowledged 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.
1-4	—		Reserved
5-7	Inbound_ackID	0b000	Input port next expected ackID value
8-15	—		Reserved
16-23	Outstanding_ackID	0x00	Output port unacknowledged ackID status. A set bit indicates that the corresponding ackID value has been used to send a packet to an attached device but a corresponding acknowledge control symbol has not been received. 0b1xxx_xxxx indicates ackID 0, 0bx1xx_xxxx indicates ackID 1, 0bxx1x_xxxx indicates ackID 2, etc.
24-28	—		Reserved
29-31	Outbound_ackID	0b000	Output port next transmitted ackID value. Software writing this value can force re-transmission of outstanding unacknowledged packets in order to manually implement error recovery.

5.8.2.7 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 5-34. 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 port 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	Port Present	0b0	The port is receiving the free-running clock on the input port.
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	Input and output ports are initialized and can communicate with the adjacent device. This bit and bit 31 are mutually exclusive (read-only).
31	Port Uninitialized	0b1	Input and output ports are not initialized and is in training mode. This bit and bit 30 are mutually exclusive (read-only).

5.8.2.8 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 5-35. Bit Settings for Port *n* Control CSRs

Bit	Name	Reset Value	Description
0	Output Port Width	see footnote[1]	Operating width of the port (read-only): 0b0 - 8-bit port 0b1 - 16-bit port
1	Output Port Enable	see footnote[2]	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
2	Output Port Driver Disable	0b0	Output port driver disable: 0b0 - output port drivers are turned on and will drive the pins normally 0b1 - output port drivers are turned off and will not drive the pins This is useful for power management.
3	—		Reserved
4	Input Port Width	see footnote[3]	Operating width of the port (read-only): 0b0 - 8-bit port 0b1 - 16-bit port
5	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
6	Input Port Receiver Disable	0b0	Input port receiver enable: 0b0 - input port receivers are enabled 0b1 - input port receivers are disabled and are unable to receive to any packets or control symbols
7	—		Reserved
8	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
9	Multicast-event Participant	see footnote[5]	Send incoming multicast-event control symbols to this port (multiple port devices only)
10-13	—		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

Table 5-35. Bit Settings for Port *n* Control CSRs (Continued)

Bit	Name	Reset Value	Description
20-27	Implementation-defined		Implementation-defined
28-30	—		Reserved
31	Port Type		This indicates the port type (read only) 0b0 - Parallel port 0b1 - Reserved

[1]The output port width reset value is implementation dependent

[2]The output port enable reset value is implementation dependent

[3]The input port width 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 6 System Clocking Considerations

6.1 Introduction

The RapidIO parallel physical interface can be deployed in a variety of system configurations. A fundamental aspect to the successful deployment of RapidIO is clock distribution. This section is provided to point out the issues of distributing clocks in a system.

6.2 Example Clock Distribution

Clock distribution in a small system is straightforward. It is assumed that clocking is provided from a single clock source (Figure 6-1).

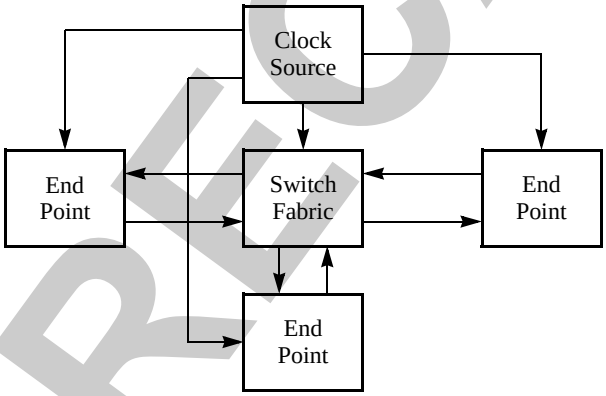


Figure 6-1. Clock Distribution in a Small System

In this case the timing budget must account for any skew and jitter component between each point. Skew and jitter are introduced owing to the end point clock regeneration circuitry (PLL or DLL) and to transmission line effects.

Distributing a clock from a central source may not be practical in larger or more robust systems. In these cases it may be desirable to have multiple clock sources or to distribute the clock through the interconnect. Figure 6-2 displays the clock distribution in a larger system.

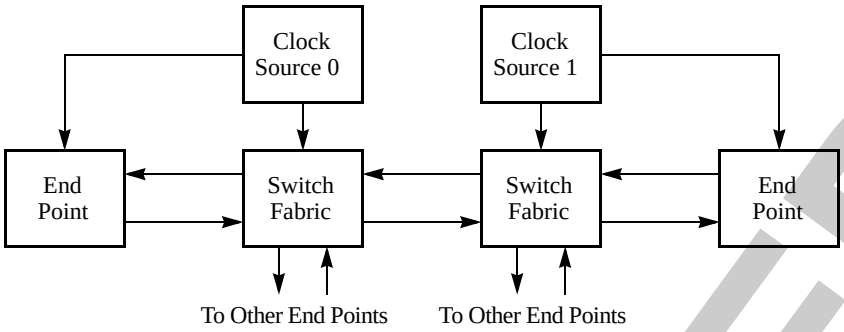


Figure 6-2. Clock Distribution in a Larger System

In such a system the clock sources may be of the same relative frequency; however, they are not guaranteed to be always at exact frequency. Clock sources will drift in phase relationship with each other over time. This adds an additional component because it is possible that one device may be slightly faster than its companion device. This requires a packet elasticity mechanism.

If the clock is transported through the interconnect as shown in Figure 6-3, then additive clock jitter must be taken into account.

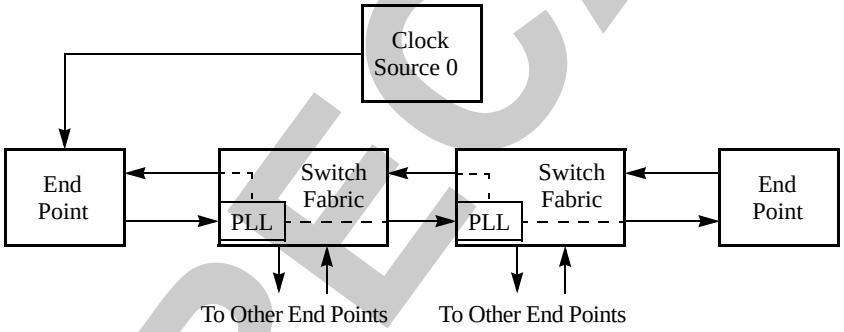


Figure 6-3. Clock Distribution Through the Interconnect

Assuming that each device gets a clock that was regenerated by its predecessor, and each device adds a certain jitter component to the clock, the resulting clock at the end point may be greatly unstable. This factor must be added to the timing budget.

6.3 Elasticity Mechanism

In systems with multiple clock sources, clocks may be of the same relative frequency but not exact. Their phase will drift over time. An elasticity mechanism is therefore required to keep devices from missing data beats. For example, if the received clock is faster than the internal clock, then it may be necessary to delete an inbound symbol. If the received clock is slower than the internal clock, then it may be necessary to insert an inbound symbol.

This RapidIO 8/16 LP-LVDS interface is source synchronous; therefore, it is guaranteed that a data element will have an associated clock strobe with which to synchronize. A clock boundary is crossed in the receive logic of the end point as the inbound data is synchronized to the internal clock. It must be guaranteed in the end point that a drift between the two clock sources does not cause a setup hold violation resulting in metastability in capturing the data.

To ensure that data is not missed, an end point implements an elasticity buffer. RapidIO uses idle control symbols as the elasticity mechanism. If a receiver needs to skip a symbol during receipt of a large packet, it can issue a throttle control symbol to cause the sender to insert an aligned pacing idle control symbol in the byte stream.

A data beat is clocked into the elasticity buffer with the external clock. The data beat is pulled out of the elasticity buffer using the internal clock delayed by a number of clocks behind the external clock event. This allows the data to become stable before it is synchronized to the internal clock. If the two clock events drift too close together then it is necessary for the synchronization logic to reset the tap and essentially skip a symbol. By guaranteeing a periodic idle control symbol, it is possible for the receive logic to skip a data beat and not miss a critical symbol element.

Blank page

Chapter 7 Board Routing Guidelines

7.1 Introduction

This chapter contains board design guidelines for RapidIO based systems. The information here is presented as a guide for implementing a RapidIO board design. It is noted that the board designer may have constraints such as standard design practices, vendor selection criteria, and design methodology that must be followed. Therefore appropriate diligence must be applied by the designer.

RapidIO is a source-synchronous differential point-to-point interconnect, so routing considerations are minimal. The very high clock rate places a premium on minimizing skew and discontinuities, such as vias and bends. Generally, layouts should be as straight and free of vias as possible using controlled impedance differential pairs.

7.2 Impedance

Interconnect design should follow standard practice for differential pairs. To minimize reflections from the receiver's 100 Ohm termination, each side of the coupled pair should have a characteristic impedence of 50 Ohms (i..e. 100 Ohms differential impedence). The two signals forming the differential pair should be tightly coupled. The differential pairs should be widely spaced, consistent with skew control and quality routing, so that the crosstalk noise is common mode.

7.3 Skew

To minimize the skew on a RapidIO channel the total electrical length for each trace within each unidirectional channel should be equal. Several layouts are suggested in

Figure 7-1.

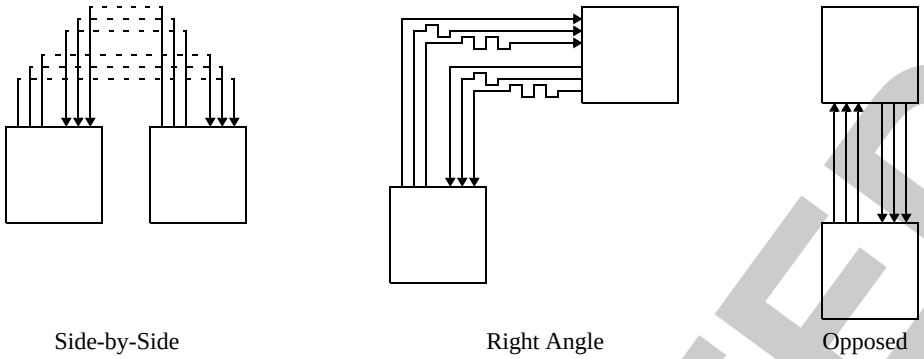


Figure 7-1. Routing for Equalized Skew for Several Placements

Because the RapidIO model is source synchronous, the total length is not critical. Best signal integrity is achieved using a clean layout between opposed parts due to routing on a single layer.

The side-by-side layout requires two routing layers and has reduced signal integrity due to the vias between layers. To keep the total electrical length equal, both layers must have the same phase velocity.

Finally, right angle routing requires meandering to equalize delay, and meandered sections reduce signal integrity while increasing radiation. It may be necessary to place meandered sections on a second routing layer to keep the routing clean.

All skew calculations should be taken to the edge of the package. The package layout and PCB breakout are co-designed to minimize skew, and a recommended PCB breakout is provided.

7.4 PCB Stackup

PCB stackup has a significant effect on EMI generated by the high frequency of operation of a RapidIO channel, so EMI control must be planned from the start. Several stackups are shown in Figure 7-2.

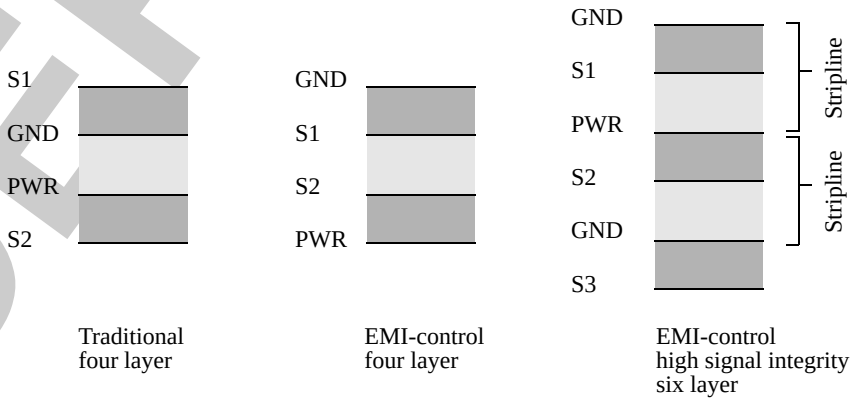


Figure 7-2. Potential PCB Stackups

The traditional four-layer stackup provides equal phase velocities on the two routing layers, but the placement of routing on the outside layers allows for easier radiation. This stackup is suitable for very short interconnects or for applications using an add-on shield.

The four-layer stackup can be rearranged to help with EMI control by placing the power and ground layers on the outside. Each routing layer still has equal phase velocities, but orthogonal routing can degrade signal integrity at very high speeds. The power distribution inductance is approximately tripled due to the larger spacing between the power and ground planes, so applications using this stackup should plan on using more and higher quality bypass capacitance.

The six-layer stackup shows one of many possible stackups. High-speed routing is on S1 and S2 in stripline, so signal quality is excellent with EMI control. S3 is for low-speed signals. Both S1 and S2 have equal phase velocities, good impedance control, and excellent isolation. Power distribution inductance is comparable to the four-layer stackup since the extra GND plane makes up for the extra (2X) spacing between PWR and GND. This example stackup is not balanced with respect to metal loading.

7.5 Termination

Depending upon the individual device characteristics and the requirements of the particular application, the board route may be required to encompass external devices such as terminating resistors or networks. The effect of such devices on the board route must be carefully analyzed and controlled.

7.6 Additional Considerations

The application environment for a RapidIO channel may place additional constraints on the PCB design.

7.6.1 Single Board Environments

A RapidIO channel completely constructed onto a single board offers the highest performance in terms of clock rate and signal integrity. The primary issues are clean routing with minimal skew. Higher clock rates put greater emphasis on the use of quality sockets (in terms of electrical performance) or on eliminating sockets altogether.

7.6.2 Single Connector Environments

The high clock rate of the 8/16 LP-LVDS physical layer requires the use of an impedance-controlled edge connector. The number of pins dedicated to power should equal the number dedicated to ground, and the distribution of power and ground pins should be comparable. If ground pins greatly outnumber power pins,

then bypass capacitors along the length of each side of the connector should be provided. Place the connector as close to one end of the RapidIO interconnect as possible.

7.6.3 Backplane Environments

With two connectors, the design considerations from the single connector environment apply but with greater urgency. The two connectors should either be located as close together or as far apart as possible.

7.7 Recommended pin escape ordering

Given the source-synchronous nature of the 8/16 LP-LVDS physical layer and the clock to data pin skew concern for maximum operating frequency, the recommended bit escape ordering (assuming the device and port orientation shown in Figure 7-1) is shown graphically in Figure 7-3 and Figure 7-4. The figures assume that the device is being viewed from the top. For BGA-style packaged devices the recommended bit escape wire route should be supplied to the board designer. The signal names are defined in Chapter 8, “Signal Descriptions”.

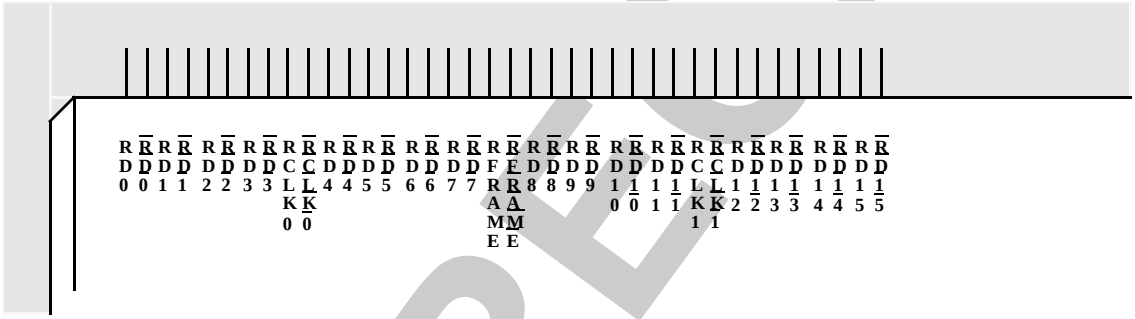


Figure 7-3. Recommended device pin escape, input port, top view of device

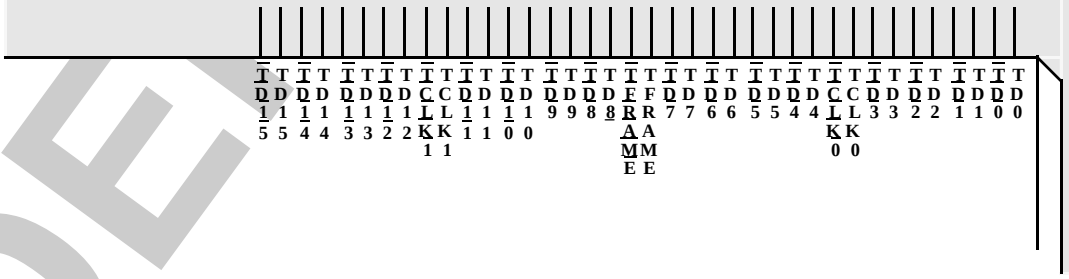


Figure 7-4. Recommended device pin escape, output port, top view of device

These pin escapes allow clean board routes that provide maximum performance connections between two devices as can be seen in the example in Figure 7-5 below.

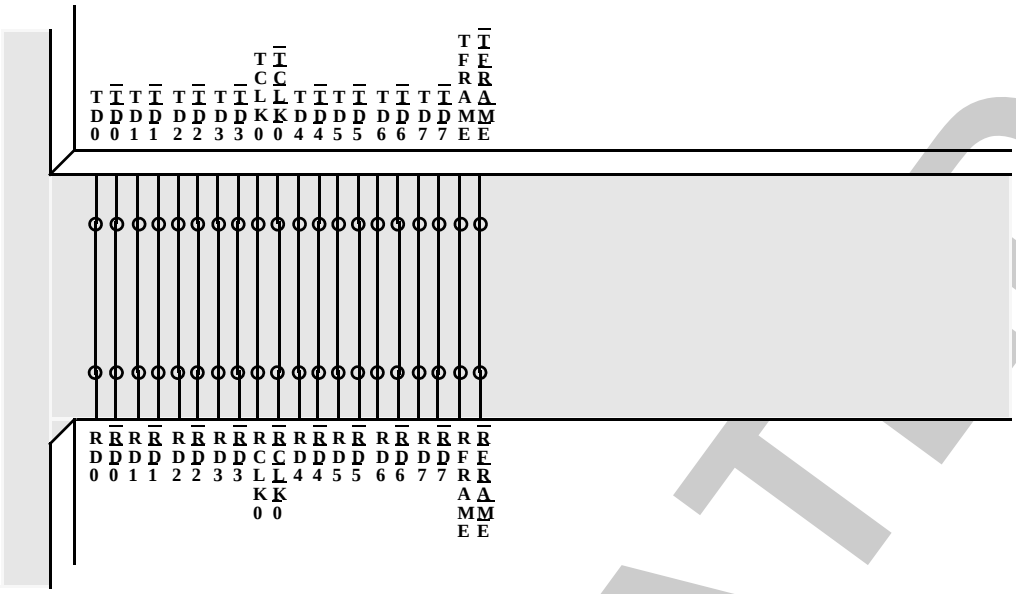


Figure 7-5. Opposed orientation, same side of board

If the attached devices are mounted with certain device orientations the bit wires become crossed. An example of this situation is shown in Figure 7-6. It is permissible for a device to also allow a bit-reversing option on the output (or input) port to support these orientations, as shown in Figure 7-6 and Figure 7-7.

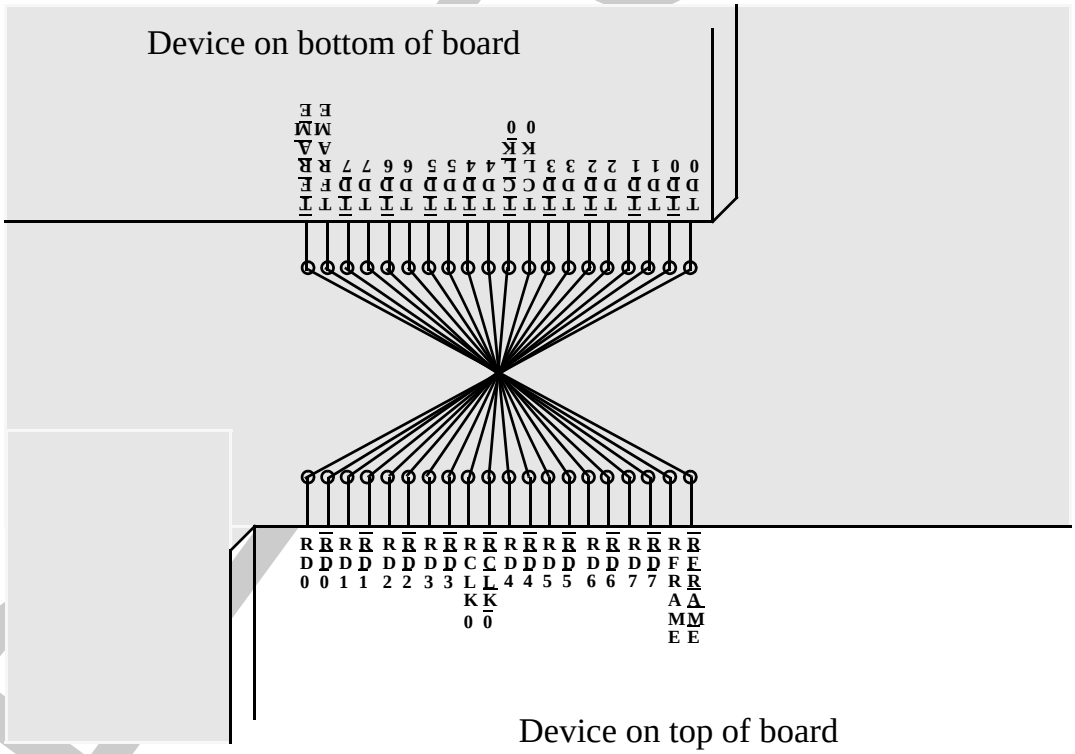


Figure 7-6. Opposed orientation, opposite sides of board

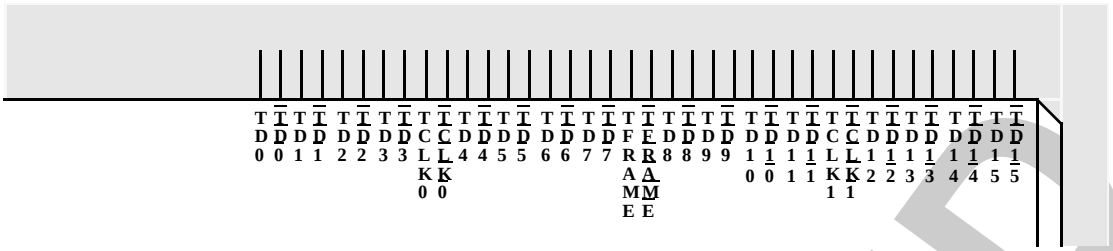


Figure 7-7. Recommended device pin escape, output port reversed, top view of device

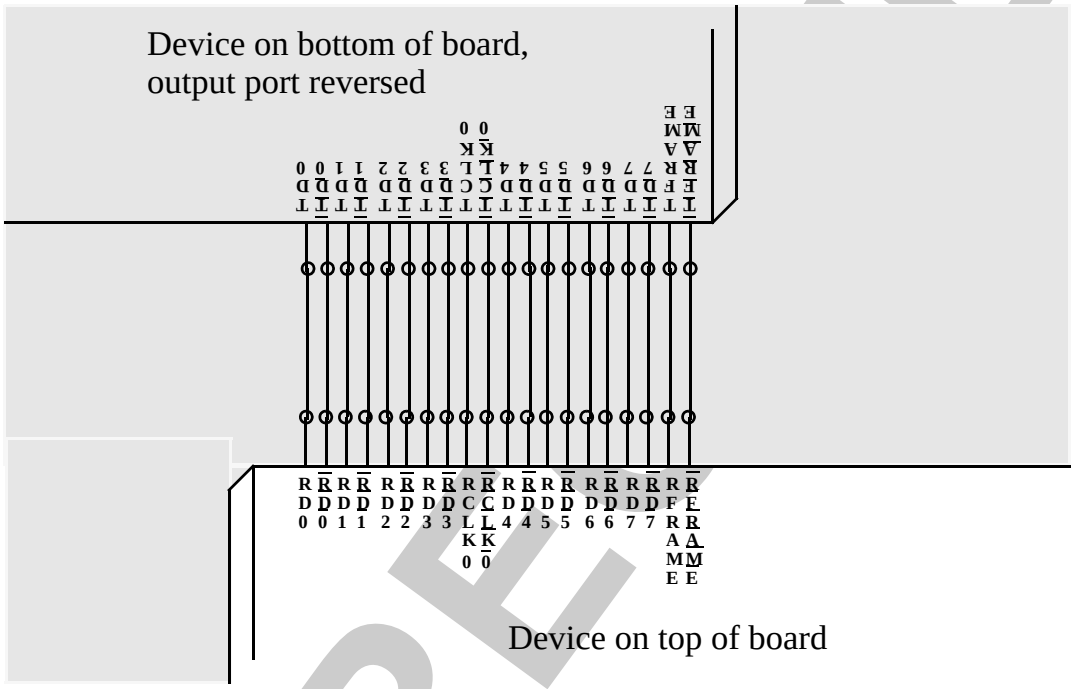


Figure 7-8. Opposed orientation, output port reversed, opposite sides of board

Chapter 8 Signal Descriptions

8.1 Introduction

This chapter contains the signal pin descriptions for a RapidIO 8/16 LP-LVDS port. The interface is defined as a parallel 10 bit full duplex point-to-point interface using differential LVDS signaling. The LVDS electrical details are described in Chapter 9, “Electrical Specifications.”

8.2 Signal Definitions

Table 8-1 provides a summary of the RapidIO signal pins as well as a short description of their functionality.

Table 8-1. 8/16 LP-LVDS Signal Descriptions

Signal Name	I/O	Signal Meaning	Timing Comments
TCLK0	O	Transmit Clock—Free-running clock for the 8-bit port and the most significant half of the 16-bit port. TCLK0 connects to RCLK0 of the receiving device.	
$\overline{\text{TCLK0}}$	O	Transmit Clock complement—This signal is the differential pair of the TCLK0 signal.	
TD[0-7]	O	Transmit Data—The transmit data is a unidirectional point to point bus designed to transmit the packet information along with the associated TCLK0 and TFRAME. The TD bus of one device is connected to the RD bus of the receiving device.	Assertion of TD[0-7] is always done with a fixed relationship to TCLK0 as defined in the AC section
$\overline{\text{TD[0-7]}}$	O	Transmit Data complement—This vector is the differential pair of TD[0-7].	Same as TD
TFRAME	O	Transmit framing signal—When issued as active this signal indicates a packet control event. TFRAME is connected to RFRAME of the receiving device.	Assertion of TFRAME is always done with a fixed relationship to TCLK0 as defined in the AC section
$\overline{\text{TFRAME}}$	O	Transmit frame complement—This signal is the differential pair of the TFRAME signal.	Same as TFRAME
TCLK1	O	Transmit Clock—Free-running clock for the least significant half of the 16-bit port (TD[8-15]). TCLK1 connects to RCLK1 of the receiving device. This signal is not used when connected to an 8-bit device.	
$\overline{\text{TCLK1}}$	O	Transmit Clock complement—This signal is the differential pair of the TCLK1 signal.	

Table 8-1. 8/16 LP-LVDS Signal Descriptions (Continued)

Signal Name	I/O	Signal Meaning	Timing Comments
TD[8-15]	O	Transmit Data—least significant half of the 16-bit port. These signals are not used when connected to an 8-bit device.	Assertion of TD[8-15] is always done with a fixed relationship to TCLK0 and TCLK1 as defined in the AC section
$\overline{\text{TD[8-15]}}$	O	Transmit Data complement—This vector is the differential pair of TD[8-15]	Same as TD[8-15]
RCLK0	I	Receive Clock—Free-running input clock for the 8-bit port and the most significant half of the 16-bit port. RCLK0 connects to TCLK0 of the transmitting device.	
$\overline{\text{RCLK0}}$	I	Receive Clock complement—This signal is the differential pair of the RCLK signal. $\overline{\text{RCLK0}}$ connects to $\overline{\text{TCLK0}}$ of the transmitting device.	
RD[0-7]	I	Receive Data—The Receive data is a unidirectional packet data input bus. It is connected to the TD bus of the transmitting device.	
$\overline{\text{RD[0-7]}}$	I	Receive Data complement—This vector is the differential pair of the RD vector.	
RFRAME	I	Receive Frame—This control signal indicates a special packet framing event on the RD pins.	RFRAME is sampled with respect to RCLK0
$\overline{\text{RFRAME}}$	I	Receive Frame complement—This signal is the differential pair of the RFRAME signal.	Same as RFRAME
RCLK1	I	Receive Clock—Free-running input clock for the least significant half of the 16-bit port (RD[8-15]). RCLK1 connects to TCLK1 of the transmitting device. This signal is not used when connected to an 8-bit device.	
$\overline{\text{RCLK1}}$	I	Receive Clock complement—This signal is the differential pair of the RCLK1 signal.	
RD[8-15]	I	Receive Data—Least significant half of the 16-bit port. These signals are not used when connected to an 8-bit device.	
$\overline{\text{RD[8-15]}}$	I	Receive Data complement—This vector is the differential pair of the RD[8-15] vector.	

8.3 RapidIO Interface Diagrams

Figure 8-1 shows the signal interface diagram connecting two 8-bit devices together with the RapidIO 8/16 LP-LVDS interconnect.

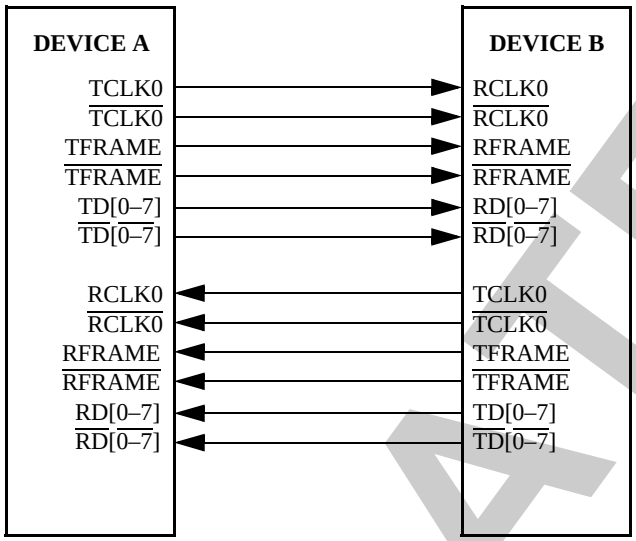


Figure 8-1. RapidIO 8-bit Device to 8-bit Device Interface Diagram

Figure 8-2 shows the connections between an 8-bit wide 8/16 LP-LVDS device and a 16-bit wide device.

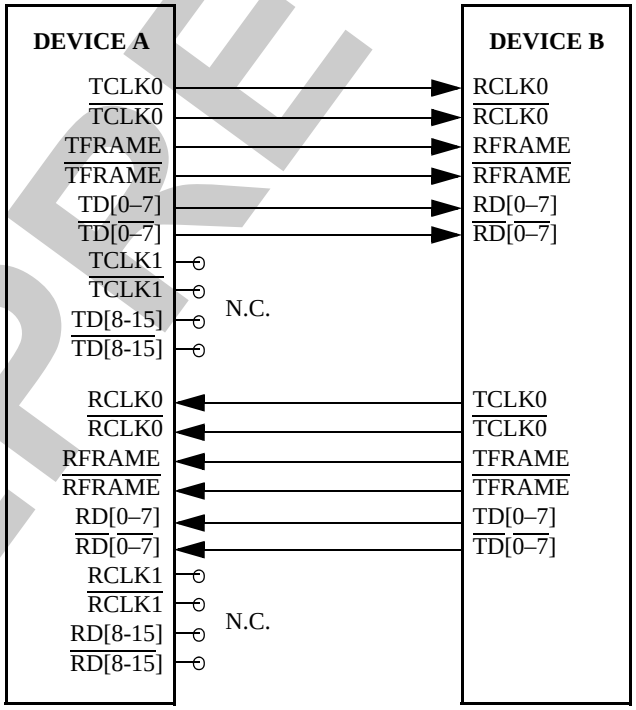


Figure 8-2. RapidIO 8-bit Device to 16-bit Device Interface Diagram

Figure 8-3 shows the connections between two 16-bit wide 8/16 LP-LVDS devices.

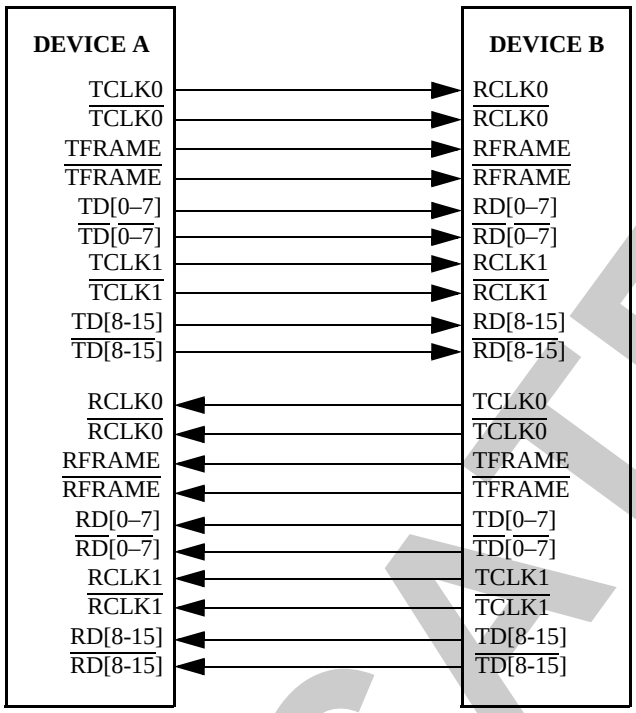


Figure 8-3. RapidIO 16-bit Device to 16-bit Device Interface Diagram

Chapter 9 Electrical Specifications

9.1 Introduction

This chapter contains the driver and receiver AC and DC electrical specifications for a *RapidIO Part 4: 8/16 LP-LVDS Physical Layer Specification* compliant device. The interface defined is a parallel differential low-power high-speed signal interface.

9.2 Overview

To allow more general compatibility with a variety of silicon solutions, the RapidIO parallel interface builds on the low voltage differential signaling (LVDS) standard. For reference refer to ANSI/TIA/EIA-644-A, *Electrical Characteristics of Low Voltage Differential Signaling (LVDS) Interface Circuits*. The goal of the interface is to allow two devices to communicate with each other within a monolithic system, and key factors in choosing an interface are electrical performance, power consumption (both at the end point and in the switch fabric), signal robustness, circuit complexity, pin count, future scalability, and industry acceptance. LVDS satisfies these requirements.

Although differential signaling requires twice as many signals as single-ended signaling, the total pin count including power and ground pins for high-speed differential and single-ended interfaces are more comparable. Single-ended interfaces require large numbers of power and ground pins to provide a low-impedance AC return path. Since LVDS uses constant-current drivers, a low-impedance AC return path is not needed, allowing for a dramatic reduction in the number of power and ground pins dedicated to the interface. The constant-current drivers also generate very small switching transients leading to lower noise and lower EMI. Differential signaling is also not as susceptible to imperfections in transmission lines and connectors.

LVDS provides for a low-voltage swing (less than 1 Volt), process independent, point-to-point differential interface. The intent of this signaling specification is for device-to-device and board-to-board applications, but it may not be suitable for cable applications owing to the stringent signal-to-signal skew requirements.

LVDS is an end point self-terminated interface. It is assumed that each receiver provides its own termination resistors. LVDS can tolerate ground potential differences between transmitter and receiver of +/- 1V.

9.3 DC Specifications

RapidIO driver and receiver DC specifications are displayed in Table 9-1 and Table 9-2. Power variation is +/- 5%. Resistor tolerances are +/- 1%.

Table 9-1. RapidIO 8/16 LP-LVDS Driver Specifications (DC)

Characteristic	Symbol	Min	Max	Unit	Notes
Differential output high voltage	V_{OHD}	247	454	mV	Bridged 100Ω load See Figure 9-1
Differential output low voltage	V_{OLD}	-454	-247	mV	Bridged 100Ω load See Figure 9-1
Differential offset voltage	ΔV_{OD}		50	mV	Bridged 100Ω load $\lvert V_{OHD}+V_{OLD}\rvert$. See Figure 9-1
Output high common mode voltage	V_{OSH}	1.125	1.375	V	Bridged 100Ω load
Output low common mode voltage	V_{OSL}	1.125	1.375	V	Bridged 100Ω load
Common mode offset voltage	ΔV_{OS}		50	mV	Bridged 100Ω load $\lvert V_{OSH}-V_{OSL}\rvert$. See Figure 9-1
Short circuit current (either output)	$\lvert I_{SS}\rvert$		24	mA	Outputs shorted to V_{DD} or V_{SS}
Bridged short circuit current	$\lvert I_{SB}\rvert$		12	mA	Outputs shorted together

Table 9-2. RapidIO 8/16 LP-LVDS Receiver Specifications (DC)

Characteristic	Symbol	Min	Max	Unit	Notes
Voltage at either input	V_I	0	2.4	V	
Differential input high voltage	V_{IHD}	100	600	mV	Over the common mode range
Differential input low voltage	V_{ILD}	-600	-100	mV	Over the common mode range
Common mode input range (referenced to receiver ground)	V_{IS}	0.050	2.350	V	Limited by V_I
Input differential resistance	R_{IN}	80	120	Ω	For on-chip termination.[1]

[1]Off-chip termination value and tolerance is vendor defined consistant with the return loss specification. Receiver input impedance shall exhibit a differential return loss better than 10 dB from DC to (1.6 * AC Clock Frequency). The differential return loss must measured at and include effects due to the receiver itself, associated circuitry such as ESD structures, chip packaging, and any external termination structures related to the receiver. The reference impedance for measurement is 100 ohms.

DC driver signal levels are displayed in Figure 9-1.

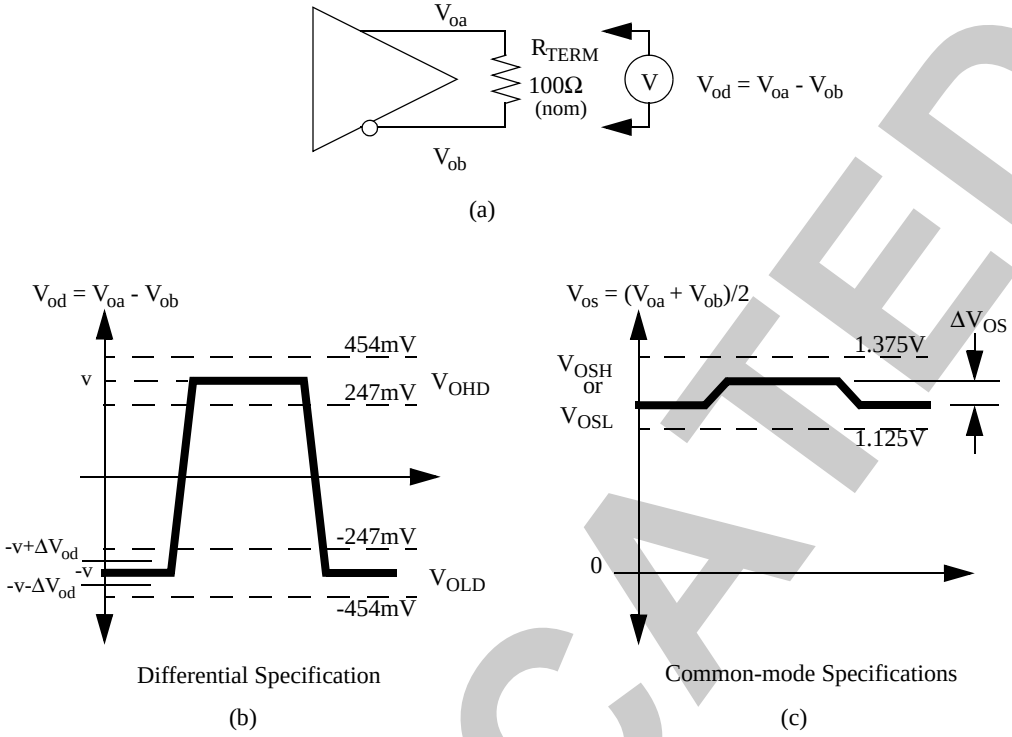


Figure 9-1. DC driver signal levels

9.4 AC Specifications

This section contains the AC electrical specifications for a RapidIO 8/16 LP-LVDS interface. The interface defined is a parallel differential low-power high-speed signal interface. RapidIO specifies operation at specific nominal frequencies only. Correct operation at other frequencies is not implied, even if the frequency is lower than the specified frequency.

9.4.1 Concepts and Definitions

This section specifies signals using differential voltages. Figure 9-2 shows how the signals are defined. The figure 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:

1. The transmitter output and receiver input signals TD, $\overline{TD}$, RD and $\overline{RD}$ each have a peak-to-peak swing of A-B Volts.
2. The differential output signal of the transmitter, V_{OD}, is defined as V_{TD}-$V_{\overline{TD}}$.
3. The differential input signal of the receiver, V_{ID}, is defined as V_{RD}-$V_{\overline{RD}}$.
4. The differential output signal of the transmitter, or input signal of the receiver, ranges from A - B Volts to -(A - B) Volts.
5. The peak differential signal of the transmitter output, or receiver input, is A - B Volts.
6. The peak to peak differential signal of the transmitter output, or receiver input, is 2*(A - B) Volts.

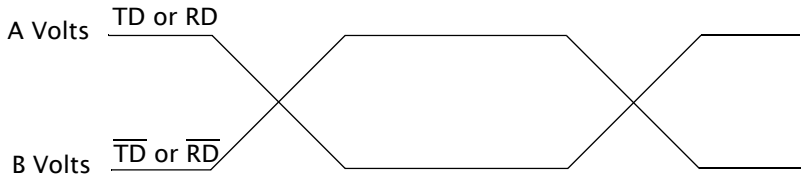


Figure 9-2. Differential Peak-Peak Voltage of Transmitter or Receiver

To illustrate these definitions using numerical values, consider the case where a LVDS transmitter has a common mode voltage of 1.2V and each signal has a swing that goes between 1.4V and 1.0V. Using these values, the peak-to-peak voltage swing of the signals TD, $\overline{TD}$, RD and $\overline{RD}$ is 400 mV. The differential signal ranges between 400mV and -400mV. The peak differential signal is 400mV, and the peak to peak differential signal is 800mV.

A timing edge is the zero-crossing of a differential signal. Each skew timing parameter on a parallel bus is synchronously measured on two signals relative to each other in the same cycle, such as data to data, data to clock, or clock to clock. A skew timing parameter may be relative to the edge of a signal or to the middle of two

sequential edges.

Static skew represents the timing difference between signals that does not vary over time regardless of system activity or data pattern. Path length differences are a primary source of static skew.

Dynamic skew represents the amount of timing difference between signals that is dependent on the activity of other signals and varies over time. Crosstalk between signals is a source of dynamic skew.

Eye diagrams and compliance masks are a useful way to visualize and specify driver and receiver performance. This technique is used in several serial bus specifications. An example compliance mask is shown in Figure 9-3. The key difference in the application of this technique for a parallel bus is that the data is source synchronous to its bus clock while serial data is referenced to its embedded clock. Eye diagrams reveal the quality (“cleanness”, “openness”, “goodness”) of a driver output or receiver input. An advantage of using an eye diagram and a compliance mask is that it allows specifying the quality of a signal without requiring separate specifications for effects such as rise time, duty cycle distortion, data dependent dynamic skew, random dynamic skew, etc. This allows the individual semiconductor manufacturer maximum flexibility to trade off various performance criteria while keeping the system performance constant.

In using the eye pattern and compliance mask approach, the quality of the signal is specified by the compliance mask. The mask specifies the maximum permissible magnitude of the signal and the minimum permissible eye opening. The eye diagram for the signal under test is generated according to the specification. Compliance is determined by whether the compliance mask can be positioned over the eye diagram such that the eye pattern falls entirely within the unshaded portion of the mask.

Serial specifications have clock encoded with the data, but the LP-LVDS physical layer defined by RapidIO is a source synchronous parallel port so additional specifications to include effects that are not found in serial links are required. Specifications for the effect of bit to bit timing differences caused by static skew have been added and the eye diagrams specified are measured relative to the associated clock in order to include clock to data effects. With the transmit output (or receiver input) eye diagram, the user can determine if the transmitter output (or receiver input) is compliant with an oscilloscope with the appropriate software.

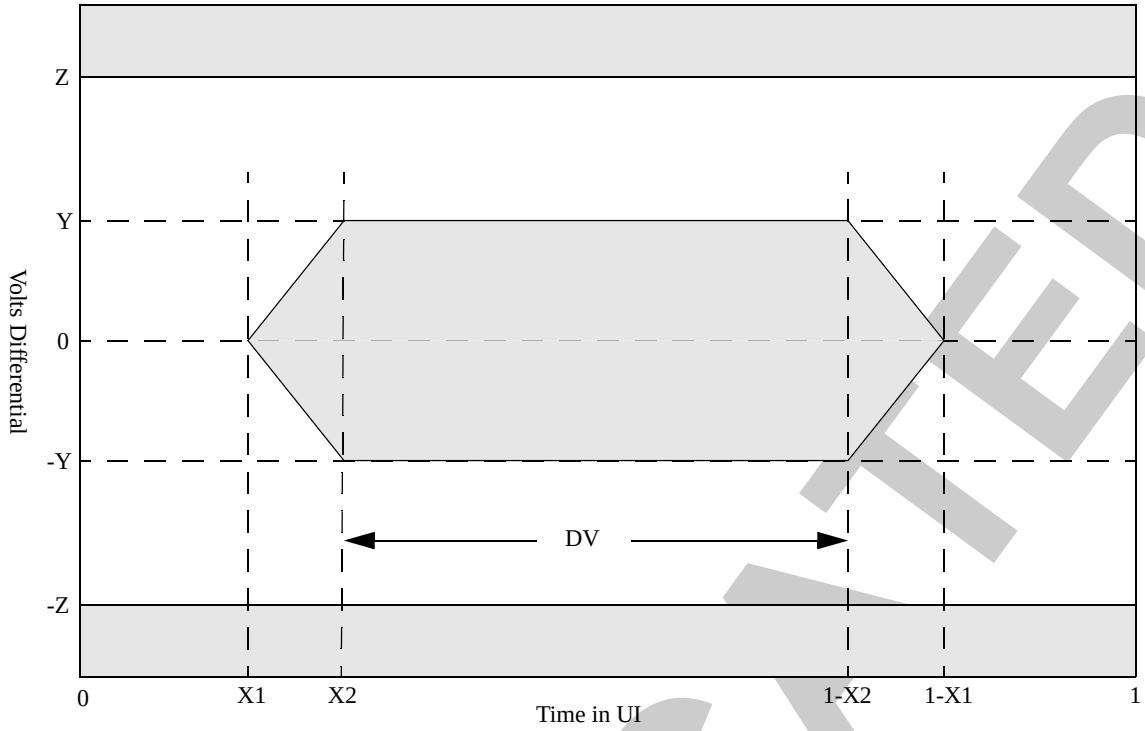


Figure 9-3. Example Compliance Mask

Y = Minimum data valid amplitude

Z = Maximum amplitude

1 UI = 1 Unit Interval = 1/Baud rate

X1 = End of zero crossing region

X2 = Beginning of Data Valid window

DV = Data Valid window = 1 - 2*X2

The waveform of the signal under test must fall within the unshaded area of the mask to be compliant. Different masks are used for the driver output and the receiver input allowing each to be separately specified.

9.4.2 Driver Specifications

Driver AC timing specifications are given in Table 9-3 through Table 9-7 below. A driver shall comply with the specifications for each data rate/frequency for which operation of the driver is specified. Unless otherwise specified, these specifications are subject to the following conditions.

The specifications apply over the supply voltage and ambient temperature ranges specified by the device vendor.

The specifications apply for any combination of data patterns on the data signals.

The output of a driver shall be connected to a 100 Ohm, +/- 1%, differential (bridged) resistive load.

Clock specifications apply only to clock signals (CLK0 and, if present, CLK1).

Data specifications apply only to data signals (FRAME, D[0-7], and, if present, D[8-15]).

FRAME and D[0-7] are the data signals associated with CLK0, D[8-5] are the data signals associated with CLK1.

Driver DC termination is not specified (in accordance with TIA/EIA-644-A), but is recommended for devices targeting higher data rates. This termination is intended to reduce data reflections in the matched data interconnect. The value and location of this termination and the methods of test and measurement are left to the individual vendor.

Table 9-3. Driver AC Timing Specifications - 500Mbps Data Rate/250MHz Clock Rate

Characteristic	Symbol	Range		Unit	Notes
		Min	Max		
Differential output high voltage	V_{OHD}	200	540	mV	See Figure 9-4
Differential output low voltage	V_{OLD}	-540	-200	mV	See Figure 9-4
Unit interval	UI	2000	2000	ps	Requires +/-100ppm long term frequency stability
Duty cycle of the clock output	DC	48	52	%	Measured at V_{OD}=0V
V_{OD} fall time, 20-80% of the peak to peak differential signal swing	t_{FALL}	.1		UI	
V_{OD} rise time, 20-80%of the peak to peak differential signal swing	t_{RISE}	.1		UI	
Data Valid	DV	.63		UI	Measured using the RapidIO Transmit Mask shown in Figure 9-4
Allowable static skew between any two data outputs within a 8 bit/9 bit group	t_{DPAIR}		.09	UI	See Figure 9-10

Table 9-3. Driver AC Timing Specifications - 500Mbps Data Rate/250MHz Clock Rate

Characteristic	Symbol	Range		Unit	Notes
		Min	Max		
Allowable static skew of data outputs to associated clock	$t_{SKEW,PAIR}$	-.09	.09	UI	See Figure 9-8, Figure 9-10
Clock to clock static skew	$t_{CSKEW, PAIR}$		.09	UI	See Figure 9-9
Clock to clock dynamic skew	$t_{CSKEW, PAIRD}$		.2	UI	See Figure 9-9

Table 9-4. Driver AC Timing Specifications - 750Mbps Data Rate/375MHz Clock Rate

Characteristic	Symbol	Range		Unit	Notes
		Min	Max		
Differential output high voltage	V_{OHD}	200	540	mV	See Figure 9-4
Differential output low voltage	V_{OLD}	-540	-200	mV	See Figure 9-4
Unit interval		1333	1333	ps	Requires +/-100ppm long term frequency stability
Duty cycle of the clock output	DC	48	52	%	Measured at V_{OD}=0V
V_{OD} fall time, 20-80% of the peak to peak differential signal swing	t_{FALL}	.1		UI	
V_{OD} rise time, 20-80%of the peak to peak differential signal swing	t_{RISE}	.1		UI	
Data Valid	DV	.6		UI	Measured using the RapidIO Transmit Mask shown in Figure 9-4
Allowable static skew between any two data outputs within a 8 bit/9 bit group	t_{DPAIR}		.1	UI	See Figure 9-10
Allowable static skew of data outputs to associated clock	$t_{SKEW,PAIR}$	-.1	.1	UI	See Figure 9-8, Figure 9-10
Clock to clock static skew	$t_{CSKEW, PAIR}$		.15	UI	See Figure 9-9
Clock to clock dynamic skew	$t_{CSKEW, PAIRD}$		.2	UI	See Figure 9-9

Table 9-5. Driver AC Timing Specifications - 1000Mbps Data Rate/500MHz Clock Rate

Characteristic	Symbol	Range		Unit	Notes
		Min	Max		
Differential output high voltage	V_{OHD}	200	540	mV	See Figure 9-4
Differential output low voltage	V_{OLD}	-540	-200	mV	See Figure 9-4
Unit interval		1000	1000	ps	Requires +/-100ppm long term frequency stability
Duty cycle of the clock output	DC	48	52	%	Measured at V_{OD}=0V
V_{OD} fall time, 20-80% of the peak to peak differential signal swing	t_{FALL}	.1		UI	

Table 9-5. Driver AC Timing Specifications - 1000Mbps Data Rate/500MHz Clock Rate

Characteristic	Symbol	Range		Unit	Notes
		Min	Max		
V_{OD} rise time, 20-80%of the peak to peak differential signal swing	t_{RISE}	.1		UI	
Data Valid	DV	.575		UI	Measured using the RapidIO Transmit Mask shown in Figure 9-4
Allowable static skew between any two data outputs within a 8 bit/9 bit group	t_{DPAIR}		.1	UI	See Figure 9-10
Allowable static skew of data outputs to associated clock	$t_{SKEW,PAIR}$	-.1	.1	UI	See Figure 9-8, Figure 9-10
Clock to clock static skew	$t_{CSKEW, PAIR}$		.15	UI	See Figure 9-9
Clock to clock dynamic skew	$t_{CSKEW, PAIRD}$		.2	UI	See Figure 9-9

Table 9-6. Driver AC Timing Specifications - 1500Mbps Data Rate/750MHz Clock Rate

Characteristic	Symbol	Range		Unit	Notes
		Min	Max		
Differential output high voltage	V_{OHD}	200	540	mV	See Figure 9-4
Differential output low voltage	V_{OLD}	-540	-200	mV	See Figure 9-4
Unit interval		667	667	ps	Requires +/-100ppm long term frequency stability
Duty cycle of the clock output	DC	48	52	%	Measured at V_{OD}=0V
V_{OD} fall time, 20-80% of the peak to peak differential signal swing	t_{FALL}	.1		UI	
V_{OD} rise time, 20-80% of the peak to peak differential signal swing	t_{RISE}	.1		UI	
Data Valid	DV	.525		UI	Measured using the RapidIO Transmit Mask shown in Figure 9-4
Allowable static skew between any two data outputs within a 8 bit/9 bit group	t_{DPAIR}		.2	UI	See Figure 9-10
Allowable static skew of data outputs to associated clock	$t_{SKEW,PAIR}$	-.2	.2	UI	See Figure 9-8, Figure 9-10
Clock to clock static skew	$t_{CSKEW, PAIR}$		.15	UI	See Figure 9-9
Clock to clock dynamic skew	$t_{CSKEW, PAIRD}$		.2	UI	See Figure 9-9

Table 9-7. Driver AC Timing Specifications - 2000Mbps Data Rate/1000MHz Clock Rate

Characteristic	Symbol	Range		Unit	Notes
		Min	Max		
Differential output high voltage	V_{OHD}	200	540	mV	See Figure 9-4
Differential output low voltage	V_{OLD}	-540	-200	mV	See Figure 9-4
Unit interval		500	500	ps	Requires +/-100ppm long term frequency stability
Duty cycle of the clock output	DC	48	52	%	Measured at V_{OD}=0V
V_{OD} fall time, 20-80% of the peak to peak differential signal swing	t_{FALL}	.1		UI	
V_{OD} rise time, 20-80% of the peak to peak differential signal swing	t_{RISE}	.1		UI	
Data Valid	DV	.5		UI	Measured using the RapidIO Transmit Mask shown in Figure 9-4
Allowable static skew between any two data outputs within a 8 bit/9 bit group	t_{DPAIR}		.2	UI	See Figure 9-10
Allowable static skew of data outputs to associated clock	$t_{SKEW,PAIR}$	-.2	.2	UI	See Figure 9-8, Figure 9-10
Clock to clock static skew	$t_{CSKEW, PAIR}$		.2	UI	See Figure 9-9
Clock to clock dynamic skew	$t_{CSKEW, PAIRD}$		.2	UI	See Figure 9-9

The compliance of driver output signals TD[0-15] and TFRAME with their minimum Data Valid window (DV) specification shall be determined by generating an eye pattern for each of the data signals and comparing the eye pattern of each data signal with the RapidIO Transmit Mask shown in Figure 9-4. The value of X2 used to construct the mask shall be $(1 - DV_{min})/2$. A signal is compliant with the Data Valid window specification if and only if the Transmit Mask can be positioned on the signal's eye pattern such that the eye pattern falls entirely within the unshaded portion of the mask.

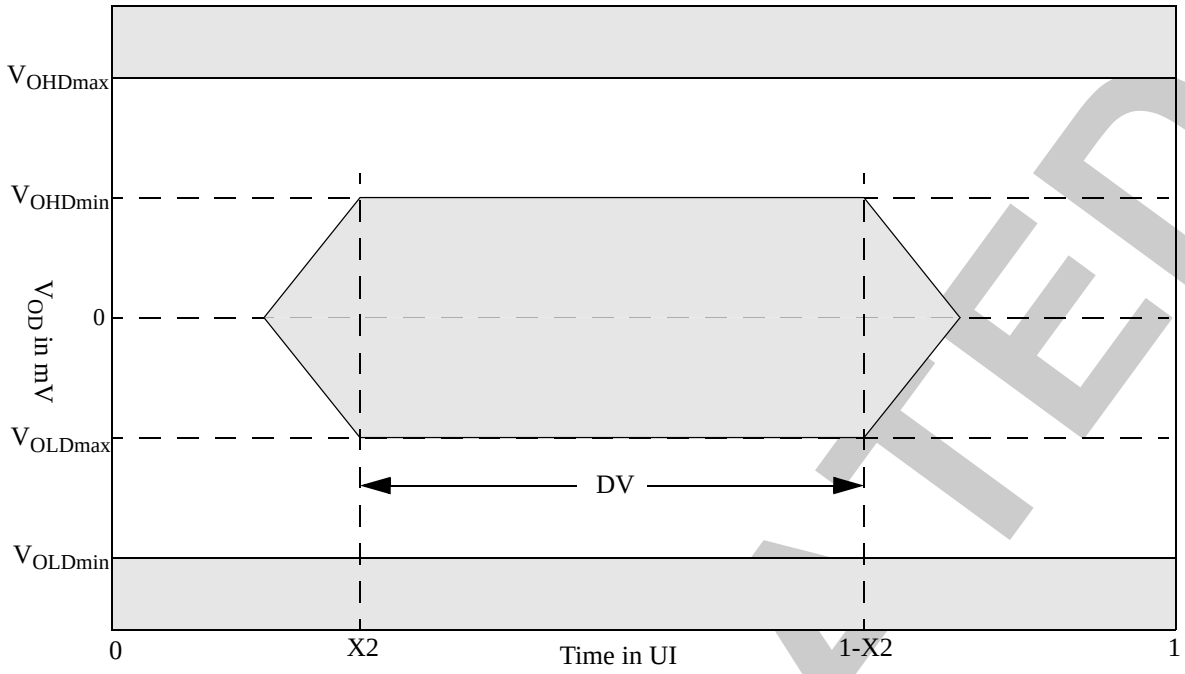


Figure 9-4. RapidIO Transmit Mask

The eye pattern for a data signal is generated by making a large number of recordings of the signal and then overlaying the recordings. The number of recordings used to generate the eye shall be large enough that further increasing the number of recordings used does not cause the resulting eye pattern to change from one that complies with the RapidIO Transmit Mask to one that does not. Each data signal in the interface shall be carrying random or pseudo-random data when the recordings are made. If pseudo-random data is used, the length of the pseudo-random sequence (repeat length) shall be long enough that increasing the length of the sequence does not cause the resulting eye pattern to change from one that complies with the RapidIO Transmit Mask to one that does not comply with the mask. The data carried by any given data signal in the interface may not be correlated with the data carried by any other data signal in the interface. The zero-crossings of the clock associated with a data signal shall be used as the timing reference for aligning the multiple recordings of the data signal when the recordings are overlaid.

While the method used to make the recordings and overlay them to form the eye pattern is not specified, the method used shall be demonstrably equivalent to the following method. The signal under test is repeatedly recorded with a digital oscilloscope in infinite persistence mode. Each recording is triggered by a zero-crossing of the clock associated with the data signal under test. Roughly half of the recordings are triggered by positive-going clock zero-crossings and roughly half are triggered by negative-going clock zero-crossings. Each recording is at least 1.9 UI in length (to ensure that at least one complete eye is formed) and begins 0.5 UI

before the trigger point (0.5 UI before the associated clock zero-crossing). Depending on the length of the individual recordings used to generate the eye pattern, one or more complete eyes will be formed. Regardless of the number of eyes, the eye whose center is immediately to the right of the trigger point is the eye used for compliance testing.

An example of an eye pattern generated using the above method with recordings 3 UI in length is shown in Figure 9-5. In this example, there is no skew between the signal under test and the associated clock used to trigger the recordings. If skew was present, the eye pattern would be shifted to the left or right relative to the oscilloscope trigger point.

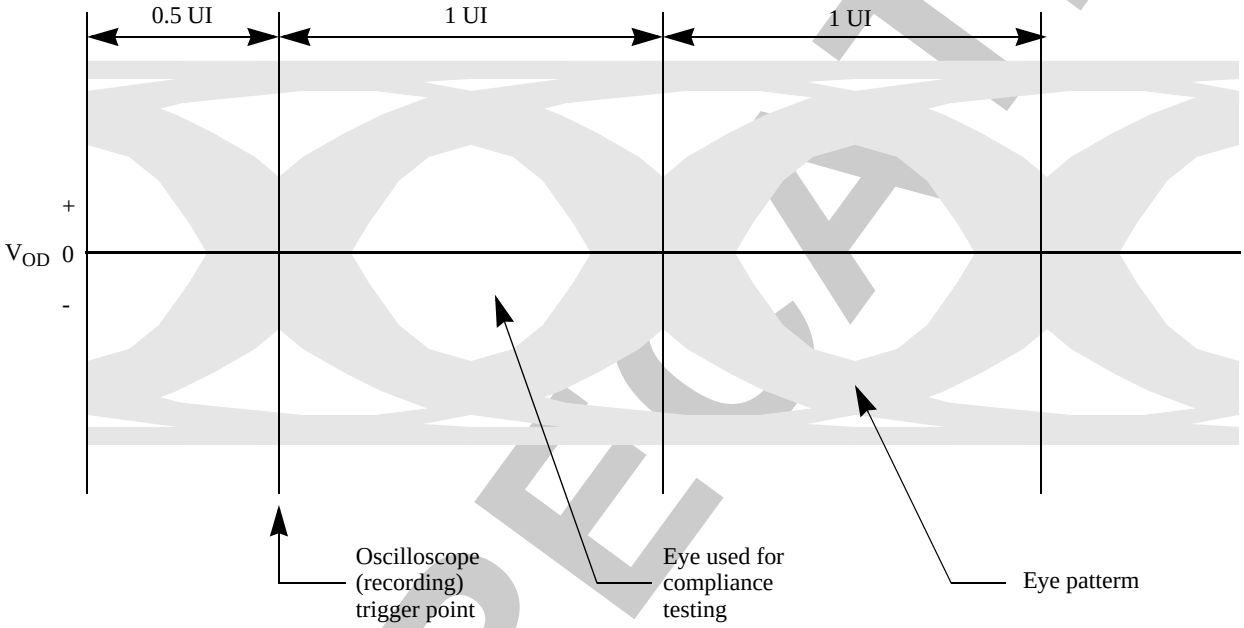


Figure 9-5. Example Driver Output Eye Pattern

9.4.3 Receiver Specifications

Receiver AC timing specifications are given in Table 9-8 through Table 9-12 below. A receiver shall comply with the specifications for each data rate/frequency for which operation of the receiver is specified. Unless otherwise specified, these specifications are subject to the following conditions.

The specifications apply over the supply voltage and ambient temperature ranges specified by the device vendor.

The specifications apply for any combination of data patterns on the data signals.

The specifications apply over the receiver common mode and differential input voltage ranges.

Clock specifications apply only to clock signals (CLK0 and, if present, CLK1).

Data specifications apply only to data signals (FRAME, D[0-7], and, if present, D[8-15]).

FRAME and D[0-7] are the data signals associated with CLK0, D[8-5] are the data signals associated with CLK1.

Table 9-8. Receiver AC Timing Specifications - 500Mbps Data Rate/250MHz Clock Rate

Characteristic	Symbol	Range		Unit	Notes
		Min	Max		
Duty cycle of the clock input	DC	47	53	%	Measured at V_{ID}=0V
Data Valid	DV	.54		UI	Measured using the RapidIO Receive Mask shown in Figure 9-6
Allowable static skew between any two data inputs within a 8 bit/9 bit group	t_{DPAIR}		.19	UI	See Figure 9-10
Allowable static skew of data inputs to associated clock	$t_{SKEW,PAIR}$	-.15	.15	UI	See Figure 9-8, Figure 9-10
Clock to clock static skew	$t_{CSKEW, PAIR}$		.14	UI	See Figure 9-9
Clock to clock dynamic skew	$t_{CSKEW, PAIRD}$		.3	UI	See Figure 9-9

Table 9-9. Receiver AC Timing Specifications - 750Mbps Data Rate/375MHz Clock Rate

Characteristic	Symbol	Range		Unit	Notes
		Min	Max		
Duty cycle of the clock input	DC	47	53	%	Measured at V_{ID}=0V
Data Valid	DV	.45		UI	Measured using the RapidIO Receive Mask shown in Figure 9-6

Table 9-9. Receiver AC Timing Specifications - 750Mbps Data Rate/375MHz Clock Rate

Characteristic	Symbol	Range		Unit	Notes
		Min	Max		
Allowable static skew between any two data inputs within a 8 bit/9 bit group	t_{DPAIR}		.3	UI	See Figure 9-10
Allowable static skew of data inputs to associated clock	$t_{SKEW,PAIR}$	-.2	.2	UI	See Figure 9-8, Figure 9-10
Clock to clock static skew	$t_{CSKEW, PAIR}$		.2	UI	See Figure 9-9
Clock to clock dynamic skew	$t_{CSKEW, PAIRD}$		.3	UI	See Figure 9-9

Table 9-10. Receiver AC Timing Specifications - 1000Mbps Data Rate/500MHz Clock Rate

Characteristic	Symbol	Range		Unit	Notes
		Min	Max		
Duty cycle of the clock input	DC	47	53	%	Measured at V_{ID}=0V
Data Valid	DV	.425		UI	Measured using the RapidIO Receive Mask shown in Figure 9-6
Allowable static skew between any two data inputs within a 8 bit/9 bit group	t_{DPAIR}		.3	UI	See Figure 9-10
Allowable static skew of data inputs to associated clock	$t_{SKEW,PAIR}$	-.2	.2	UI	See Figure 9-8, Figure 9-10
Clock to clock static skew	$t_{CSKEW, PAIR}$		.2	UI	See Figure 9-9
Clock to clock dynamic skew	$t_{CSKEW, PAIRD}$		.3	UI	See Figure 9-9

Table 9-11. Receiver AC Timing Specifications - 1500Mbps Data Rate/750MHz Clock Rate

Characteristic	Symbol	Range		Unit	Notes
		Min	Max		
Duty cycle of the clock input	DC	47	53	%	Measured at V_{ID}=0V
Data Valid	DV	.375		UI	Measured using the RapidIO Receive Mask shown in Figure 9-6
Allowable static skew between any two data inputs within a 8 bit/9 bit group	t_{DPAIR}		.4	UI	See Figure 9-10
Allowable static skew of data inputs to associated clock	$t_{SKEW,PAIR}$	-.25	.25	UI	See Figure 9-8, Figure 9-10
Clock to clock static skew	$t_{CSKEW, PAIR}$		.3	UI	See Figure 9-9
Clock to clock dynamic skew	$t_{CSKEW, PAIRD}$		.3	UI	See Figure 9-9

Table 9-12. Receiver AC Timing Specifications - 2000Mbps Data Rate/1000MHz Clock Rate

Characteristic	Symbol	Range		Unit	Notes
		Min	Max		
Duty cycle of the clock input	DC	47	53	%	Measured at V_{ID}=0V
Data Valid	DV	.35		UI	Measured using the RapidIO Receive Mask shown in Figure 9-6
Allowable static skew between any two data inputs within a 8 bit/9 bit group	t_{DPAIR}		.4	UI	See Figure 9-10
Allowable static skew of data inputs to associated clock	$t_{SKEW,PAIR}$	-.25	.25	UI	See Figure 9-8, Figure 9-10
Clock to clock static skew	$t_{CSKEW, PAIR}$		.3	UI	See Figure 9-9
Clock to clock dynamic skew	$t_{CSKEW, PAIRD}$		.3	UI	See Figure 9-9

The compliance of receiver input signals RD[0-15] and RFRAME with their minimum Data Valid window (DV) specification shall be determined by generating an eye pattern for each of the data signals and comparing the eye pattern of each data signal with the RapidIO Receive Mask shown in Figure 9-6. The value of X2 used to construct the mask shall be (1 - DV_{min})/2. The +/- 100mV minimum data valid and +/- 600mV maximum input voltage values are from the DC specification. A signal is compliant with the Data Valid window specification if and only if the Receive Mask can be positioned on the signal's eye pattern such that the eye pattern falls entirely within the unshaded portion of the mask.

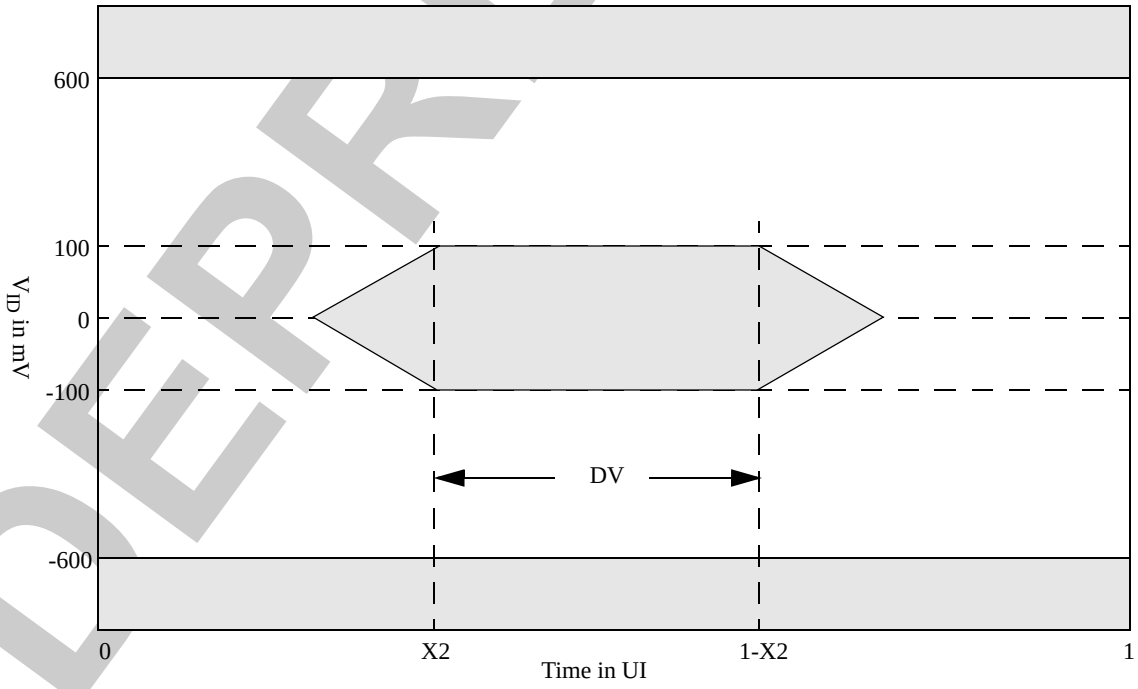


Figure 9-6. RapidIO Receive Mask

The eye pattern for a data signal is generated by making a large number of recordings of the signal and then overlaying the recordings. The number of recordings used to generate the eye shall be large enough that further increasing the number of recordings used does not cause the resulting eye pattern to change from one that complies with the RapidIO Receive Mask to one that does not. Each data signal in the interface shall be carrying random or pseudo-random data when the recordings are made. If pseudo-random data is used, the length of the pseudo-random sequence (repeat length) shall be long enough that increasing the length of the sequence does not cause the resulting eye pattern to change from one that complies with the RapidIO Receive Mask to one that does not comply with the mask. The data carried by any given data signal in the interface may not be correlated with the data carried by any other data signal in the interface. The zero-crossings of the clock associated with a data signal shall be used as the timing reference for aligning the multiple recordings of the data signal when the recordings are overlaid.

While the method used to make the recordings and overlay them to form the eye pattern is not specified, the method used shall be demonstrably equivalent to the following method. The signal under test is repeatedly recorded with a digital oscilloscope in infinite persistence mode. Each recording is triggered by a zero-crossing of the clock associated with the data signal under test. Roughly half of the recordings are triggered by positive-going clock zero-crossings and roughly half are triggered by negative-going clock zero-crossings. Each recording is at least 1.9 UI in length (to ensure that at least one complete eye is formed) and begins 0.5 UI before the trigger point (0.5 UI before the associated clock zero-crossing). Depending on the length of the individual recordings used to generate the eye pattern, one or more complete eyes will be formed. Regardless of the number of eyes, the eye whose center is immediately to the right of the trigger point is the eye used for compliance testing.

An example of an eye pattern generated using the above method with recordings 3 UI in length is shown in Figure 9-7. In this example, there is no skew between the signal under test and the associated clock used to trigger the recordings. If skew was present, the eye pattern would be shifted to the left or right relative to the oscilloscope trigger point.

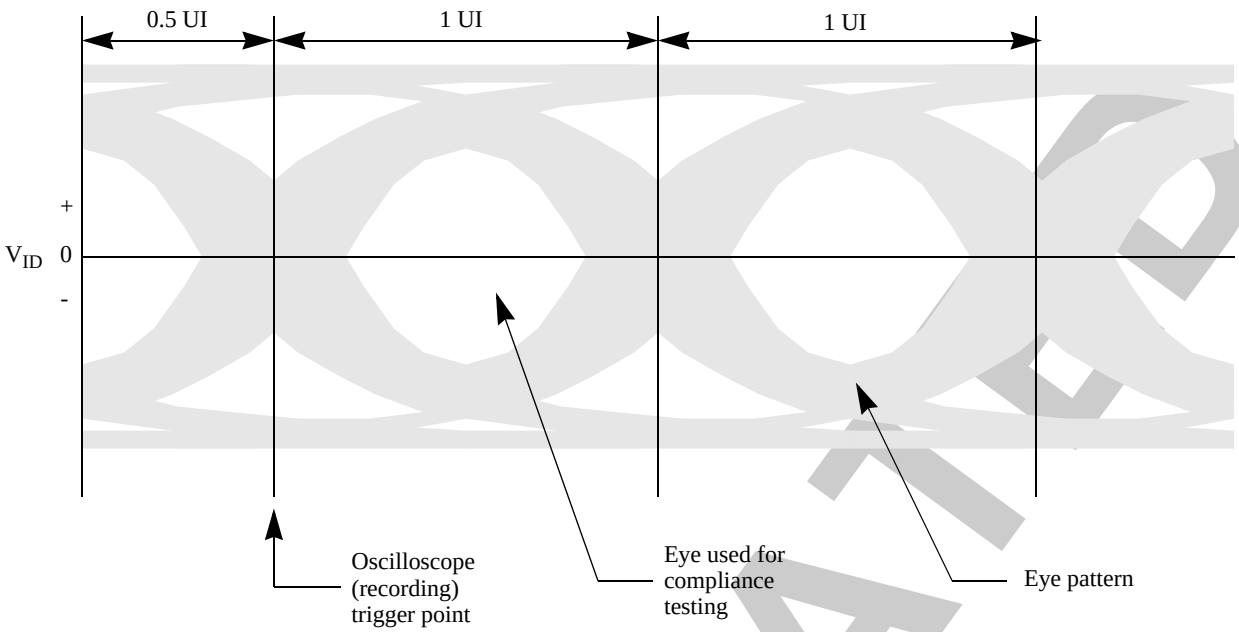


Figure 9-7. Example Receiver Input Eye Pattern

Figure 9-8 shows the definitions of the data to clock static skew parameter $t_{SKEW,PAIR}$ and the Data Valid window parameter DV. The data and frame bits are those that are associated with the clock. The figure applies for all zero-crossings of the clock. All of the signals are differential signals. V_D represents V_{OD} for the transmitter and V_{ID} for the receiver. The center of the eye is defined as the midpoint of the region in which the magnitude of the signal voltage is greater than or equal to the minimum DV voltage.

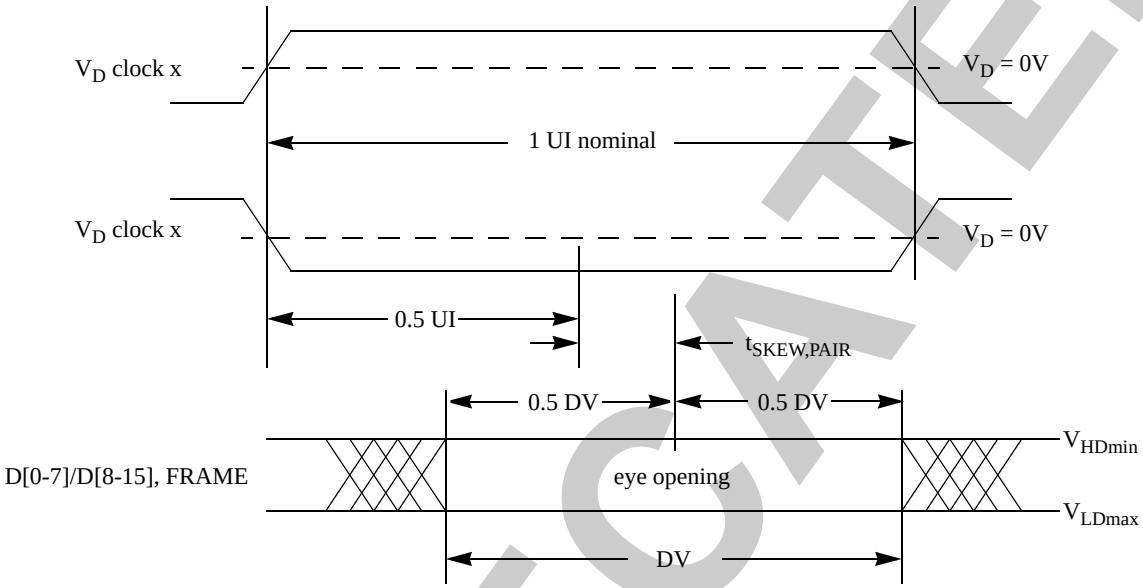


Figure 9-8. Data to Clock Skew

Figure 9-9 shows the definitions of the clock to clock static skew parameter $t_{CSKEW, PAIR}$ and the clock to clock dynamic skew parameter $t_{CSKEW, PAIRD}$. All of the signals shown are differential signals. V_D represents V_{OD} for the transmitter and V_{ID} for the receiver. These two parameters, $t_{CSKEW, PAIR}$ and $t_{CSKEW, PAIRD}$, only apply to 16 bit interfaces.

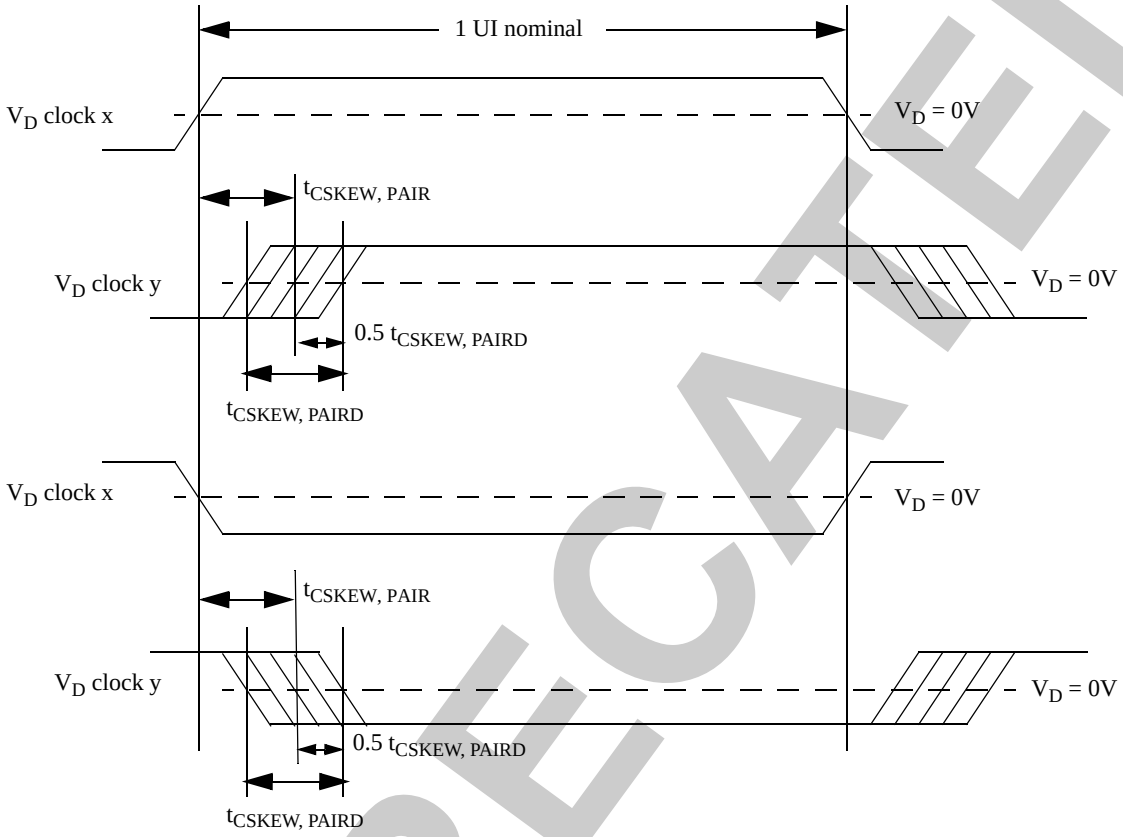


Figure 9-9. Clock to Clock Skew

Figure 9-10 shows the definition of the data to data static skew parameter t_{DPAIR} and how the skew parameters are applied.

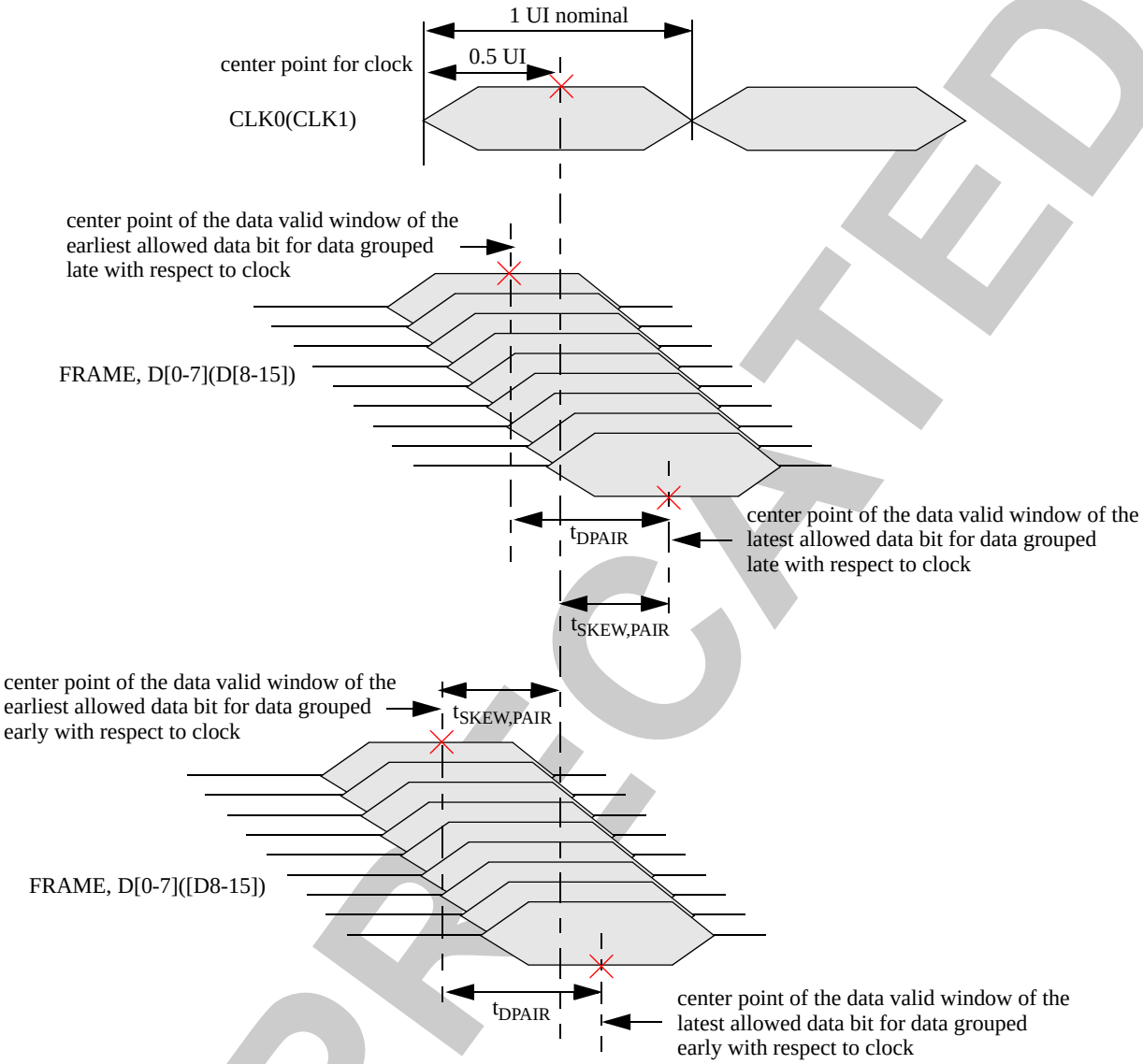


Figure 9-10. Static Skew Diagram

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 4: 8/16 LP-LVDS 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 Link Initialization and Maintenance Mechanism

This section contains the link training and initialization state machine referred to in Section 3.7.1.1, “Sampling Window Alignment.” Training takes place in two circumstances; when coming out of reset and after the loss of reliable input port sampling during system operation.

Link initialization and maintenance actually requires two inter-dependent state machines in order to operate, one associated with the input port and the other with the output port. The two state machines work together to complete the link training. The state machines are intended for a device with an 8-bit port or a device with a 16-bit port. The port can only transition from the “Port Uninitialized” status to the “Port OK” status in the Port n Error and Status CSR when both halves of the state machine are in their OK state.

A.2.1 Input port training state machine

Figure A-1 illustrates the input port training state machine. Error conditions are only detectable while in the “OK” states (OK and OK_maint_trn). The optional OK_maint_trn state, shaded in Figure A-1, is used to adjust the device input port sampling circuitry during system operation.

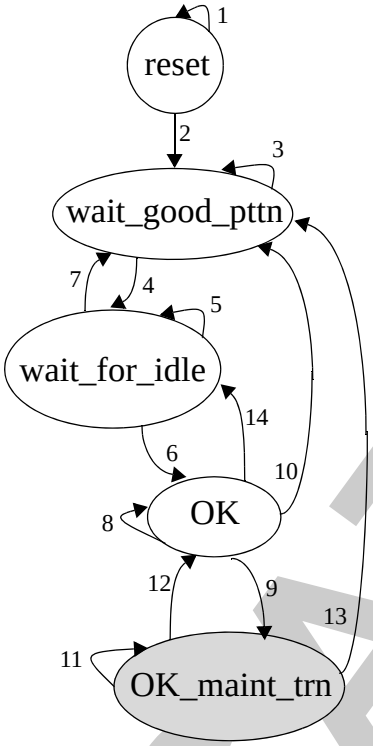


Figure A-1. Input port training state machine

Table A-1 describes the state transition arcs for Figure A-1.

Table A-1. Input port training state machine transition table

Arc	Current State	Next state	cause	Comments
1	reset	reset	Start training condition not met.	Remain in the reset state until the start training condition is met. Typically, this is after reset has been applied to the device and all other necessary initialization activity has completed.
2	reset	wait_good_pttn	Start training condition met.	This state is entered after all initialization activity has completed for the device.
3	wait_good_pttn	wait_good_pttn	Wait for the sampling circuitry to indicate that it is calibrated.	Remain in this state until the sampling circuitry is calibrated.
4	wait_good_pttn	wait_for_idle	Sampling circuitry is calibrated and the defined training pattern has been received.	Upon recognizing the defined training pattern, a 16-bit port can decide whether it's output port needs to be downgraded to drive in 8-bit mode. Request the output port to start sending idle control symbols.
5	wait_for_idle	wait_for_idle	Remain in this state until an exit condition occurs.	In this state, only training patterns and link-request/send-training control symbols are legal.

Table A-1. Input port training state machine transition table (Continued)

Arc	Current State	Next state	cause	Comments
6	wait_for_idle	OK	Idle control symbol has been received	This transition indicates that the input port is ready to start receiving packets and other control symbols. Due to input/link delays the input port may see an extra idle/training pattern sequence when finishing the alignment sequence.
7	wait_for_idle	wait_good_pttn	The input port receives something besides a training pattern, idle, or link-request/send-training control symbol, or the sampling circuitry is no longer calibrated.	Receiving something unexpected or when the sampling circuitry is no longer able to reliably sample the device pins causes both the input port and output port to start restart the training sequence.
8	OK	OK	Sampling circuitry remains calibrated and is not drifting.	This is a functional state in which packets and control symbols can be accepted. Errors are also reported in this state.
9	OK	ready_maint_trn	Sampling circuitry drift.	This transition takes place when the sampling circuitry can still reliably sample the device pins, but adjustment is required to prevent eventual loss of calibration.
10	OK	wait_good_pttn	Sampling circuitry is no longer calibrated.	Both the input port and output port restart the training sequence when the sampling circuitry is no longer able to reliably sample the device pins. This error invokes the error recovery algorithm when the OK state is re-entered to attempt to recover possible lost data.
11	OK_maint_trn	OK_maint_trn	Training patterns have not been received, and the sampling circuitry is still calibrated.	This is a functional state in which packets and control symbols can be accepted. Errors are also reported in this state. In this state, the device adjusts the sampling circuitry when the training patterns are received.
12	OK_maint_trn	OK	The complete sequence of 256 training patterns followed by an idle has been received and the sampling circuitry is still calibrated.	Sampling circuitry has been adjusted.
13	OK_maint_trn	wait_good_pttn	Sampling circuitry is no longer calibrated.	Both the input port and output port restart the alignment sequence when the sampling circuitry is no longer able to reliably sample the device pins. This error invokes the error recovery algorithm when the ready state is re-entered to attempt to recover possible lost data.
14	OK	wait_for_idle	The input port receives a link-request/send-training control symbol immediately followed by a training pattern	The attached device is no longer calibrated and has re-started the alignment sequence.

A.2.2 Output port training state machine

Figure A-2 illustrates the output port training state machine. Packets can only be transmitted when both the input port and output port are in their "OK" states (OK and OK_maint_trn for the input port, and OK, OK_send_trn_req and OK_send_trn_pttn for the output port). The optional OK_send_trn state, lightly shaded in Figure A-2, is used to adjust the device input port sampling circuitry during system operation, and is associated with the OK_maint_trn state in the input port state machine.

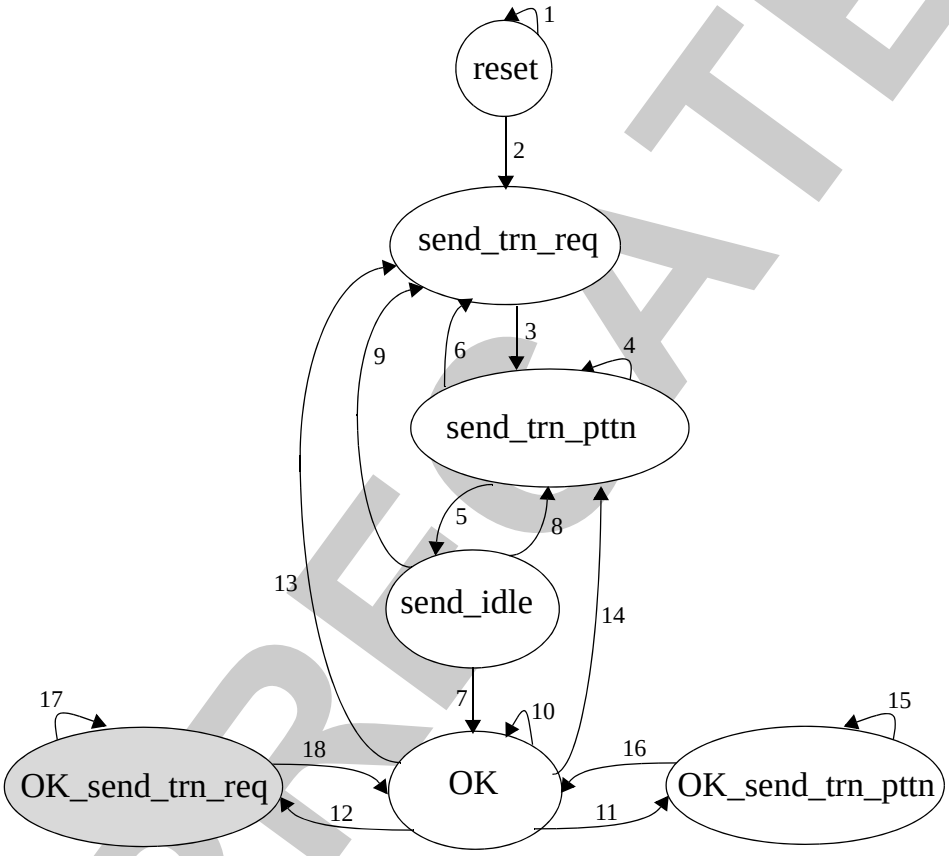


Figure A-2. Output port training state machine

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

Table A-2. Output port training state machine transition table

Arc	Current State	Next state	cause	Comments
1	reset	reset	Start training condition not met.	Remain in the reset state until the start training condition is met. Typically, this is after reset has been applied to the device and all other necessary initialization activity has completed.
2	reset	send_trn_req	Start training condition met.	This state is entered after all initialization activity has completed for the device. The output port will send a link-request/send-training control symbol
3	send_trn_req	send_trn_pttn	Unconditional transition.	The output port will send 256 iterations of the training pattern
4	send_trn_pttn	send_trn_pttn	The 256 iterations of the training pattern is not completed.	The input port is waiting to calibrate and receive the defined training pattern. The output port is sending training patterns.
5	send_trn_pttn	send_idles	The 256 iterations of the training pattern is completed and the input port has requested to send idle control symbols.	The input port sampling circuitry is calibrated. In the send_idle state, one idle control symbol is sent out on the output port.
6	send_trn_pttn	send_trn_req	The 256 iterations of the training pattern are completed but the input port has not requested to send idle control symbols.	Remain in the send_trn_req - send_trn_pttn loop until the input port sampling circuitry is calibrated and the input port recognizes the defined training pattern and then requests to send idle control symbols. A link-request/send-training control symbol is sent out in state send_trn_req.
7	send_idle	OK	The input port is in state OK	Ready to start sending packets and any control symbol.
8	send_idle	send_trn_pttn	The input port is not in OK or wait_good_pttn state	The output port will send 256 iterations of the of the training pattern
9	send_idle	send_trn_req	The input port is in state wait_good_pttn	Transition to send_trn_req and start over.
10	OK	OK	A link-request/send-training is not received on the input port and the input port does not ask for a reset to the beginning of the training sequence.	This is a functional state in which packets and control symbols are transmitted. Errors are detected and reported in this state.
11	OK	OK_send_trn_pttn	link-request/send-training followed by a packet or control symbol is received on the input port.	This transition occurs when in the OK state and a maintenance training request is received from the attached device.
12	OK	OK_send_trn_req	The input port wants the attached device to send 256 iterations of the training pattern.	This transition occurs when in the OK state and input port sampling circuitry needs to be adjusted, and is associated with the optional input port OK_maint_trn state.

Table A-2. Output port training state machine transition table (Continued)

Arc	Current State	Next state	cause	Comments
13	OK	send_trn_req	The input port asks for a reset to the beginning of the training sequence.	Transition to send_trn_req and start over. This occurs when the sampling circuitry is no longer able to reliably sample the device pins.
14	OK	send_trn_pttn	A link-request/send-training followed by the training pattern is received on the input port.	The attached device has lost synchronization.
15	OK_send_trn_pttn	OK_send_trn_pttn	The 256 iterations of the training pattern is not completed.	The output port is sending training patterns. Errors are detected and reported in this state. Must send at least one idle control symbol after the 256 iterations.
16	OK_send_trn_pttn	OK	The 256 iterations of the training pattern are completed and followed by at least one idle control symbol.	This is a normal operating case where the attached device requested that we send training patterns yet it maintained alignment.
17	OK_send_trn_req	OK_send_trn_req	Waiting to send the link-request/send-training	Might have to wait for the end of the current packet because link-request control symbols can not be embedded. Errors are detected and reported in this state.
18	OK_send_trn_req	OK	link-request/send-training sent out on the output port as requested by the input port.	Input port is requesting training patterns from the other end to adjust its sampling circuitry.

A.3 Packet Retry Mechanism

This section contains the example packet retry mechanism state machine referred to in Section 2.3.3, “Transaction and Packet Delivery”.

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.3.1 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-3.

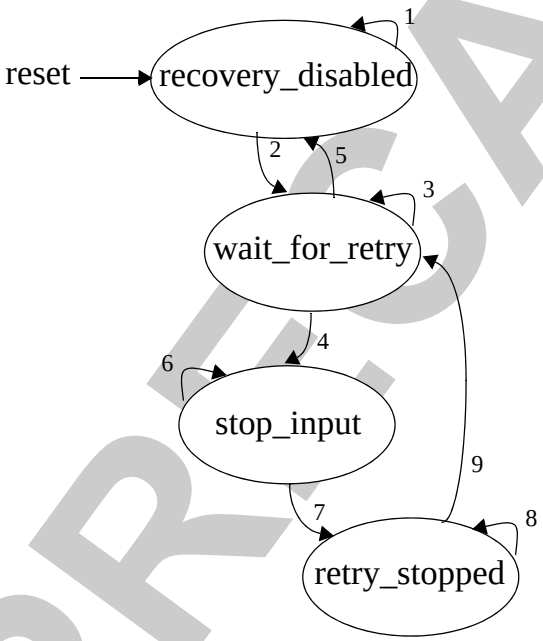


Figure A-3. Input port retry recovery state machine

Table A-3 describes the state transition arcs for Figure A-3. The states referenced in the comments in quotes are the RapidIO 8/16 LP-LVDS defined status states, not states in this state machine.

Table A-3. 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 training 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 ready" state and set the "Input Retry-stopped" state.
7	stop_input	retry_stopped	Input port stop activity is complete.	
8	retry_stopped	retry_stopped	Remain in this state until a restart-from-retry or 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 restart-from-error control symbol or an input port error is encountered.	The restart-from-error control symbol is a link-request/input-status control symbol. Clear the "Input Retry-stopped" state and set the "Port ready" state. An input port error shall cause a clean transition between the retry recovery state machine and the error recovery state machine.

A.3.2 Output port retry recovery state machine

On receipt of an error-free packet-retry acknowledge control symbol, the attached output port follows the behavior shown in 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.

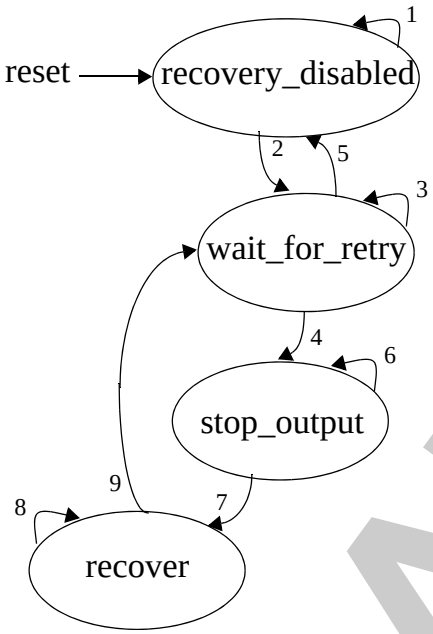


Figure A-4. Output port retry recovery state machine

Table A-4 describes the state transition arcs for Figure A-4.

Table A-4. 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 training 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 ready" state, set the "Output Retry-stopped" state, and stop transmitting new packets.
7	stop_output	recover	Output port stop procedure is complete.	

Table A-4. Output port retry recovery state machine transition table (Continued)

Arc	Current State	Next state	cause	Comments
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 re-transmitted. 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 ready" 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.
9	recover	wait_for_retry	Internal recovery procedure is complete.	Re-transmission has started, so return to the wait_for_retry state to wait for the next packet-retry control symbol.

A.4 Error Recovery

This section contains the error recovery state machine referred to in Section 2.4.5, “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.4.1 Input port error recovery state machine

There are a variety of recoverable error types described in detail in Section 2.4.5, “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 where the second 16 bits are not an inversion of the first 16 bits. The recovery state machine for the input port of a RapidIO link is shown in Figure A-5.

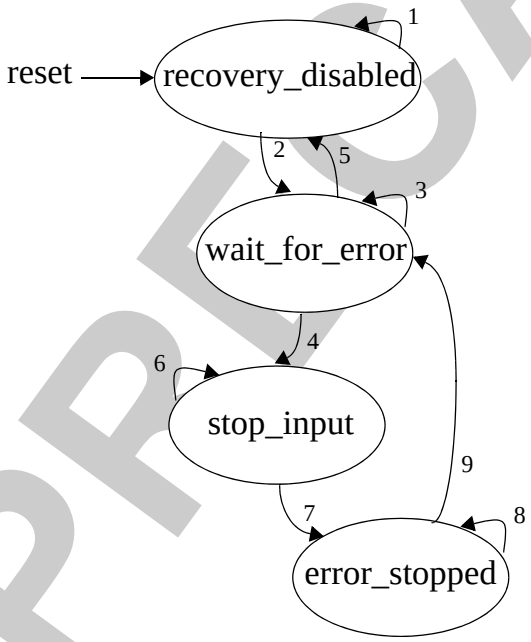


Figure A-5. Input port error recovery state machine

Table A-5 describes the state transition arcs for Figure A-5. 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-5. 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 training 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 ready" 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 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 restart-from-error control symbol.	The restart-from-error control symbol is a link-request/input-status control symbol. Clear the "Input Error-stopped" state and set the "Port ready" state, which will put the input port back in normal operation.

A.4.2 Output port error recovery state machine

The second recoverable group of errors described in Section 2.4.5, "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. Another example is the receipt of a link-request/send-training control symbol, which should cause the error recovery procedure to be followed after responding to the

request. The state machine for the output port is shown in Figure A-6.

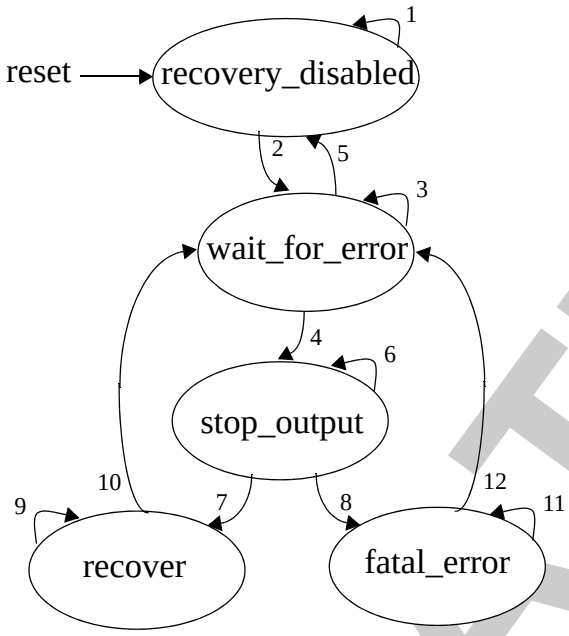


Figure A-6. Output port error recovery state machine

Table A-6 describes the state transition arcs for Figure A-6. 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-6. 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 training 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-6. 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 ready” 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 timeout several times before regarding a timeout 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 ackIDs are 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 re-transmitted. 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 ready” state to restart the output port.
10	recover	wait_for_error	The internal recovery procedure is complete.	Re-transmission (if any was necessary) has started, so return to the wait_for_error state to wait for the next error.

Table A-6. 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 even though the output port is shut off.

Blank page

Glossary of Terms and Abbreviations

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

A

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.

C

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

CCITT. Consultive Communication for International Telegraph and Telephone.

Command and status registers (CSRs). A set of registers that allows 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 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 an end point processing element connected to the RapidIO interconnect.

Direct Memory Access (**DMA**). The process of accessing memory in a device by specifying the memory address directly.

DLL. Delay lock loop.

Doorbell. A port on a device that is capable of generating an interrupt to a processor.

Double-data-rate clock. A data reference signal that indicates new valid data on both low-to-high and high-to-low transitions of the clock.

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.

EOP. End of packet.

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

F

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

First symbol. The leading 16 bits of a packet.

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.

Host. A processing element responsible for exploring and initializing all or a portion of a RapidIO based system.

I

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

I/O. Input-output.

L

LVDS. Low voltage differential signaling.

M

Multicast. The concept of sending a packet to more than one processing elements in a system.

N

NRZ signal. No return to zero signal.

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.

PCB. Printed circuit board.

PLL. Phase lock loop.

Port-write. An address-less maintenance 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

SECDED. Single error correction, double error detection.

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

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.

Symbol. A 16-bit quantity.

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.