XILINX® LogiCORE™

RapidIO Processor Buffer

DS241 (v1.0) December 20, 2002

Interface Specification

Introduction

The RapidIO Processor Buffer provides an interface between the Xilinx Processor Local Bus—Intellectual Property Interface (PLB-IPIF) and the Xilinx 8-bit LP/LVDS RapidIO Physical Layer module.

The RapidIO Physical Layer implements an 8-bit parallel LVDS port for data transfer off chip. The RapidIO Processor Buffer interfaces with the processor through the PLB-IPIF module using the IP Interconnect (IPIC).

This document explains the RapidIO Processor Buffer, and guides users through a design example.

Features

The PLB RapidIO LVDS is a soft IP core designed for Xilinx FPGAs incorporating the IBM PowerPC™ 405 and MicroBlaze™ processing elements. The design provides these features:

- Front end Interface to the IBM® CoreConnect™ PLB Bus
  - Supports PLB signaling per the IBM 64-Bit Processor Local Bus, Architectural Specification (v3.x)
  - Integrates easily with the Xilinx Platform Generator for PPC405 System Development
  - 64-bit wide data transfers
  - 512x64 Tx and Rx Packet Buffers (up to 8 maximally sized packets can be queued for Tx and Rx)
  - PLB Cacheline and Burst Transfer Interface Support with Packet Buffers
  - Parameterized System Address Block
- Back end Interface to RapidIO Bus
  - Incorporates Xilinx LogiCORE™ RapidIO Physical Layer
  - Supports RapidIO Physical Layer 8/16 LP-LVDS Interconnect Specification v1.1
  - 8-bit LVDS Physical Layer (Tx and Rx functions)
  - 500 MBytes/sec Peak Transfer Rate at the Physical Layer Tx and Rx ports
- Processor Accessible Registers
  - Interrupt Enable and Status Registers
  - S/W Reset/MIR Register
  - RapidIO Physical Layer Link Status Register
  - RapidIO Physical Layer Management Register Set
- System Interrupt Support
  - Tx and Rx Flow Control Interrupts
  - Programmable Enables/Disables
  - Interrupt Status Registers can support S/W Polled Mode control flow in place of Interrupt Control Flow
- PLB System clock frequency up to 100 MHz

LogiCORE™ Facts				
Core Specifics				
Supported Device	2VP7FF896-6			
Resources Used	**I/O**[1]	**LUTs**[2]	**FFs**[2]	**Block RAMs**
	271	5480	2600	4
Provided with Core				
Documentation	RapidIO Processor Buffer Interface Specification			
Design File Formats	VHDL, Xilinx specific netlists			
Constraints File	rio_proc.ucf			
Verification	Simulation models provided for the PLB-IPIF and RapidIO Physical Layer			
Instantiation Template	rio_top.vhd			
Design Tool Requirements				
Xilinx Implementation Tools	Xilinx 5.1i SP2			
Synthesis	Xilinx Synthesis Tool (XST) 5.1i SP2			
Support				
Support provided by Xilinx, Inc.				

Notes:
1. Total number of RapidIO I/O is 40; remaining 221 I/Os are the PLB interface.
2. Includes the 8-bit LP/LVDS RapidIO Physical Layer and IPIF Block

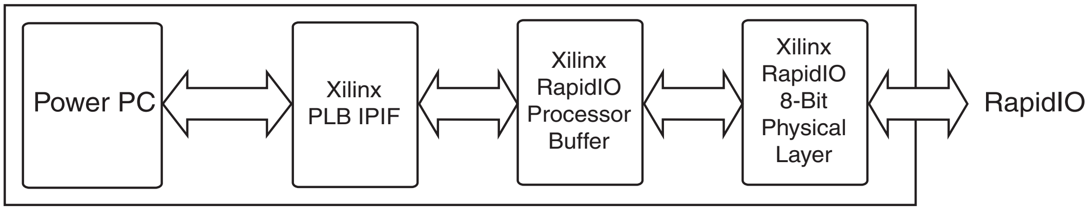

Figure 1: **RapidIO Processor Solution**

General Description

The RapidIO Processor Buffer stores RapidIO transaction packets between the RapidIO Physical Layer and the IBM PowerPC 405 processor. This buffering is needed to implement the reliable transport features of the RapidIO specification.

The RapidIO Processor Buffer interfaces to the IBM processor through the Xilinx Processor Local Bus Intellectual Property Interface (PLB-IPIF). The PLB-IPIF architecture specification facilitates the connection of Xilinx or customer IP modules to the IBM Processor Local Bus (PLB). The PLB is part of IBM's CoreConnect™ family of data buses and associated infrastructure. CoreConnect is intended for use in system-on-chip environments, including Xilinx Virtex-II Pro FPGAs with the embedded PowerPC processor.

The RapidIO Processor Buffer interfaces to the RapidIO Physical Layer interface using the Xilinx LocalLink™ Interface, which is a high-performance, synchronous, point-to-point interface, designed to serve as a user interface to the Xilinx system interfaces IP solutions. The interface defines a set of protocol agnostic signals that allow the transfer of protocol data units.

In addition to implementing the data and buffer management functions, the buffer acts as a clock domain transformer by supporting these three independently clocked interfaces:

- The Intellectual Property Interconnect (IPIC) - synchronized by the Processor Bus Clock
- The Transmit Interface - sourced by the Physical Layer Tx LocalLink Interface at 62.5 MHz
- The Physical Layer Receive Interface - sourced by the Physical Layer Rx LocalLink Interface at 62.5MHz

Figure 1 shows the RapidIO Processor solution: the PLB-IPIF, the RapidIO Processor Buffer, and the RapidIO 8-bit Port Physical Layer.

Functional Description

This section describes the four interfaces of the RapidIO Processor Buffer, as shown in Figure 2. These are the interfaces:

- Intellectual Property Interconnect (IPIC)
- Management Bus Interface (MGMT)
- Transmit Interface (Tx)
- Receive Interface (Rx)

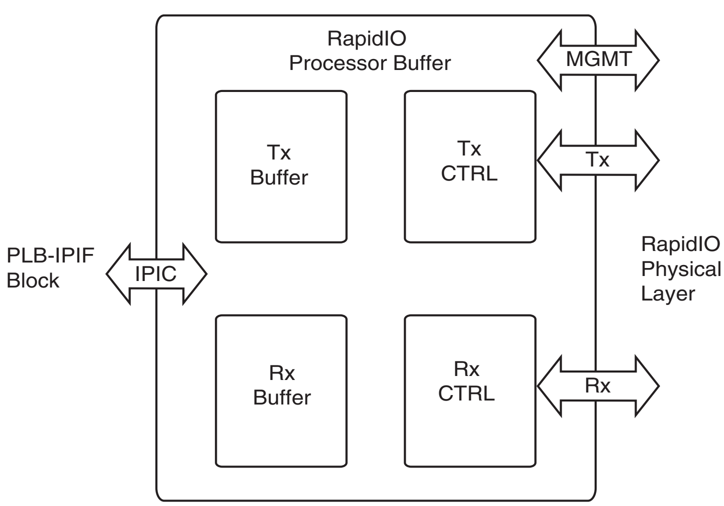

Figure 2: **RapidIO Processor Buffer Block Diagram**

Figure 3 shows a more detailed block diagram of the RapidIO Processor Buffer.

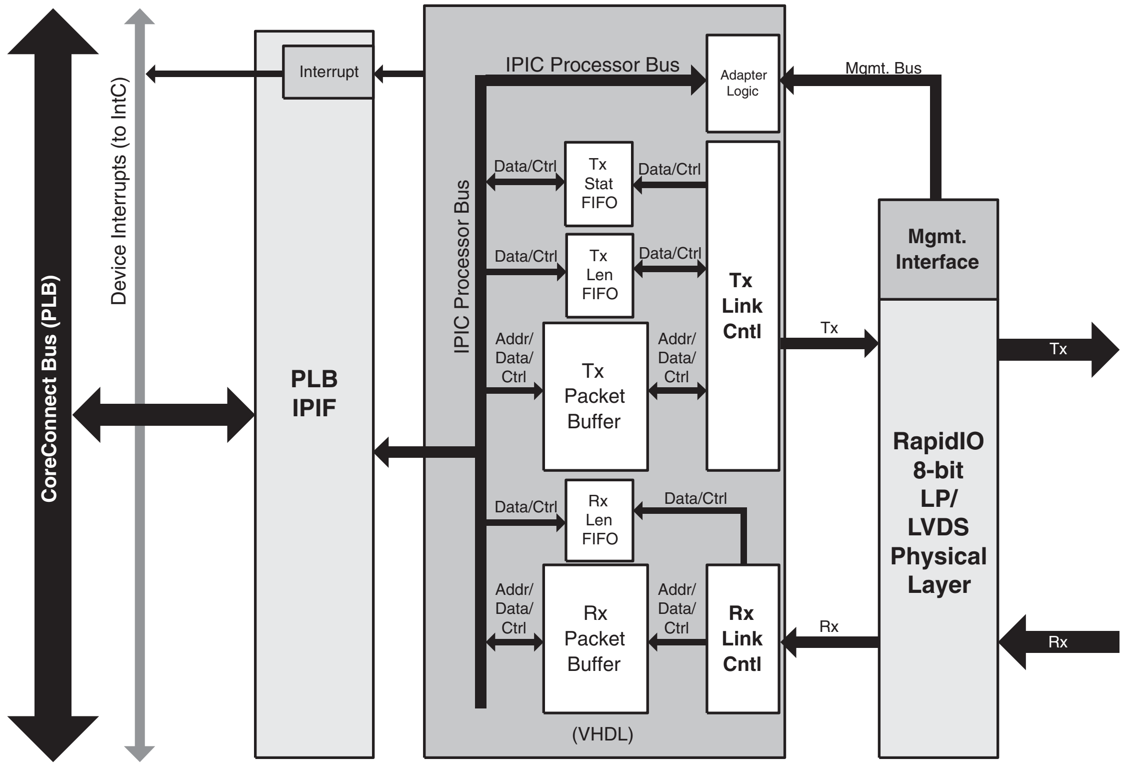

Figure 3: **RapidIO Processor Buffer Detailed Block Diagram**

Intellectual Property Interconnect (IPIC) Interface

The IPIC interface is used to connect the RapidIO Processor Buffer to the PLB-IPIF block. IPIC is a common structure used to connect a variety of IP to the Xilinx IPIF blocks. IPIF is a standardized block developed by Xilinx to link IP modules to IBM CoreConnect bus structures. For each variant of CoreConnect, a different IPIF is required. The RapidIO Processor solution is targeted to the PLB and therefore used the PLB-IPIF.

Management Interface

The Processor Buffer provides a simple translation function to adapt the IPIC protocol and signal set to that needed by the RapidIO Physical Layer Management Interface. Since the Physical Layer Management Interface is provided the PLB Bus clock for synchronization, the Processor Buffer does not have to provide a clock domain transformation.

Transmit Interface

The Processor Buffer Transmit Interface has four major functions:

- Tx Packet Buffer
- Tx Status FIFO
- Tx Length FIFO
- Tx Link Controller

Tx Packet Buffer

The Tx Packet Buffer is a temporary storage agent for RapidIO Transmit Packet Data.The buffer is comprised of 512 word X 64-bit Xilinx BlockRAM that are each broken down into 8 bins of 64 quadwords. This is a dual port configuration, which supports independent clocking of two ports, A and B. Transmit packet data is written into port A from the processor bus and is then read from port B of the buffer by the Tx Link Controller logic and transferred to the Physical Layer Tx LocalLink Interface.

Tx Status FIFO

The Tx Status FIFO is a 15 word deep and 8 bits wide status register for the Tx Link Controller, which informs the processor bus that a Tx Packet transmission is complete. When a Packet transmission completes, the Rx Link controller writes the Acknowledgement ID (Ack ID) of the Packet. The presence of one or more entries to the Tx Status FIFO causes an interrupt to be asserted to the User Application. The Interrupt enables the User Application to mange its Tx Packet completion queue.

Tx Length FIFO

The User Application uses the Tx Length FIFO to tell the Tx Link Controller that a Tx Packet is available in the Tx Packet Buffer and can be transmitted.

Tx Link Controller

The Tx Link Controller (TLC) writes Tx packet data into the Tx Link Port of the RapidIO Physical Layer. When the Tx Length FIFO specifies that at least one Tx Packet is present in the Tx Packet Buffer, the TLC reads the Length FIFO. The TLC sequentially reads the packet words from the Tx Packet Buffer and writes it to the Physical Layer LocalLink Interface.

Receive Interface

The Processor Buffer Receive Interface consists of four functions:

- Rx Packet Buffer
- Rx Length FIFO
- Rx Acknowledge Port
- Rx Link Controller

Rx Packet Buffer

The Rx Packet Buffer is a temporary storage agent for RapidIO Receive Packet Data. It consists of 512 words X 64-bits Xilinx Block RAM, divided into 8 bins of 64 quadwords. This is a dual port configuration with independent clocking on Port A and B. The Rx Link controller receives Packet Data from the Physical Layer Rx LocalLink Interface and writes it into the Rx Packet Buffer Port B. The User Application reads the data from the Rx Packet Buffer Port A.

Rx Length FIFO

The Rx Length FIFO is used by the Rx Link Controller to communicate with the User Application. An entry into the FIFO issues an interrupt to the User Application, signaling that a readable Rx Packet is available in the Rx Packet Buffer. The FIFO is a dual clock design and provides clock domain transformation between the processor bus clock and the RapidIO Physical Layer Rx Clock. The FIFO is written by the Rx Link Controller.

Rx Acknowledge Port

The User Application acknowledges the reception of an Rx Packet by writing to the Rx Acknowledge Port. This port has no data value associated with it so any data value can be output during the write. The act of writing to this port causes the Rx Link Controller to update its Rx Packet Buffer Bin pointers and the buffer availability status to the Physical Layer.

Rx Link Controller

The Processor Buffer manages the Physical Layer Rx LocalLink Interface using the Rx Link Controller (RLC). The RLC receives data from the Physical Layer interface and writes to the Rx Packet Buffer. Up to eight packets can be stored in the Rx Packet Buffer. If the Rx Packet Buffer becomes full, the RLC stops accepting data from the RapidIO Physical Layer. The RLC provides count value for RapidIO Physical Layer that specifies available bins to write. When a packet is transferred, the RLC writes associated packet descriptor data into the Rx Length FIFO. This issues an interrupt to the User Application, which must write to the acknowledge port when data is retrieved. This specifies that a write-able bin is empty.

Design Port Specifications

The following tables list and give descriptions of the Processor Buffer interface signal set.

Table 1: **Rapid IO Processor Buffer IPIC Signal Set**

Name	Direction	Description
Bus_Clk	In	Input clock from the Processor Bus
Bus_Reset	In	Master reset for the Processor Bus
Bus2Mgmt_RdCE	In	Active high chip enable used to select the Physical Layer Management Interface during processor read operations.
Bus2Mgmt_WrCE	In	Active high chip enable used to select the Physical Layer Management Interface during processor write operations
Bus2RxBuf_RdCE	In	Active high chip enable used to select the Processor Buffer Rx Packet Buffer during processor read operations of that function
Bus2RxBuf_WrCE	In	Active high chip enable used to select the Processor Buffer Rx Packet Buffer during processor write operations to that function
Bus2RxAck_WrCE	In	Active high chip enable used to select the Processor Buffer Rx Acknowledge Port during processor write operations to that function
Bus2RxLen_RdCE	In	Active high chip enable used to select the Processor Buffer Rx Length FIFO during processor read operations of that function
Bus2TxBuf_RdCE	In	Active high chip enable used to select the Processor Buffer Tx Packet Buffer during processor read operations of that function
Bus2TxBuf_WrCE	In	Active high chip enable used to select the Processor Buffer Tx Packet Buffer during processor write operations to that function
Bus2TxLen_WrCE	In	Active high chip enable used to select the Processor Buffer Tx Length FIFO during processor write operations to that function
Bus2TxStat_RdCE	In	Active high chip enable used to select the Processor Buffer Tx Status FIFO during processor read operations of that function
Bus2IP_Addr [0:31]	In	Input address bus from the processor interface used to address the Packet Buffers and the Physical Layer Management Interface Registers
Bus2IP_BE[0:7]	In	Active high Byte Enable signals from the processor bus used when writing or reading words or long words to/from the Packet Buffers
Bus2IP_Data [0:63]	In	Processor write data bus
Bus2IP_RdReq	In	Active high signal used to qualify the Read chip Enable input signals
Bus2IP_WrReq	In	Active high signal used to qualify the Write chip Enable input signals
IP2Bus_Data [0:63]	Out	Processor read data bus
IP2Bus_RdAck	Out	Active high output signal indicating that the requested read data is valid and stable on the read data bus
IP2Bus_WrAck	Out	Active high output signal indicating that the requested write of the data on the write data bus has been accepted
IP2Bus_Retry	Out	Active high bus operation retry request. This is tied low, and currently not used by the Processor Buffer.
IP2Bus_Error	Out	Active high retry request for the currently active bus operation. This is tied low, and currently not used by the Processor Buffer.

Table 1: **Rapid IO Processor Buffer IPIC Signal Set *(Continued)***

Name	Direction	Description
IP2Bus_ToutSup	Out	Active high request for suppression of the watchdog time out function monitoring the currently active bus operation. This is tied low, and currently not used by the Processor Buffer.
IP2Bus_IntEvent(0)	Out	Interrupt event 0—this active high interrupt output is connected to the Rx_lnk_rst_int signal, and is asserted whenever the Processor Buffer detects a Rx Link Reset assertion on the Physical Layer Rx LocalLink Interface.
IP2Bus_IntEvent(1)	Out	Interrupt event 1—this active high interrupt output is connected to the rxlen_pkt_avail_int signal. This interrupt indicates that there is at least 1 entry in the Rx Length FIFO. A corresponding Rx packet is available in the Rx Packet Buffer. The Bin# of the packet is in the data word read from the Rx Length FIFO.
IP2Bus_IntEvent(2)	Out	Interrupt event 2—this active high interrupt output is connected to the txstat_tack_int signal, and indicates that the Tx Status FIFO has at least one entry. This interrupt occurs when the RapidIO Physical Layer acknowledges the completed transmission of a packet and the Tx Link Controller writes the ackID (source Bin#) of the completed Tx packet.
IP2Bus_IntEvent(3)	Out	Interrupt event 3—this active high interrupt output is connected to the mgmt_int signal, and is asserted whenever the Physical Layer Management Interface asserts its interrupt request signal.
IP2Bus_IntEvent(4)	Out	Interrupt event 4—this active high interrupt output is connected to the rxlen_wr_err_int signal, and is asserted whenever the Rx Link Controller attempts to write to the Rx Length FIFO when it is already “Full”. This interrupt is provided as a system debugging aid and is not used operationally.
IP2Bus_IntEvent(5)	Out	Interrupt event 5—this active high interrupt output is connected to the txstat_rd_err_int signal, and is asserted whenever the User Application attempts to read the Tx Status FIFO when it is already “Empty”. This interrupt is provided as a system debugging aid and is not used operationally.
IP2Bus_IntEvent(6)	Out	Interrupt event 6—this active high interrupt output is connected to the txlen_wr_err_int signal. This interrupt occurs if the User Application attempts to write an entry into the Tx Length FIFO when it is already “Full”. This interrupt is provided as a system debugging aid and is not used operationally.
IP2Bus_IntEvent(7)	Out	Interrupt event 7—this active high interrupt output is connected to the Tx_Coherency_Err_Int signal. This interrupt occurs if the Bin # read out of the Tx Length FIFO by the Tx Link Controller (TLC) does not match the Lnk_RNext_FM value being output by the Rx Link port of the RapidIO Physical Layer. This condition means that the User Application and the Physical Layer are “out of sync” on the Packet Buffer Bin numbers being used for the Tx Packet data.

Table 2: **RapidIO Processor Buffer Tx LocalLink Interface Signal Set**

Name	Direction	Description
TxLnk2Buf_Clk	In	Input clock (62.5MHz) from the Physical Layer that is used for all Tx LocalLink Interface synchronization
TxLnk2Buf_Rdy_n	In	Active low signal indicating that the Physical Layer Tx channel is operational
TxLnk2Buf_LastAck [0:2]	In	This 3 bit bus indicates the last successfully transmitted packet ID. The packet ID is the same as the Tx Packet Buffer Bin # used for sourcing the packet data.
TxLnk2Buf_Next_FM [0:2]	In	This 3 bit bus indicates the Bin # to the Tx Link Controller that is to be used for the next Tx Packet to be transmitted.
TxLnk2Buf_Dst_Rdy_n	In	This active low signal indicates that the Physical Layer Tx LocalLink Interface is ready to receive Tx packet data.
TxLnk2Buf_Dst_Dsc_n	In	This active low signal indicates that the Physical Layer Tx LocalLink Interface is disconnecting and is unable to accept any packet data. The Tx Link controller attempts retransmission of the entire packet when the Physical Layer is ready.
Buf2TxLnk_Src_Rdy_n	Out	This active low signal indicates that the Processor Buffer Tx LocalLink Interface is outputting Tx packet data on the Buf2TxLnk_Data [0:63] data bus.
Buf2TxLnk_Src_Dsc_n	Out	This active low signal indicates that the Processor Tx Link Controller is disconnecting and is unable to send any packet data. This signal is currently not asserted by the Processor Buffer.
Buf2TxLnk_Data [0:63]	Out	This bus (64bits) is the Tx Packet Data transfer medium between the Processor Buffer and the Physical Layer.
Buf2TxLnk_SOF_n	Out	This active low signal indicates the Start Of Frame (the first word) of a Tx Packet.
Buf2TxLnk_EOF_n	Out	This active low signal indicates the End Of Frame (the last word) of a Tx Packet.
Buf2TxLnk_Rem	Out	This bus indicates the Remainder (REM) value of the last word of the Tx Packet and is valid when Buf2TxLnk_EOF_n is asserted.

Table 3: **RapidIO Processor Buffer Rx LocalLink Interface Signal Set**

Name	Direction	Signal Description
RxLnk2Buf_Clk	In	Input clock (62.5MHz) from the Physical Layer used for all Rx LocalLink Interface synchronization
RxLnk2Buf_Reset_n	In	Active low signal indicating that the Physical Layer has detected an externally initiated RapidIO link reset
RxLnk2Buf_Rdy_n	In	Active low signal indicating that the Physical Layer Rx channel is operational
RxLnk2Buf_Data [0:63]	In	This bus (64bits) is the Rx Packet Data transfer medium between the Physical Layer and the Processor Buffer.
RxLnk2Buf_Rem	In	This bus indicates the Remainder (REM) value of the last word of the Rx Packet and is valid when RxLnk2Buf_EOF_n is asserted.
RxLnk2Buf_SOF_n	In	This active low signal indicates the Start Of Frame (the first word) of a Rx Packet.
RxLnk2Buf_EOF_n	In	This active low signal indicates the End Of Frame (the last word) of a Tx Packet.
RxLnk2Buf_Src_Rdy_n	In	This active low signal indicates that the Physical Layer Rx LocalLink Interface is ready to send Tx packet data to the Processor Buffer.

Table 3: **RapidIO Processor Buffer Rx LocalLink Interface Signal Set *(Continued)***

Name	Direction	Signal Description
RxLnk2Buf_Src_Dsc_n	In	This active low signal indicates that the Physical Layer Rx LocalLink Interface is disconnecting and is unable to send any more Rx packet data. The Rx Link controller aborts the packet reception.
Buf2RxLnk_Buf_Stat [0:3]	Out	This bus indicates to the Physical Layer Rx LocalLink Interface how many Rx Packet Buffer Bins are available for filling with Rx Packet Data.
Buf2RxLnk_Dst_Rdy_n	Out	This active low signal indicates that the Processor Buffer Rx LocalLink Interface is ready to accept Rx packet data from the RxLnk2Buf_Data [0:63] data bus.
Buf2RxLnk_Dst_Dsc_n	Out	This active low signal indicates that the Processor Rx Link Controller is disconnecting and is unable to receive any packet data. This signal is currently not asserted by the Processor Buffer.

Table 4: **RapidIO Processor Buffer Management Interface Signal Set**

Name	Direction	Signal Description
Mgmt2Phy_Data [0:31]	Out	This bus (32bits) is used to write data to the RapidIO Physical Layer's Management Interface.
Mgmt2Phy_Sel_n	Out	This active low signal is used to select the RapidIO Physical Layer's Management Interface during read or write operations.
Mgmt2Phy_Rd_n	Out	This active low signal is asserted in conjunction with the Mgmt2Phy_Sel_n signal to initiate a read operation of the RapidIO Physical Layer's Management Interface.
Mgmt2Phy_Wr_n	Out	This active low signal is asserted in conjunction with the Mgmt2Phy_Sel_n signal to initiate a write operation of the RapidIO Physical Layer's Management Interface.
Mgmt2Phy_Addr [0:11]	Out	This bus is used to supply the desired register address to the Physical Layer during read or write operations of the RapidIO Physical Layer's Management Interface.
Phy2Mgmt_Data [0:31]	In	This data bus (32bits) returns the read data requested of the Physical Layer during read operations.
Phy2Mgmt_Rdy_n	In	This active low signal indicates to the Processor Buffer that the requested operation of the Physical Layer's Management Interface is being accepted (in case of a write) or the data is valid on the Phy2Mgmt_Data [0:31] bus (in case of reads).
Phy2Mgmt_Int_n	In	This active low interrupt signal is an indication that the RapidIO Physical Layer's Management Interface is requesting service.

RapidIO Processor Buffer Design Example

The RapidIO Processor Buffer is part of the suite of processor IP delivered by the Xilinx Embedded Development Toolkit (EDK). For the purpose of this design example, it is delivered as a standalone IP with the RapidIO Physical Layer IP as well as the pre-configured PLB-IPIF block.

Directory Structure:

Along with this application note comes a downloadable `.zip` file that can be used as a design example. The zip file maintains the following directory structure when extracted:

- **rio_proc**: the main project directory, which contains these four directories:
  - **docs**: this folder contains all of the documentation that comes with the RapidIO example.
  - **src_ref**: the source folder contains two directories: `netlist_src` and `vhdl`.
    The `netlist_src` directory contains the COREGen core netlists as well as the RapidIO Physical Layer netlist and the PLB-IPIF netlist.
    The `vhdl` directory contains all code for the RapidIO Processor Buffer, as well as the top level files of instantiate the RapidIO processor buffer, PLB-IPIF, and the 8-bit LP/LVDS RapidIO Physical Layer.
  - **impl**: this folder contains the 5.1i ISE project examples. `rio_proc_pc.npl` should be used when the operating system is Windows-based, and `rio_proc_unix.npl` for a UNIX-based operating system.
    These two folders are contained in this folder as well:
    - **src**: Contains all of the source files and netlists for use by the ISE tools. The source files are the same as those found in the main `/rio_proc/src_ref` folder. They have been provided in this directory for ease of use.
    - **ucf**: Contains the `.ucf` constraints file `rio_proc.ucf`. This constraints file contains all the RapidIO Physical Layer constraints as well as the any additional constraints for the PLB-IPIF and the RapidIO Processor buffer.
      Note: For the purpose of this design example, the source file `src/rio_proc.vhd` instantiates the RapidIO Processor solution in an FPGA with all RapidIO and PLB interface signals tied to input and output buffers. The user may instantiate this design with other modules connected to the PLB interface and, therefore, would not connect the PLB interface signals to the input and output buffers of the FPGA.
  - **sim**: This folder contains the simulation models for the PLB-IPIF and the RapidIO 8-bit LP/LVDS Physical Layer.

Implementation:

The user can synthesize and implement the design example in the `rio_pro/impl` directory using one of the supported flows. The user can run the Xilinx ISE tools on a Windows based OS using the GUI tools, or run on a UNIX based OS using either the GUI tools or the script based flow. Instructions for the Unix and Windows based GUI flow are provided below.

After opening up 5.1i ISE Project Navigator, select File > Open Project from the top menu bar, browse to location of the `rio_proc` directory and select the corresponding `.npl` file, and then click Open. Project Navigator then lists all the source files necessary for this project including the user constraints file.

In the Process window, the user can synthesize the design by right-clicking on the Synthesize process and selecting Run. The same can be done for the Implement process, which invokes the Translate, Map, and Place & Route process. Each processes should complete without errors.

Simulation:

This folder contains the simulation models for the RapidIO Physical Layer and the PLB-IPIF configured for this design example. However, a simulation environment for design example is currently not supported. These simulation blocks have been provided to allow the user to set up their own simulation environment where they may add any additional blocks that interface to the Processor Local Bus (PLB) with the use of EDK.

EDK also contains a placeholder for an element of the IBM CoreConnect Toolkit that allows the user to do a full functional simulation. The CoreConnect Toolkit is only available to CoreConnect licensees. EDK integrates seamlessly with the IBM CoreConnect Toolkit. This toolkit is not included with the EDK, but it is required if bus functional simulation is desired. The toolkit provides a number of features that enhance design productivity and allow you to get the most from the Embedded Development Kit.

To obtain the IBM CoreConnect Toolkit, you must be a licensee of the IBM CoreConnect Bus Architecture. Licensing CoreConnect gives you access to a wealth of documentation, Bus Functional Models, Hardware IP, and the Toolkit.

Xilinx provides a Web-based licensing mechanism that allows you to obtain the CoreConnect Toolkit from the Xilinx web site. To license CoreConnect, use an Internet browser to access:

http://www.xilinx.com/ipcenter/processor_central/register_coreconnect.htm.

The simulation directory is broken up into the following two folders that contain the corresponding files:

- `ipif_sim_model`
  `plb_ipif_rio.vhd`: The back-annotated VHDL simulation model for the PLB-IPIF. This file cannot be synthesized or used in any other manner other than simulation.
- `phy_sim_model`
  `phy_8_lvds.v`: The back-annotated Verilog simulation model for the LVDS Physical layer. This file cannot be synthesized, or used in any other manner other than simulation.

Revision History

The following table shows the revision history for this document.

Date	Version	Revision
12/20/02	1.0	Initial Xilinx release.