RapidIO 8-bit Port Physical Layer v3.0.2

DS243 February 10, 2005 Product Specification

Introduction

The LogiCORE RapidIO Physical Layer Interface, a fixed-netlist solution for the RapidIO interconnect, is a pre-implemented and fully tested module for Xilinx Virtex™-II and Virtex™-II Pro FPGAs.

Xilinx Virtex-II series FPGAs enable the design of fully RapidIO compliant systems. The FPGA devices meet the required electrical and timing parameters, including AC output drive characteristics, setup, hold, and clock to output as stated in the 250 MHz RapidIO AC specification.

Features

- RapidIO Physical Layer Interface compliant with RapidIO Interconnect Specification v1.2
- 8-bit LVDS port with 64-bit internal data path
- Peak performance of 8 Gbps / port
- 250 MHz clock rate, 500Mbps / LVDS pin pair
- Supports Packet Retry, stomp, transmission error recovery, throttle-based flow control, and CRC Time-of-Day (TOD) sync
- Separate clock domain to control management interface for reading and writing configuration registers
- Flexible buffer management scheme
- Automatic phase alignment of input clock to avoid delay matching in board design
- Reduced product development time
  - Predefined device pinout and relative placement of critical logic
  - Critical paths are controlled by a constraints file ensuring predictable timing
- Xilinx Smart-IP™ technology
- Available under terms of the SignOnce™ IP License

LogiCORE™ Facts			
Core Specifics			
Supported Device Family	Virtex-II 2V1000FF896-4 Virtex-II 2V2000FF896-4 Virtex-II Pro 2VP7FF896-5 Virtex-II Pro 2VP20F896-5		
Resources Used[1]	I/O	LUTs	FFs
	20 LVDS pairs	~5000	~2500
Provided with Core			
Documentation	RapidIO Physical Layer Interface Design Guide and Quick Start Guide		
Design File Formats	Verilog Simulation Models NGO Netlist		
Constraints File	User Constraints File (.ucf)		
Example Design	Verilog Buffer Design, Register Manager Design, Processor Buffer Design Example		
Design Tool Requirements			
Xilinx Tools	Xilinx ISE 8.1i SP2		
Tested Entry and Verification Tools[2]	Xilinx Synthesis Technology Synplicity Synplify Cadence Verilog XL Model Technology ModelSim		
Support			
Xilinx, Inc. provides technical support for this product when used as described in the Design Guide and the Getting Started Guide. Xilinx cannot guarantee timing, funcitionality, or support of the product in devices not listed above, or if customized beyond that allowed in the product documentation, or if any changes are made in sections of the design marked "DO NOT MODIFY".			

1. The exact number of slices depends on user configuration of the interface and the level of resource sharing with adjacent logic.
2. See the implementation guide or product release notes for current supported versions.

Applications

- Gigabit, terabit and faster routers
- Layer 4–7 switches and multiprotocol routers
- Bridges to general purpose and network processors, proprietary and other standard interfaces
- Embedded systems
- Image and signal processing applications
- Hig-bandwidth memory interfaces
- Interfaces to encryption and compression engines

Functional Description

The LogiCORE RapidIO Physical Layer interface performs several functions. It is partitioned into two modules, the OSI Physical Layer Module (OPLM) and the OSI Link Layer Module (OLLM). The OPLM is responsible for the serialization / deserialization function, transmit and internal clock generation, link initialization and training. The OLLM is responsible for CRC generation and verification, symbol generation and decoding, packet exchange protocol handshake and buffer management. Figure 1 illustrates the OLLM and OPLM in the RapidIO Physical Layer.

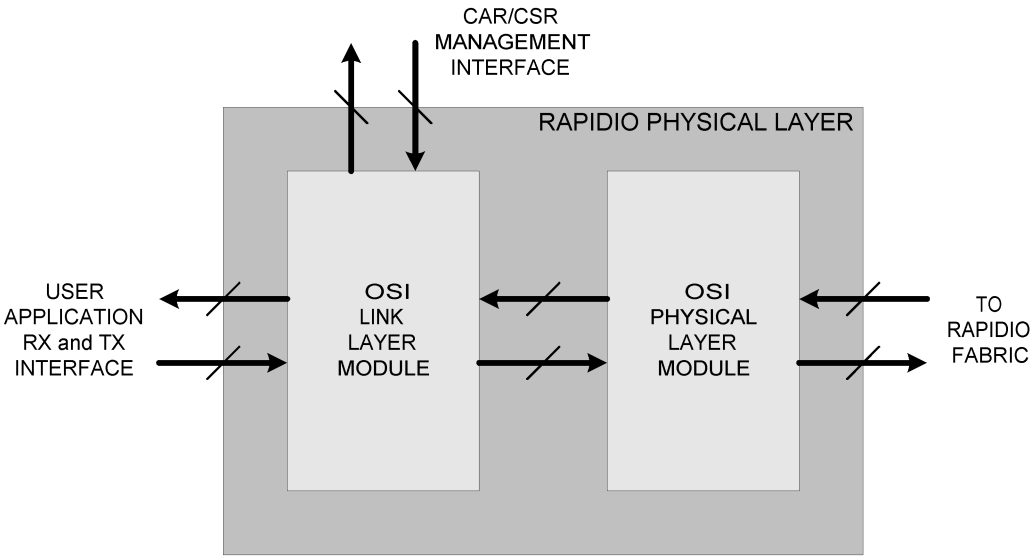

Figure 1: **RapidIO Physical Layer Block Diagram**

OPLM

The OPLM contains four main blocks: The Rx and Tx SERDES and the Rx and Tx Training State Machines. Clock generation is done within the Rx and Tx SERDES. Figure 2 illustrates the OPLM in the RapidIO Physical Layer.

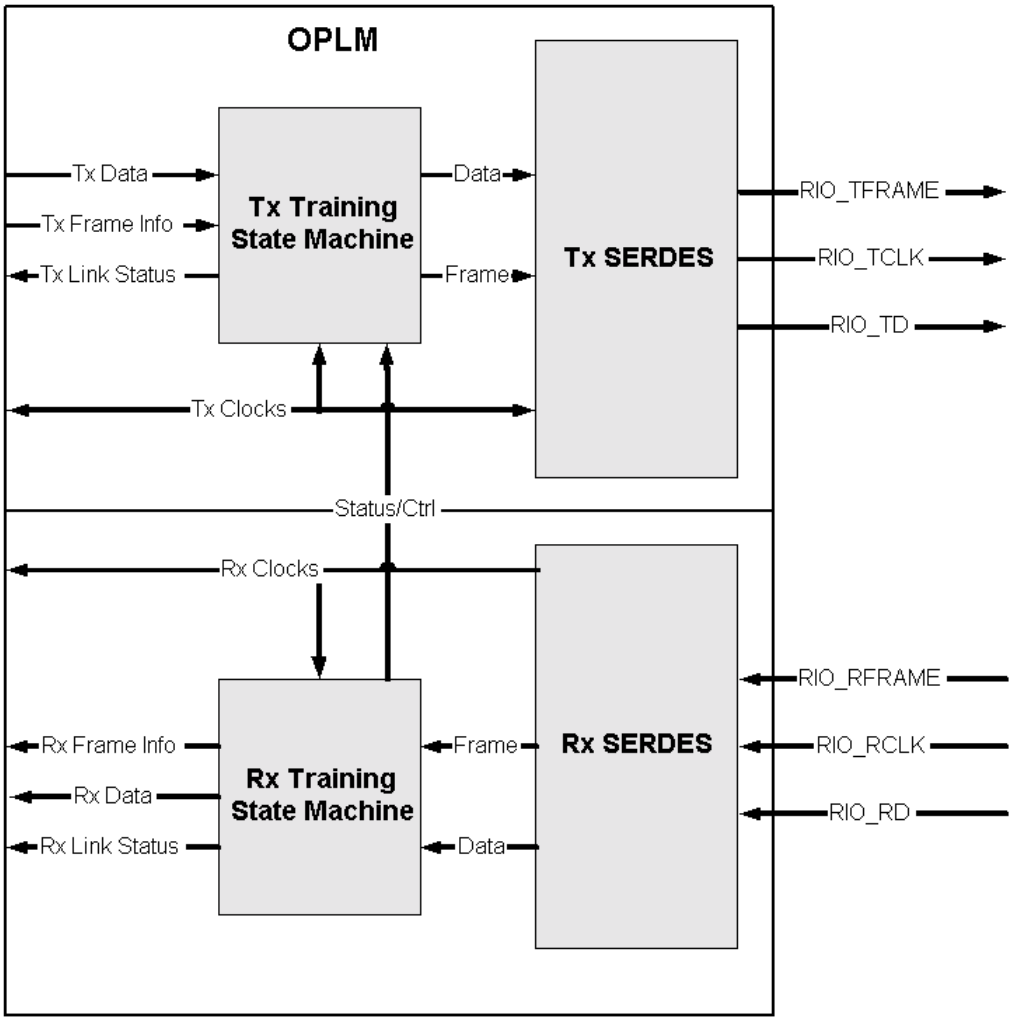

Figure 2: **Physical Layer-OPLM Block Diagram**

Rx/Tx Training State Machines

The Rx and Tx training state machines are responsible for negotiating a valid link with the connected device. The training state machines implement the training protocol outlined in the RapidIO Interconnect Specification. Status of the Tx and Rx link is communicated to the OLLM. The Rx Training State Machine monitors the incoming Rx data stream for valid training patterns and requests. If necessary, the Tx Training State Machine will insert valid training patterns or training requests in the outgoing data stream.

Rx/Tx SERDES

The Rx and Tx SERDES block are responsible for the serialization and deserialization of the data and clock generation for all internal clocks and the transmit RapidIO Link Clock.

On the receive path, the 8-bit double data rate words are received by the SERDES block, where it is converted to 64-bit si gle data rate words. In addition to the data, there are additional signals that indicate where the frame begins in the 64-bit word. The received RapidIO Rx Link Clock is used to capture data received on the RapidIO receive port. Divisibles of this clock (/2 and /4) are used for the internal receive logic path.

On the transmit path, the 64-bit data from the OLLM or the link training-related control symbols are presented to the SE DES block, where it is converted to the 8-bit data. The 8-bit data is transmitted with the appropriate frame and clock to the RapidIO transmit port.

OLLM

The OLLM contains several blocks to handle the CRC generation and verification, symbol generation and decoding, packet exchange protocol handshake, maintenance register management and buffer management. Figure 3 illustrates the OLLM block partitioning.

Rx Error Detection (upper and lower 32-bit word)

These blocks handle symbol error detection. They detect bit mismatches on packet symbols. If an unrecoverable error is detected, they assert an error signal to the symbol decoders.

Rx Symbol Decoder (upper and lower 32-bit word)

These blocks decode symbols from the incoming data stream. They provide data and symbol decode information to the Rx Symbol Strippers. The following symbols are decoded: Time of day synchronization, packet-accepted, packet-not-accepted, packet-retry, throttle, idle, restart- from-retry, end of packet, stomp, link request and link response control symbols. These blocks also decode the incoming ackID and the number of pacing idles requested in the throttle request control symbol for processing by the other blocks.

Rx Symbol Stripper

RapidIO supports the transmission of control symbols within the data stream. Since the control symbol can be transmitted on the upper and/or lower word of an incoming 64-bit data stream, this block handles the multiplexing of that data so that a pure 64-bit data word is outputted and the symbol is stripped from the stream. The symbol information is then passed to the various other blocks for processing. The remaining data is passed to the Rx Align FIFO and CRC checker.

Rx Link Request State Machine

Using information from the Rx Symbol Stripper, the Rx Link Request State Machine processes all incoming Link Request commands and provides status and control information to the various other OLLM blocks. This information may require the scheduling of symbols or responses in the outgoing data path.

Rx Control State Machine

The control state machine block consists of three main functions: frame level flow control, error detection/recovery and the retry mechanism. Using information from the detected symbols and the incoming data stream, status and control are pr vided to the various other OLLM blocks for responses in the outgoing data stream. For example, the Rx control state machine may provide frame level flow control by generating throttle requests in response to Rx Align FIFO's ability to accept data.

Rx CRC Checker

The Rx CRC Checker uses the incoming stream of data to calculate the expected CRC in parallel, as data is stored in the OLLM Rx Align FIFO. The checker then compares the expected CRC with the received CRC in order to determine if a bit error has occurred. Error detection is indicated to the Rx Control State Machine. Data packets with unrecoverable CRC errors are indicated to the user application.

Rx Align FIFO

The Rx Align FIFO is used to properly align data with respect to the input/output controls signals to and from the User and create elasticity in the receive path.

Tx CRC Generator

Using the outgoing data stream from the user application, the Tx CRC Generator calculates the outgoing CRC and inserts it in the outgoing data stream.

Tx Control State Machine

The Tx control state machine is responsible for sequencing the transmission pipeline and has several functions. These fun tions include handling frame generation, error recovery, throttle generation and packet scheduling. The Tx Control State Machine uses a link scheduler to arbitrate what goes out on the link between control packets and data packets. The link scheduler also determines when the CRC should be embedded in a data packet.

Tx Frame Generator

Using input from the outgoing data stream and the Tx Control State Machine, the Tx Frame Generator is responsible for generating the frame information to the OPLM for the Tx data stream. The OPLM uses this information to drive the signal transition on the **RIO_TFRAME** output.

Tx Symbol Generator (upper and lower 32-bit word)

These blocks generate 32-bit symbols based on input from the OLLM Tx control state machine.

Tx Mux (upper and lower 32-bit word)

These blocks multiplex between the outgoing data packet from the Tx CRC generator and outgoing symbols from the Tx symbol generators.

Management Interface Block

The Management Interface Block implements the Physical Layer Command and Status Registers (CSRs). This block is responsible for serving access requests from the management port and synchronizing them with local updates coming from the Rx and Tx Control State Machines. Maintenance packet reads and writes originating from the connected end point to these registers are done through the management port. The Rx and Tx Control State Machines handle updates to these registers as a result of Rx and Tx link activity.

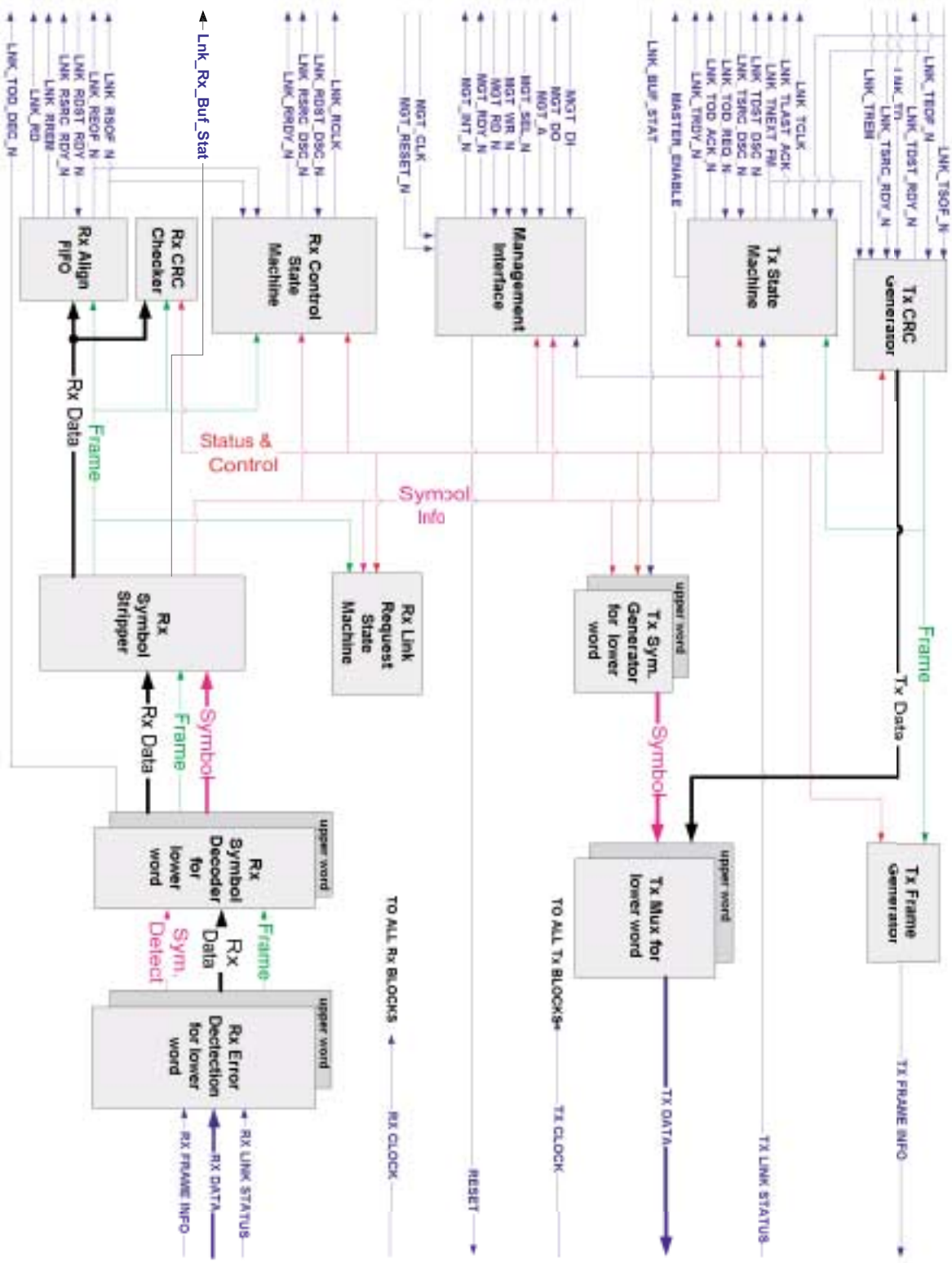

Figure 3: **Physical Layer-OLLM Block Diagram**

Pinout

In compliance with the RapidIO Interconnect Specification v1.2, the RapidIO Physical Layer core provides an 8-bit LVDS interface with separate LVDS signals for the RapidIO clock and RapidIO frame signals.

Signal names and descriptions are shown in Table 1.

Table 1: **RapidIO LVDS Interface Signals**

Name	Direction	Type	Description
RIO_TCLK	Output	Differential	RIO LVDS transmit clock.
RIO_TD [0:7]	Output	Differential	8-bit bus for the RIO LVDS transmit data.
RIO_TFRAME	Output	Differential	RIO LVDS transmit frame signal. This signal toggles for the first transfer of each data packet and for the first transfer of each aligned control symbol. It also toggles for all embedded control and idle symbols in a packet.
RIO_RCLK	Input	Differential	RIO LVDS receive clock.
RIO_RD[0:7]	Input	Differential	8-bit bus for the RIO LVDS receive data.
RIO_RFRAME	Input	Differential	RIO LVDS receive frame signal. This signal toggles for the first symbol of each packet and for the first control symbol of each aligned control symbol. It also toggles for all embedded control and idle symbols in a packet.

Signal Waveforms

The following waveforms demonstrate the operation of the RapidIO Interface signals. Each waveform illustrates the RapidIO Clock, Data and Frame signals. RapidIO uses DDR technology; therefore, data is clocked at every clock edge. This operation is identical for both the Rx and Tx direction.

Figure 3 demonstrates the start of data packet with an embedded control symbol. This is signified by the transition on **RIO_FRAME** on the positive edge of clock 1 to indicate a start of packet. The data is presented on the positive and negative edges of clock 1 and 2. On the positive edge of clock 3, **RIO_FRAME** transitions again to indicate an embedded control sym bol. The control symbol is presented on **RIO_D** on the positive and negative edges of clock 3 and 4. On the positive of edge of clock 5, there is no transition on frame to indicate the continuation of the data packet.

Figure 4 demonstrates the completion of one packet, with an idle control symbol and the beginning of a new packet. The end of packet is indicated by the transition on **RIO_FRAME** on the positive edge of clock 2, followed by the transfer of a control symbol on **RIO_D** on the positive and negative edges of clock 2 and 3. Immediately following the control symbol, an idle is transmitted on **RIO_D** with a corresponding transition on **RIO_FRAME** on the positive edge of clock 4. The transition on **RIO_FRAME** on the positive edge of clock 6, with the transfer of data on the positive edge of that clock signifies the begin ning of a new packet.

Figure 5 demonstrates the completion of one packet followed by the start of a new packet. This is illustrated by the transition on **RIO_FRAME** on the positive edge of clock 3 to signify the transfer of a control symbol. However, the transition, on the positive edge of clock 5, on **RIO_FRAME**, without a corresponding control symbol, would indicate the start of a new data packet.

User Application Interface Description

The RapidIO Physical Layer backend interface to the user application is designed to be a simple yet flexible interface. It consists of two unidirectional data buses with separate control signals for each direction. The control signals allow the User Application to stall or disconnect the data in each direction. The interface also allows the user to operate each data path in its own clock domain. However, the Physical Layer core does manage the synchronization between the clock domains of any necessary communication between the two data paths.

In addition to the Rx and Tx data interface, the Physical Layer provides a separate management interface. This is used to control the RapidIO defined capabilities and control/status registers (CAR and CSR) for the Physical Layer. This interface is also designed with a separate input clock to allow the user to manage these registers in a separate clock domain.

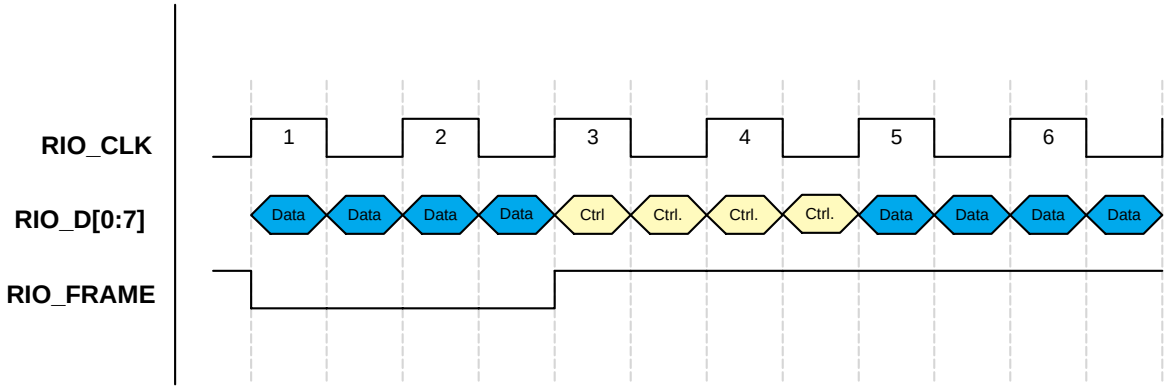

Figure 4: **RapidIO Start of Packet with Embedded Control Symbol**

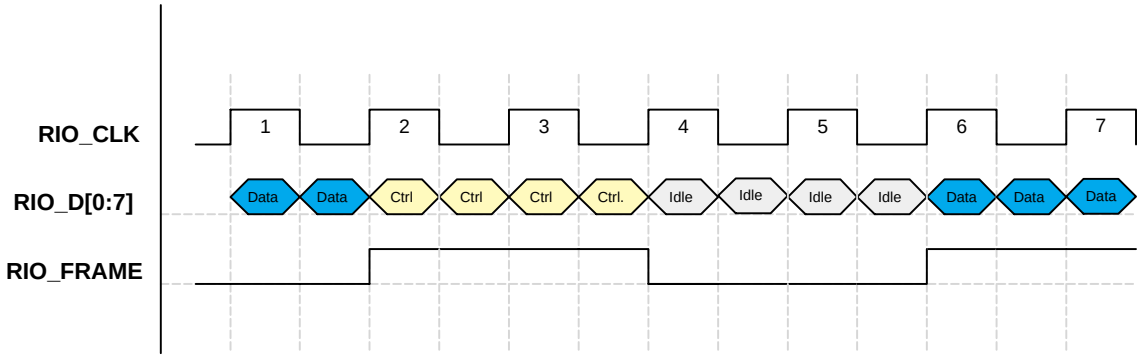

Figure 5: **RapidIO End of Packet, Idle, and Start of New Data Packet**

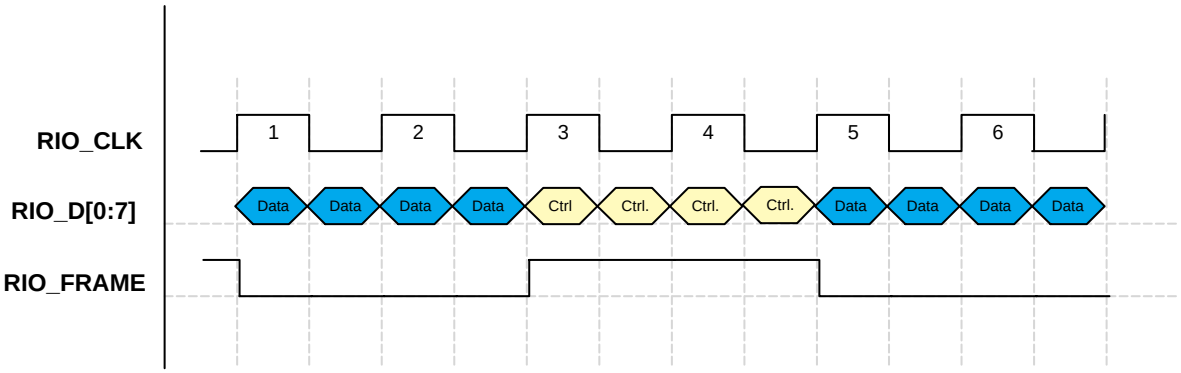

Figure 6: **RapidIO End of Packet and Start of New Packet**

Signal Description

The following tables list the User Application interface signals. Table 2 lists the system inputs to the core. Table 3 lists the incoming or Rx User interface. Table 4 lists the outgoing or Tx user interface. Table 5 lists the Time-of-Day signal interface and Table 6 lists the management interface for the RapidIO configuration registers.

Table 2: **System Input Signals**

Name	Direction	Type	Description
SYS_RESET_N	Input	Active Low	Asserted resets the Physical Layer Core.
TX_250	Input	Active High	DDR Clock
TX_250I	Input	Active High	DDR Inverted Clock
TX_62	Input	Active High	Internal Transmit Data Clock
TX_LOCKED	Input	Active High	TX DCM Locked

Table 3: **Rx Interface - Physical Layer to User**

Name	Direction	Type	Description
LNK_RESET_N	Output	Active Low	Resets the link. During startup, this signal will remain asserted until both LNK_RCLK and LNK_TCLK are stable. During normal operation, this signal will assert if four ininterrupted Link Request Reset control symbols are received or any of the DCMs lose lock.
LNK_RX BUF_STAT [0:3]	Output	n/a	Buffer Status. 4-bit bus which indicates the number of maximum sized packets that the connected device can receive.
LNK_RRDY_N	Output	Active Low	Receive Ready. When asserted indicates that the RapidIO receive port is operational.

Table 3: **Rx Interface - Physical Layer to User *(Continued)***

Name	Direction	Type	Description
LNK_RCLK	Output	n/a	Receive Data Clock. Clock derived from **RIO_RCLK** for the Rx end (forwarded clock). Data and control signals on this link must meet setup and hold times with respect to the rising edge of this clock.
LNK_RD [0:63]	Output	n/a	Data Out. 64-bit data bus from the Physical Layer to the User Application.
LNK_RREM [0:2]	Output	n/a	Data Remainder. 3-bit bus that indicates remaining bytes of data being presented by the Physical Layer to the User Application in the last data word when **LNK_REOF_N** is asserted. Number of valid bytes is equal to the value on **LNK_RREM** + 1.
LNK_RSOF_N	Output	Active low	Start of Frame indication. Used to indicate that the data being presented on the **LNK_RD** bus is the first word of the packet.
LNK_REOF_N	Output	Active low	End of Frame indication. Used to indicate that the data being presented on the **LNK_RD** bus is the last word of the packet.
LNK_RSRC_RDY_N	Output	Active low	Source Ready indication. Used to indicate that the source is presenting valid data on **LNK_RD**.
LNK_RSRC_DSC_N	Output	Active low	Source Discontinue indication. Used to indicate that the Physical Layer wants to discontinue the transfer of the current packet.
LNK_RDST_RDY_N	Input	Active low	Destination Ready indication. Used to indicate that the User Application can accept the data presented on **LNK_RD** during this clock cycle.
LNK_RDST_DSC_N	Input	Active low	Destination Discontinue indication. Used to indicate that the User Application wants to discontinue the transfer of the current packet. Upon recognition of this signal, the Physical Layer will stop the transfer of this packet.

Table 4: **Tx Interface - Physical Layer from User**

Name	Direction	Type	Description
LNK_TCLK	Output	n/a	Transmit Data Clock. Data and control signals on this link must meet setup and hold times with respect to the rising edge of this clock.
LNK_BUF_STAT [0:3]	Input	n/a	Buffer Status. 4-bit bus which indicates the number of maximum sized packets that the Buffer is ready to receive.
LNK_TRDY_N	Output	Active Low	Transmit Ready. Used to indicate that the RapidIO transmit port of the link is operational.
LNK_TD [0:63]	Input	n/a	Data In. 64-bit data bus from the user application to the Physical Layer.
LNK_TREM [0:2]	Input	n/a	Data remainder indication. 3-bit bus that indicates remaining bytes of data being presented by the User Application to the Physical Layer in the last data word when the **LNK_TEOF_N** is asserted. Number of valid bytes is equal to the value on **LNK_TREM** + 1.
LNK_TSOF_N	Input	Active Low	Start of Frame indication. Used to indicate that the data being presented on the **LNK_TD** bus is the first word of the packet.
LNK_TEOF_N	Input	Active Low	End of Frame indication. Used to indicate that the data being presented on the **LNK_TD** bus is the last word of the packet.
LNK_TSRC_RDY_N	Input	Active Low	Source Ready indication. Used to indicate that the source is presenting valid data on **LNK_TD**.
LNK_TSRC_DSC_N	Input	Active Low	Source Discontinue indication. Used to indicate that the user application wants to discontinue the transfer of the current packet.
LNK_TDST_RDY_N	Output	Active Low	Destination Ready indication. Used to indicate that the Physical Layer can accept the data presented on **LNK_TD** during this clock cycle.
LNK_TDST_DSC_N	Output	Active Low	Destination Discontinue indication. Used to indicate that the Physical Layer wants to discontinue the transfer of the current packet. Upon recognition of this signal, the user application should stop the transfer of this packet.
LNK_TLAST_ACK[0:2]	Output	n/a	Last Valid Acknowledgement. 3-bit bus that indicates to the user application the ackID of the last frame to be accepted. The user application may then release for reuse the buffer that holds the frame.

Table 4: **Tx Interface - Physical Layer from User *(Continued)***

Name	Direction	Type	Description
LNK_TNEXT_FM[0:2]	Output	n/a	Next Frame to Transmit. 3-bit bus that indicates to the user application the ackID of the next frame to be transmitted if the link is idle. If the link is not idle, this bus indicates to the User Application the ackID of the packet currently being transmitted.
LNK_TFORCE_TRAIN_N	Input	Active Low	Asserting this signal for one LNK_TCLK cycle will cause the core to re-enter the link phasing / training process. All data in the transmit direction should be stopped before this signal is asserted. Any data inbound should be recovered via the RapidIO link protocol. This signal may be asserted if multiple errors are seen on the link. If the RIO_RCLK has drifted over time, asserting this signal will re-align the receive clock.
MASTER_ENABLE	Output	n/a	Master Enable. Reflects the state of bit 1 in the Port General Control CSR.

Table 5: **Time of Day Signal Interface**

Name	Direction	Type	Description
LNK_TOD_REQ_N	Input	Active Low	Time of Day Request. Asserting this signal results in the transmission of a Time-of-day synchronization packet control symbol. This signal should be asserted until the Physical Layer acknowledges it with the **LNK_TOD_ACK_N** signal.
LNK_TOD_ACK_N	Output	Active Low	Time of Day Acknowledgement. This signal is asserted by the Physical Layer when it has transmitted a Time-of-day synchronization packet control symbol.
LNK_TOD_DEC_N	Output	Active Low	Time of Day Decode. The assertion of this signal indicates that a time-of-day synchronization packet control symbol has been decoded.

Table 6: **CSR and CAR Management Interface**

Name	Direction	Type	Description
MGT_CLK	Input	n/a	Management Clock. Management read and write cycles are made with respect to this clock.
MGT_RESET_N	Input	Active Low	Reset Signal. Initializes the OLLM and asserts the reset to the OPLM.
MGT_RD_N	Input	Active Low	Management Register Read.
MGT_WR_N[0:3]	Input	Active Low	Management Register Write. 4-bit bus that indicates the valid byte lanes to write.

Table 6: **CSR and CAR Management Interface *(Continued)***

Name	Direction	Type	Description
MGT_SEL_N	Input	Active Low	Management Device Select. Asserted enables the Management read or write operations to the Physical Layer.
MGT_A[0:11]	Input	n/a	Management Register Address. 12-bit address bus to access the Management registers.
MGT_DI[0:31]	Input	n/a	Data In.
MGT_DO[0:31]	Output	n/a	Data out.
Mgt_rdy_n	Output	Active low	Management Ready indication. Asserted indicates that the Management Interface is ready for data.
Mgt_int_n	Output	Active low	Management Interrupt.

Waveforms

The following waveforms demonstrate the operation of the Physical Layer user/application Interface signals. Figure 6 illustrates the transfer of three data packets from the Physical Layer to the user application. **LNK_RDST_RDY_N** and **LNK_RDST_DSC_N** are signals driven by the user while the remaining signals are driven by the Physical Layer. **LNK_RSOF_N** signals the start of a packet and **LNK_REOF_N** signals the end of a packet. These signals have to be asserted in conjunction with **LNK_RSRC_RDY_N** to indicate that a valid 64-bit data word is present on the data bus. The fig ure illustrates the start of packet one during clock 1. Since the user has asserted **LNK_RDST_RDY_N** during this clock, the data is accepted on clock edge 2. During clock 2, **LNK_RSRC_RDY_N** is deasserted to indicate that the Physical Layer is not qualifying valid data. During clock 3, **LNK_RSRC_RDY_N** is asserted. Since **LNK_RDST_RDY_N** is also asserted du ing this clock, data is accepted on clock edge 4. This operation continues until **LNK_REOF_N** is asserted in conjunction with **LNK_RSRC_RDY_N** and **LNK_RDST_RDY_N** during clock 7 to accept the last piece of data on clock edge 8. It is important to note that during clock 7, **LNK_RSOF_N** is also asserted. This signals a start of a new packet. When **LNK_RSOF_N** and **LNK_REOF_N** are asserted on the same clock edge, the upper 32-bit word belongs to the current packet (packet one) and the lower 32-bit word belongs to the next packet (packet two). The completion of the second packet transfer occurs on clock edge 11, where **LNK_REOF_N** is asserted, but **LNK_RSOF_N** is not asserted to indicate that the complete 64-bit word belongs to the second packet. The third packet transfer begins during clock 11, where **LNK_RSOF_N** is asserted. This transfer completes on clock edge 14.

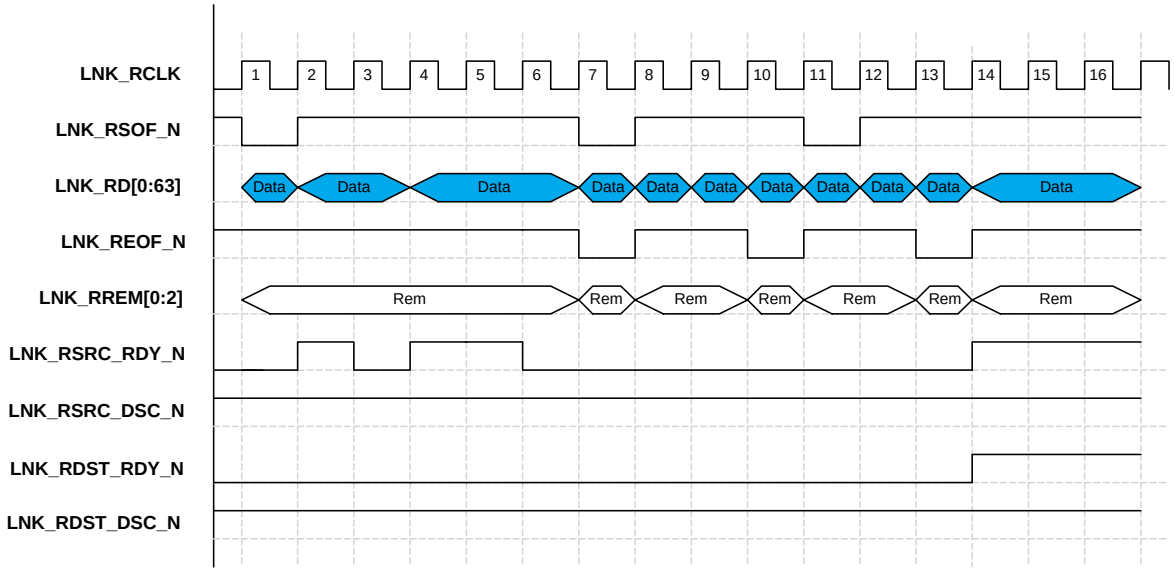

Figure 7: **Transfer of Data from Physical Layer to User Interface - Rx Path**

Figure 7 illustrates the transfer of three data packets from the user application to the Physical Layer. **LNK_TDST_RDY_N** and **LNK_RDST_DSC_N** are signals driven by the Physical Layer while the remaining signals are driven by the user. **LNK_TSOF_N** signals the start of the packet and **LNK_REOF_N** signals the end of a packet. These signals have to be asserted in conjunction with **LNK_TSRC_RDY_N** to indicate that a valid 64-bit data word is present on the data bus. The figure illustrates the start of packet one during clock 1. Since the Physical Layer has asserted **LNK_TDST_RDY_N**, the data is accepted on clock edge 2. Data continues to transfer during clock 2 and clock 3. During clock 4, the Physical Layer deasserts **LNK_TDST_RDY_N**. Data is held on the data bus until the Physical Layer asserts **LNK_TDST_RDY_N** and data is accepted on clock edge 6. During clock 7, the User indicates the end of a packet by asserting **LNK_TEOF_N**. The last piece of data for packet one is accepted on clock edge 8. During clock 8, the User drives **LNK_TSOF_N** asserted but deasserts **LNK_TSRC_RDY_N**. Since both signals are required to signal the start of a packet, the first 64-bit data word of packet two is not accepted until clock edge 10. The last data word of packet two is accepted on clock edge 11 when **LNK_TEOF_N**, **LNK_TSRC_RDY_N** and **LNK_TDST_RDY_N** are asserted. The transfer of packet three begins during clock 12 and co pletes on clock edge 14.

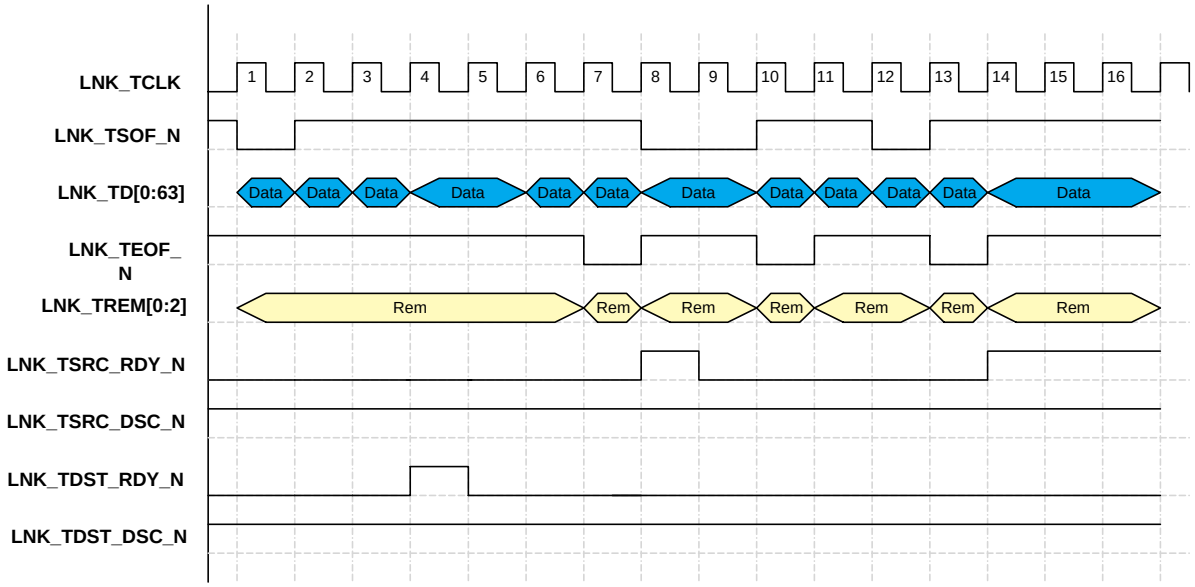

Figure 8: **Transfer of Data to Physical Layer from User Interface - Tx Path**

Figure 8 illustrates reading and writing to the Physical Layer CAR/CSR space through the Management Interface. The user initiates a read to a management register during clock 2 by asserting **MGT_RD_N** with a valid address on **MGT_A[0:11]**. During clock 3, the user drives the next address to be read even though it has not yet received the read data for AD1. The Physical Layer responds to the request during clock 3 by asserting **MGT_RDY_N** and providing the read data on **MGT_DO[0:31]**. The Physical Layer also samples **MGT_A[0:11]** and **MGT_RD_N** to see next request and provides the corresponding data on **MGT_DO[0:31]** during clock 4 while asserting **MGT_RDY_N**.

A write to the management registers is illustrated during clocks 6-9. The User initiates a write to a management register dur ing clock 6 by asserting all byte enables on **MGT_WR_N[0:3]** and driving the address and data on **MGT_A[0:11]** and **MGT_DI[0:31]** respectively. Unlike a read, the user must wait for the Physical Layer to accept the data before advancing to the next address and data. The Physical Layer accepts the data on clock edge 8 by asserting **MGT_RDY_N**. The user then drives the next address and data during clock 8. The Physical Layer accepts the write on clock edge 9.

Configuration Space Support

The Physical Layer core is configured as a generic end-point device with one port. Table 7 lists the complete configuration register map as defined in the RapidIO Physical Layer 8/16 LP-LVDS Specification. The shaded region contains registers for additional ports and is not supported. Definition of the registers can be found in the RapidIO Physical Layer 8/16 LP-LVDS Specification.

Table 7: **RapidIO Configuration Space Register Map**

Port	Register Name (word 0)	Register Name (word 1)	Block byte Offset - 0xh
General	8 / 16 LP-LVDS Port Maintenance Block Header		**000h**
	Reserved		**008-018h**
	Port Link Time-Out Control CSR	Port Response Time-Out Control CSR	**020h**
	Reserved		**028h**
	Reserved		**030h**
	Reserved	Port general Control CSR	**038h**
Port 0	Reserved		**040h**
	Reserved		**048h**
	Reserved		**050h**
	Port 0 Error and Status CSR	Port 0 Control CSR	**058h**
Port 1	Reserved		**060h**
	Reserved		**068h**
	Reserved		**070h**
	Port 1 Error and Status CSR	Port 1 Control CSR	**078h**
Port 2-14	Assigned to port 2 – 14 CSRs		**080-218h**
Port 15	Reserved		220h
	Reserved		228h
	Reserved		230h
	Port 15 Error and Status CSR	Port 15 Control CSR	238h

Supported Commands

Stype	Sub_type[0:2]	Control Symbol Command	RapidIO End-point
000	N/A	Packet - Accepted	Yes
001	N/A	Packet - Retry	Yes
010	N/A	Packet - Not Accepted	Yes
100	000	Idle	Yes
100	001	Stomp	Yes
100	010	End Of Packet (eop)	Yes
100	011	Restart- From - Retry	Yes

Stype	Sub_type[0:2]	Control Symbol Command	RapidIO End-point
100	100	Throttle	Yes
100	101	Time-Of-Day Synchronization	Yes
101	-	Link Request	Yes
110	-	Link Response	Yes

Verification

Xilinx has developed an internal system level testbench that allows the simulation of an open RapidIO environment. This environment makes use of the RapidIO Bus Functional Model (BFM) originally developed by Motorola. In this environment, the RapidIO Physical Layer Interface can be tested by itself or with other simulatable RapidIO agents. Included in these agents are a switch, an end point, a signal recorder, and a protocol monitor. This powerful internal test environment has been used to verify the RapidIO Physical Layer Interface. The RapidIO BFM is available to members of the RapidIO Trade Association. Please visit http://www.rapidio.org for more information on how to obtain this model.

Ordering Information

This Xilinx LogiCORE module is provided under the SignOnce IP Site License. A free evaluation version of the module is available.

Once purchased, the core may be downloaded from the Xilinx IP Center for use with the Xilinx CORE Generator system v6.3i SP2 and later. The Xilinx CORE Generator system is bundled with all Alliance and Foundation Series software packages, at no additional charge.

Please contact your local Xilinx sales representative or visit the Xilinx Silicon Xpresso Cafe for pricing and availability on Xi inx LogiCORE modules and software. Information on additional Xilinx LogiCORE modules is available on the Xilinx IP Center.

Revision History

The following table shows the revision history for this document.

Date	Version	Revision
07/02	1.1	Initial Xilinx Release
01/03	1.3	Converted to new template.
07/03	1.3.1	Updated to Version 1.3.1.
11/15/03	1.3.2	Updated to Version 1.3.2.
05/17/04	2.1	Updated for Version 2.1
11/11/04	2.2	Updated to support Xilinx software v6.3i SP2.
12/15/04	3.0	Updated document to support core v3.0 and Xilinx software v6.3i SP3.
6/29/05	3.0.1	Updated version number to 3.0.1.