LogiCORE IP AXI External Peripheral Controller (EPC) (v1.00.a)

DS809 July 25, 2012 **Product Specification**

Introduction

This specification defines the architecture and interface requirements for the Xilinx® LogiCORE™ IP External Peripheral Controller (AXI EPC IP Core). The controller supports data transfers between the AXI4 (Advanced eXtensible Interface) and the external synchronous and/or asynchronous peripheral devices such as USB and LAN devices, which have processor interface.

Examples of peripheral devices supported by the AXI EPC include the 10/100 non-PCI Ethernet single chip (SMSC LAN91C111) from SMSC and CY7C67300 USB Controller from Cypress Semiconductor devices.

Features

- Connects as a 32-bit slave on AXI4-Lite Slave
- Byte enable support
- Parameterized support of up to four external peripheral devices with each device configured with separate base address and high address range
- Supports both synchronous and asynchronous access modes of peripheral devices with the support for a separate clock domain for synchronous peripheral devices
- Supports both multiplexed and non-multiplexed address and data buses
- The data width of peripheral devices is independently configured to 8-bit, 16-bit or 32-bit with the provision to enable data width matching when the AXI data width is greater than that of peripheral device
- Configurable timing parameters for peripheral bus interface
- Tested with the SMSC LAN91C111 and the Cypress CY7C67300 USB Controller device

LogiCORE IP Facts Table	
Core Specifics	
Supported Device Family[1]	Zynq™-7000[2], Virtex®-7[3], Kintex™-7[3], Artix™-7[3], Virtex-6[4], Spartan®-6 [5]
Supported User Interface	AXI4-Lite
Resources	See Table 5 through Table 9
Provided with Core	
Design File	ISE®: VHDL Vivado™: RTL
Example Design	Not Provided
Test Bench	Not Provided
Constraints File	None
Simulation Model	None
Supported S/W Driver	N/A
Tested Design Flows[6]	
Design Entry	Xilinx Platform Studio (XPS) Vivado Design Suite[7]
Simulation	Mentor Graphics ModelSim
Synthesis	Xilinx Synthesis Technology (XST) Vivado Synthesis
Support	
Provided by Xilinx@ www.xilinx.com/support	

Notes:

1. For a complete list of supported derivative devices, see the Embedded Edition Derivative Device Support.
2. Supported in ISE Design Suite implementations only.
3. For more information on the 7 series devices, see the *7 Series FPGAs Overview* [Ref 6].
4. For more information on the Virtex-6 devices, see the *Virtex-6 Family Overview* [Ref 5].
5. For more information on the Spartan-6 devices, see the *Spartan-6 Family Overview* [Ref 4].
6. For the supported versions of the tools, see the Xilinx Design Tools: Release Notes Guide.
7. Supports only 7 series devices.

Functional Description

The AXI External Peripheral Controller (AXI EPC) interface diagram shown in Figure 1 depicts the overall interfaces of the core design.

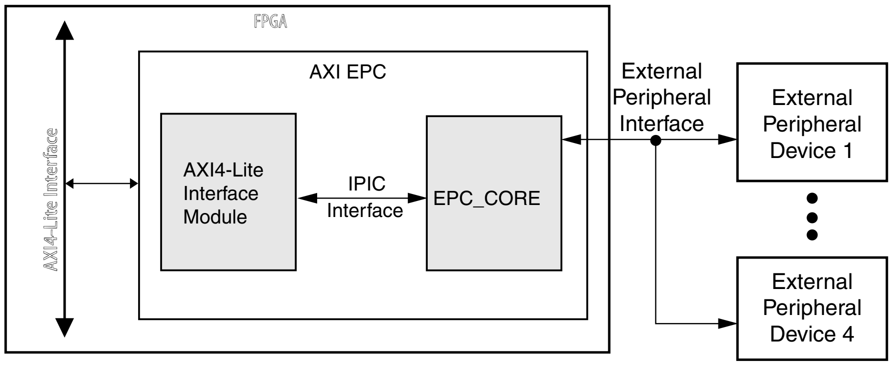
FPGA
AXI4-Lite Interface
AXI EPC
AXI4-Lite Interface Module
IPIC Interface
EPC_CORE
External Peripheral Interface
External Peripheral Device 1
External Peripheral Device 4

Figure 1: **AXI EPC Interface Diagram**

The AXI EPC IP Core design provides a general purpose interface to external peripheral devices and the AXI4. It is a AXI4 slave device. The AXI EPC IP Core can be configured to provide support for multiple external peripherals (non-memory peripherals like USB, LAN, and so on) up to a maximum of four devices and each device is independently configured to respond either in synchronous or in asynchronous mode. The timing parameters governing the access cycles such as setup/hold time, cycle access time, and cycle recovery time are configured by the user. It receives read or write operation commands from the AXI4 and generates a corresponding access cycle to one of the four peripheral devices. Xilinx recommends that peripherals such as LAN and USB, which have embedded interfaces, be used with the core.

The AXI EPC IP Core is comprised of the following modules:

- AXI4-Lite Interface Module
- EPC CORE

AXI4-Lite Interface Module

The AXI4-Lite Interface Module provides an interface between the EPC CORE and the AXI4. The AXI4-Lite interface module implements the basic functionality of the AXI4 interface operation and performs the necessary protocol and timing translation between the AXI4 and the IPIC interface.

EPC CORE

The EPC CORE provides an interface between the IPIC interface and the external peripheral devices. The EPC CORE consists of the logic necessary to convert the access cycles on the IPIC interface to the corresponding access cycles on the peripheral bus adhering to the device specific timing parameters.

The block diagram for the AXI EPC is shown in Figure 2.

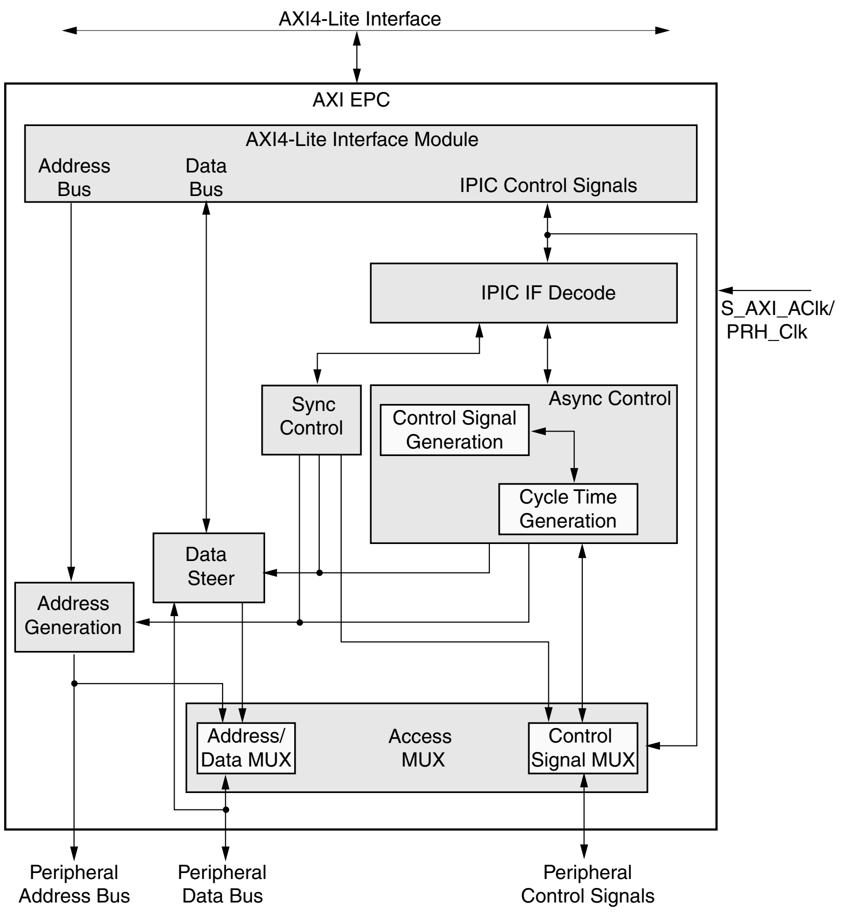

Figure 2: **AXI EPC Block Diagram**

The AXI EPC core consists of the following:

- AXI4-Lite Interface Module – Provides the address decoding logic and necessary interface between AXI4 and EPC core signals
- IPIC IF Decode Module – Provides decoding of IPIC signals of AXI4-Lite Interface Module and synchronization of control signals
- Sync Control Module – Implements the state machine which controls the synchronous interface
- Async Control Module – Implements the state machine which controls the asynchronous interface including the asynchronous timing parameters
- Data Steer Module – Provides the data bus width matching and data steering logic
- Address Generation Module – Provides the generation of the lower address bits
- Access MUX Module – Provides the multiplexing of address, data, and other control signals

A detailed description of these modules is provided in the following sections.

IPIC IF Decode Module

The IPIC IF Decode module implements the interface to the AXI. It also configures the EPC CORE and interfaces to the Async Control module and Sync Control module by driving the necessary control signals based on user parameter settings.

Sync Control and Async Control Modules

In Figure 2 the Sync Control and Async Control modules depict the synchronous and asynchronous paths of the EPC CORE. This ensures that the read and write accesses to the external device(s) adheres to the specific timing parameters defined for the external device(s). Implementation of the Sync Control and Async Control modules is dependent on the parameter C_PRHx_SYNC. If synchronous and asynchronous external peripheral devices exist simultaneously, both the Sync Control and Async Control modules are implemented.

The Sync Control module operates either on the AXI clock (`S_AXI_Clk`) or on the external peripheral clock (`PRH_Clk`) depending on the generic C_PRH_CLK_SUPPORT. If C_PRH_CLK_SUPPORT = 1, then the Sync Control module operates on external device peripheral clock (`PRH_Clk`) that is different from the AXI clock. Peripheral clock (`PRH_Clk`) frequency should always be less than the AXI clock frequency. If more than one device is synchronous, then the frequency for the `PRH_Clk` should be chosen as the minimum of the operating frequencies of those devices. The IPIC control signals that are inputs to the Sync Control module are synchronized to the `PRH_Clk` in the IPIC IF Decode module to indicate the start of a transaction. Similarly, the control signals from the Sync Control module to the IPIC interface such as data acknowledge are synchronized to the AXI clock in the IPIC IF Decode module. If C_PRH_CLK_SUPPORT = 0, the Sync Control module operates on the AXI clock and the IPIC Decode module interface does not perform synchronization of control signals.

The Async Control module operates on the AXI clock only. This module generates the control signals to the initiate read and write access cycles to the external peripheral device based on the asynchronous timing parameters set by the user.

Data Steer and Address Generation Modules

The data bus of the external device must be less than or equal to the AXI data width and can be 8, 16, or 32 bits. When the width of the external peripheral data bus is less than that of AXI and if C_PRHx_DWIDTH_MATCH = 1 for a particular device, then the Data Steer Module generates multiple read or write cycles to the external device to match a single access on the AXI.

To map a single 32-bit AXI access to multiple 8-bit or 16-bit accesses, the lower bits of the address bus are internally generated within the Address Generation module to provide the correct address to the external peripheral device. The address bus increments as each transaction completes.

For example, if the external device is 8 bits wide, four read or write cycles to the device are performed to match a single 32-bit read or write transaction of AXI. For a write cycle, the first byte of the AXI data bus (`S_AXI_WDATA[7:0]`) is presented on the peripheral data bus (`PRH_Data[0:7]`). When the external device accepts the transaction, a new write cycle is generated and the second byte of the AXI data bus (`S_AXI_WDATA[17:8]`) is presented on the peripheral data bus (`PRH_Data[0:7]`) and so on. When the last byte of the AXI data bus (`S_AXI_WDATA[31:24]`) is accepted by the peripheral, the data acknowledge signal is generated and sent to the AXI4-Lite Interface module to indicate that the access is complete on the peripheral interface.

Similarly, for a read cycle, when the external device indicates it is ready to complete the transaction, the data on the peripheral data bus (`PRH_Data[0:7]`) is internally registered as the first byte to be presented to AXI data bus (`S_AXI_RDATA[7:0]`) followed by initiation of new read cycle on the peripheral interface. The second read access on the peripheral data bus (`PRH_Data[0:7]`) is internally registered as the second byte to be presented to AXI data bus (`S_AXI_RDATA[15:8]`) and so on. When all four bytes are read from the external device, an acknowledge is generated to the AXI4-Lite Interface module to indicate that the data is ready to be transferred to AXI data bus.

When support for data width match is enabled for any of the external devices, then the access to that device should respect data alignment that is a half word access should be aligned to a 16-bit boundary and a word access should be aligned to a 32-bit boundary.

Access MUX Module

The interface to the external peripherals supports both multiplexed and non-multiplexed address and data bus to the external devices. The Access MUX module controls the multiplexing of the peripheral address and data buses based on the parameter C_PRHx_BUS_MULTIPLEX. If C_PRHx_BUS_MULTIPLEX = 1, the address and the data bus are multiplexed and presented to the corresponding external device on PRH_Data bus. The address is valid on the `PRH_Data` bus as long as the address strobe is active (PRH_ADS). This access is performed in two phases (address phase and data phase). The data phase is followed by the address phase. If C_PRHx_BUS_MULTIPLEX = 0, the address and data are presented to the device on separate buses (`PRH_Addr` bus and `PRH_Data` bus) and the access cycle contains only one phase.

Design Parameters

To allow the user to create the AXI EPC that is uniquely tailored for the user's system, certain features can be parameterized in the AXI EPC design. Some of these parameters control the interface to the AXI while others control the interface to the peripheral devices. This allows the user to have a design that utilizes only the minimum resources required by the system and runs at the best possible performance.

The features that are parameterizable in the AXI EPC core are shown in Table 1.

Inferred Parameters

In addition to the parameters listed in Table 1, there are also parameters that are inferred for each AXI interface in the Embedded Development Kit (EDK) tools. Through the design, these EDK-inferred parameters control the behavior of the AXI Interconnect.

For a complete list of the interconnect settings related to the AXI interface, see DS768, *LogiCORE IP AXI Interconnect Data Sheet*.

Table 1: **AXI EPC IP Core Design Parameters**

Generic	Feature/Description	Parameter Name	Allowable Values	Default Value	VHDL Type
System Parameter					
G1	Target FPGA family	C_FAMILY	Spartan-6, Virtex-6	Virtex-6	string
AXI Parameters					
G2	AXI address bus width	C_S_AXI_ADDR_WIDTH	32	32	integer
G3	AXI data bus width	C_S_AXI_DATA_WIDTH	32	32	integer
AXI EPC Interface Parameters					
G4	AXI clock period	C_S_AXI_CLK_PERIOD_PS(1)	Integer number of picoseconds	10000	integer
G5	Peripheral clock period	C_PRH_CLK_PERIOD_PS(24)	Integer number of picoseconds	20000	integer

Table 1: **AXI EPC IP Core Design Parameters *(Cont'd)***

Generic	Feature/Description	Parameter Name	Allowable Values	Default Value	VHDL Type
G6	Number of peripherals	C_NUM_PERIPHERALS	1 to 4	1	integer
G7	Maximum of address bus width of all external peripherals	C_PRH_MAX_AWIDTH	3 to 32	32	integer
G8	Maximum of data bus width of all external peripherals	C_PRH_MAX_DWIDTH	8, 16, 32	32	integer
G9	Maximum of data bus width of all peripherals and address bus with of peripherals employing address/data multiplexing	C_PRH_MAX_ADWIDTH[2]	8 to 32	32	integer
G10	Peripheral clock support	C_PRH_CLK_SUPPORT[1]	0 = Peripheral device interface operates at the AXI clock 1 = Peripheral device interface operates at external peripheral clock	0	integer
G11	AXI burst support	C_PRH_BURST_SUPPORT[3]	0 = No burst support from AXI	0	integer
		AXI EPC Peripheral Address Space			
G12	External peripheral base address	C_PRHx_BASEADDR[4]	Valid address[5][6]	User must set values[6]	std_logic_vector
G13	External peripheral high address	C_PRHx_HIGHADDR[4]	Valid address[5][6]	User must set values[6]	std_logic_vector
		AXI EPC Peripheral Interface Parameters			
G14	Support for access to FIFO within the external peripheral	C_PRHx_FIFO_ACCESS[4]	0 = No support for the external peripheral FIFO access 1 = Access to FIFO structures within the external peripheral device is supported	0	integer
G15	External peripheral FIFO offset from peripheral base address	C_PRHx_FIFO_OFFSET[4][8][9]	Any valid offset within the base and high address range assigned to the peripheral device	0	integer
G16	Address bus width of peripherals	C_PRHx_AWIDTH[4]	3 to 32	32	integer
G17	Data bus width of peripherals	C_PRHx_DWIDTH[4]	8, 16, or 32	32	integer
G18	Support for data width match when the peripheral device data width is less than the AXI data width	C_PRHx_DWIDTH_MATCH[4][10]	0 = No multiple cycles on the peripheral interface for single AXI read or write cycle 1 = Run multiple cycles on the peripheral interface for single AXI read or write cycle	0	integer

Table 1: **AXI EPC IP Core Design Parameters *(Cont'd)***

Generic	Feature/Description	Parameter Name	Allowable Values	Default Value	VHDL Type
G19	Peripheral access mode	C_PRHx_SYNC(1)(4)	0 = External device is asynchronous 1 = External device is synchronous	1	integer
G20	Peripheral bus type	C_PRHx_BUS_ MULTIPLEX(4)	0 = External device has separate address and data bus 1 = External device has multiplexed address and data bus	0	integer
AXI EPC Timing Parameters					
G21	Address bus (PRH_Addr) setup with respect to rising edge of address strobe (PRH_ADS) or falling edge of read/write (PRH_Rd_n/ PRH_Wr_n)	C_PRHx_ADDR_ TSU(4)(11)	Integer number of picoseconds	User must set values.(12) See targeted device's data sheet.	integer
G22	Address bus (PRH_Addr) hold with respect to falling edge of address strobe (PRH_ADS) or rising edge of read/write (PRH_Rd_n/ PRH_Wr_n)	C_PRHx_ADDR_ TH(4)(13)(17)(20)	Integer number of picoseconds	User must set values.(12) See targeted device's data sheet.	integer
G23	Minimum pulse width of address strobe (PRH_ADS)	C_PRHx_ADS_ WIDTH(4)	Integer number of picoseconds	User must set values.(12) See targeted device's data sheet.	integer
G24	Chip select (PRH_CS_n) setup with respect to falling edge of read/write (PRH_Rd_n/ PRH_Wr_n)	C_PRHx_CSN_ TSU(4)(11)	Integer number of picoseconds	User must set values.(14) See targeted device's data sheet.	integer
G25	Chip select (PRH_CS_n) hold with respect to rising edge of read/write (PRH_Rd_n/ PRH_Wr_n)	C_PRHx_CSN_ TH(4)(17)(20)	Integer number of picoseconds	User must set values.(14) See targeted device's data sheet.	integer
G26	Minimum pulse width of write signal (PRH_Wr_n)	C_PRHx_WRN_ WIDTH(4)(15)(16)(17)	Integer number of picoseconds	User must set values.(14) See targeted device's data sheet.	integer

Table 1: **AXI EPC IP Core Design Parameters *(Cont'd)***

Generic	Feature/Description	Parameter Name	Allowable Values	Default Value	VHDL Type
G27	Cycle time of write signal (PRH_Wr_n)	C_PRHx_WR_CYCLE(4)(16)(17)	Integer number of picoseconds	User must set values.(14) See targeted device's data sheet.	integer
G28	Data bus (PRH_Data) setup with respect to falling edge of write signal (PRH_Wr_n)	C_PRHx_DATA_TSU(4)(15)	Integer number of picoseconds	User must set values.(14) See targeted device's data sheet.	integer
G29	Data bus (PRH_Data) hold with respect to rising edge of write signal (PRH_Wr_n)	C_PRHx_DATA_TH(4)(17)	Integer number of picoseconds	User must set values.(14) See targeted device's data sheet.	integer
G30	Minimum pulse width of read signal (PRH_Rd_n)	C_PRHx_RDN_WIDTH(4)(18)(19)(20)	Integer number of picoseconds	User must set values.(14) See targeted device's data sheet.	integer
G31	Cycle time of read signal (PRH_Rd_n)	C_PRHx_RD_CYCLE(4)(19)(20)	Integer number of picoseconds	User must set values.(14) See targeted device's data sheet.	integer
G32	Data bus (PRH_Data) validity from falling edge of read signal (PRH_Rd_n)	C_PRHx_DATA_TOUT(4)(18)	Integer number of picoseconds	User must set values.(14) See targeted device's data sheet.	integer
G33	Data bus (PRH_Data) high impedance from rising edge of read (PRH_Rd_n)	C_PRHx_DATA_TINV(4)(20)	Integer number of picoseconds	User must set values.(14) See targeted device's data sheet.	integer

Table 1: **AXI EPC IP Core Design Parameters *(Cont'd)***

Generic	Feature/Description	Parameter Name	Allowable Values	Default Value	VHDL Type
G34	Device ready (PRH_Rdy) validity from the falling edge of read or write (PRH_Rd_n/ PRH_Wr_n)	C_PRHx_RDY_ TOUT(4)(21)(23)	Integer number of picoseconds	User must set values.(14) See targeted device's data sheet.	integer
G35	Maximum period of device ready signal (PRH_Rdy) to wait before device timeout	C_PRHx_RDY_ WIDTH(4)(22)(23)	Integer number of picoseconds	User must set values.(23) See targeted device's data sheet.	integer

Table 1: **AXI EPC IP Core Design Parameters *(Cont'd)***

Generic	Feature/Description	Parameter Name	Allowable Values	Default Value	VHDL Type

Notes:

1. The generic C_PRH_CLK_SUPPORT is relevant only when the device is configured for synchronous access (that is, C_PRHx_SYNC = 1). If more than one device is synchronous, then the frequency for the peripheral clock (PRH_Clk) should be chosen as the minimum of the operating frequencies of those devices.
2. The C_PRH_MAX_ADWIDTH determines the size of the data bus. For all non multiplexed devices the C_PRH_MAX_ADWIDTH reflects the maximum of data bus width of all external devices (that is, C_PRH_MAX_ADWIDTH equals C_PRH_MAX_DWIDTH). However, if any of the devices is configured for multiplexed address and data bus, then C_PRH_MAX_ADWIDTH should be set as the maximum of data bus of all external devices and the address bus of device(s) employing multiplexed address and data bus.
3. Current version of the AXI EPC do not support the burst transactions to/from the AXI.
4. 'x' in the generic refers to the number of the peripheral device and takes a value in the range of 0 to C_NUM_PERIPHERALS - 1.
5. AXI EPC design can accommodate up to four peripheral devices. The address range for the devices are designated as C_PRHx_BASEADDR and C_PRHx_HIGHADDR.
6. The range specified by C_PRHx_BASEADDR and C_PRHx_HIGHADDR must comprise a contiguous range and the size of the range must be a power of two (that is, size of range = 2^m). Further, the "m" least significant bits of C_PRHx_BASEADDR must be zero. The base and high address range assigned to different peripherals must be mutually exclusive.
7. No default value is specified to ensure that the actual value is set (that is, if the value is not set, a compiler error will be generated).
8. C_PRHx_FIFO_ACCESS must be set to 1 if the support for access to FIFO within external peripheral device is to be included.
9. C_PRHx_FIFO_OFFSET is the byte offset of the external peripheral FIFO from the base address (C_PRHx_BASEADDR) of the peripheral irrespective of the data width of the peripheral device. If C_PRHx_FIFO_ACCESS = 1, then C_PRHx_FIFO_OFFSET must be set to a valid offset within the address range assigned to the peripheral.
10. The generic C_PRHx_DWIDTH_MATCH is relevant only when the width of the peripheral data (PRH_Data) bus is less than the AXI data bus (S_AXI_WDATA). The generic C_PRHx_DWIDTH_MATCH must be set to "1" in such cases.
11. Address setup time is with respect to falling edge of address strobe (PRH_ADS) if the address and the data bus are multiplexed (that is, C_PRHx_BUS_MULTIPLEX = 1 and must be set both in synchronous and asynchronous mode). Address setup time is with respect to falling edge of read/write signals (PRH_Wr_n/PRH_Rd_n), if the device is non multiplexed and is relevant only if the access mode is asynchronous.
12. Value for the parameter must be assigned if the address and the data bus are multiplexed (that is C_PRHx_BUS_MULTIPLEX = 1). This parameter assignment is applicable for synchronous and asynchronous devices.
13. Address hold time is with respect to falling edge of address strobe (PRH_ADS) if the address and the data bus are multiplexed (that is, C_PRHx_BUS_MULTIPLEX = 1 and must be set both in synchronous and asynchronous mode). Address hold time is with respect to rising edge of read/write signals (PRH_Wr_n/PRH_Rd_n) if the address and the data bus are separate and is relevant only if the access mode is asynchronous.
14. Value must be assigned if the access mode of the peripherals is asynchronous (that is, C_PRHx_SYNC = 0). If the access mode of the peripheral is synchronous, that is C_PRHx_SYNC = 1, then the zero should be assigned to the parameter.
15. Write signal (PRH_Wr_n) low time is the maximum of C_PRHx_WRN_WIDTH and C_PRHx_DATA_TSU.
16. The value of C_PRHx_WRN_WIDTH must be smaller than C_PRHx_WR_CYCLE. The C_PRHx_WR_CYCLE time is considered for the buffer period between consecutive writes.
17. In non-multiplexed address and data bus mode, write recovery time is the maximum of C_PRHx_ADDR_TH, C_PRHx_CSN_TH, C_PRHx_DATA_TH, and PRH_Wr_n high time (that is, C_PRHx_WR_CYCLE minus C_PRHx_WRN_WIDTH). If the peripheral uses multiplexed address and data bus, then the write recovery time is the maximum of C_PRHx_CSN_TH, C_PRHx_DATA_TH, and PRH_Wr_n high time.
18. Read signal (PRH_Rd_n) low time is the maximum of C_PRHx_RDN_WIDTH and C_PRHx_DATA_TOUT.
19. The value of C_PRHx_RDN_WIDTH must be smaller than C_PRHx_RD_CYCLE. The C_PRHx_RD_CYCLE time is considered for the buffer period between consecutive reads.
20. In non-multiplexed address and data bus mode, read recovery time is the maximum of C_PRHx_ADDR_TH, C_PRHx_CSN_TH, C_PRHx_DATA_TINV, and PRH_Rd_n high time (that is, C_PRHx_RD_CYCLE minus C_PRHx_RDN_WIDTH). If the peripheral uses multiplexed address and data bus, then the read recovery time is the maximum of C_PRHx_CSN_TH, C_PRHx_DATA_TINV, and PRH_Rd_n high time.
21. Device ready validity period (C_PRHx_RDY_TOUT) must be set as the maximum of the device ready values specified for read and write transactions for that device.
22. Device ready pulse width (C_PRHx_RDY_WIDTH) must be set as the maximum of the device ready values specified for read and write transactions for that device.The read and write transactions starts from the falling edge and ends with raising edge of chip select (PRH_CS_n) in synchronous non multiplex mode and asynchronous mode. In synchronous multiplex mode, the transaction (address phase, data Phase) starts from the falling edge and ends with raising edge of chip select (PRH_CS_n).
23. Device ready validity (C_PRHx_RDY_TOUT) period must be less than device ready signal width (C_PRHx_RDY_WIDTH). Device ready signal width (C_PRHx_RDY_WIDTH) must be set for both synchronous and asynchronous access mode to prevent the device from holding the AXI indefinitely.
24. C_PRH_CLK_PERIOD_PS should be greater than C_S_AXI_CLK_PERIOD_PS and is valid only when C_PRH_CLK_SUPPORT is set.

I/O Signals

The AXI EPC IP Core I/O signals are listed and described in Table 2.

Table 2: **AXI EPC IP Core I/O Signal Description**

Port	Signal Name	Interface	Signal Type	Initial State	Description
System Signals					
P1	S_AXI_ACLK	AXI	I	-	AXI clock
P2	S_AXI_ARESETN	AXI	I	-	AXI reset. Active-Low.
AXI Write Address Channels Signals					
P3	S_AXI_AWADDR[C_S_AXI_ADDR_WIDTH-1:0]	AXI	I	-	AXI Write address. The write address bus gives the address of the write transaction.
P4	S_AXI_AWVALID	AXI	I	-	Write address valid. This signal indicates that valid write address is available.
P5	S_AXI_AWREADY	AXI	O	0x0	Write address ready. This signal indicates that the slave is ready to accept an address.
AXI Write Channel Signals					
P6	S_AXI_WDATA[C_S_AXI_DATA_WIDTH - 1:0]	AXI	I	-	Write data
P7	S_AXI_WSTB[C_S_AXI_DATA_WIDTH/8-1:0]	AXI	I	-	Write strobes. This signal indicates which byte lanes to update in memory.(1)
P8	S_AXI_WVALID	AXI	I	-	Write valid. This signal indicates that valid write data and strobes are available.
P9	S_AXI_WREADY	AXI	O	0x0	Write ready. This signal indicates that the slave can accept the write data.
AXI Write Response Channel Signals					
P10	S_AXI_BRESP[1:0]	AXI	O	0x0	Write response. This signal indicates the status of the write transaction. 00 = OKAY 10 = SLVERR
P11	S_AXI_BVALID	AXI	O	0x0	Write response valid. This signal indicates that a valid write response is available.
P12	S_AXI_BREADY	AXI	I	-	Response ready. This signal indicates that the master can accept the response information.
AXI Read Address Channel Signals					
P13	S_AXI_ARADDR[C_S_AXI_ADDR_WIDTH -1:0]	AXI	I	-	Read address. The read address bus gives the address of a read transaction.
P14	S_AXI_ARVALID	AXI	I	-	Read address valid. This signal indicates, when HIGH, that the read address is valid and remains stable until the address acknowledgement signal, S_AXI_ARREADY, is high.
P15	S_AXI_ARREADY	AXI	O	0x1	Read address ready. This signal indicates that the slave is ready to accept an address.

Table 2: **AXI EPC IP Core I/O Signal Description *(Cont'd)***

Port	Signal Name	Interface	Signal Type	Initial State	Description
AXI Read Data Channel Signals					
P16	S_AXI_RDATA[C_S_AXI_DATA_ WIDTH -1:0]	AXI	O	0x0	Read data
P17	S_AXI_RRESP[1:0]	AXI	O	0x0	Read response. This signal indicates the status of the read transfer. 00 = OKAY 10 = SLVERR
P18	S_AXI_RVALID	AXI	O	0x0	Read valid. This signal indicates that the required read data is available and the read transfer can complete.
P19	S_AXI_RREADY	AXI	I	-	Read ready. This signal indicates that the master can accept the read data and response information.
AXI EPC Signals					
P20	PRH_Clk(1)	EPC	I	-	External peripheral clock input
P21	PRH_Rst	EPC	I	-	External peripheral reset input
P22	PRH_CS_n[0:C_NUM_ PERIPHERALS - 1]	EPC	O	1	External peripheral chip select. Active-Low signal
P23	PRH_Addr[0:C_PRH_ MAX_AWIDTH - 1](2)	EPC	O	-	External peripheral address bus
P24	PRH_ADS	EPC	O	0	External peripheral address strobe in case of multiplexed address and data bus. Active-High signal
P25	PRH_BE[0:C_PRH_ MAX_DWIDTH/8 - 1]	EPC	O	-	External peripheral byte enables
P26	PRH_RNW	EPC	O	1	External peripheral read/write signal for synchronous access mode
P27	PRH_Rd_n	EPC	O	1	External peripheral read signal for asynchronous access mode. Active-Low signal
P28	PRH_Wr_n	EPC	O	1	External peripheral write signal for asynchronous access mode. Active-Low signal
P29	PRH_Burst	EPC	O	0	Burst cycle indication to external peripheral. Active-High signal
P30	PRH_Rdy[0:C_NUM_ PERIPHERALS - 1]	EPC	I	-	Peripheral ready signal. This signal is used by the external peripheral to extend the transaction. Active-High signal
P31	PRH_Data_I[0: C_PRH_MAX_ADWIDTH - 1]	EPC	I	-	External peripheral input data bus
P32	PRH_Data_O[0: C_PRH_MAX_ADWIDTH - 1]	EPC	O	-	External peripheral output data bus

Table 2: **AXI EPC IP Core I/O Signal Description *(Cont'd)***

Port	Signal Name	Interface	Signal Type	Initial State	Description
P33	PRH_Data_T[0: C_PRH_MAX_ADWIDTH - 1]	EPC	O	-	3-state control for external peripheral output data bus

Notes:

1. PRH_Clk is utilized only when C_PRH_CLK_SUPPORT = 1 and the interface to the external peripheral is synchronous.
2. External peripheral devices are considered as byte addressable irrespective of the data bus width.

Parameter – I/O Signal Dependencies

The dependencies between the AXI EPC design parameters and the I/O ports are shown in Table 3. The width of the AXI EPC signals depend on some of the parameters. In addition, when certain features are parameterized out of the design, the related logic is no longer a part of the design. The unused input signals and unrelated output signals are set to a specified value.

Table 3: **AXI EPC IP Core Parameter – I/O Signal Dependencies**

Generic or Port	Name	Affects	Depends	Description
Design Parameters				
G2	C_S_AXI_ADDR_ WIDTH	P3, P13	-	Affects the number of bits in address bus
G3	C_S_AXI_DATA_W IDTH	P6, P7, P16	G6	Affects the number of bits in address bus
G6	C_NUM_ PERIPHERALS	P22, P30	P31-	The number of peripherals in the system determines the number of chip select signals driven. The number of device ready inputs decoded by the AXI external peripheral controller.
G7	C_PRH_MAX_ AWIDTH	P23	P24-	The width of the peripheral address bus is set to the maximum of address bus width of all peripheral devices
G8	C_PRH_MAX_ DWIDTH	P25	-	The number of byte enables for the peripheral data bus is determined by the maximum of data bus width of the peripheral devices
G9	C_PRH_MAX_ ADWIDTH	P31, P32, P33	P34-	The width of the peripheral input data bus, peripheral output data bus, and the peripheral output data bus 3-state control are determined by the maximum of data bus width of all peripherals. The address bus width of peripherals employing address/data bus multiplexing.
G10	C_PRH_CLK_ SUPPORT	P20	-	If C_PRH_CLK_SUPPORT = 0, then all external devices operate on AXI clock. The input PRH_Clk must be driven high externally.
G18	C_PRHx_DWIDTH _MATCH	P29	-	If the device is configured for synchronous mode with data width matching enabled, then PRH_Burst is driven high until all but the last byte of data is flushed to the device. For asynchronous mode and synchronous mode with data width match disabled, PRH_Burst is driven low.
G19	C_PRHx_SYNC	P20, P26, P27, P28, P29	P30-	If the device is configured for asynchronous mode: • With peripheral clock support, the peripheral clock input (PRH_Clk) must be driven high externally • PRH_RNW is driven high as PRH_Rd_n and PRH_Wr_n should be used as read and write strobe • With data width matching enabled, the AXI EPC drives PRH_Burst to its default low If the device is configured for synchronous mode: • The AXI EPC drives PRH_Rd_n and PRH_Wr_n outputs to their default state of high because PRH_RNW output is used as read/write strobe
G20	C_PRHx_BUS_ MULTIPLEX	P24	-	PRH_ADS is driven low if the device does not use multiplexed address and data bus (that is, C_PRHx_BUS_MULTIPLEX = 0)
I/O Signals				
P3	S_AXI_AWADDR[C _S_AXI_ADDR_ WIDTH-1:0]	-	G2	Port width depends on the generic C_S_AXI_ADDR_WIDTH
P6	S_AXI_WDATA[C_ S_AXI_DATA_ WIDTH-1:0]	-	G3	Port width depends on the generic C_S_AXI_DATA_WIDTH

Table 3: **AXI EPC IP Core Parameter – I/O Signal Dependencies *(Cont'd)***

Generic or Port	Name	Affects	Depends	Description
P7	S_AXI_WSTB[C_S_AXI_DATA_WIDTH/8-1:0]	-	G3	Port width depends on the generic C_S_AXI_DATA_WIDTH
P13	S_AXI_ARADDR[C_S_AXI_ADDR_WIDTH -1:0]	-	G2	Port width depends on the generic C_S_AXI_ADDR_WIDTH
P16	S_AXI_RDATA[C_S_AXI_DATA_WIDTH -1:0]	-	G3	Port width depends on the generic C_S_AXI_DATA_WIDTH
P20	PRH_Clk	-	G10, G19	PRH_Clk is selected as operating clock only if the device is configured for synchronous mode with peripheral clock support (that is, C_PRHx_SYNC = 1 and C_PRH_CLK_SUPPORT = 1). Asynchronous interface always operates on AXI_Clk. Therefore, if no device is configured for synchronous mode with peripheral clock support enabled, then the input PRH_Clk must be driven high.
P22	PRH_CS_n	-	G6	The number of chip select signals driven by the AXI external peripheral controller is determined by the C_NUM_PERIPHERALS parameter
P23	PRH_Addr	-	G7	The width of the peripheral address bus is determined by the C_PRH_MAX_AWIDTH parameter
P24	PRH_ADS	-	G20	PRH_ADS is driven low if the device does not use multiplexed address and data bus (that is, C_PRHx_BUS_MULTIPLEX = 0)
P25	PRH_BE	-	G8	The number of byte enables for the peripheral data bus is determined by the C_PRH_MAX_DWIDTH parameter
P26	PHR_RNW	-	G19	If the device is synchronous (that is, C_PRHx_SYNC = 1), then PRH_RNW should be used as the read/write control signal. For asynchronous mode of operation, PRH_RNW is driven high.
P27	PRH_Rd_n	-	G19	If the device is asynchronous (that is, C_PRHx_SYNC = 0), then PRH_Rd_n should be used as the read strobe. For synchronous mode of operation, PRH_Rd_n is driven high.
P28	PRH_Wr_n	-	G19	If the device is asynchronous (that is, C_PRHx_SYNC = 0), then PRH_Wr_n should be used as the write strobe. For synchronous mode of operation, PRH_Wr_n is driven high.
P29	PRH_Burst	-	G18, G19	If the device is synchronous with data width matching enabled (that is, C_PRHx_SYNC = 1 and C_PRHx_DWIDTH_MATCH = 1), then PRH_Burst is driven high until all but the last byte of data is flushed to the device. For asynchronous mode and synchronous mode with data width match disabled, PRH_Burst is driven low
P30	PRH_Rdy	-	G6	The number of device ready inputs decoded by the AXI external peripheral controller is determined by the C_NUM_PERIPHERALS parameter

Table 3: **AXI EPC IP Core Parameter – I/O Signal Dependencies *(Cont'd)***

Generic or Port	Name	Affects	Depends	Description
P31	PRH_Data_I	-	G9	The peripheral input data bus width of AXI external peripheral controller is determined by the C_PRH_MAX_ADWIDTH parameter
P32	PRH_Data_O	-	G9	The peripheral output data bus width of AXI external peripheral controller is determined by the C_PRH_MAX_ADWIDTH parameter
P33	PRH_Data_T	-	G9	The peripheral output data bus 3-state control width of AXI external peripheral controller is determined by the C_PRH_MAX_ADWIDTH parameter

Allowable Parameter Combinations

When the peripheral devices are configured in non-multiplexed mode, the C_PRH_MAX_AWIDTH should be the maximum of C_PRHx_AWIDTH of all peripherals in the system (that is, if C_NUM_PERIPHERALS is 2), then C_PRH_MAX_AWIDTH should be maximum of C_PRH0_AWIDTH and C_PRH1_AWIDTH.

C_PRH_MAX_DWIDTH should be the maximum of C_PRHx_DWIDTH of all peripherals in the system (that is, if C_NUM_PERIPHERALS is 2), then C_PRH_MAX_DWIDTH should be maximum of C_PRH0_DWIDTH and C_PRH1_DWIDTH.

If any of the peripheral devices are configured for a multiplexed address and data buses, then the parameter C_PRH_MAX_ADWIDTH should be set as the maximum of the data bus and address bus of the device(s) employing a multiplexed address and data bus. If all devices employ non-multiplexed address and data buses, then C_PRH_MAX_ADWIDTH reflects the maximum of data bus width of all external devices. Therefore, for any configuration the parameter C_PRH_MAX_ADWIDTH should be greater than or equal to C_PRH_MAX_DWIDTH.

The generic C_PRH_CLK_SUPPORT is relevant only when the devices are configured for the synchronous access (that is, C_PRHx_SYNC = 1). If more than one of the devices are synchronous, then the frequency for the peripheral clock should be chosen as the minimum of the operating frequencies of those devices.

The range specified by C_PRHx_BASEADDR and C_PRHx_HIGHADDR parameters must comprise a contiguous range and the size of the range must be a power of two (that is, size of range = 2^m). Furthermore, the "m" least significant bits of C_PRHx_BASEADDR must be zero. The base and high address range assigned to different peripherals must be mutually exclusive. No default value is specified for C_PRHx_BASEADDR and C_PRHx_HIGHADDR to enforce that the user configures these parameters with the actual values. If the values are not set for C_PRHx_BASEADDR and C_PRHx_HIGHADDR, a compiler error is generated.

To access FIFO like structures within the external peripheral devices, C_PRHx_FIFO_ACCESS must be set to 1. When FIFO access is enabled, C_PRHx_FIFO_OFFSET specifies the offset of the FIFO in terms of number of byte locations from the base address (C_PRHx_BASEADDR) of the peripheral and should be set to an offset that lies within the address range assigned to the peripheral device (that is, C_PRHx_BASEADDR + C_PRHx_FIFO_OFFSET ≤ C_PRHx_HIGHADDR). Furthermore, the FIFO offset should be within the range addressable by the address bus (that is, C_PRHx_FIFO_OFFSET < 2^n where n = C_PRHx_AWIDTH).

The width of the read/write strobe must be less than the cycle time of the corresponding access (that is, C_PRHx_WRN_WIDTH < C_PRHx_WR_CYCLE and C_PRHx_RDN_WIDTH < C_PRHx_RD_CYCLE). The value of device ready validity period must be less than the read/write strobe width and the maximum pulse width of the device ready signal (that is, C_PRHx_RDY_TOUT < minimum of C_PRHx_WRN_WIDTH, C_PRHx_RDN_WIDTH, and C_PRHx_RDY_WIDTH). The time period between two consecutive writes (when the data width matching is enabled) is decided by maximum of (C_PRHx_WR_CYCLE - C_PRHx_WRN_WIDTH), C_PRHx_CSN_TH, and C_PRHx_DATA_TH parameters. Similar calculation is applicable for consecutive reads.

Make sure that the user has configured the timing parameters properly while using AXI EPC IP Core at different frequencies on various FPGA devices.

Design Considerations

The AXI EPC IP Core is AXI slave device. It receives read or write instructions from the processor and generates a corresponding access cycle on the peripheral interface. Examples of read and write accesses are illustrated in the Timing Diagrams section. The user must take the following considerations into account while designing with the AXI EPC.

Timing Parameters in Async Mode of Operation

Internally in the AXI EPC IP Core, the timing parameters are inter-related and some calculation is done based upon the values provided by the user. In the asynchronous mode of operation, the following parameters are used:

- C_PRHx_ADDR_TSU – Address set up time
- C_PRHx_ADDR_TH – Address hold time
- C_PRHx_ADS_WIDTH – Address strobe width
- C_PRHx_CSN_TSU – Chip select set up time
- C_PRHx_CSN_TH – Chip select hold time
- C_PRHx_WRN_WIDTH – Write control width time
- C_PRHx_WR_CYCLE – Write cycle time (that is, time between two consecutive writes)
- C_PRHx_DATA_TSU – Data set up time
- C_PRHx_DATA_TH – Data hold time
- C_PRHx_RDN_WIDTH – Read control width time
- C_PRHx_RD_CYCLE – Read cycle time (that is, time between two consecutive reads)
- C_PRHx_DATA_TOUT – Time taken by the data to be out at read condition
- C_PRHx_DATA_TINV – Data line 3-state after the read control signal is de-activated
- C_PRHx_RDY_TOUT – Time for device ready signal to go high, after device is selected
- C_PRHx_RDY_WIDTH – Maximum time until the AXI EPC IP Core can wait for device to be ready

These parameters are linked in the design as follows:

- Address hold time – Parameter is calculated as:

  (C_PRHx_ADDR_TH/C_BUS_CLOCK_PERIOD_PS)
- Chip select/Data/Address hold time – Three parameters are used to calculate the hold time as: ((max2(max2(C_PRHx_DATA_TH,C_PRHx_CSN_TH),C_PRHx_ADDR_TH)/ C_BUS_CLOCK_PERIOD_PS)
- Read control signal width time – Parameter is calculated as: (C_PRHx_RDN_WIDTH/C_BUS_CLOCK_PERIOD_PS)
- Write control signal width time – Parameter is calculated as: (C_PRHx_WRN_WIDTH/C_BUS_CLOCK_PERIOD_PS)

- Address strobe signal width time – Three parameters are used to calculate the address strobe width as: ((max2(max2(C_PRHx_ADDR_TSU, C_PRHx_ADS_WIDTH), C_PRHx_CSN_WIDTH)/ C_BUS_CLOCK_PERIOD_PS)
- Write recovery time for non-multiplexed case is calculated as: ((max2(max2(C_PRHx_CSN_TH, C_PRHx_DATA_TH),max2(C_PRHx_ADDR_TH,PRH_Wr_nx))/C_BUS_CLOCK_PERIOD_PS) Where PRH_Wr_nx = (C_PRHx_WR_CYCLE - C_PRHx_WRN_WIDTH)
- Read recovery time for non-multiplexed case is calculated as: ((max2(max2(C_PRHx_CSN_TH, C_PRHx_DATA_TINV),max2(C_PRHx_ADDR_TH,PRH_Rd_nx))/C_BUS_CLOCK_PERIOD_PS)
- Write recovery time for multiplexed case is calculated as: ((max2(C_PRHx_CSN_TH, C_PRHx_DATA_TH),PRH_Wr_nx))/C_BUS_CLOCK_PERIOD_PS) Where PRH_Wr_nx = (C_PRHx_WR_CYCLE - C_PRHx_WRN_WIDTH)
- Read recovery time for multiplexed case is calculated as: ((max2(C_PRHx_CSN_TH, C_PRHx_DATA_TINV),PRH_Rd_nx))/C_BUS_CLOCK_PERIOD_PS)

  Where PRH_Rd_nx = (C_PRHx_RD_CYCLE - C_PRHx_RDN_WIDTH)
- Device Ready signal time – Parameter is calculated as: (C_PRHx_RDY_TOUT/C_BUS_CLOCK_PERIOD_PS)
- Maximum time for Device Ready signal to be active – Parameter is calculated as: (C_PRHx_RDY_WIDTH/C_BUS_CLOCK_PERIOD_PS)

Some of the calculations are repeated for different uses for internal purpose. There are some dummy states included in between the states to meet the design requirements. The user should make sure that these timing parameter value calculations do not exceed the AXI timeout value (C_DPHASE_TIMEOUT) if C_DPHASE_TIMEOUT > 0.

Table 4 indicates the ranges for the mentioned parameters.

Table 4: **Example of Timing Ranges for AXI EPC IP Core in Asynchronous Mode of Operation**

Parameter Name	Timing Parameter's Range		
C_PRHx_ADDR_TSU	6000 (set 1)	20000 (set 2)	40000 (set 3)
C_PRHx_ADDR_TH	6000	20000	30000
C_PRHx_ADS_WIDTH	10000	20000	40000
C_PRHx_CSN_TSU	6000	20000	40000
C_PRHx_CSN_TH	6000	20000	30000
C_PRHx_WRN_WIDTH	15000	30000	30000
C_PRHx_WR_CYCLE	30000	60000	60000
C_PRHx_DATA_TSU	10000	20000	30000
C_PRHx_DATA_TH	5000	20000	30000
C_PRHx_RDN_WIDTH	15000	30000	30000
C_PRHx_RD_CYCLE	30000	60000	60000
C_PRHx_DATA_TOUT	5000	5000	15000
C_PRHx_DATA_TINV	10000	15000	25000

Table 4: **Example of Timing Ranges for AXI EPC IP Core in Asynchronous Mode of Operation *(Cont'd)***

Parameter Name	Timing Parameter's Range		
C_PRHx_RDY_TOUT	10000(set 1)	50000(set 2)	60000(set 3)
C_PRHx_RDY_WIDTH	50000	100000	120000

Notes:

1. The above range of timing parameters (set 1) are used for internal device utilization and while testing on USB and LAN. While using the parameters, care has been taken that all parameters of same set are used.
2. The above timing sets (set 2 and set 3) are just an example of how the timing parameters can be given to AXI EPC IP Core. The user can have different timing parameter values as per the targeted device's data sheet. In such cases, note the calculation given above.

FIFO Transactions

When C_PRHx_FIFO_ACCESS = 1, the AXI EPC IP Core supports access to FIFOs within the external peripheral devices. When the FIFO within the external device is accessed, then the peripheral address bus (`PRH_Addr`) must remain constant representing the FIFO address within the address range assigned to the peripheral device. When data width matching is enabled and the access corresponds to the FIFO, the AXI EPC IP Core does not increment the peripheral address (`PRH_Addr`) bus.

Abnormal Terminations

If `PRH_Rdy` is deasserted for more than the maximum period as specified by C_PRHx_RDY_WIDTH, then the AXI EPC terminates the current access to the external device and signals slave error to the AXI master by asserting either `S_AXI_BRESP` or `S_AXI_RRESP` on the AXI. In this case, the access to the external device is terminated abnormally. Therefore, the external device can be in an indeterminate state and the exceptions should be handled appropriately at the system level.

Interrupt Handling

If the external device has interrupt capability, then the interrupt outputs of the external device should be connected directly to the system interrupt controller.

Device Ready Signal (PRH_Rdy Signal)

The AXI EPC IP Core read/write access cycles are executed only when the external device assert the device ready signal (`PRH_Rdy`).

As the `PRH_Rdy` signal becomes active, this indicates that the peripheral device is ready for communication. After the `PRH_Rdy` goes active, it is expected that it remains in the same active state until that particular transaction is completed by the AXI EPC IP Core. If the AXI EPC IP Core is implemented with data width match enabled, then the activeness of the `PRH_Rdy` signal is always checked for every transaction, regardless of it being active throughout the transaction.

The user should see the data sheet of the external peripheral, to define the value of `PRH_Rdy` time period (that is, C_PRHx_RDY_TOUT parameter value should be filled based on the mentioned information). It is also expected that the user provides maximum waiting period for `PRH_Rdy` signal. This waiting period is indicated by C_PRHx_RDY_WIDTH parameter. The C_PRHx_RDY_TOUT parameter value should be less than the C_PRHx_RDY_WIDTH parameter value. The C_PRHx_RDY_WIDTH time period should be lower than the AXI IPIF timeout value if C_DPHASE_TIMEOUT > 0 (that is, C_PRHx_RDY_TOUT < C_PRHx_RDY_WIDTH < C_DPHASE_TIMEOUT). Starting from the falling edge of the `PRH_WR_`*n* or, `PRH_RD_`*n* signal, the internal counter for C_PRHx_RDY_TOUT and C_PRHx_RDY_WIDTH starts. The AXI EPC IP Core expects that the `PRH_Rdy` signal should appear before any of these timer ends. If the PRH_Rdy does not become active before the C_PRHx_RDY_TOUT counter ends, then the AXI EPC IP Core waits until the end of C_PRHx_RDY_WIDTH counter. In case PRH_Rdy becomes active, then internal state machine proceeds to complete the further process steps. If PRH_Rdy does not become active even before the C_PRHx_RDY_WIDTH ends, the AXI EPC IP Core generates an error and terminate the transaction. The user can re-initiate the same transaction later on.

Figure 3 shows the diagrammatic representation of how the async state machine reacts with the PRH_Rdy signal. Consider this figure as representation only. In actual design, there are some dummy states added to meet the design requirements. In case of asynchronous reset to the core, all states go to IDLE by default.

If the external peripheral device does not have a device ready signal, then the PRH_Rdy input of the AXI EPC for that particular device must be tied to logic high.

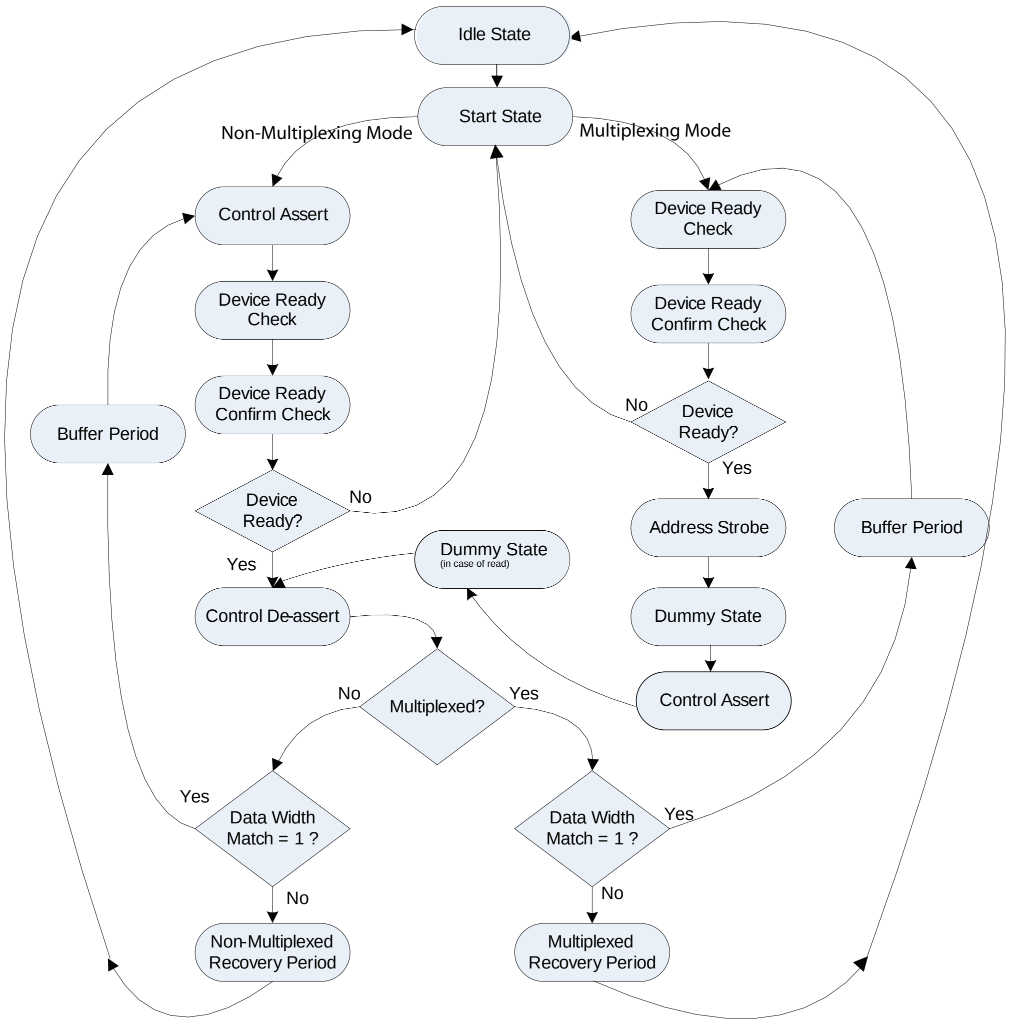

Figure 3: **Diagrammatic Representation of Async State Machine Flow**

Peripheral Data Bus Mapping

The peripheral data bus (`PRH_Data`) uses big endian bit labelling (that is, bit-0 is Most Significant Bit (MSB) and bit-31 is Least Significant Bit (LSB) for a 32-bit bus) and is sized according to the C_PRH_MAX_ADWIDTH parameter. Peripherals that have smaller widths should connect to this bus starting at bit-0 (MSB). For example, if three external devices are present in the system with data bus width of 8, 16, and 32 bits, the 8-bit device should connect to `PRH_Data[0:7]`, the 16-bit wide peripheral should connect to `PRH_Data[0:15]`, and the 32-bit peripheral should connect to `PRH_Data[0:31]`.

The bit and byte labeling for the big endian data types is shown in Figure 4.

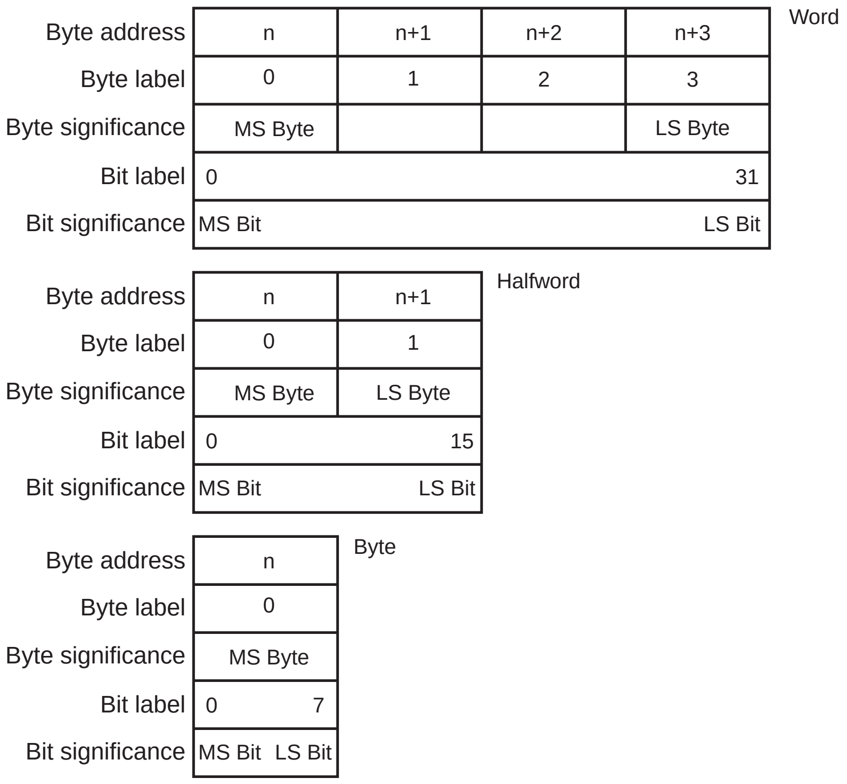

Figure 4: **AXI EPC Big Endian Data Type**

Unsupported Features

Many peripheral devices have the device specific input/output ports such as status, remote reset, remote wake-up, and interrupts. The AXI EPC IP Core does not have any provision to support these device specific input/output ports. Therefore, if the external device has any such device specific ports, then these input/output ports can be connected directly to system general purpose input output controller or to the system interrupt controller. If the external device has interrupt capability, then the interrupt outputs of the external device should be connected directly to the system interrupt controller.

Many peripheral devices support DMA capability. However, the AXI EPC IP Core is a AXI slave device and does not support DMA operations from the external peripheral devices.

The current version of the AXI EPC IP Core does not support burst transactions to/from the AXI. Consequently, the parameter C_PRH_BURST_SUPPORT must be always assigned to 0.

External Peripheral Connections

The AXI EPC IP Core interface to the external device is based upon the width of the AXI data bus, address and data width of the peripheral subsystem, number of peripherals in the system, address/data multiplex support, mode of operation (synchronous or asynchronous) and if synchronous device, the operating clock for the synchronous datapath.

Determining Address and Data Width

The address bus width of the peripheral subsystem is the maximum width of the address bus of all peripheral devices connected to the AXI EPC IP Core. If all devices employ non-multiplexed address and data bus, the data bus width of the peripheral subsystem is the maximum of the data bus width of all external devices. If any of the devices are configured for multiplexed address and data bus, then the data bus width of the peripheral subsystem should be set as the maximum of the data bus and the address bus of the device(s) employing a multiplexed address and data bus.

Endian Considerations

The peripheral address and data bus of the AXI EPC IP Core is labeled with big endian bit labeling (D0 is the MSB and D31 is the LSB for a 32-bit bus) while most peripheral devices are either endian agnostic (that is, they can be connected either way or little endian (D31 is the MSB and D0 is the LSB for a 32-bit data bus)). Caution must be exercised with the connection to the external peripheral devices to avoid incorrect data and address connections. See Data Steer and Address Generation Modules for a data steering example.

Clock Generation

When connecting to a synchronous external device, the AXI EPC IP Core can operate either on the AXI clock (`S_AXI_ACLK`) or on a peripheral clock (`PRH_Clk`). The operating clock for the synchronous interface is based on the generic C_PRH_CLK_SUPPORT. If C_PRH_CLK_SUPPORT = 1, the peripheral clock is used else the AXI clock is used. The external devices connected to the AXI EPC IP Core determines the operating frequency of the peripheral clock. The minimum of the operating frequencies of various devices connected to AXI EPC IP Core should be used as the operating frequency of the peripheral clock (`PRH_Clk`).

Figure 5 illustrates a block diagram of one of the methods for generating a device specific clock source using two DCMs. The DCM modules are located within the FPGA but are external to the AXI EPC IP Core. An external clock source is used to generate the AXI clock (`S_AXI_ACLK`) to the AXI EPC. The device clock source (`PRH_Clk`) is also generated from the same DCM (DCM0) using the CLKFX output and must be routed on the board to be fed back to the FPGA. The parameters of DCM0 like C_CLKFX_MULTIPLY and C_CLKFX_DIVIDE should be set according to the required frequency output at CLKFX pin of DCM0. Another DCM (DCM1) is used to synchronize the device peripheral clock to the `PRH_Clk` signal of AXI_EPC.

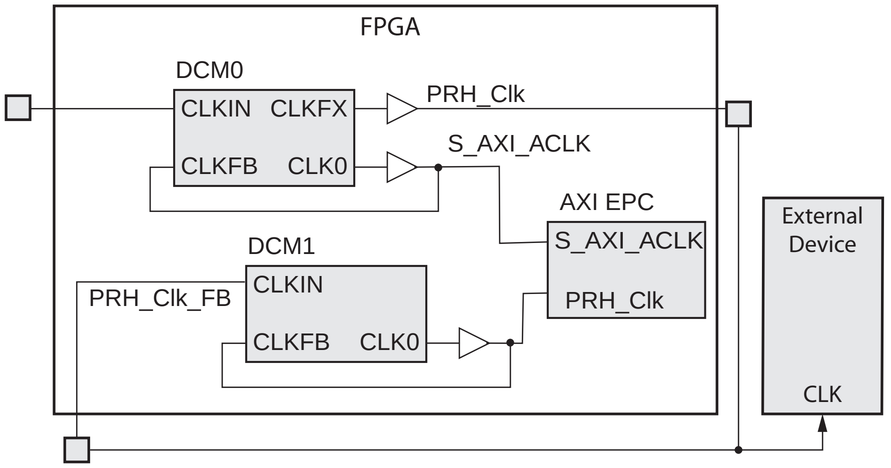

Figure 5: **External Peripherals Clocked by FPGA Output with Feedback**

Example Peripheral Connections

Example: SMSC LAN91C111 10/100 Non-PCI Ethernet Single Chip MAC + PHY

An example peripheral device supported by the AXI EPC IP Core is the SMSC 10/100 non-PCI Ethernet Single Chip MAC + PHY LAN91C111. The AXI EPC IP Core allows AXI masters to access the device both in synchronous and asynchronous mode. The device connections for synchronous and asynchronous mode of operations are outlined in the Asynchronous Mode and Synchronous Mode sections.

Asynchronous Mode

Figure 6 illustrates the SMSC LAN91C111 connections to the AXI EPC IP Core in asynchronous mode with non-multiplexed address and data buses. In asynchronous mode, the input clock source (`PRH_Clk`) to the external device must be tied to Vcc. When the SMSC LAN91C111 is accessed through the asynchronous mode, the rising edge of `PRH_Rd_`*n* and `PRH_Wr_`*n* signals are used to control the read and the write operations, respectively. As the device uses Asynchronous Ready (ARDY) signal to indicate its readiness in asynchronous mode. This signal is connected to the PRH_Rdy input of the AXI EPC IP Core. Since the address and the data buses are not multiplexed, `PRH_ADS` is driven low by the AXI EPC IP Core. The timing parameters should be set according to the SMSC LAN91C111 Async mode specifications.

The following parameter settings apply for the connections shown in Figure 6.

- C_PRH_CLK_SUPPORT = 0
- C_PRH0_AWIDTH = 16
- C_PRH0_DWIDTH = 32
- C_PRH0_SYNC = 0
- C_PRH0_DWIDTH_MATCH = 0
- C_PRH0_BUS_MULTIPLEX = 0

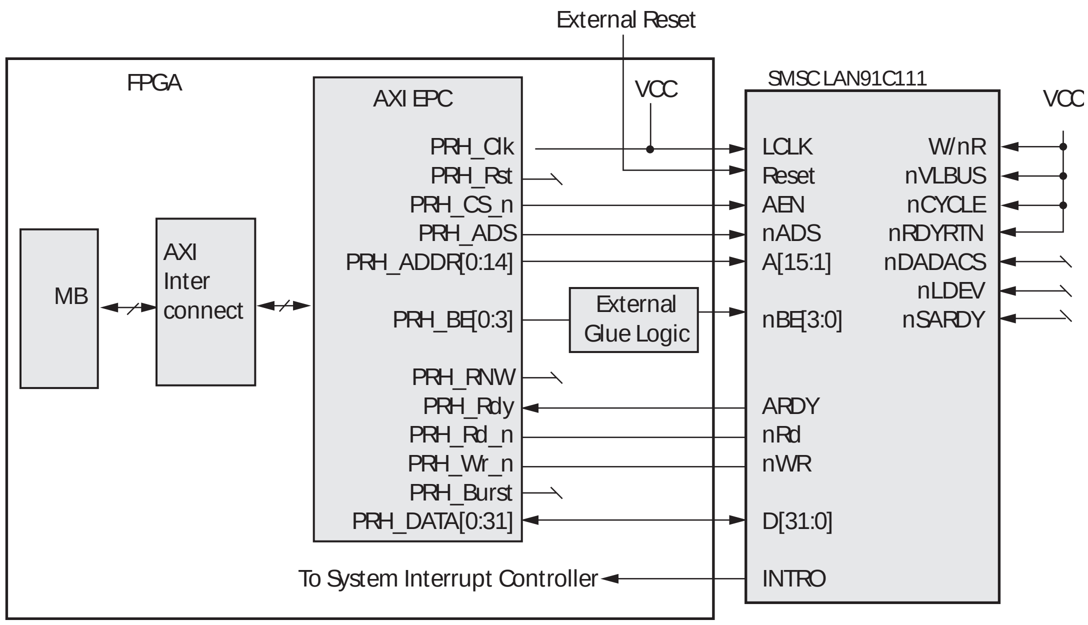

Figure 6: **SMSC LAN91C111 Connection to AXI EPC IP Core in Asynchronous Mode**

External FPGA Logic

Special attention must be given while interfacing the AXI EPC IP Core to the SMSC LAN91C111 in asynchronous mode. The AXI EPC IP Core drives the PRH_BE active-High while the SMSC LAN91C111 requires the input byte enable to be active-Low. To match this requirement, external FPGA logic is required to invert the PRH_BE signals before they are interfaced with the byte enable signals of the SMSC LAN91C111.

Synchronous Mode

Figure 7 illustrates the example for SMSC LAN91C111 connection to the AXI EPC IP Core in synchronous VL Bus mode with non-multiplexed address and data buses. In synchronous mode, the device requires an input clock source to the LCLK pin with a maximum operating frequency of 50 MHz. The local clock generated by the DCM for the peripheral interface must be routed to the LCLK input of SMSC LAN91C111 and PRH_Clk of AXI EPC IP Core. When the device is accessed, PRH_RNW is used as the control signal to indicate a read/write access. When PRH_RNW is high, it indicates a read operation and when low, it indicates a write operation. The SMSC LAN91C111 uses nSRDY signal to indicate the device readiness. This signal is connected to PRH_Rdy input of the AXI_EPC. As the address and the data bus are not multiplexed, PRH_ADS is driven low.

The following parameter settings apply for the connections shown in Figure 7.

- C_PRH_CLK_SUPPORT = 1
- C_PRH0_AWIDTH = 16
- C_PRH0_DWIDTH = 32
- C_PRH0_SYNC = 1
- C_PRH0_DWIDTH_MATCH = 0
- C_PRH0_BUS_MULTIPLEX = 0

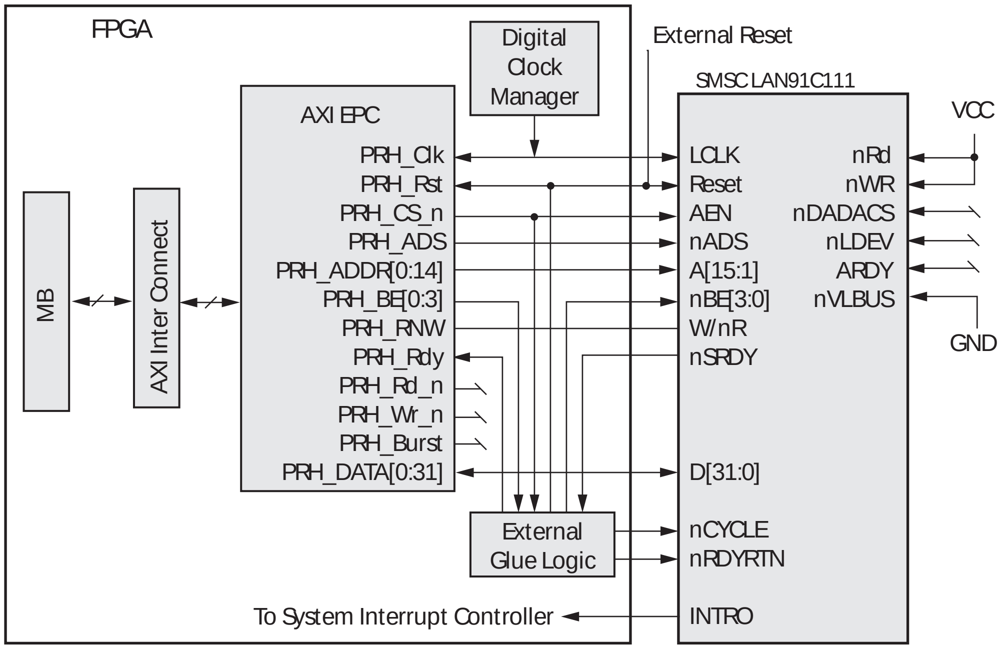

Figure 7: **SMSC LAN91C111 Connection to AXI EPC IP Core in Synchronous Mode**

External FPGA Logic

The SMSC LAN91C111 device requires two inputs, `nCYCLE` and `nRDYRTN`, to control synchronous operation. The definition of these signals is outside the scope of this document and can be found in the documents listed in the References section. The `nCYCLE` and `nRDYRTN` signals are generated in external FPGA logic. `nCYCLE` can be generated from `PRH_CS_`*n* and is driven low for a single clock cycle on a high to low transition on `PRH_CS_`*n*. `nRDYRTN` is the registered value of the `nSRDY` signal.

In case of synchronous multiplexing mode, the generation of `nCYCLE` signal is little different. The `PRH_CS_`*n* signal is generated twice from the core, first during the address phase and second time during the data phase. User should take care while writing the logic for `nCYCLE` signal generation in external FPGA logic. The `nCYCLE` should be generated in data phase when there is `PRH_CS_`*n* asserted and `PRH_ADS` is not present.

Design Considerations

When the AXI EPC IP Core is interfaced to the SMSC LAN91C111 with data width match enabled (C_PRHx_DWIDTH_MATCH = 1) and the internal SRAM of the peripheral is being accessed, the device has to be configured in the address auto increment mode. Address auto increment mode can be configured by setting "AUTO INCR" bit in the Bank 2 Pointer Register of SMSC LAN91C111. More information can be obtained by referring to the registers of Bank 2 in the SMSC LAN91C111 data sheet. See the References section.

Example: Cypress Semiconductor's CY7C67300 EZ-Host Programmable Embedded USB Host/Peripheral Controller

The Cypress Semiconductor's CY7C67300 EZ-Host™ Programmable Embedded USB Host/Peripheral Controller is another example of a peripheral device supported by the AXI EPC IP Core. Since the Cypress USB controller operates in asynchronous mode, the AXI EPC IP Core must be configured to support asynchronous mode operation.

Asynchronous Mode

Figure 8 illustrates the CY7C67300 USB Controller connection to the AXI EPC IP Core in asynchronous mode with non-multiplexed address and data buses. This interface is tested in hardware.

The following parameter settings apply for the connection shown in Figure 8.

- C_PRH_CLK_SUPPORT = 0
- C_PRH0_AWIDTH = 4
- C_PRH0_DWIDTH = 16
- C_PRH0_SYNC = 0
- C_PRH0_DWIDTH_MATCH = 1
- C_PRH0_BUS_MULTIPLEX = 0

Configuration of CY7C67300 USB Controller

The CY7C67300 USB Controller is configured as a Host Processor Interface (HPI). The boot-up sequence code for the CY7C67300 USB Controller in this particular example is present on the external CompactFlash memory device and is accessed through the AXI SYSACE interface. This code is downloaded to the CY7C67300 USB Controller through the HPI interface. After this boot-up sequence is executed by the CY7C67300 USB Controller, it becomes ready to operate in normal HPI mode with the external host controller (AXI EPC IP Core). The AXI EPC address bits which corresponds to word boundary are only used in the 2-bit address bus interface with the Cypress Semiconductor's USB device.

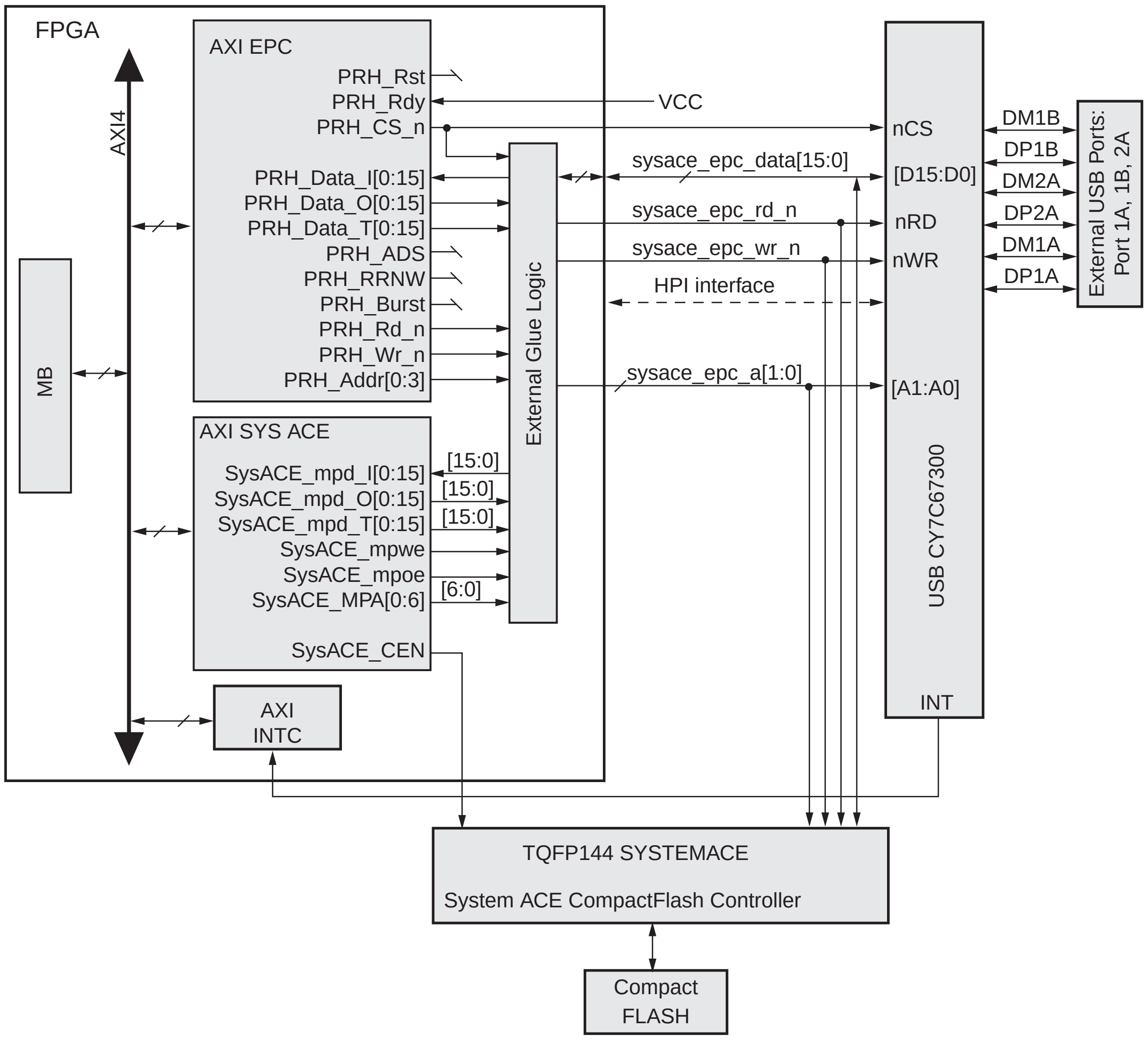

Figure 8: **CY7C67300 USB Controller Connection to AXI EPC in Asynchronous Mode**

External FPGA Logic

In the interface diagram example shown in Figure 8, the AXI EPC IP Core, as well as the AXI SYSACE share the interface for data, address and control lines on the board. This requires external multiplex logic to be placed outside of IP cores in the FPGA. The AXI EPC IP Core control signals `PRH_RD_`*n* and `PRH_WR_`*n* are MUXed with the `MEM_CEN` and `MEM_WEN` signals of the AXI SYSACE. When CY7C67300 USB Controller is accessed through the asynchronous mode, the rising edge of `sysace_epc_rd_`*n* and `sysace_epc_wr_`*n* signals are used to control the read and the write operations respectively. See the References section for more information on AXI SYSACE.

Design Constraints

Timing Constraints

Timing constraints must be placed on the system clock and the peripheral clock, setting the frequency to meet the bus timing requirements. An example is shown in Figure 9.

Following constraint must be placed on the system clock:

```
NET S_AXI_ACLK TNM_NET = S_AXI_ACLK;
TIMESPEC TS_S_AXI_ACLK = PERIOD S_AXI_ACLK 5 ns HIGH 50%;
```

If C_PRH_CLK_SUPPORT=1, then the following constraint must be placed on the peripheral clock:

```
NET PRH_Clk TNM_NET = PRH_Clk;
TIMESPEC TS_PRH_Clk = PERIOD PRH_Clk 10 ns HIGH 50%;

NET "S_AXI_ACLK" TNM_NET = FFS "GRP_S_AXI_ACLK";
NET "PRH_Clk" TNM_NET = FFS "GRP_PRH_Clk ";

TIMESPEC "TS_AXI_PRHCLK_FALSE_PATH" = FROM "GRP_S_AXI_ACLK " TO " GRP_PRH_Clk" TIG;
TIMESPEC "TS_PRHCLK_AXI_FALSE_PATH" = FROM " GRP_PRH_Clk" TO " GRP_S_AXI_ACLK " TIG;
```

Figure 9: **AXI EPC Timing Constraints**

Design Implementation

Timing Diagrams

The timing diagrams in the figures show various AXI transactions and the resulting access cycles on the peripheral interface. Timing diagrams are not shown for all possible combinations of generics and the resulting access cycles. However, the timing diagrams shown are sufficient for the basic understanding of various access cycles supported by the AXI EPC IP Core.

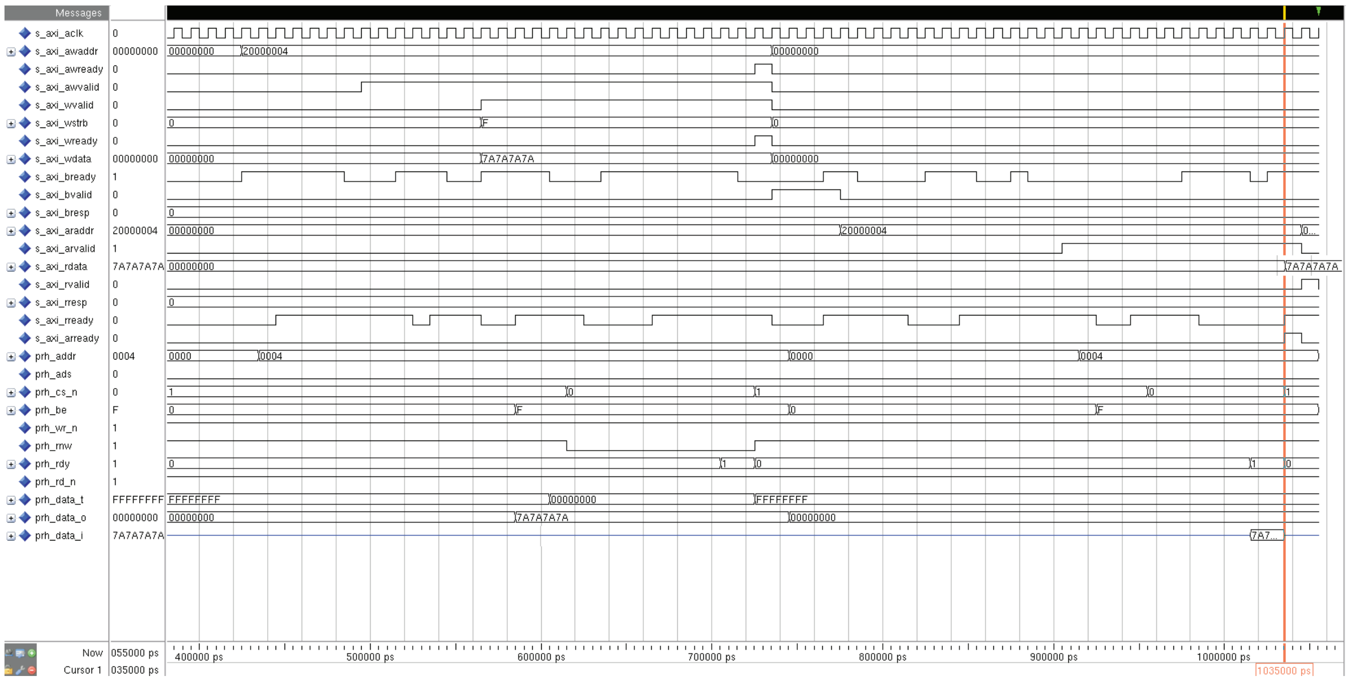

Figure 10: **Synchronous Write-Read Transactions to Device Memory When Bus is Not Multiplexed and Data Width Matching is Disabled (C_PRH_CLK_SUPPORT = 0), Assuming the Peripheral Device is Ready**

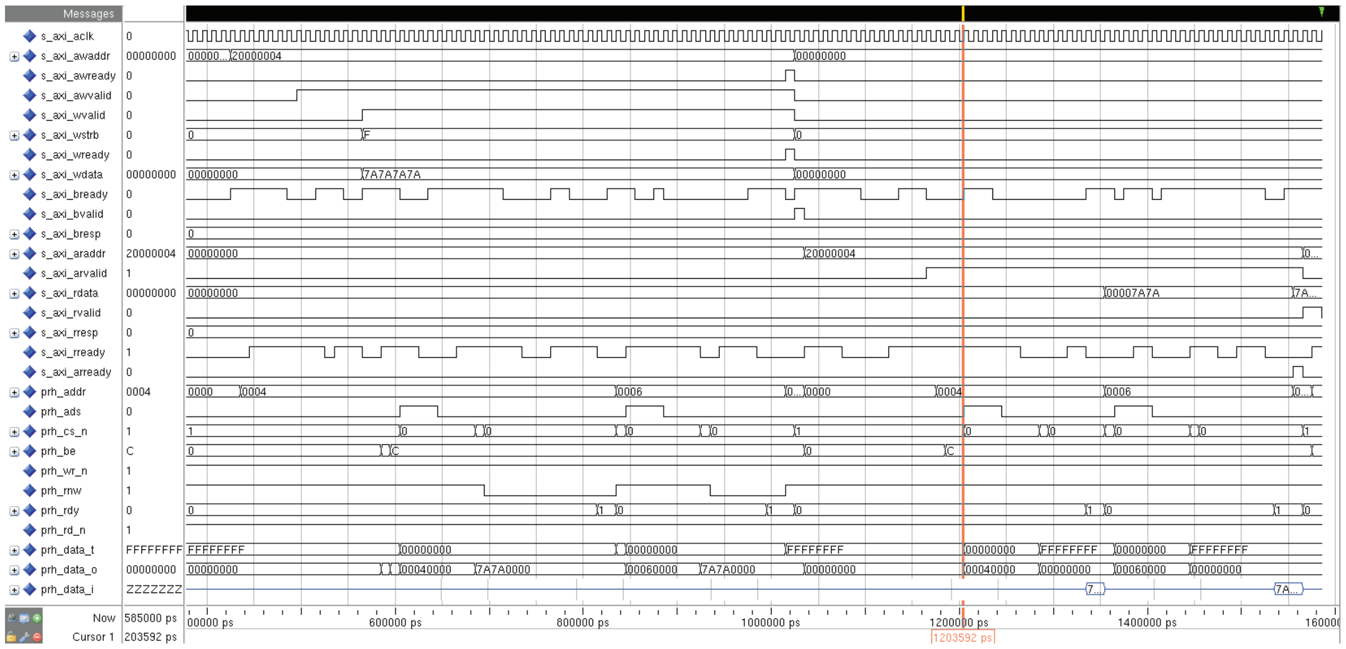

Figure 11: **Synchronous Write and Read Transactions to Device Memory When Bus is Multiplexed and Data Width Matching is Enabled (C_PRH_CLK_SUPPORT = 0)**

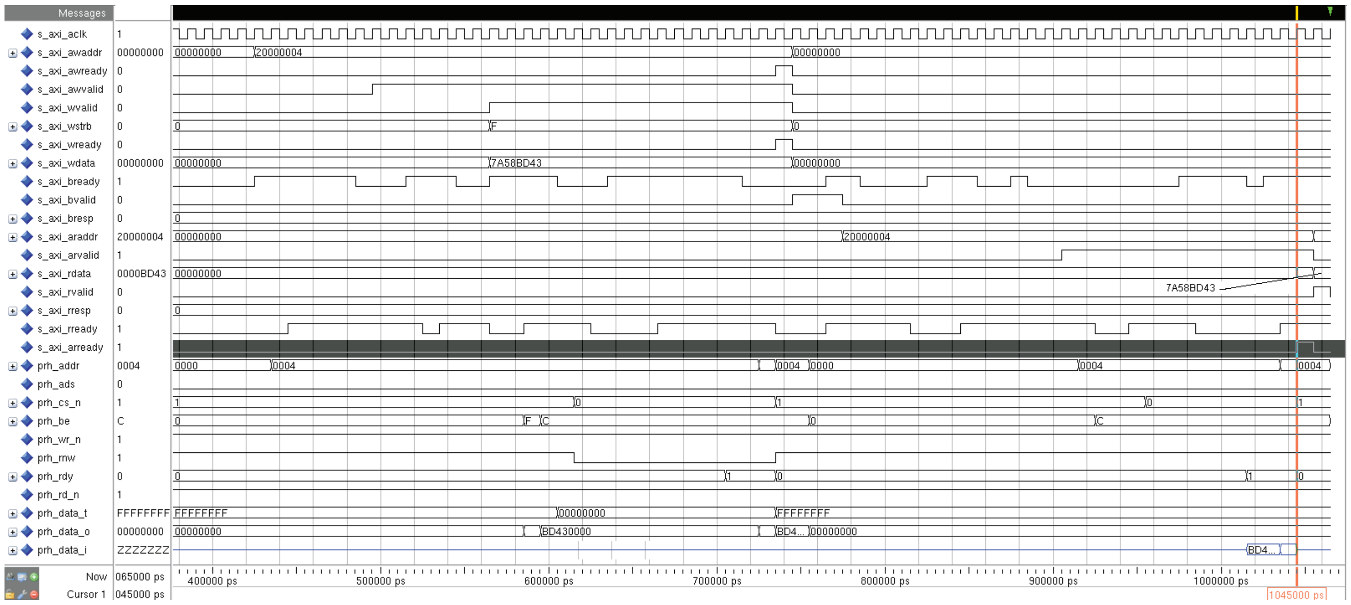

Figure 12: **Synchronous Read and Write Transactions to Device Memory When Bus is Not Multiplexed and Data Width Matching is Enabled (C_PRH_CLK_SUPPORT = 0)**

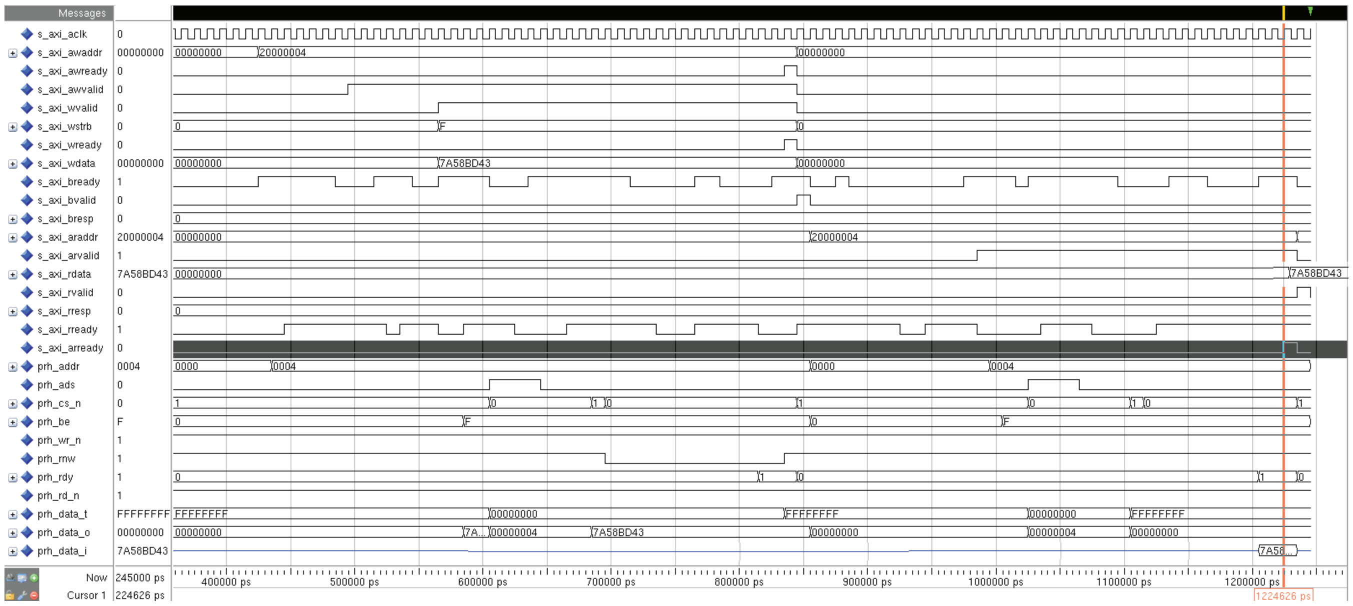

Figure 13: **Synchronous Read and Write Transactions to Device Memory When Bus is Multiplexed and Data Width Matching is Disabled (C_PRH_CLK_SUPPORT = 0)**

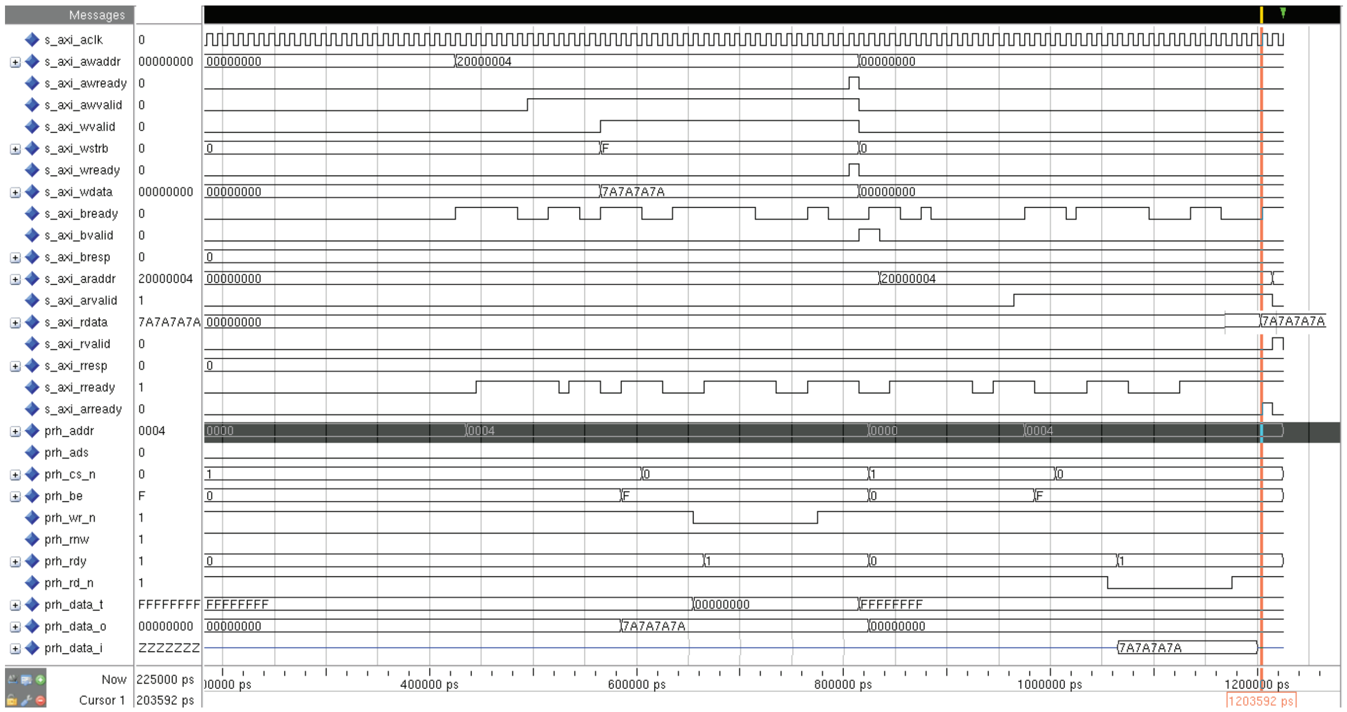

Figure 14: **Asynchronous Write-Read Transactions to Device Memory When Bus is Not Multiplexed and Data Width Matching is Disabled (C_PRH_CLK_SUPPORT = 0)**

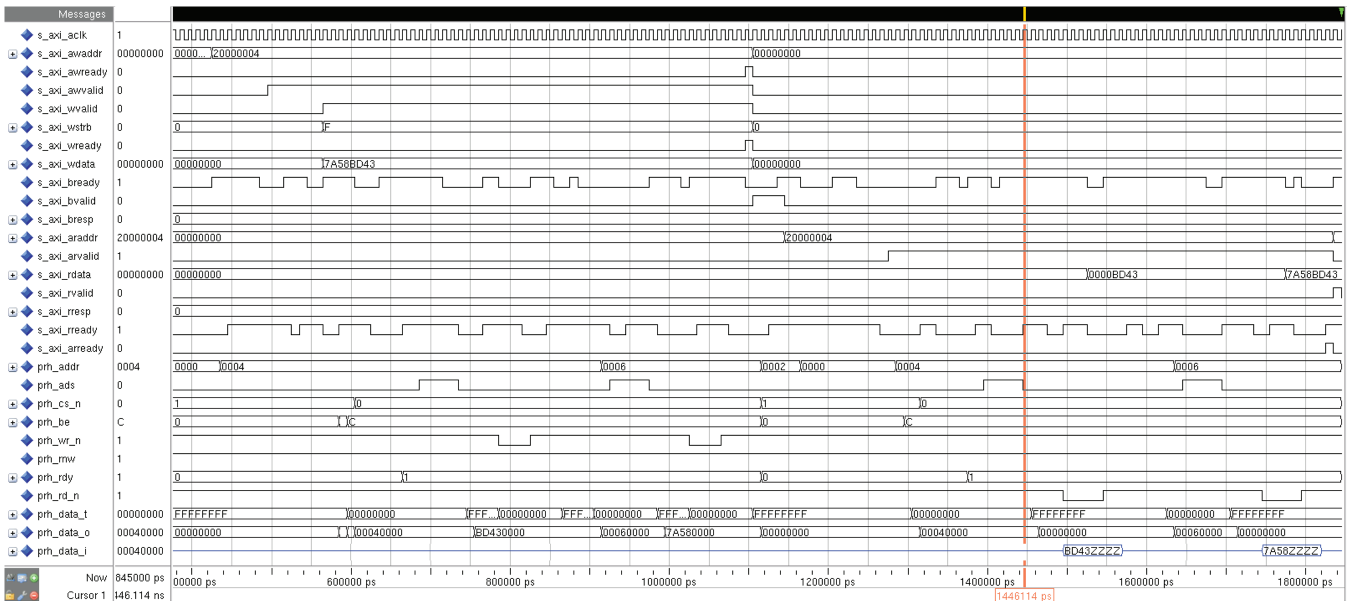

Figure 15: **Asynchronous Read and Write Transactions to Device Memory When Bus is Multiplexed and Data Width Matching is Enabled (C_PRH_CLK_SUPPORT = 0)**

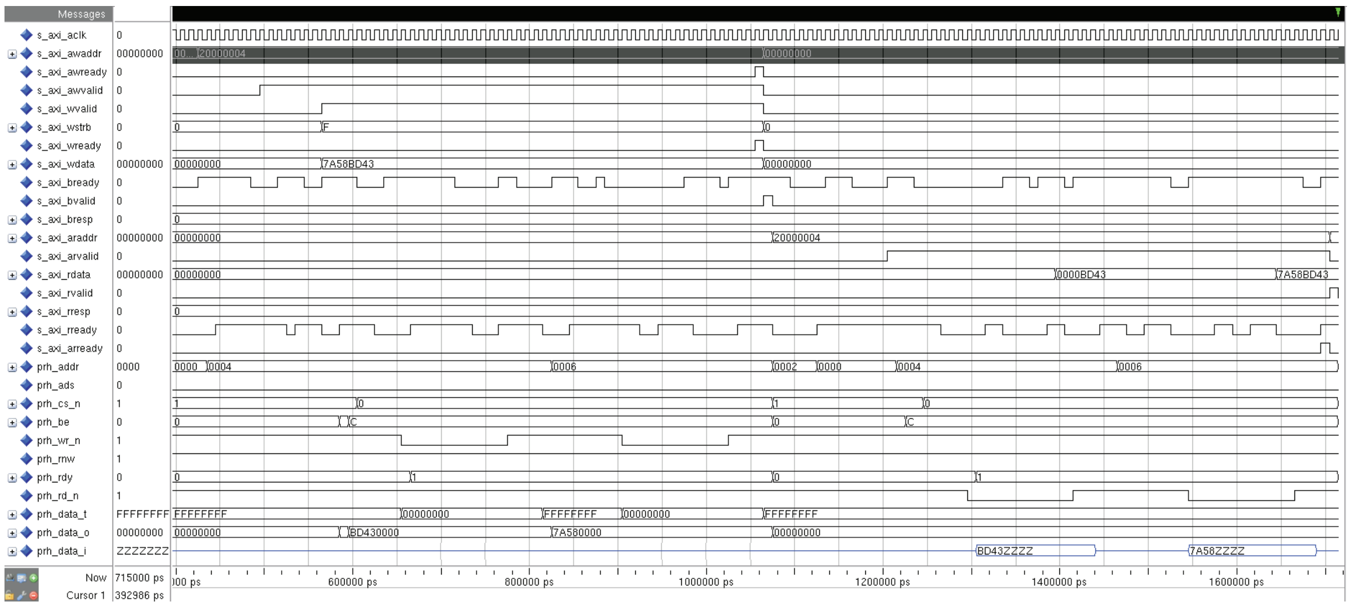

Figure 16: **Asynchronous Read and Write Transactions to Device Memory When Bus is Not Multiplexed and Data Width Matching is Enabled**

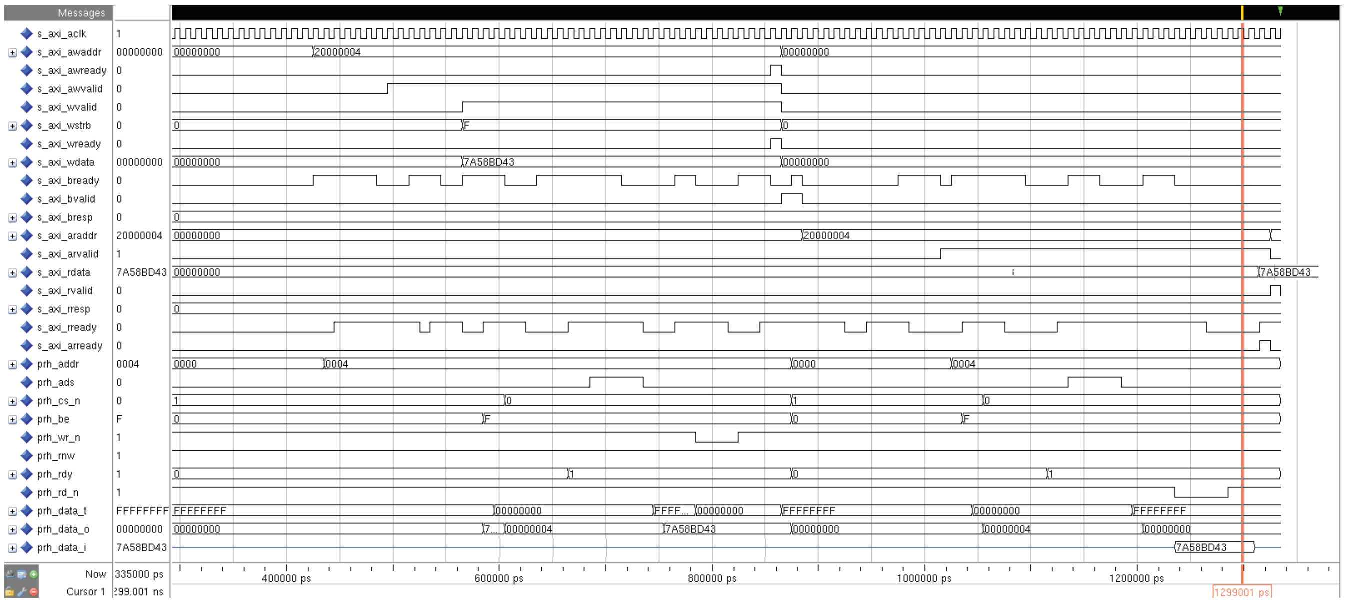

Figure 17: **Asynchronous Read and Write Transactions to Device Memory When Bus is Multiplexed and Data Width Matching is Disabled**

Scenario When the External Device is Not Ready for Transaction

In scenarios where the AXI EPC IP Core has initiated the transaction, but the external peripheral is not ready (PRH_Rdy signal is not activated), the control signals from the core is extended until the internal C_PRHx_RDY_WIDTH counter expires. The core completes the transaction with error signal generation.

Target Technology

The intended target technology is listed in the Supported Device Family field of the LogiCORE IP Facts Table.

Device Utilization and Performance Benchmarks

Core Performance

Because the AXI EPC module is used with other design modules in the FPGA, the utilization and timing numbers reported in this section are estimates. When the AXI EPC module is combined with other designs, the utilization of FPGA resources and timing of the AXI EPC IP Core design varies from the results reported here.

The AXI EPC IP Core resource utilization for various parameter combinations measured with Virtex-7 as the target device are detailed in Table 5.

Table 5: **Performance and Resource Utilization Benchmarks for Virtex-7 FPGA (XC7V855T-FFG1157-3)**

Parameter Values (Other Parameters at Default Value)							Device Resources			Performance
C_NUM_PERIPHERALS	C_PRH_CLK_SUPPORT	C_PRHx_AWIDTH	C_PRHx_DWIDTH	C_PRHx_SYNC	C_PRHx_BUS_MULTIPLEX	C_PRHx_DWIDTH_MATCH	Slices	Slice Flip- Flops	LUTs	F_{Max} (MHz)
1	0	3	32	0	0	0	54	153	109	285.796
2	0	12,12	32,32	0,0	0,1	0,0	82	187	182	201.939
2	1	12,12	32,32	1,1	0,1	0,0	63	150	135	217.77
3	0	6,12,12	32,16,8	0,0,0	0,0,0	1,1,1	63	123	132	239.866
3	0	12,12,12	32,16,8	1,1,1	0,0,0	1,1,1	135	179	323	200.844
3	1	12,12,12	32,16,8	1,1,1	1,1,1	1,1,1	116	156	296	201.329
4	0	12,12,12,12	16,8,16,32	0,0,0,0	0,0,0,0	1,1,1,1	133	171	334	217.25
4	0	12,12,12,12	32,8,32,8	0,0,0,0	0,0,0,0	0,0,0,0	143	187	332	207.426

Notes:

The bus multiplex timing parameters and asynchronous timing parameters are set to typical values for all devices. The values used for various timing parameters are as follows:

- C_PRHx_ADDR_TSU = 6 ns
- C_PRHx_ADDR_TH = 6 ns
- C_PRHx_ADS_WIDTH = 10 ns
- C_PRHx_CSN_TSU = 6 ns
- C_PRHx_CSN_TH = 6 ns
- C_PRHx_WRN_WIDTH = 15 ns
- C_PRHx_WR_CYCLE = 30 ns
- C_PRHx_DATA_TSU = 10 ns
- C_PRHx_DATA_TH = 5 ns
- C_PRHx_RDN_WIDTH = 15 ns
- C_PRHx_RD_CYCLE = 30 ns
- C_PRHx_DATA_TOUT = 5 ns
- C_PRHx_DATA_TINV = 10 ns
- C_PRHx_RDY_TOUT = 10 ns
- C_PRHx_RDY_WIDTH = 50 ns

The AXI EPC IP Core resource utilization for various parameter combinations measured with Kintex-7 and Zynq-7000 as the target devices are detailed in Table 6.

Table 6: **Performance and Resource Utilization Benchmarks for Kintex-7 FPGA[1] and Zynq-7000 Device**

Parameter Values (Other Parameters at Default Value)							Device Resources			Performance
C_NUM_PERIPHERALS	C_PRH_CLK_SUPPORT	C_PRHx_AWIDTH	C_PRHx_DWIDTH	C_PRHx_SYNC	C_PRHx_BUS_MULTIPLEX	C_PRHx_DWIDTH_MATCH	Slices	Slice Flip- Flops	LUTs	F_{Max} (MHz)
1	0	3	32	0	0	0	59	153	108	239.234
2	0	12,12	32,32	0,0	0,1	0,0	81	187	194	204.583
2	1	12,12	32,32	1,1	0,1	0,0	63	150	135	222.965
3	0	6,12,12	32,16,8	0,0,0	0,0,0	1,1,1	62	123	132	223.764
3	0	12,12,12	32,16,8	1,1,1	0,0,0	1,1,1	139	179	323	223.015
3	1	12,12,12	32,16,8	1,1,1	1,1,1	1,1,1	116	156	296	219.491
4	0	12,12,12,12	16,8,16,32	0,0,0,0	0,0,0,0	1,1,1,1	132	171	333	234.082
4	0	12,12,12,12	32,8,32,8	0,0,0,0	0,0,0,0	0,0,0,0	150	187	333	212.089

Notes:

The bus multiplex timing parameters and asynchronous timing parameters are set to typical values for all devices. The values used for various timing parameters are as follows:

- C_PRHx_ADDR_TSU = 6 ns
- C_PRHx_ADDR_TH = 6 ns
- C_PRHx_ADS_WIDTH = 10 ns
- C_PRHx_CSN_TSU = 6 ns
- C_PRHx_CSN_TH = 6 ns
- C_PRHx_WRN_WIDTH = 15 ns
- C_PRHx_WR_CYCLE = 30 ns
- C_PRHx_DATA_TSU = 10 ns
- C_PRHx_DATA_TH = 5 ns
- C_PRHx_RDN_WIDTH = 15 ns
- C_PRHx_RD_CYCLE = 30 ns
- C_PRHx_DATA_TOUT = 5 ns
- C_PRHx_DATA_TINV = 10 ns
- C_PRHx_RDY_TOUT = 10 ns
- C_PRHx_RDY_WIDTH = 50 ns

1. Kintex-7 (XC7K410T-FFG676-3)

The AXI EPC IP Core resource utilization for various parameter combinations measured with Artix-7 and Zynq-7000 as the target devices are detailed in Table 7.

Table 7: **Performance and Resource Utilization Benchmarks for Artix-7 FPGA[1] and Zynq-7000 Device**

Parameter Values (Other Parameters at Default Value)							Device Resources			Performance
C_NUM_PERIPHERALS	C_PRH_CLK_SUPPORT	C_PRHx_AWIDTH	C_PRHx_DWIDTH	C_PRHx_SYNC	C_PRHx_BUS_MULTIPLEX	C_PRHx_DWIDTH_MATCH	Slices	Slice Flip- Flops	LUTs	F_{Max} (MHz)
1	0	3	32	0	0	0	59	153	111	257.334
2	0	12,12	32,32	0,0	0,1	0,0	94	187	207	202.143
2	1	12,12	32,32	1,1	0,1	0,0	59	149	139	215.378
3	0	6,12,12	32,16,8	0,0,0	0,0,0	1,1,1	59	123	130	212.224
3	0	12,12,12	32,16,8	1,1,1	0,0,0	1,1,1	122	178	317	188.111
3	1	12,12,12	32,16,8	1,1,1	1,1,1	1,1,1	110	155	296	217.817
4	0	12,12,12,12	16,8,16,32	0,0,0,0	0,0,0,0	1,1,1,1	126	171	334	164.799
4	0	12,12,12,12	32,8,32,8	0,0,0,0	0,0,0,0	0,0,0,0	131	185	326	186.359

Notes:

The bus multiplex timing parameters and asynchronous timing parameters are set to typical values for all devices. The values used for various timing parameters are as follows:

- C_PRHx_ADDR_TSU = 6 ns
- C_PRHx_ADDR_TH = 6 ns
- C_PRHx_ADS_WIDTH = 10 ns
- C_PRHx_CSN_TSU = 6 ns
- C_PRHx_CSN_TH = 6 ns
- C_PRHx_WRN_WIDTH = 15 ns
- C_PRHx_WR_CYCLE = 30 ns
- C_PRHx_DATA_TSU = 10 ns
- C_PRHx_DATA_TH = 5 ns
- C_PRHx_RDN_WIDTH = 15 ns
- C_PRHx_RD_CYCLE = 30 ns
- C_PRHx_DATA_TOUT = 5 ns
- C_PRHx_DATA_TINV = 10 ns
- C_PRHx_RDY_TOUT = 10 ns
- C_PRHx_RDY_WIDTH = 50 ns

1. Artix-7 FPGA (XC7A355TDIE)

The AXI EPC IP Core resource utilization for various parameter combinations measured with Virtex-6 as the target device are detailed in Table 8.

Table 8: **Performance and Resource Utilization Benchmarks for Virtex-6 FPGA (XC6VLX130T-FF1156-1)**

Parameter Values (Other Parameters at Default Value)							Device Resources			Performance
C_NUM_PERIPHERALS	C_PRH_CLK_SUPPORT	C_PRHx_AWIDTH	C_PRHx_DWIDTH	C_PRHx_SYNC	C_PRHx_BUS_MULTIPLEX	C_PRHx_DWIDTH_MATCH	Slices	Slice Flip- Flops	LUTs	F_{Max} (MHz)
1	0	3	32	0	0	0	48	153	100	275
2	0	12,12	32,32	0,0	0,1	0,0	83	187	208	222
2	1	12,12	32,32	1,1	0,1	0,0	53	121	128	216
3	0	6,12,12	32,16,8	0,0,0	0,0,0	1,1,1	114	179	317	213
3	0	12,12,12	32,16,8	1,1,1	0,0,0	1,1,1	113	160	299	205
3	1	12,12,12	32,16,8	1,1,1	1,1,1	1,1,1	119	169	326	201
4	0	12,12,12,12	16,8,16,32	0,0,0,0	0,0,0,0	1,1,1,1	119	189	347	232
4	0	12,12,12,12	32,8,32,8	0,0,0,0	0,0,0,0	0,0,0,0	77	173	213	326

Notes:

The bus multiplex timing parameters and asynchronous timing parameters are set to typical values for all devices. The values used for various timing parameters are as follows:

- C_PRHx_ADDR_TSU = 6 ns
- C_PRHx_ADDR_TH = 6 ns
- C_PRHx_ADS_WIDTH = 10 ns
- C_PRHx_CSN_TSU = 6 ns
- C_PRHx_CSN_TH = 6 ns
- C_PRHx_WRN_WIDTH = 15 ns
- C_PRHx_WR_CYCLE = 30 ns
- C_PRHx_DATA_TSU = 10 ns
- C_PRHx_DATA_TH = 5 ns
- C_PRHx_RDN_WIDTH = 15 ns
- C_PRHx_RD_CYCLE = 30 ns
- C_PRHx_DATA_TOUT = 5 ns
- C_PRHx_DATA_TINV = 10 ns
- C_PRHx_RDY_TOUT = 10 ns
- C_PRHx_RDY_WIDTH = 50 ns

The AXI EPC IP Core resource utilization measured with Spartan-6 as the target device is shown in the following Table 9.

Table 9: **Performance and Resource Utilization Benchmarks for Spartan-6 FPGA (XC6SLX45T-FGG484-2)**

Parameter Values (Other Parameters at Default Value)							Device Resources			Performance
C_NUM_PERIPHERALS	C_PRH_CLK_SUPPORT	C_PRHx_AWIDTH	C_PRHx_DWIDTH	C_PRHx_SYNC	C_PRHx_BUS_MULTIPLEX	C_PRHx_DWIDTH_MATCH	Slices	Slice Flip- Flops	LUTs	F_{Max} (MHz)
1	0	3	32	0	0	0	55	153	98	178
2	0	12,12	32,32	0,0	0,1	0,0	88	190	197	157
2	1	12,12	32,32	1,1	0,1	0,0	63	124	129	141
3	0	6,12,12	32,16,8	0,0,0	0,0,0	1,1,1	124	189	330	170
3	0	12,12,12	32,16,8	1,1,1	0,0,0	1,1,1	103	167	298	158
3	1	12,12,12	32,16,8	1,1,1	1,1,1	1,1,1	128	169	326	118
4	0	12,12,12,12	16,8,16,32	0,0,0,0	0,0,0,0	1,1,1,1	131	197	343	158
4	0	12,12,12,12	32,8,32,8	0,0,0,0	0,0,0,0	0,0,0,0	76	176	215	162

Notes:

The bus multiplex timing parameters and asynchronous timing parameters are set to typical values for all devices. The values used for various timing parameters are as follows:

- C_PRHx_ADDR_TSU = 6 ns
- C_PRHx_ADDR_TH = 6 ns
- C_PRHx_ADS_WIDTH = 10 ns
- C_PRHx_CSN_TSU = 6 ns
- C_PRHx_CSN_TH = 6 ns
- C_PRHx_WRN_WIDTH = 15 ns
- C_PRHx_WR_CYCLE = 30 ns
- C_PRHx_DATA_TSU = 10 ns
- C_PRHx_DATA_TH = 5 ns
- C_PRHx_RDN_WIDTH = 15 ns
- C_PRHx_RD_CYCLE = 30 ns
- C_PRHx_DATA_TOUT = 5 ns
- C_PRHx_DATA_TINV = 10 ns
- C_PRHx_RDY_TOUT = 10 ns
- C_PRHx_RDY_WIDTH = 50 ns

System Performance

To measure the system performance (F_{Max}) of this core, axi_epc_v1_00_a core is added to the Spartan-6 and Virtex-6 series as the Device Under Test (DUT) as shown in Figure 18.

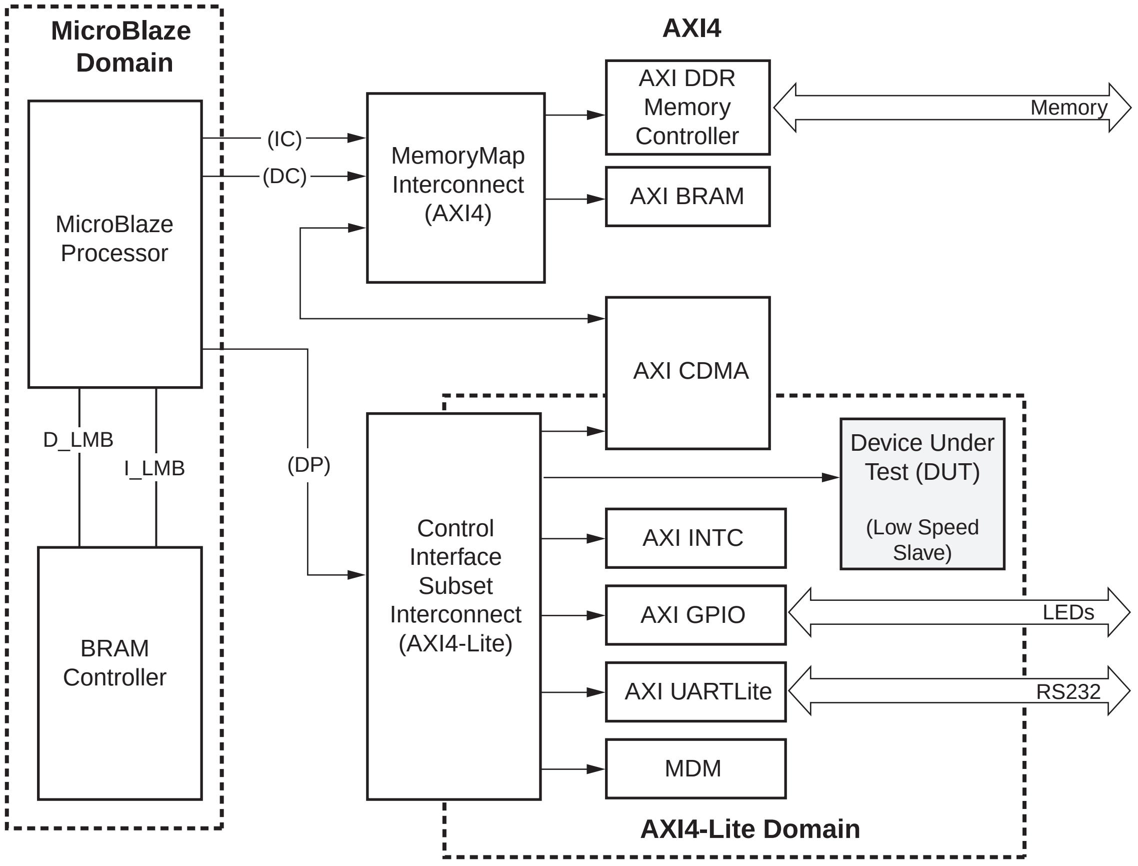

Figure 18: **Virtex-6 and Spartan-6 Devices F_{Max} Margin System**

The target FPGA was then filled with logic to drive the LUT and BRAM utilization to approximately 70% and the I/O utilization to approximately 80%. Using the default tool options and the slowest speed grade for the target FPGA, the resulting target F_{Max} numbers are shown in Table 10.

Table 10: **AXI EPC System Performance**

Target FPGA	Target F_{Max} (MHz)
Artix-7	110
Virtex-7	180
Kintex-7	180
Virtex-6	180
Spartan-6	110

The target F_{Max} is influenced by the exact system and is provided for guidance. It is not a guaranteed value across all systems.

References

To search for Xilinx documentation, go to www.xilinx.com/support.

1. DS583, *XPS SYSACE (System ACE™) Interface Controller*
2. DS765, *LogiCORE IP AXI4-Lite IPIF (v1.01.a) Data Sheet*
3. DS768, *LogiCORE IP AXI Interconnect Data Sheet*
4. DS160, *Spartan-6 Family Overview*
5. DS150, *Virtex-6 Family Overview*
6. DS180, *7 Series FPGAs Overview*
7. UG814, *Vivado Design Suite Getting Started Guide*
8. LAN91C111, *10/100 Non-PCI Ethernet Single Chip MAC + PHY Data Sheet, SMSC*
9. CY7C67300, *EZ-Host™ Programmable Embedded USB Host/Peripheral Controller Data Sheet*, Cypress Semiconductor
10. *ARM® AMBA® AXI4 Protocol Version: 2.0 Specification* http://www.arm.com/products/system-ip/amba/amba-open-specifications.php

Technical Support

Xilinx provides technical support at www.xilinx.com/support for this LogiCORE IP product when used as described in the product documentation. Xilinx cannot guarantee timing, functionality, or support of product if implemented in devices that are not defined in the documentation, if customized beyond that allowed in the product documentation, or if changes are made to any section of the design labeled DO NOT MODIFY.

See the IDS Embedded Edition Derivative Device Support web page (www.xilinx.com/ise/embedded/ddsupport.htm) for a complete list of supported derivative devices for this core.

Licensing and Ordering Information

This Xilinx LogiCORE IP module is provided at no additional cost with the Xilinx Vivado Design Suite and ISE Design Suite Embedded Edition tools under the terms of the Xilinx End User License.

Information about this and other Xilinx LogiCORE IP modules is available at the Xilinx Intellectual Property page. For information on pricing and availability of other Xilinx LogiCORE IP modules and tools, contact your local Xilinx sales representative.

Revision History

Date	Version	Revision
03/01/11	1.0	Initial Xilinx release
06/22/11	2.0	Updated for Xilinx tools v13.2. Added support for Artix-7, Virtex-7, and Kintex-7 devices.
07/25/12	3.0	• Updated for Vivado and Zynq features with minor document updates for v14.2 • Added Peripheral clock frequency should always be less to Sync Control Module • Added note to C_PRH_CLK_PERIOD_PS

Notice of Disclaimer