Parallel RapidIO Physical Layer Interface IP Core

User's Guide

October 2005
ipug37_02.0

Introduction

RapidIO is a high performance, low pin count, packet switched, full duplex, system level interconnect architecture. The architecture addresses the need for faster bus speeds in an intra-system interconnect for microprocessors, digital signal processors, communications, and network processors. It also offers error management and provides a well defined architecture for recovering from, and reporting, transmission errors.

RapidIO systems contain endpoint and switch processing elements. The RapidIO interconnect architecture is partitioned into a layered hierarchy of specifications which includes Logical, Common Transport and Physical layers. The Logical layer defines the operations and transactions by which endpoint processing elements communicate with each other. The Common Transport layer defines how the transactions are routed from one endpoint processing element to another through switch processing elements. The Physical layer defines the interface between two devices and the packet transport mechanism, flow control and electrical characteristics. Currently there are two defined Physical layer specifications, an 8-bit or 16-bit wide parallel specification and a 1-lane (1x) or 4-lane (4x) serial specification.

This user's guide describes Lattice's Parallel RapidIO Physical Layer specification and interface.

The Parallel RapidIO Physical Layer core comes with the following documentation and files:

- Data sheet
- User's guide
- Lattice evaluation gate level netlist
- Simulation model for evaluation
- Core instantiation template
- Testbench and testbench coding template

The following experience is recommended for the user to implement a design using this IP core:

- Familiarity with the LatticeECP™/LatticeEC™ FPGA architecture
- Familiarity with simulation, synthesis and Lattice ispLEVER® tools
- Knowledge of the *RapidIO Interconnect Specification* Rev 1.2, June 2002

Features

- Supports 8/16-bit LP-LVDS protocol
- 500Mbps data rate (250MHz clock) per LVDS pin pair with 8-bit wide PHY data
- 250Mbps data rate (125MHz clock) per LVDS pin pair with 16-bit wide PHY data
- Peak performance of 4Gbps throughput at receive side, 4Gbps at transmit side
- Supports automatic detection of an 8/16-bit RapidIO port
- Supports throttle based flow control
- Link validation is supported in hardware
- Compliant with the *RapidIO Interconnect Specification* Rev 1.2
- Simple user interface for easy integration into user logic
- Targets LatticeECP/EC FPGAs

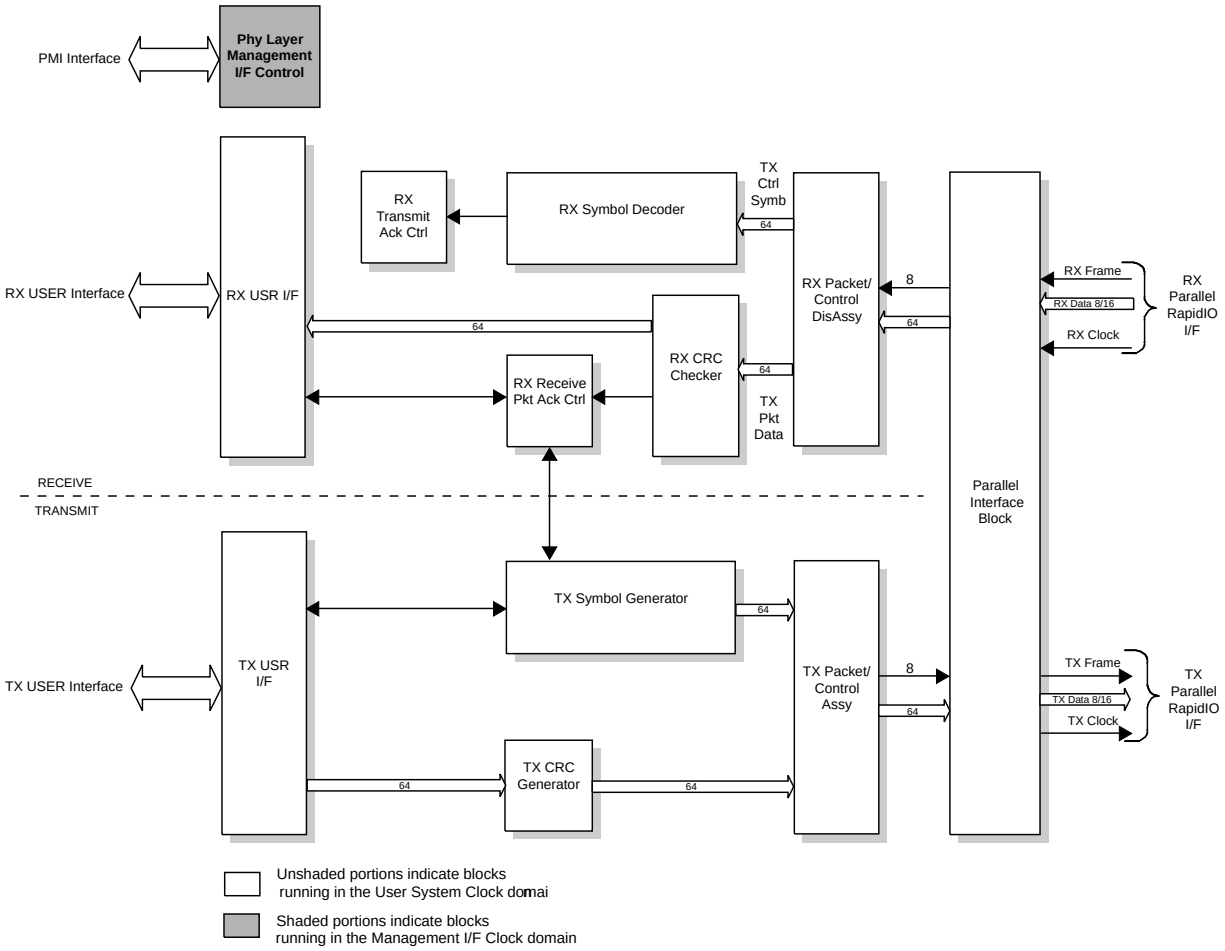
General Description

The Parallel RapidIO core supports the physical layer specification as defined in the *RapidIO Specification* Rev 1.2. The Parallel RapidIO Physical layer defines a protocol for packet delivery between parallel RapidIO devices and other devices; including packet transmission, flow control, error management and link maintenance protocols. The core supports an 8/16-bit bus running at a maximum of 500Mbps for 8-bit wide PHY data, and 250Mbps for 16-bit wide PHY data. This bus performance is available on each LVDS pin pair, giving a total peak performance of 4Gbps at Received Side, 4Gbps at Transmit side.

This parallel RapidIO core is optimized to support the LatticeECP/EC family of FPGAs. For more information on Lattice products, refer to the Lattice website at www.latticesemi.com.

Block Diagram

Figure 1. Physical Layer Block Diagram



Functional Description

Figure 1 shows the block diagram of the Parallel RapidIO Physical Layer core. Data from the user, which is received through the TX User Interface, is first passed through a CRC (Cyclic Redundancy Code) generation block to the TX Packet/Control symbol assembly block. The control symbols for this block are received from the TX Symbol Generator. The assembled data which contains both packet and control symbols are sent to the transmit side of the Parallel Interface block.

The incoming data from the RX Parallel RapidIO Interface are processed by the receive side of the Parallel Interface block. Then the data are passed to the RX Packet and Control Disassembly block. This block separates the data packets from the control symbols. The data packets are sent to the RX CRC Checker and the control symbols are sent to the RX Symbol Decoder.

The decoded control symbols are passed to the RX Transmit Ack Ctrl and RX Receive Packet Ack Ctrl blocks, which take the appropriate actions. Finally, the data from the RX CRC Checker is sent out to the user through the RX User Interface.

PHY Layer Management Interface (PMI) Block

This block contains the Command and Status Registers (CSRs) that allow the user to configure and read the capabilities, configuration and status of the input and output ports. For more detailed information about registers, refer to the Registers Description section of this document.

Transmit Physical Layer Block

TX User Interface

The TX User Interface generates all the control signals necessary to interface to user logic or the Logical and Transport layers of the RapidIO stack. The main function of this block is to indicate that the TX PHY is ready to receive data from the user. It also controls the discard signal `tx_riop_discard`, which is used to discard a packet being sent to the core.

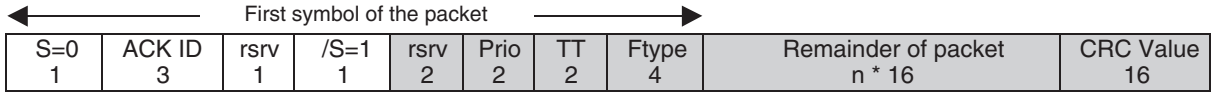
This block also generates the `tx_next_ackid` signal [ACK ID = 0 to 7], which indicates that the data associated with this ACK ID has to be sent. Since the user buffer can only store up to eight packets, this block generates a `tx_release_ackid` signal [ACK ID = 0 to 7] indicating that the packet designated by `tx_release_ackid` was sent and received successfully. The block interacts with the ACK Queue Management block to decide the next ACK ID value.

A detailed description of the operation of each signal in this interface is explained in the User Interface section of this document.

TX CRC Generator

This block generates the 16-bit CRC using the CCITT CRC-16 polynomial $X^{16}+X^{12}+X^5+1$. The CRC is generated over all of the packet header and the entire data payload, except for the first six bits of the added physical layer fields (which are treated as logical 0s) as shown below. The initial value of the CRC is 0xFFFF, or all logic 1s. For the CRC calculation, the first six bits are assumed to be logic 0. This structure allows the ACK ID to be changed on a link-by-link basis as the packet is transported, without requiring the CRC to be recomputed for each link.

Figure 2. CRC Generator



Notes:
1. n = number of transport and logical data fields.
2. The TX CRC generator is similar in fuctionality as the RX CRC Checker
3. For more information on the field format above, refer to *RapidIO Interconnect Specification,* section 1.2.1 titled "Flow Control Fields Format".

For a packet that has less than or equal 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 at the end of the packet.

For packets longer than 80 bytes (including physical layer 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 the RapidIO RX side to regard the embedded CRC value as 2 bytes of packet payload for CRC checking purposes. 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 post appended to the packet. The pad of logic 0s allows the CRC check to always be aligned at the 32-bit boundary.

TX Symbol Generator and Control Symbols
This block generates the appropriate control symbols based on the control symbol generation request and writes them into the Symbol Queue FIFO.

This block stores the control symbols that are to be sent out in a synchronous FIFO. All the control symbols are aligned to 32 bits with the last 16 bits as a bit-wise inverse of the first 16. Since the data width is 64 bits, and each control symbol is 32 bits, it can send up to two control symbols on every clock. Hence the FIFO stores up to two control symbols in one location. Outlined below is a list and description of the different Acknowledgement and Packet Control Symbols that are used in Parallel RapidIO IP core:

- **Packet-Accepted Control Symbol** indicates that the receiving device has taken responsibility for sending the packet to its final destination and that the 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.
- **Packet-Retry Control Symbol** indicates that the receiving device was not able to accept the packet due to a temporary resource conflict, such as insufficient buffering, and the sender should retransmit the packet.
- **Packet-Not-Accepted Control Symbol** indicates to the sender that the receiving port did not accept the packet. The cause field indicates the reason for not accepting 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. This will avoid wasting bandwidth by transmitting the entire rejected packet.
- **Stomp Control Symbol** is used to cancel a partially transmitted packet. Sending a stomp control signal will terminate packet transmission immediately.
- **Restart-From-Retry Control Symbol** is generated in response to a packet-not-accepted control symbol reception. It is used to enable the receiver to start accepting packets after the receiver has retried a packet.
- **Throttle Control Symbol** specifies the number of aligned pacing idles that the sender inserts in a packet. This symbol is generated by the TX Throttle logic.
- **End-of-Packet Control Symbol** is used to mark the end of the packet.

Another type of control symbol is Link Control Symbol. This is different from Acknowledgement and Packet Control Symbols since Link Control symbols cannot be embedded in a packet. There are three types of Link Control Symbols:

- **Send-training Control Symbol** can be transmitted on three occasions. When coming out of reset, after the loss of reliable input port sampling during system operation (loss of the lock signal from I/O blocks) or from the user signal.
- **Reset Control Symbol** is sent upon user request. Four reset commands are sent in a row without any other intervening packets or control symbols, except idle control symbols. This prevents spurious reset commands resetting a device inadvertently.
- **Input-status (Request) Control Symbol** is sent to get the ACK ID value and the input buffer status expected by the receiver.

For more detailed information on control symbols, refer to *RapidIO Interconnect Specification* Part IV: Physical Layer 8/16 LP-LVDS Specification.

TX Packet/Control Assembly

The TX Packet/Control Assembly block gets packet data from the user, appends physical layer header information within the relevant physical fields and sends the data to the Parallel Interface block. It also adds control symbols received from the TX Symbol Generator block between packets. The following formats show data request and response.

Figure 3. Request Packet Format

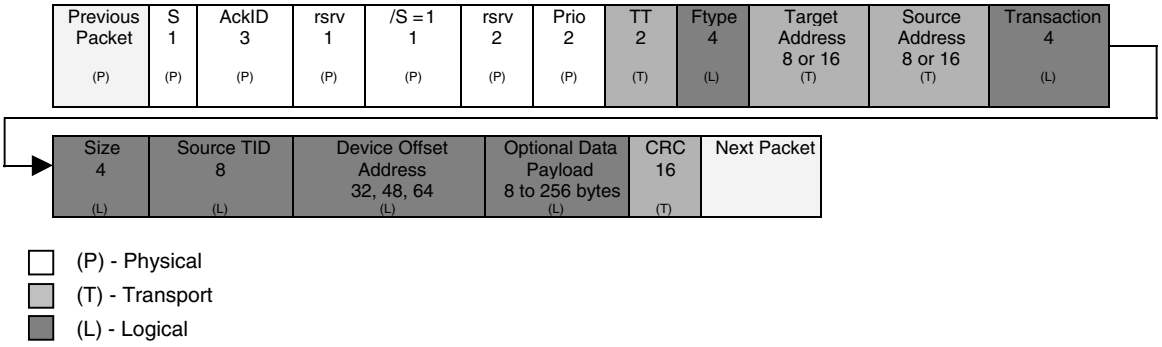
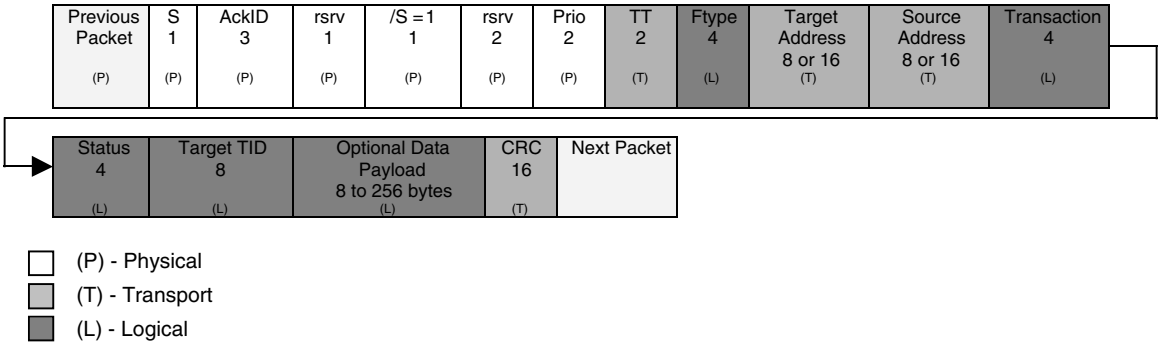
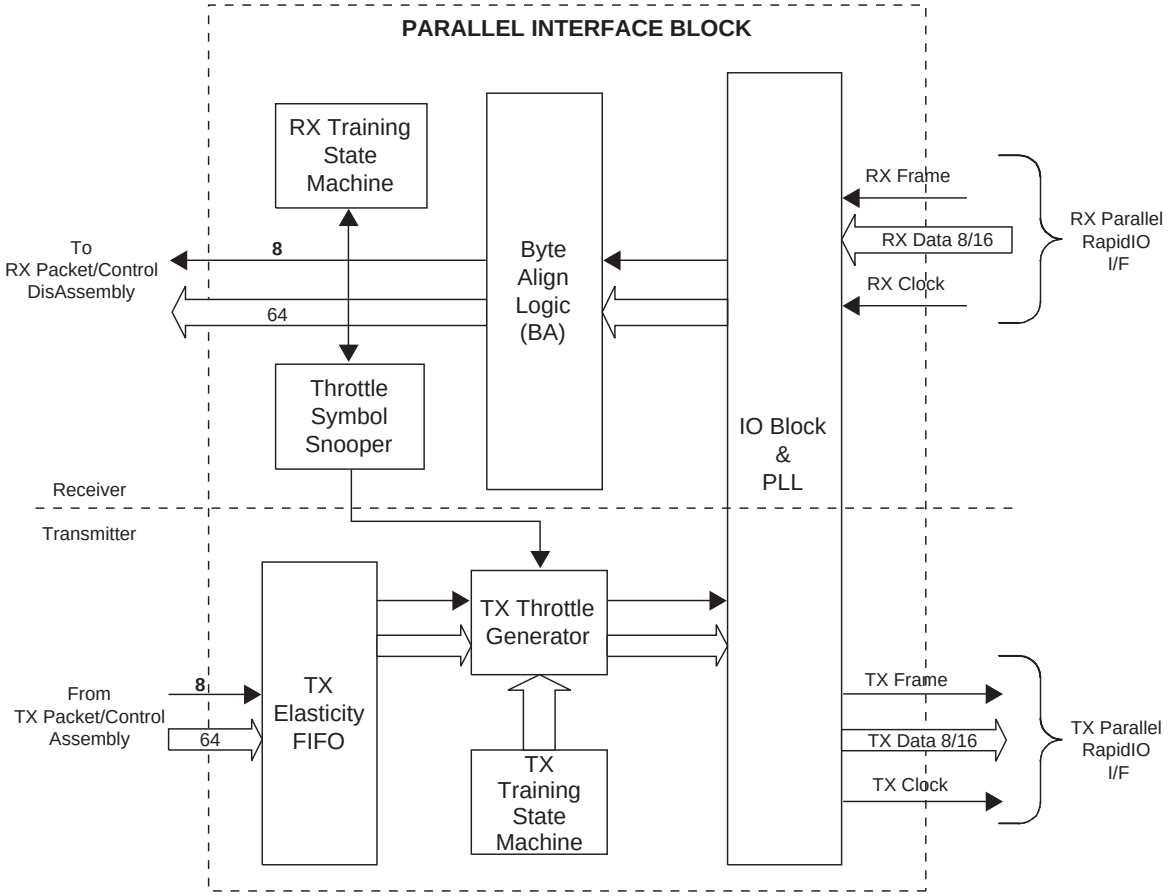


Figure 4. Response Packet Format



Note: These two kinds of packets are handled in the same manner. There is no difference between these two types of packets as far as physical layer is concerned.

Figure 5. Parallel Interface Block



Parallel Interface Block – TX Side

TX Elasticity FIFO

The TX elasticity asynchronous FIFO serves two purposes. First, it is used to translate data from the TX user clock to the RX user clock domain. Second, it provides some elasticity in case there are any differences between the TX user clock and the RX user clock. This is done by telling the TX user interface to stop sending data when the FIFO is getting full and to send idle control symbols out when FIFO has become empty. (Note that the RX user clock is derived from the RX Clock received on the Parallel RapidIO Interface).

TX Throttle Generator

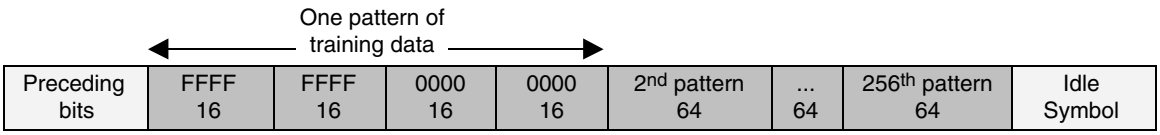
The receiver of a packet may request that the sender insert pacing idle control symbols within its data stream by sending a throttle control symbol, specifying the number of aligned pacing idle control symbols to be inserted in the packet. So whenever the RX throttle symbol snooper finds a throttle control, the TX throttle logic inserts idle symbols (aligned pacing idle control symbols). If additional delay is needed, the receiver can send another throttle control symbol to request the insertion of more idle symbols in the packet.

TX Training State Machine (TTS)

The RX Symbol Decoder block decodes send-training symbols and requests the transmitter to suspend normal operation and begin transmitting a special training pattern. The TX Training State Machine transmits a total of 256 iterations of the training pattern followed by at least one idle control symbol and then resumes operation. The send-training symbol does not generate a link-response control symbol (ACK).

The send-training symbol is issued at power up, reset and can also be issued at the user's request.

Figure 6. TX or RX Training Pattern Format



For more detailed information on the training state machine, refer to Appendix A of the *RapidIO Interconnect Specification,* Part IV: Physical Layer 8/16 LP-LVDS Specification.

Parallel Interface Block – RX Side

RX IO Logic Block (I/O)
This block takes the RapidIO physical data bus (8/16 bits wide), frame signal and receive clock and converts them into 64 bits of data and 8 bits of frame data. Here the data is captured (both edges) using the incoming RapidIO line clock and transferred to 1/4 clock (for 8-bit data) or to 1/2 clock (for 16-bit data). The internal clock is derived from the input RapidIO clock.

Once bit alignment is achieved, a lock signal is provided. The bit alignment logic will guarantee up to ±1 bit of data in each byte of the 64-bit data word. This means that a few bits in each byte of the 32-bit data word could belong to the previous byte or the following byte. This anomaly is corrected in the RX byte align logic block during training.

The latency of this logic is eight external RX clock cycles for an 8-bit bus, and four clock cycles for 16-bit bus.

RX Byte Align Logic (BA)
This block takes the received data stream and aligns the bytes to the correct 32-bit boundaries. This byte alignment takes place only when the training pattern is being received. The latency of this logic is two internal clock cycles.

RX Training State Machine (RTS)
The RX training state machine recognizes the training pattern and communicates information about the training pattern to the RX byte align block. The RX byte align block continues to correct the data until the training state machine goes to the "OK" state. See Figure 6 for the training pattern format.

Training can take place in three circumstances:

1. When the RapidIO core is coming out of reset

2. After the loss of the lock signal from I/O blocks

3. When there is a training request by the user

Throttle Symbol Snooper
The Throttle Symbol Snooper looks for Throttle Control Symbols in the 32-bit data words. Once it detects a throttle symbol, it requests that the TX throttle generation logic inserts 'n' number of IDLE control symbols in the frame. The receiver of a packet may request that the sender insert idle control symbols on its behalf, by sending a throttle control symbol specifying the number of idle control symbols to insert. The packet sender then inserts the number of aligned idles into the packet stream. If additional delay is needed, the receiver can send another throttle control symbol.

Receive Physical Layer Block

RX Packet/Control Disassembly (RPD)
The RX Packet and Control Disassembly block separates the data packets and control symbols from the received data. It sends the separated data to the RX CRC Checker block while it sends the RX control symbols to the RX Symbol Decoder block. This block also filters the idle control symbols received.

The received packet data from the RX Packet and Control Disassembly block contains gaps in packet stream due to the removal of packet physical headers and embedded control symbols. Before sending this data to the RX User Interface, this block aligns the start of packet so that the packet begins at the 64-bit boundary of the data bus.

This block checks the Link Requests/Responses and appropriately blocks the incoming traffic under error conditions. This block is also responsible for generation of errors such as s-bit errors, corrupted control symbols and link request/response errors.

RX Symbol Decoder (RSD)
The RX Symbol Decoder block stores any control symbols that are received into a synchronous FIFO. This block then reads the data from the FIFO, decodes the control symbols and passes them to the RX Transmit ACK Control Block (RTA). During error conditions the FIFO is emptied and the contents are ignored until the error has been resolved.

The different kinds of control symbols used are:

- Acknowledgement Control Symbols (Acks)
  - Packet-accepted
  - Packet-retry
  - Packet-not-accepted
- Link Maintenance Control Symbols
  - Link Request
    - Send Training
  - Link Response
- MC Event Control Symbol

For more detailed information about control symbols, refer to *RapidIO Interconnect Specification,* Part IV: Physical Layer 8/16 LP-LVDS Specification.

RX Transmit ACK Ctrl (RTA)
The RX Transmit ACK control block keeps track of the acknowledgement control symbols for packets transmitted. This is implemented by maintaining an ACK received queue. Once a packet is accepted, an ACK ID number [ACK ID = 0 to 7] is sent to the user to release the packet buffer. If a packet needs to be resent, a packet-retry ACK ID [ACK ID = 0 to 7] will be sent to the transmit block to send the packet once again. Similarly, for packet-not-accepted conditions, an ACK ID [ACK ID = 0 to 7] will be sent to the transmit block to cause it to resend the packet.

If a packet is still in transmission, a packet-retry or packet-not-accepted acknowledgment may cause the sender to cancel transmission of the packet. The sending device can use the stomp, 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.

This block is also responsible for maintaining the timers once a packet is sent. There can be eight outstanding packets in Parallel RapidIO and once all the packets are sent, the core starts waiting for the ACK CS from the other device. Upon the expiration of any timer, Retry Recovery starts and all the packets that are not acknowledged are sent again.

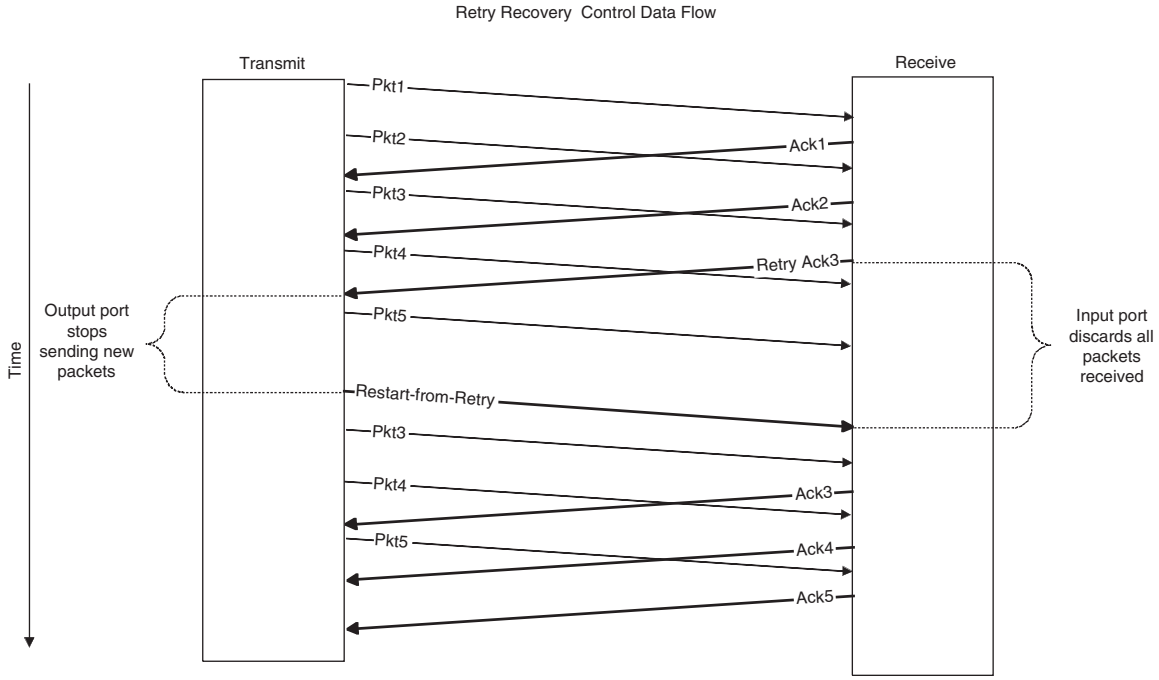
This block generates the appropriate responses to the following control symbol errors. These errors are listed below:

- Error on control symbol
  - Packet-not-accepted control symbol
  - Packet-retry/packet-ack control symbol with unexpected ACK ID
  - Corrupt control symbol
- Time out waiting for control symbol

Retry Recovery: The retry recovery state machine is part of the RX Transmit ACK Control (RTA) block. Its purpose is to attempt to recover data for a retry condition. It is triggered when a device receives a packet-retry control symbol to stop sending new packets. After sending a packet-retry control symbol, the state machine will then send a restart-from-retry control symbol to resend the data, for which the packet-retry control symbol was originally sent.

Figure 7 shows an example of the Retry Recovery flow control operation. In this example, after the transmitter transmits several packets, the receiver sends a packet-retry control symbol (RetryAck3) on packet #3 to the transmitter. It asks the transmitter to resend packet #3 again. The receiver discards all packets received after packet #3. Once the transmitter detects the packet-retry control symbol, it stops sending new packets until it sends an input status of restart-from-retry to the receiver to prepare the receiver for incoming packets.

Figure 7. Retry Recovery Control Data Flow

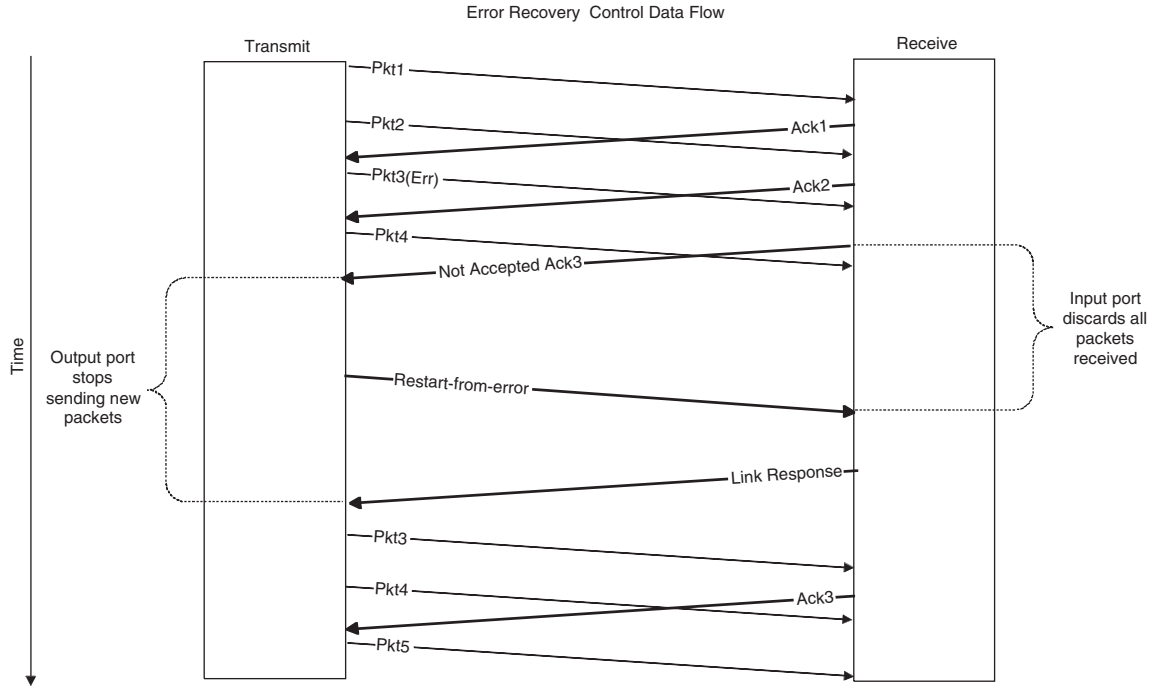


It is possible that by the time the transmitter receives the packet-retry control symbol (RetryAck3), it is still transmitting packets to the receiver due to the transmitter latency in processing the packet-retry control symbol. This condition is acceptable since the receiver discards all incoming packets received after it sends packet-retry control symbols. For more information on the retry recovery state machine, refer to Appendix A of the *RapidIO Interconnect Specification* Part IV: Physical Layer 8/16 LP-LVDS Specification.

Error Recovery: The error recovery state machine is also part of RX Transmit ACK Control (RTA) block, and will attempt to recover data for error conditions. When this state machine detects an output error, such as an unexpected ACK ID, the state machine immediately transitions into an output error-stopped state, stops sending new packets and issues a restart-from-error control symbol to the sender. The restart-from-error control symbol receives a response from the receiver including the ACK ID. This ACK ID indicates where the sender should begin re-transmission. For more information on the error recovery state machine, refer to Appendix A of the *RapidIO*

Interconnect Specification Part IV: Physical Layer 8/16 LP-LVDS Specification. Figure 8 shows an example of the Error Recovery flow control operation.

Figure 8. Error Recovery Flow Diagram



RX Receiver Packet ACK Ctrl (RRA)
This block monitors the incoming packets and RX related Control symbols generated by the RX Packet/Control Disassembly (RPD) block. This block decides whether to accept the packet sent by the other device based on the CRC correctness the ACK ID of the packet and other conditions, such as maximum payload. Once the packet is accepted or rejected, this block will inform the TX Symbol Generator block (TSG) to send the packet-accepted, packet-not-accepted or packet- retry control symbol to the sender.

This state machine interacts with the TX retry recovery State Machine of the link device and they work together to attempt recovery for a retry condition. This RX retry recovery state machine is triggered when it returns a packet-retry control symbol to the sender, due to some temporary internal condition, and shall silently discard all new incoming packets until it receives a restart-from-retry control symbol from the sender.

The following are the different types of errors present in RapidIO and are grouped as input and output errors.

Input Errors:

- Error on Packet
  - Unexpected ACK ID on Packets
  - Corrupt Packet indicated by CRC
  - Packet that overruns defined packet boundary.

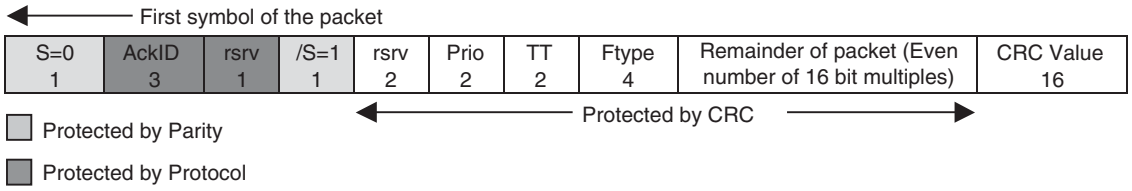
Output Errors:

- Error on Control Symbol
  - Packet-not-accepted control symbol
  - Packet-retry/packet-accepted control symbol with unexpected ACK ID.
  - Corrupt Control symbol
- Intermediate Errors (S bit error)
- Time out waiting for control symbol
- Received Send-training control symbol

RX CRC Checker: The RX CRC Checker generates a separate CRC for each data packet and compares it to the appended CRC transmitted with each packet, to identify any errors.

This CRC is generated over all of a packet header and the entire data payload except the first six bits of the added physical layer fields (which are treated as logical 0s) as shown below. The initial value of the CRC is 0xFFFF, or all logic 1s. For purposes of simplifying CRC calculation, the uncovered first six bits are assumed to be logic 0s.

Figure 9. CRC Checking



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.

For a packet that has less than or equal 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 (including physical layer 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 the devices to regard the embedded CRC value as two bytes of packet payload for CRC checking purposes. 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 post appended to the packet. The pad of logic 0s allows the CRC check to always be done at the 32-bit boundary.

RX User Interface

The RX User Interface generates all the control signals necessary to interface to the RX user logic or the Logical and Transport layers of the RapidIO stack. This block generates the signals indicating that the RX PHY is ready to send data. This block also controls the discard signal, which is used to discard a packet being sent to the user if packet errors are detected.

The block is used by the user logic to receive data and core status from the Parallel RapidIO core and to interface to a simple FIFO interface. The signals generated and used by this block are described in the User Interface and Signal Description sections of this document.

User Interface

This section explains in detail the user interface signals present in the Parallel RapidIO PHY IP core. The following are the different user interfaces:

- TX User Interface
- RX User Interface
- PMI Interface

TX User Interface

The TX User Interface consists of two types of interface: TX User Control Interface and TX Generic FIFO Bridge Interface. Both run synchronously on the clock received on `tx_usr_clk` signal.

The TX User Control Interface generates all the control signals required for the TX user such as tx_riop_ready, tx_next_ackid, tx_release_ackid, tx_riop_discard and tx_release_val. The TX Generic FIFO Bridge Interface is a standard FIFO interface used to control data flow coming from the user side.

After device power up and the reset_n signal is asserted, The Parallel RapidIO core goes through an internal initialization/training cycle (the user has to program the CSR registers). Once this is successful, the tx_riop_ready signal is asserted indicating that the core is ready for transmitting Parallel RapidIO packets.

Note: During the transmit process, the status of tx_riop_ready must be monitored continuously. When tx_riop_ready is de-asserted, all user interface signals should be regarded as invalid input or invalid output.

Once the tx_riop_ready is active, the tx_next_ackid[2:0] signal bus indicates to the user the ACK ID of the buffer to be transmitted. These buffers are numbered from 0 to 7 and are maintained by the user. Each of these buffers contains a packet that has to be transmitted through the core. The implementation of these buffers is left to the user and is not specified in this document.

If a packet has been successfully transmitted by the core (i.e. after the core receives a acknowledgement from the far end receiver), the signal tx_release_val goes active indicating that the user buffer associated with the tx_release_ackid[2:0] value can be released. This means that the user can now store a different packet into the buffer that has been released. The buffer should not be updated with a new packet until a release signal tx_release_ackid has been received on that buffer.

Once a complete packet is available at the user end for transmission, the user must indicate this to the core by activating the sine_tx_data_avail signal. On seeing the sine_tx_data_avail signal active, the core activates the read signal sine_tx_read.

Once the sine_tx_read is activated, the user should provide the requested buffer data on these signals such as tx_prio[1:0], sine_tx_sof, sine_tx_data[63:0], sine_tx_dataen[3:0] and sine_tx_eof.

The sine_tx_sof signal should be asserted at the start of the packet and sine_tx_eof should be asserted at the end of the packet. The sine_tx_sof and sine_tx_eof signals should only be asserted for one clock cycle.

The tx_prio[1:0] indicates the priority field of the Parallel RapidIO packet. The sine_tx_dataen[3:0] indicates the number of bytes valid at the end of the packet. During all other times, this signal is not valid.

The sine_tx_alm_empty signal should be asserted whenever the buffer contains only one 64-bit data left. This signal is used to deactivate the sine_tx_read signal to prevent the core from issuing illegal reads.

During a data transfer from the user to the core (i.e. when sine_tx_read is asserted), the user can terminate the transfer by asserting the tx_usr_discard. Similarly, the core can terminate the packet by asserting the tx_rios_discard signal. The reasons for which the user can terminate the packet transfer are user-dependent and are not specified in this document.

RX User Interface

The RX user interface consists of the RX User Control Interface and the RX Generic FIFO Bridge Interface. Both run synchronously on the rx_usr_clk signal. The RX User Control Interface generates all the control signals required for the RX user such as rx_riop_ready and rx_riop_discard.

After the device is powered up and when the reset_n signal is activated, the Parallel RapidIO core runs through an internal initialization/training cycle (the user has to program the CSR registers). Once this is successful the rx_riop_ready signal goes active, indicating that the other control signals on the RX interface are valid.

After rx_riop_ready is active, whenever the core receives a good packet, the core generates a sine_tx_write signal indicating that the data transfer to the user is active. The other signals associated with this data transfer are sine_rx_sof, sine_rx_data[63:0], sine_rx_hwen[3:0] and sine_rx_eof.

The sine_rx_sof will always be asserted at the start of the packet and sine_rx_eof will be asserted at the end of the packet. However, for the end of the packet case, the sine_rx_eof signal may or may not be asserted.

There are two conditions indicating that the transfer of one packet has ended:

1. The assertion of the sine_rx_eof signal at the end of the packet

2. The assertion of the sine_rx_sof signal to indicate that a new packet is started and the previous packet has been ended.

The sine_rx_hwen[3:0] indicates the number of bytes valid at the end of the packet. During all other times this signal is not valid. When the packet is being transferred, invalid and valid data could be sent on the same packet. Thus, it is important for user to use both the sine_rx_hwen[3:0] signals and the sine_rx_data bus together to determine the validity of data in the current packet.

During a data transfer from the core to the user (i.e. when sine_rx_write is active), the user can terminate the transfer by asserting the rx_usr_discard. Similarly the core can terminate the packet by asserting the rx_phy_discard. The reasons for which the user can terminate the packet transfer are user-dependent and are not specified in this document.

The user provides the rx_buf_th signal which indicates that the user receive buffer is almost full. The actual threshold level on which this signal is activated is not specified in this document.

The user also provides the rx_buf_status[3:0] signal, which indicates number of available packet buffers in user logic. The encoding value specifies the number of packet buffers the receiving device has currently available. "0000" specifies that the buffer is full, "1110" specifies 30 spaces available and "1111" specifies an undefined number of spaces available. The core uses this buffer status signal whenever a control symbol is generated and sent out.

PMI Interface

The Physical Layer Management Interface (PMI) is used to read the registers present in the core.

This interface uses a separate clock signal called pmi_usr_clk. Thus, the interface can run independently and at a different frequency compared to Transmit or Receive blocks in the core.

After the device is powered up and has gone through a reset cycle using the reset_n signal, the pmi_ready signal will be de-asserted, indicating to the user that the interface is ready to accept read/write cycles.

Once the pmi_ready is deasserted or inactive, the user can perform a read operation by pulsing the pmi_sel signal along with the pmi_addr signal, which contains the address of the register to be read, and the signal pmi_wr should be active low indicating a read command. After the above sequence the user logic should wait for the pmi_ready signal to go high, indicating successful completion of the read cycle and the data read out from the register will be available on the pmi_dataout[31:0] data bus.

To perform a write cycle, the user logic first must check to see if the pmi_ready is low. Then the write operation can be performed by pulsing the pmi_sel signal along with pmi_addr, which contains the address of the register to be written. The bus pmi_datain[31:0] should contain the data to be written into the register and the pmi_wr signal should be active high, indicating a write command. After the above sequence, the user logic will wait for the pmi_ready signal to go high indicating successful completion of the write cycle. The data on the pmi_datain[31:0] bus should remain valid during the time that the pmi_ready signal is asserted.

Register Descriptions

This section describes the different registers available in this core. Table 1 shows the overall global memory map of the IP and different register sets implemented.

Table 1. Register Memory MAP

Address	Description
`USR_RIOPHY_BASE +16'h0xxx	PHY CSR Registers
`USR_RIOPHY_BASE +16'h1xxx	User Registers

PHY CSR Registers: These implement the PHY Command and Status Registers as defined in the Parallel RapidIO Specification.

USER Registers: These implement the registers required by the IP core, and give status information about the IP core.

PHY CSR Registers

Table 2. PHY CSR Register Addresses

Offset Address	Name	Description
'h000	PHY_HEADER0	Port Maintenance Block Header 0
'h004	PHY_HEADER1	Port Maintenance Block Header 1
'h020	PHY_LINK_TO	Port Link Time-Out Control CSR
'h024	PHY_RESP_TO	Port Response Time-Out Control CSR
'h03C	PHY_GEN_CTL	Port General Control CSR
'h058	PHY_ERR_STAT	Port Error and Status CSR
'h05C	PHY_CONTROL	Port 0 Control CSR

Table 3. RapidIO PHY Register MAP

Port	Block Byte Offset	Register Name (Word 0)	Register Name (Word 1)
General	0x0	8/16 LP-LVDS Port Maintenance Block Header	
	0x8-18	Reserved	
	0x20	Port Link Time-Out Control CSR	Port Response Time-Out Control CSR
	0x28	Reserved	
	0x30	Reserved	
	0x38	Reserved	Port General Control CSR
Port 0	0x40	Reserved	
	0x48	Reserved	
	0x50	Reserved	
	0x58	Port 0 Error and Status CSR	Port 0 Control CSR
Port 1	0x60	Reserved	
	0x68	Reserved	
	0x70	Reserved	
	0x78	Port 1 Error and Status CSR	Port 1 Control CSR
Port 2 –Port 14	0x80 - 218		
Port 15	0x220	Reserved	
	0x228	Reserved	
	0x230	Reserved	
	0x238	Port 15 Error and Status CSR	Port 15 Control CSR

Table 4. PHY Register Descriptions

Name	Bits	Reset Value	Access	Description
PHY_HEADER0				
EF_PTR	15:0	`h0000	R	Hard wired pointer to the next block in the data structure, if one exists
EF_ID	31:16	16'd1	R	Hard wired Extended Features ID. Bit 16 is 1'b1 Bit 31:17 are zeros.
PHY_HEADER1				
—	31:0	—	—	Reserved
PHY_LINK_TO				
time-out_value	23:0	'hFFFF	RW	Time-out interval value. Bit 0 is considered the LSB. Bit 23 is considered the MSB.
—	31:24	—	—	Reserved
PHY_RESP_TO				
resp_time_out	23:0	'hFFFF	RW	Time-out interval value. Bit 0 is considered the LSB. Bit 23 is considered the MSB.
—	31:24	—	—	Reserved
PHY_GEN_CTL				
host	0	0	RW	A host device is responsible for system exploration, initialization and maintenance. Host devices typically initialize agent or slave devices. 'b0 – agent or slave device 'b1 – host device
master_enable	1	0	RW	This bit controls whether a device is allowed to issue a request into the system. If the Master Enable is not set, the device may only respond to requests. 'b0 – processing element cannot issue a request. 'b1 – processing element can issue a request.
discovered	2	0	RW	This device has been located by the processing element responsible for system configuration. 'b0 – The device has not been previously discovered. 'b1 – The device has been discovered by another processing element.
—	31:3	—	—	Reserved
PHY_ERR_STAT				
—	10:0	—	—	Reserved
output_retry_enc	11	0	RW	The output port has encountered a retry condition. This bit is set until it is written with logic 1 to clear.
output_retried	12	0	R	The output port has received a packet-retry control symbol and cannot make 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

Table 4. PHY Register Descriptions (Continued)

Name	Bits	Reset Value	Access	Description
output_retry_stop	13	0	R	The output port has received a packet-retry control symbol and is in the "output-retry-stopped" state.
output_err_enc	14	0	RW	The output port has encountered (and possibly recovered from) a transmission error. This bit is set when bit 15 is set. Once set, it will remain set until it's written with logic 1 to clear.
output_err_stop	15	0	R	The output port is in the "output error-stopped" state.
—	20:16	—	—	Reserved
input_retry_stop	21	0	R	The input port is in the "input retry-stopped" state.
input_err_enc	22	0	RW	The input port has encountered (and possibly recovered from) a transmission error. This bit is set when bit 23 is set. Once set, it will remain set until it's written with logic 1 to clear.
input_err_stop	23	0	R	Input port is in the "input error-stopped" state.
—	26:24	—	—	Reserved
port_write_pend	27	0	RW	Port has encountered a condition, which required it to initiate a Maintenance Port-write transaction. This bit is only valid if the device is capable of issuing a maintenance port-write transaction. Once set, it will remain set until it's written with logic 1 to clear.
—	28	—	—	Reserved
port_error	29	0	RW	Input or output port has encountered an error from which hardware was unable to recover. Once set, it will remain set until it's written with logic 1 to clear.
port_ok	30	0	R	Input and output ports are initialized and can communicate with the adjacent device. This bit and bit 31 are mutually exclusive.
port_uninit	31	1	R	Input and output ports are not initialized and are in training mode. This bit and bit 30 are mutually exclusive.
PHY_CONTROL				
Output_port_width	0	0	R	Operating width of the port (read-only): 'b0 - 8-bit port 'b1 - 16-bit port
Output_port_enable	1	0	RW	Output port transmit enable: 'b0 - port is stopped and not enabled to issue any packets except to route or respond to I/O logical MAINTENANCE packets, depending upon the functionality of the processing element. Control symbols are not affected and are sent normally. b1 - port is enabled to issue any packets
Output_port_driver_disable	2	0	RW	Output port driver disable: 'b0 - output port drivers are turned on and will drive the pins normally. 'b1 - output port drivers are turned off and will not drive the pins. This is useful for power management.

Table 4. PHY Register Descriptions (Continued)

Name	Bits	Reset Value	Access	Description
—	3	—	—	Reserved
Input_port_width	4	0	R	Operating width of the port (read only): 'b0 - 8-bit port 'b1 - 16-bit port
Input_port_en	5	0	RW	Input port receive enable: 'b0 - port is stopped and only enabled to route or respond to I/O logical MAINTENANCE packets, depending upon the functionality of the processing element. 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. 'b1 - port is enabled to respond to any packet
Input_port_receiver_dis	6	0	RW	Input port receiver enable: `b0 - input port receivers are enabled. `b1 - input port receivers are disabled packets or control symbols.
—	7	—	—	Reserved
err_checking_dis	8	0	RW	
multicast_partp	9	0	R	This bit disables all RapidIO transmission error checking 'b0 - Error checking and recovery is enabled. 'b1 - Error checking and recovery is disabled. Device behavior when error checking and recovery is disabled and an error condition occurs is undefined.
—	10:31	—	—	Reserved
port_type	31	1	R	This indicates the port type, parallel or serial 0- Parallel port 1- Serial port

USER Registers

The following table shows the different USER registers implemented in this IP core.

Table 5. User Register Addresses

Offset Address	Name	Description
'h00	USR_STATUS	Status Register
'h04	USR_SYMBOL	Symbol Register
'h08	USR_TP_CNT	Training Pattern Request Count Register
'h1C	USR_CONSIST_CNT	Consistency Count Register

Table 6. User Register Descriptions

Name	Bits	Reset Value	Access	Description
USR_STATUS				
err_noidle	0	0	RW	When asserted indicates that the RX hasn't got IDLE control symbol after requesting.
err_ackid	1	0	RW	Indicates Link Response Fatal Error. The link response contains the AckID that is not outstanding.
cs_sent	2	1	R	Requested Control Symbol is sent.
init_done	3	0	RW	Management has configured the registers so ports are ready.
reset_cs	4	0	RW	Reset control symbol received
naccptd_cause	7:5	0	R	Packet not accepted 000: Encountered internal error 001: Received unexpected AckID on packet 010: Received error on control symbol 011: Non-maintenance packet reception is stopped 100: Received bad CRC on packet 101: Received S bit parity error on packet/control symbol 110: Reserved 111: General Error
mc_event	8	0	RW	Active high indicates MC Event received, user can clear this bit by writing a zero.
—	31:9	-	—	Reserved
USR_SYMBOL				
cs	15:0	'h0000	RW	Control symbol that need to be send (given by the user).
—	31:16	—	—	Reserved
USR_TP_CNT				
tp_req_cnt	2:0	'h7	RW	This counter value indicates the maximum number of times the "Training State Machine" can request training patterns.
—	31:3	—	—	Reserved
USR_CONSIST_CNT				
consist_cnt	3:0	4'b100	RW	Minimum number of times data received to be same before finding the shifted bits.
—	31:4	—	—	Reserved

Signal Descriptions

Table 7. Parallel RapidIO Phy Interface Signals

Signal Name	Active State	Signal Direction (I/O)	Signal Description
Transmit Interface			
tclk0_p, tclk0_n	—	O	LVDS Transmit Clock – Free-running clock for the 8-bit port. tclk0 connects to rclk0 of the receiving device.
tframe_p, tframe_n	—	O	LVDS Transmit framing signal – When issued as active, this signal indicates a packet control event. tframe is connected to rframe of the receiving device.
td_p[0-7]/td_p[0:15] td_n[0-7]/td_n[0-15]	—	O	LVDS 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.
Receive Interface			
rclk0_p, rclk0_n	—	I	LVDS Receive Clock – Free-running input clock for the 8-bit port. rclk0 connects to tclk0 of the transmitting device.
rframe_p, rframe_n	—	I	LVDS Receive framing signal – When asserted as active, this signal indicates a packet control event. rframe is connected to tframe of the transmitting device.
rd_p[0-7]/rd_p[0:15] rd_n[0-7]/rd_n[0:15]	—	I	LVDS Receive Data – The Receive data is a unidirectional packet data input bus. It is connected to the td bus of the transmitting device.

Table 8. User Interface Signals

Signal Name	Active State	Signal Direction (I/O)	Signal Description
reset_n	Low	I	Active low system reset signal
rx_usr_clk	—	O	Receive side (RX) user clock
RX User Control Interface (RUI)			
rx_buf_status[3:0]	—	I	Specifies the number of available packet buffers in the receiving device. The encoding value specifies the number of packet buffers the receiving device has currently available. "0000" specifies that the buffer is full "1111" specifies 15 spaces are available.
rx_buf_th	High	I	Indicates that the user receive buffer is becoming full.
rx_ready	High	O	Ready signal indicating the TX RapidIO-PHY is ready to receive data from the user.
rx_phy_discard	High	O	Indicates that the RapidIO-PHY wants to terminate the current frame being received from the user interface.
rx_usr_discard	High	I	Indicates that the user interface wants to terminate the current frame being sent to the RapidIO-PHY.
sine_rx_data [63:0]	—	O	Indicates data was sent from the RX RapidIO-PHY side to the user interface. This contains the logical/transport layer data.
sine_rx_hwen[3:0]	—	O	Indicates the valid half word of the data bus sine_rx_data of the current frame transfer.
sine_rx_sof	—	O	SOF signal indicating the start of sine_rx_data.
sine_rx_eof	—	O	EOF signal indicating the end of sine_rx_data.

Table 8. User Interface Signals (Continued)

Signal Name	Active State	Signal Direction (I/O)	Signal Description
TX User Control Interface			
tx_riop_ready	High	O	Ready signal indicating the TX RapidIO-PHY is ready to receive data from the user. When tx_riop_ready is de-asserted, all user interface signals are regarded as invalid input or output.
tx_next_ackid[2:0]	—	O	Indicates that the buffer data associated with this "AckID" should be sent by the user.
tx_release_ackid[2:0]	—	O	Indicates that the buffer data associated with this "AckID" can be released and reused by the user interface.
tx_release_val	—	O	Indicates that tx_release_ackid is valid.
tx_riop_discard	High	O	Indicates that the RapidIO-PHY wants to terminate the current frame being received from the user interface.
tx_usr_discard	High	I	Indicates the user interface wants to terminate the current frame being sent to the RapidIO-PHY interface.
tx_prio[1:0]	—	I	Indicates the priority of the packet being received.
sine_tx_data[63:0]	—	I	This bus contains the data read from the user interface
sine_tx_alm_empty	High	I	This signal is asserted when the User Transmit FIFO is almost empty.
sine_tx_read	High	O	This signal is asserted when the TX RapidIO- PHY requests data from the user transmit FIFO.
sine_tx_sof	High	I	This signal is asserted along with the first data transferred to indicate the start of a new frame.
sine_tx_eof	High	I	This signal is asserted along with the last data transferred to indicate the end of the frame.
sine_tx_dataen[3:0]	—	I	This signal indicates the valid bytes of the data bus in the last data transfer of the current frame. For all other data transfers, the FIFO is expected to pack the data bus. The width of this bus depends on the width of the data bus.
sine_tx_data_avail	High	I	This signal is asserted as long as there is at least one full frame/ predetermined amount of data in the user Transmit FIFO.
Physical Management Register Interface (PMI)			
pmi_usr_clk	—	I	Management interface clock
pmi_sel	High	I	Management Device Select. Asserting this signal can access Management registers.
pmi_ready	High	O	Management is ready. When asserted, indicates that Management has taken/kept the data from/to the data bus.
pmi_wr	High	I	Management register write. Valid when pmi_sel is asserted. Low indicates read.
pmi_addr [`USR_ADDR_WIDTH-1:0]	—	I	Management address bus for register access.
pmi_datain[31:0]	—	I	Register Data In
pmi_dataout[31:0]	—	O	Register Data Out
pmi_int	High	O	Management Interrupt

Parameter Descriptions

User-configurable parameters are shown in Table 9. These parameters are configured using IPexpress™, included with Lattice's ispLEVER® design tools.

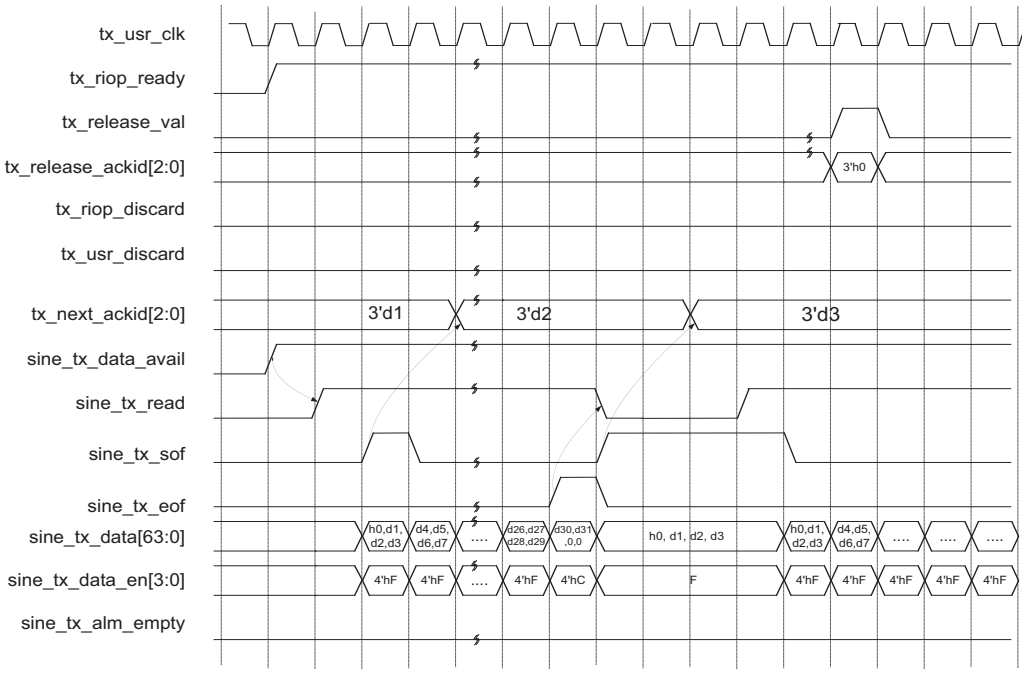
Table 9. User Configurable Parameters

Parameter	Description	Range	Default
PHY_DSIZE_8	Selects 8-bit RapidIO data width port. Otherwise, selects 16-bit RapidIO data width port.	8 bit or 16 bit	8
INT_DSIZE	Internal Data Bus Width	64	64 bits [Pre-selected]
USR_RIOPHY_BASE	PMI Phy Register Base Address	0 - 'hFFFF_FFFF	32'h0000_0000
USR_ADD_WIDTH	PMI Interface Address Bus Width	0 - 64	32
LRTMOUT_CTR	Link Timeout Counter Width	16	16 [Pre-selected]
LNK_RESP_TMOUT	Link Timeout Value	0 - 'hFFFF	16'h004B (75 Internal clocks)
RX_REC_IDLE_LIMIT	Receive Idle Limit	0 - 'hFF	8'hFF
MAX_PKT_SIZE	Max packet size allowed for reception in bytes.	0 - 9'h114	9'h114 (276 bytes)
TSG_PE_ADDR	Partial Empty Flag setting for Symbol FIFO in TX Path	0 - 63	1
TSG_PF_ADDR	Partial Full Flag setting for Symbol FIFO in TX Path.	0 - 63	31

Timing Diagrams

Transmit User Interface

Figure 10. TX User Data Transfer of a Packet



Note: Data with data valid on two half words during the `sine_tx_eof`.

Figure 11. TX Discarding a Packet by tx_usr_discard

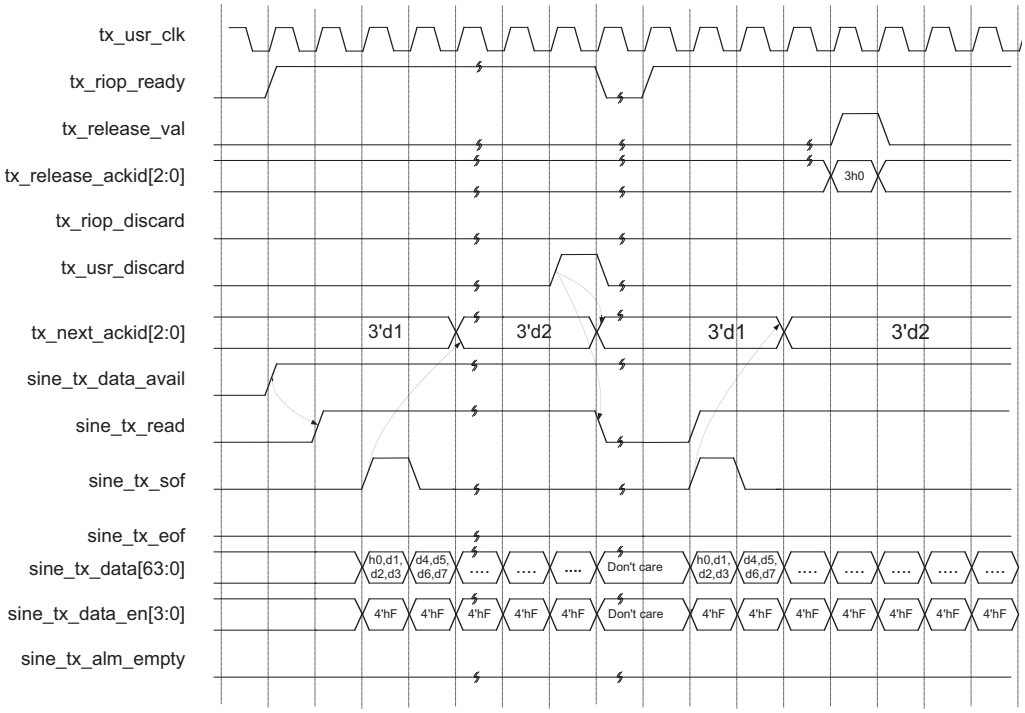
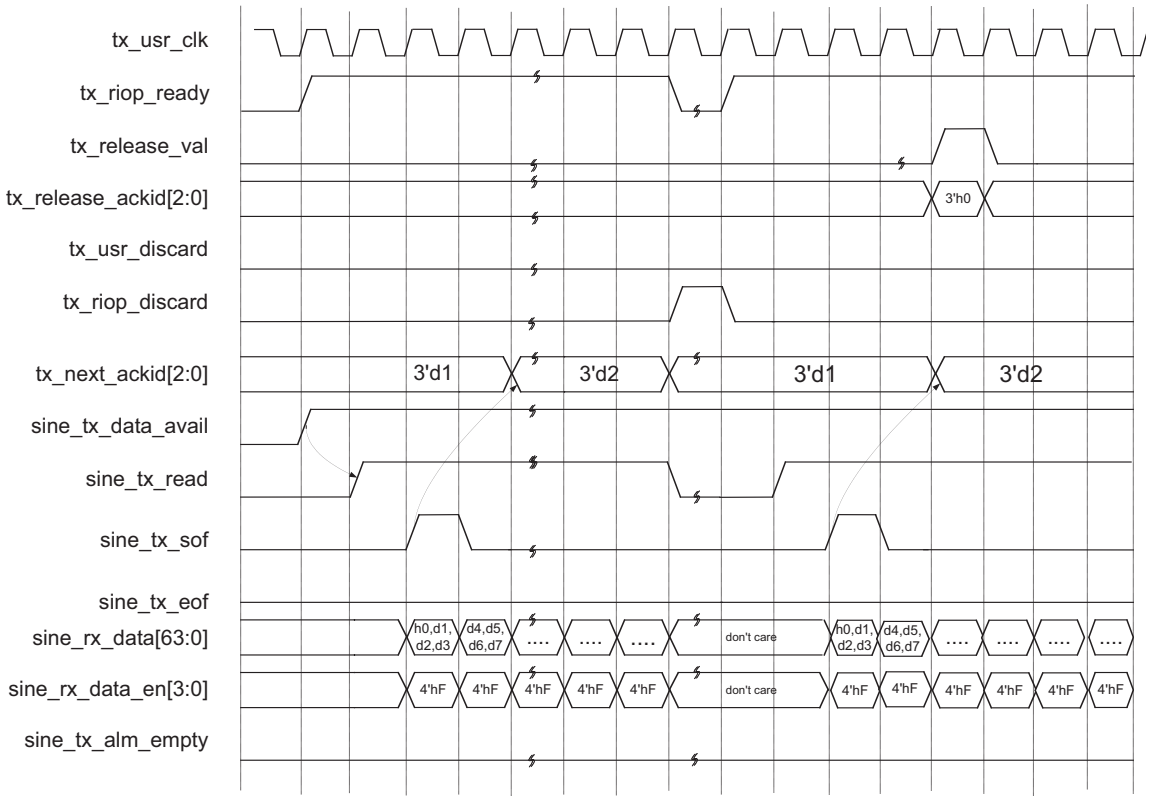


Figure 12. TX Discarding a Packet by tx_riop_discard



Receive User Interface

Figure 13. RX Transfer of Data Without Discards

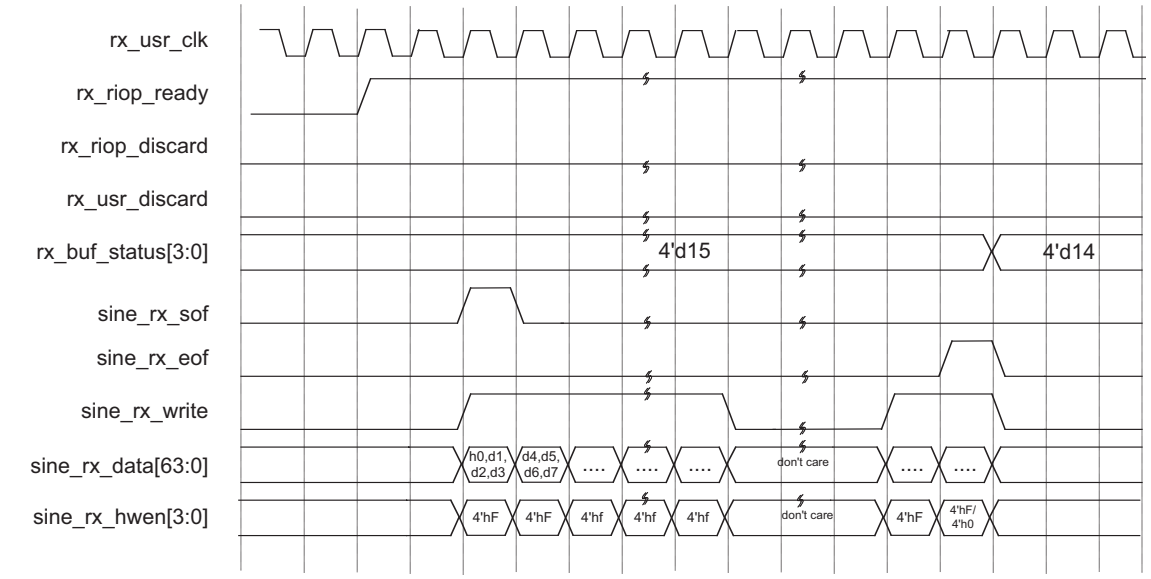


Figure 14. Discarding the Packet Using rx_riop_discard

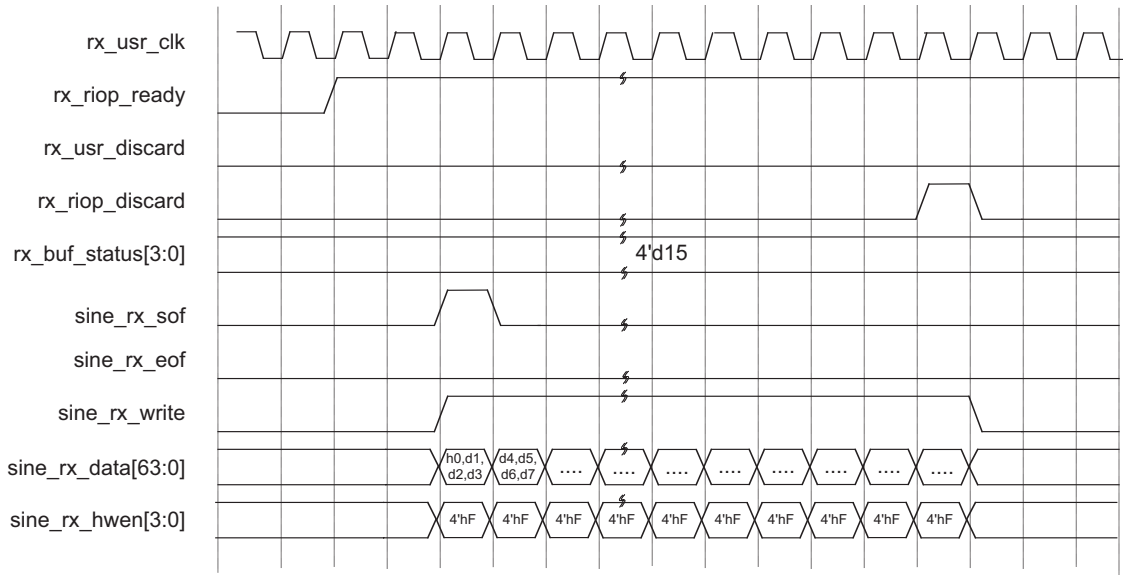
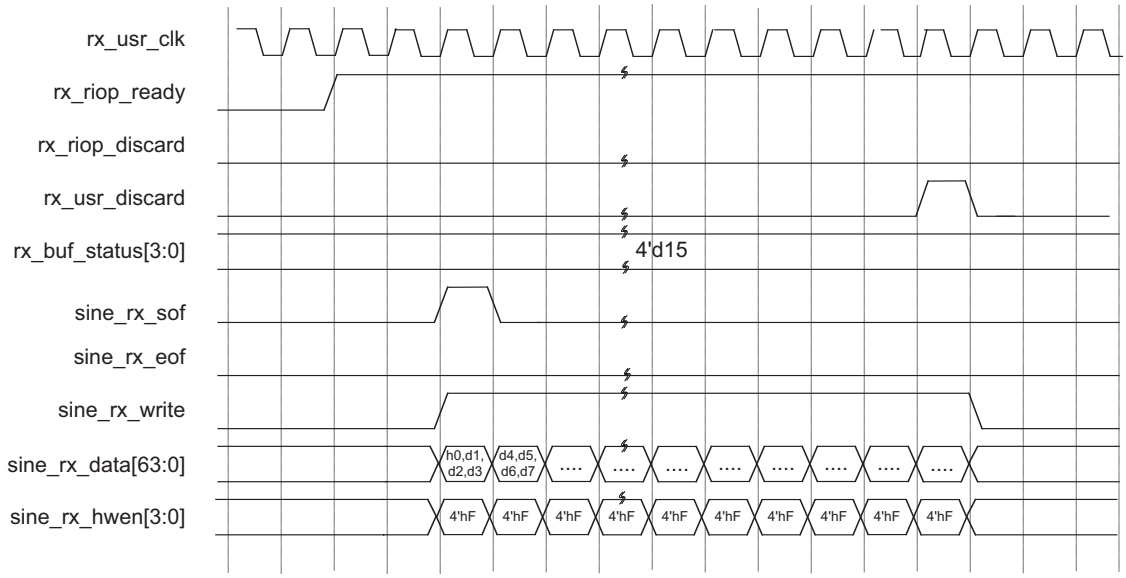
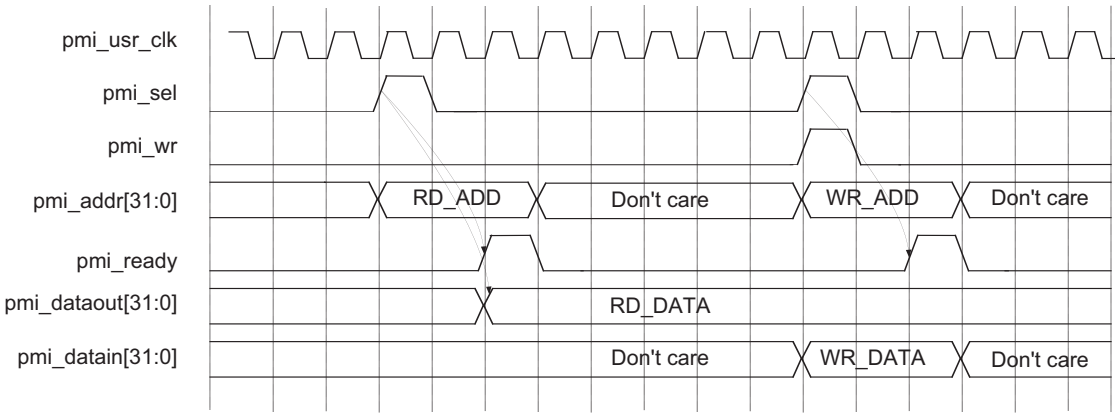


Figure 15. Discarding the Packet Using rx_usr_discard



PMI User Interface

Figure 16. PMI Read and Write Cycles



Reference Information

- ispLEVER Software Online Help Manual
- IP Evaluation Tutorials

Technical Support Assistance

Hotline: 1-800-LATTICE (North America)
+1-503-268-8001 (Outside North America)
e-mail: techsupport@latticesemi.com
Internet: www.latticesemi.com

Appendix for LatticeECP and LatticeEC™ Devices

Table 10. Performance and Resource Utilization[1]

Parameter Filename	Parameter Settings	SLICEs	LUTs	Registers	I/Os	sysMEM EBRs	f_{MAX}
rio_para_e2_1_001_1.lpc	See Table 11	3788	4841	4080	288	2	250MHz (PHY clocks) 62.5MHz (User clocks)

1. Performance and utilization characteristics are generated using LFEC20E-5F672C in Lattice's ispLEVER 4.2 software. When using this IP core in a different density, speed, or grade within the LatticeECP/EC family, performance may vary.

Supplied Netlist Configurations

The Ordering Part Number (OPN) for the Parallel RapidIO Physical Layer Interface IP on LatticeECP/EC devices is RIO-PARA-E2-N1 (for all configurations of the netlist package). Table 11 lists the evaluation netlists that can be downloaded from the Lattice web site at www.latticesemi.com.

You can use the IPexpress software tool to help generate new configurations of this IP core. IPexpress is the Lattice IP configuration utility, and is included as a standard feature of the ispLEVER design tools. Details regarding the usage of IPexpress can be found in the IPexpress and ispLEVER help system. For more information on the ispLEVER design tools, visit the Lattice web site at: www.latticesemi.com/software.

Table 11. Parameter Settings of the Evaluation Packages

PHY Size	Bus Width	PHY Register Base Address	User Address Width	Link Timeout Counter Width	Link Timeout Value	Receive Idle Limit	Maximum Packet Size	Partial Empty Flag Address	Partial Full Flag Address
8	64	00000000	32	16	004B	FF	114	1	32