LogiCORE IP XPS Universal Serial Bus 2.0 Device (v7.01a)

DS639 October 16, 2012 **Product Specification**

Introduction

The Xilinx® Universal Serial Bus 2.0 High Speed Device with Processor Local Bus (PLB) v4.6 enables Universal Serial Bus (USB) connectivity to a user design with a minimal amount of resources. This interface is suitable for USB-centric, high-performance designs, bridges, and legacy port replacement operations.

Features

- 32-bit Processor Local Bus (PLB) Master and Slave Interface based on the PLB v4.6 specification
- Compliant with the USB 2.0 Specification
- Supports High Speed and Full Speed
- Supports Universal Transceiver Macrocell Interface (UTMI) + Low Pin Interface (ULPI) to external USB physical-side interface (PHY)
- Supports ULPI PHY register access
- Supports Parameterized ULPI PHY Reset
- Supports Resume and Reset detection in low-power mode
- Supports Resuming of Host from Low-power mode with Remote Wake-up signalling
- Supports eight endpoints, including one control endpoint 0. Endpoints 1 - 7 can be bulk, interrupt, or isochronous. Endpoints are individually configurable
- Supports two ping-pong buffers for each endpoint except for endpoint 0
- USB Error detection, updates Error Count and generates Error interrupt
- Direct Memory Access (DMA) mode to increase throughput during the data transfers

LogiCORE IP Facts Table				
Core Specifics				
Supported Device Family[1]	Virtex®-6, Spartan®-6, Virtex-5, Virtex-4, Spartan-3			
Supported User Interfaces	PLBv46 Master/Slave			
Resources Used				
	LUTs	**Block RAMs**	**FFs**	**DSP Slices**
See Table 26 and Table 27				
Provided with Core				
Documentation	Product Specification			
Design Files	VHSIC Hardware Description Language (VHDL)			
Example Design	Not Provided			
Test Bench	Not Provided			
Constraints File	User Constraints File (UCF)			
Simulation Model	ModelSim/NC protect encrypted			
Supported S/W Driver[2]	Standalone			
Tested Design Flows[3]				
Design Entry	Embedded Development Kit (EDK) 14.3			
Simulation	Mentor Graphics ModelSim			
Synthesis	Xilinx Synthesis Technology (XST)			
Support				
Provided by Xilinx @ www.xilinx.com/support				

1. For a complete list of supported derivative devices, see Embedded Edition Derivative Device Support.
2. Standalone driver details can be found in the EDK or SDK directory (<*install_directory*>/doc/usenglish/xilinx_drivers.htm). Linux OS and driver support information is available from //wiki.xilinx.com.
3. For the supported versions of the tools, see the Xilinx Design Tools: Release Notes Guide.

© Copyright 2007–2012 Xilinx, Inc. Xilinx, the Xilinx logo, Artix, ISE, Kintex, Spartan, Virtex, Vivado, Zynq, and other designated brands included herein are trademarks of Xilinx in the United States and other countries. All other trademarks are the property of their respective owners.

Functional Description

The USB 2.0 protocol multiplexes many devices over a single, half-duplex, serial bus. The bus runs at 480 Mb/s (High Speed) or at 12 Mb/s (Full Speed) and is designed to be plug-and-play. The host always controls the bus and sends tokens to each device specifying the required action. Each device has an address on the USB 2.0 bus and has one or more endpoints that are sources or sinks of data. All devices have the system control endpoint (endpoint 0).

The Xilinx Platform Studio (XPS) USB 2.0 Device has eight endpoints, one control endpoint (endpoint 0) and seven user endpoints.

Endpoint 0 of the USB 2.0 Device has different requirements than the seven user endpoints. Endpoint 0 handles control transactions only, which start with an 8-byte setup packet and then are followed by zero or more data packets. The setup packet is always stored in a dedicated location in the Dual Port Random Access Memory (DPRAM) at an address offset of 0x80. When a setup packet is received, the SETUPPkt bit of the Interrupt Status Register (ISR) is set. Data packets are a maximum of 64 bytes. These data packets are stored in a single bidirectional data buffer set up by the configuration memory of Endpoint 0 located at the address offset 0x0 in the DPRAM. When a data packet is transmitted or received successfully, the FIFOBufFree and FIFOBufRdy bits of the Interrupt Status Register (ISR) are set respectively.

The seven user endpoints of the USB 2.0 Device can be configured as bulk, interrupt or isochronous. In addition, endpoints can be configured as INPUT (to the host) or OUTPUT (from the host). Each of these endpoints has two ping-pong buffers of the same size for endpoint data. The user endpoints data buffers are unidirectional and are configured by the Endpoint Configuration and Status register of the respective endpoint. The size of the buffers can be configured from 0 to 512 bytes for bulk, 64 bytes for interrupt, and up to 1,024 bytes for isochronous endpoints.

The XPS USB 2.0 High Speed Device core with the PLB and ULPI interfaces is shown in Figure 1 and described in the subsequent sections.

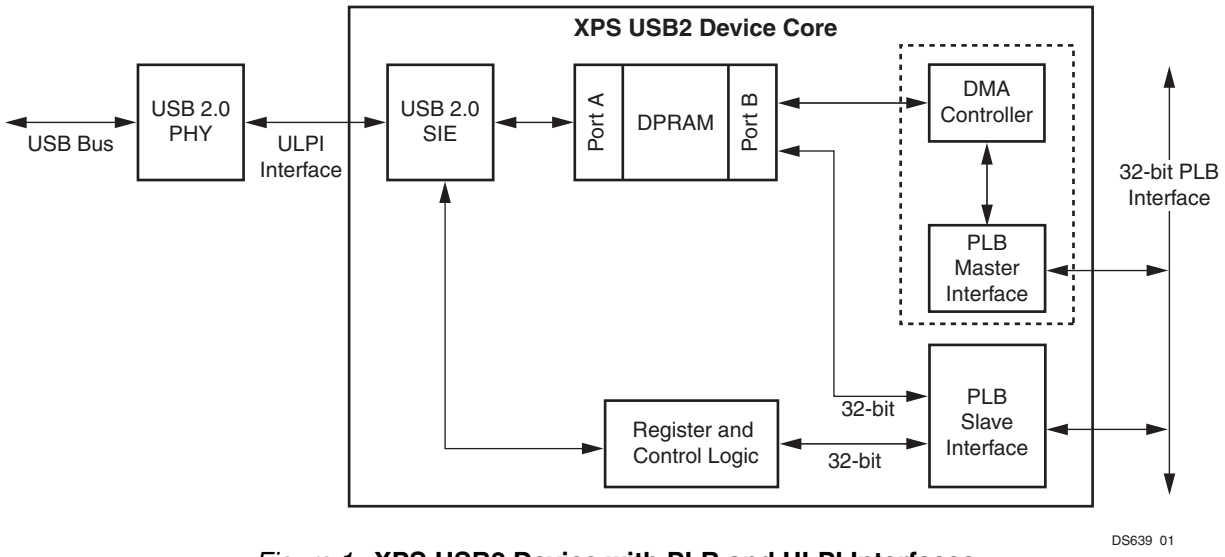

Figure 1: **XPS USB2 Device with PLB and ULPI Interfaces**

When the host wants to send data to an endpoint of the device, it sends a token, which consists of an OUT PID along with the address of the device and the endpoint number, followed by the data. The device uses handshake packets to report the status of the data transaction if, and only if, the token and data are received without any USB errors, such as Bit Stuff, PID and Cyclic Redundancy Check (CRC) errors. Handshake packets indicate successful reception of data, flow control, and halt conditions.

To receive data, the host sends a token that consists of an IN PID along with the device address and endpoint number and then waits for data from the device. The device responds with the requested data to the host if and only if the token is received without any USB errors. The device then waits for a handshake packet.

Register and Control Logic

The XPS USB 2.0 Device includes a few 32-bit registers that provide control and status information of the core. These registers are accessible from the PLB Slave Interface. An Interrupt Enable Register (IER) allows the generation of a PLB interrupt based on specific Interrupt Status Register (ISR) bits. For more information on the registers see Register Description.

USB 2.0 Serial Interface Engine (SIE)

The USB 2.0 Serial Interface Engine (SIE) handles the serialization and de-serialization of USB traffic at the byte level and the multiplexing and demultiplexing of USB data to and from the endpoints of the core. The SIE also handles USB 2.0 state transitions, such as suspend, resume, USB reset, and remote wake-up signalling. (To wake-up the host from suspend mode).

The SIE interfaces to the PHY using a ULPI interface that requires 12 pins. Data to the Field Programmable Gate Array (FPGA) from the USB is received from the PHY, error checked, and loaded into the appropriate area of the DPRAM. Data from the FPGA that is to be sent over the USB is loaded from the DPRAM, protocol wrapped. Then when the protocol allows, is presented to the PHY, one byte at a time. Details of the ULPI interfaces is beyond the scope of this document.

If any packet has USB errors, the SIE ignores and discards it. The SIE increases the respective USB error count in the Error Count Register (ECR) and sets the respective USB error bits in the Interrupt Status Register (ISR).

The status of the current USB transactions are signalled by the SIE to the Interrupt Status Register (ISR). Certain conditions can be enabled through the Interrupt Enable Register (IER) to generate an interrupt.

Control of the SIE is from the following four sources:

- The lower 64 bytes of the DPRAM contain the control and status locations for each endpoint.
- The Control and Status Registers provide overall start and stop, status indication, enabling of interrupts using status register bits, address control on USB, current Start of Frame timing information, and endpoint Buffer Ready indication.
- The logic of the SIE module is coded to reflect the requirements of Chapter 8 of the USB 2.0 Specification.
- The USB physical bus is connected to the SIE over the ULPI PHY interface. The SIE natively implements the ULPI protocol.

Dual Port Block RAM (DPRAM)

The DPRAM is the data storage area between the SIE and PLB interface. Port A of the DPRAM is used by the SIE and Port B is used by the Processor/DMA controller. Both ports are 32-bit wide.

The XPS USB2 Device uses two sets of four block RAMs implemented as 64 x 8 bits (DPRAM1) and 2 K x 8 bits each (DPRAM2), with dual asynchronous clock ports.

Data from the USB 2.0 Device is stored in the appropriate locations in the DPRAM by the SIE through Port A. The firmware or hardware being utilized by the user accesses the data through Port B over the PLB Slave Interface. Data to the USB 2.0 Device is loaded by the user through the PLB Slave Interface to Port B, into appropriate locations in the DPRAM. When the host requests data from the device, the SIE accesses this data from Port A.

The DPRAM is seen by the SIE as DPRAM1 and DPRAM2. DPRAM2 has seven endpoint First In First Outs (FIFOs) for endpoint 1-7. DPRAM1 has the endpoint 0 FIFO and the control register area that defines how the memory is arranged and reports the status of each FIFO buffer (ready, not ready, and count).

Each FIFO is double-buffered to help support the high throughput possible with USB 2.0. One buffer can be used for a current USB transaction, while the other buffer is available to the user application for processing. The storage areas are treated as FIFOs only by the SIE. The firmware or hardware utilized by the user can access the storage as ordinary RAM over the PLB.

PLB Slave Interface

The PLB Slave interface in the core performs the following operations:

- Responds to PLB transactions to read-from or write-into the 32-bit control registers, status registers, and DPRAM.
- Supports byte, half word, and word transfers for the DPRAM, but only word transfers are supported for the registers.

PLB Master Interface

The PLB Master interface in the core performs the following operations:

- Performs read and write transactions as a PLB master in DMA mode.
- In DMA mode, interrupts are generated based on DMA done and DMA error conditions.

DMA Controller

The DMA Controller is included in the XPS USB 2.0 Device core if the C_INCLUDE_DMA parameter is set to 1. The DMA Controller provides simple Direct Memory Access services to the DPRAM2 and external memory device or peripheral on the PLB and vice versa. The DMA controller transfers data from a source address to a destination address without processor intervention for a given length. It provides programmable registers for direction (read-from/write-into DPRAM2), source address, destination address, and transfer length.The source and destination addresses counters are increment-only. It also supports PLB burst transfers. The DMA Controller in the core does the data transfers from/to DPRAM2 to/from external memory only.

USB 2.0 PHY

The USB PHY can be any ULPI compliant PHY. The primary function of the PHY is to manage the bit-level serialization and de-serialization of USB 2.0 traffic. To do so, it must detect and recover the USB clock. The clock runs at 480 Mega Hertz (MHz), a speed that is too fast for practical implementation on the FPGA. Because 480 MHz is also too fast for the USB SIE clock, the PHY interfaces to the SIE on a byte serial basis and generates a 60 MHz clock which runs the SIE side of the USB 2.0 Device.

Input/Output (I/O) Signals

A description of the I/O signals for the XPS USB 2.0 Device core is provided in Table 1.

Table 1: **XPS USB 2.0 Device I/O Signals**

Port	Signal Name	Interface	I/O	Initial State	Description
PLB Master Interface Signals					
P1	MPLB_Clk	PLB	I	-	PLB master clock
P2	MPLB_Rst	PLB	I	-	PLB master reset
P3	M_ABus[0:C_MPLB_ AWIDTH - 1]	PLB	O	0	Master address bus
P4	M_BE[0:C_MPLB_DWIDTH/8 - 1]	PLB	O	0	Master byte enables
P5	M_wrDBus[0:C_MPLB_DWIDTH - 1]	PLB	O	0	Master write data bus
P6	M_request	PLB	O	0	Master bus request
P7	M_RNW	PLB	O	0	Master read not write
P8	M_priority[0:1]	PLB	O	0	Master bus request priority
P9	M_rdBurst	PLB	O	0	Master burst read transfer indicator
P10	M_type[0:2]	PLB	O	0	Master transfer type
P11	M_size[0:3]	PLB	O	0	Master transfer size
P12	M_wrBurst	PLB	O	0	Master burst write transfer indicator
P13	M_MSize[0:1]	PLB	O	0	Master data bus size
P23	M_BusLock	PLB	O	0	Master bus lock
P14	MPLB_MRdDBus[0:C_MPLB_DWIDTH -1]	PLB	I	-	PLB master read data bus
P15	MPLB_MRdErr	PLB	I	-	PLB master slave read error indicator
P16	MPLB_MWrErr	PLB	I	-	PLB master slave write error indicator
P17	MPLB_MWrBterm	PLB	I	-	PLB master terminate write burst indicator
P18	MPLB_MWrDAck	PLB	I	-	PLB master write data acknowledge
P19	MPLB_MAddrAck	PLB	I	-	PLB master address acknowledge
P20	MPLB_MRdBTerm	PLB	I	-	PLB master terminate read burst indicator
P21	MPLB_MRdDAck	PLB	I	-	PLB master read data acknowledge
P22	MPLB_MRearbitrate	PLB	I	-	PLB master bus rearbitrate indicator
P24	MPLB_MSSize[0:1]	PLB	I	-	PLB slave data bus size
P25	MPLB_MTimeout	PLB	I	-	PLB master bus time out
Unused PLB Master Interface Signals					
P26	M_TAttribute[0:15]	PLB	O	0	Master Transfer Attribute bus
P27	M_lockErr	PLB	O	0	Master lock error indicator
P28	M_abort	PLB	O	0	Master abort bus request indicator
P29	M_UABus[0:31]	PLB	O	0	Master upper address bus
P30	MPLB_MBusy	PLB	I	-	PLB master slave busy indicator
P31	MPLB_MIRQ	PLB	I	-	PLB master slave interrupt indicator
P32	MPLB_MRdWdAddr[0:3]	PLB	I	-	PLB master read word address

Table 1: **XPS USB 2.0 Device I/O Signals *(Cont'd)***

Port	Signal Name	Interface	I/O	Initial State	Description
PLB Slave Interface Signals					
P33	SPLB_Clk	PLB	I	-	PLB clock
P34	SPLB_Rst	PLB	I	-	PLB reset
P35	SPLB_ABus[0:C_SPLB_AWIDTH - 1]	PLB	I	-	PLB address bus
P36	SPLB_type[0:2]	PLB	I	-	PLB transfer type
P37	SPLB_size[0:3]	PLB	I	-	PLB transfer size
P38	SPLB_BE[0:C_SPLB_ DWIDTH/8 - 1]	PLB	I	-	PLB byte enables
P39	SPLB_wrDBus[0:C_SPLB_DWIDTH - 1]	PLB	I	-	PLB write data bus
P40	SPLB_RNW	PLB	I	-	PLB read not write
P41	SPLB_PAValid	PLB	I	-	PLB primary address valid indicator
P42	SPLB_masterID[0: C_ SPLB_MIDWIDTH - 1]	PLB	I	-	PLB current master identifier
P43	Sl_rdDBus[0:C_SPLB_ DWIDTH - 1]	PLB	O	0	Slave read bus
P44	Sl_addrAck	PLB	O	0	Slave address acknowledge
P45	Sl_SSize[0:1]	PLB	O	0	Slave data bus size
P46	Sl_MWrErr[0:C_SPLB_NUM_MASTERS - 1]	PLB	O	0	Slave write error indicator
P47	Sl_MRdErr[0:C_SPLB_NUM_MASTERS - 1]	PLB	O	0	Slave read error indicator
P48	Sl_wait	PLB	O	0	Slave wait indicator
P49	Sl_rearbitrate	PLB	O	0	Slave rearbitrate bus indicator
P50	Sl_wrDAck	PLB	O	0	Slave write data acknowledge
P51	Sl_wrComp	PLB	O	0	Slave write transfer complete indicator
P52	Sl_rdDAck	PLB	O	0	Slave read data acknowledge
P53	Sl_rdComp	PLB	O	0	Slave read transfer complete indicator
P54	Sl_MBusy[0:C_SPLB_NUM_MASTERS - 1]	PLB	O	0	Slave busy indicator
Unused PLB Slave Interface Signals					
P55	SPLB_UABus[0:31]	PLB	I	-	PLB address bus
P56	SPLB_SAValid	PLB	I	-	PLB secondary address valid indicator
P57	SPLB_rdPrim	PLB	I	-	PLB secondary to primary read request indicator
P58	SPLB_wrPrim	PLB	I	-	PLB secondary to primary write request indicator
P59	SPLB_abort	PLB	I	-	PLB abort bus request indicator
P60	SPLB_busLock	PLB	I	-	PLB lock
P61	SPLB_MSize[0:1]	PLB	I	-	PLB master data bus size
P62	SPLB_lockerr	PLB	I	-	PLB lock error indicator
P63	SPLB_rdBurst	PLB	I	-	PLB burst read transfer indicator
P64	SPLB_wrBurst	PLB	I	-	PLB burst write transfer indicator
P65	SPLB_rdpendReq	PLB	I	-	PLB pending bus read request indicator
P66	SPLB_wrpendReq	PLB	I	-	PLB pending bus write request indicator

Table 1: **XPS USB 2.0 Device I/O Signals *(Cont'd)***

Port	Signal Name	Interface	I/O	Initial State	Description
P67	SPLB_rdpendPri[0:1]	PLB	I	-	PLB pending read request priority
P68	SPLB_wrpendPri[0:1]	PLB	I	-	PLB pending write request priority
P69	SPLB_reqpri[0:1]	PLB	I	-	PLB current request priority
P70	SPLB_TAttribute[0:15]	PLB	I	-	PLB Transfer Attribute bus
P71	Sl_wrBTerm	PLB	O	0	Slave terminate write burst transfer
P72	Sl_rdBTerm	PLB	O	0	Slave terminate read burst transfer
P73	Sl_rdWdAddr[0:3]	PLB	O	0	Slave read word address
P74	Sl_MIRQ[0:C_SPLB_NUM_MASTERS - 1]	PLB	O	0	Slave interrupt indicator
System Signals					
P75	USB_Irpt	System	O	0	USB Interrupt
USB Specific Signals					
P76	ULPI_Clock	USB	I	-	All USB protocol interface signals are synchronous to this clock
P77	ULPI_Dir	USB	I	-	Direction of Data flow between PHY and SIE
P78	ULPI_Next	USB	I	-	Indicator of when the PHY is ready for the next bit
P79	ULPI_Stop	USB	O	0	Indicator that last byte transmission is complete
P80	ULPI_Reset(1)	USB	O	0	Active-High/Low Reset to the PHY
P81	ULPI_Data_I(7:0)	USB	I	-	Input Data from PHY to SIE
P82	ULPI_Data_O(7:0)	USB	O	0	Output Data from SIE to PHY
P83	ULPI_Data_T	USB	O	0	ULPI_Data is a 3-state port with ULPI_Data_I as the IN port, ULPI_Data_O as the OUT port and ULPI_Data_T as the 3-state output
Optional Ports Used for Debug Purpose					
P84	Configured	USB	O	0	Used for USB 2.0 Certification - Test mode 2
P85	Spare1	USB	O	0	Used for USB 2.0 Certification -Test mode 0
P86	Spare2	USB	O	0	Used for USB 2.0 Certification - Test mode 1
P87	Vbus_detect	USB	O	0	0 = Indicates valid V_{BUS} has not been detected 1 = Indicates Valid V_{BUS} has been detected
P88	Show_currentspeed	USB	O	0	0 = indicates Full-speed 1 = indicates High-speed
P89	Running	USB	O	0	0 = indicates that the SIE in reset state and does not respond to USB traffic 1= indicates that SIE finished USB reset and ready to respond to USB traffic
P90	Suspended	USB	O	0	0 = Indicates XPS USB 2.0 Device has been suspended 1 = Indicates XPS USB 2.0 Device has not been suspended
P91	Disconnected	USB	O	0	0 = Indicates XPS USB 2.0 Device connected 1 = Indicates XPS USB 2.0 Device disconnected

Notes:

1. The polarity of the ULPI Reset port is user configurable. The C_PHY_RESET_TYPE parameter defines the type of the ULPI Reset as either active-High or active-Low.

Design Parameters

To obtain an XPS USB 2.0 Device that is uniquely tailored to the user system requirements, certain features can be parameterized in the XPS USB 2.0 Device design. The features that can be parameterized in the Xilinx XPS USB2 Device design are shown in Table 2.

Table 2: **Design Parameters**

Generic	Feature/ Description	Parameter Name	Allowable Values	Reset Value	VHDL Type
System Parameter					
G1	Device family	C_FAMILY	aspartan3, spartan3, spartan3a, spartan3e, aspartan3a, aspartan3e, aspartan3adsp, spartan6, virtex4, qrvirtex4, qvirtex4, virtex5, virtex6	virtex4	string
PLB Parameters					
G2	XPS USB20 Device Base Address	C_BASEADDR	Valid Address(1),(2)	None	std_logic_vector
G3	XPS USB20 Device High Address	C_HIGHADDR	Valid Address(1),(2)	None	std_logic_vector
G4	PLB master data width	C_MPLB_DWIDTH	32,64,128	32	integer
G5	PLB master address width	C_MPLB_AWIDTH	32	32	integer
G6	PLB address width	C_SPLB_AWIDTH	32	32	integer
G7	PLB data width	C_SPLB_DWIDTH	32, 64, 128	32	integer
G8	PLB Master ID Bus Width	C_SPLB_MID_WIDTH	$\log_2$(C_SPLB_NUM_MASTERS) with a minimum value of 1	1	integer
G9	Number of PLB Masters	C_SPLB_NUM_MASTERS	1 - 16	1	integer
G10	Width of the Slave Data Bus	C_SPLB_NATIVE_WIDTH	32	32	integer
G11	Width of the Master Data Bus	C_MPLB_NATIVE_WIDTH	32	32	integer
USB Core Specific Parameters					
G12	Implementation of DMA	C_INCLUDE_DMA	0-1(3)	1	integer
G13	Polarity of ULPI Reset	C_PHY_RESET_TYPE	0-1(4)	1	integer
G14	Includes USB ERROR interrupt logic	C_INCLUDE_USBERR_LOGIC	0-1(5)	0	integer

Notes:

1. Address range specified by C_BASEADDR and C_HIGHADDR must be at least 0x8000 and must be a power of 2. C_BASEADDR must be a multiple of the range, where the range is C_HIGHADDR - C_BASEADDR + 1.
2. No default value is specified to ensure that the actual value is set, that is, if the value is not set, a compiler error is generated. The address range must be at least 0x7FFF. For example, C_BASEADDR = 0x80000000, C_HIGHADDR = 0x80007FFF.
3. If C_INCLUDE_DMA = 1, the DMA controller logic is included in the core.The user can access the RAM for endpoint 1-7 buffers through the DMA controller only; enabling DMA logic disconnects the AXI slave interface from endpoint 1-7 buffers.
4. If C_PHY_RESET_TYPE = 1, the core puts the active-High reset condition on the ULPI Reset port. Otherwise, the core drives active-Low hard reset condition.
5. If C_INCLUDE_USBERR_LOGIC = 1 includes USB ERROR interrupt generation logic, otherwise the core excludes the USB ERROR interrupt generation logic from the core. This parameter is added to reduce the resources used for the generation of USB ERROR interrupt and the respective counters from the core.

Allowable Parameter Combinations

The address-range size specified by C_BASEADDR and C_HIGHADDR must be a power of 2 and must be at least 0x7FFF. For example, if C_BASEADDR = 0x80000000, C_HIGHADDR must be at least = 0x80007FFF.

C_PLB_MIDWIDTH depends on C_PLB_NUM_MASTERS. It must be set to the maximum of 1 or the smallest integer greater than or equal to $\log_2$(C_PLB_NUM_MASTERS).

Port Dependencies

The width of some of the XPS USB 2.0 Device signals depends on parameters selected in the design. The dependencies between the XPS USB 2.0 Device design parameters and I/O signals are shown in Table 3.

Table 3: **Parameter Port Dependencies**

Generic or Port	Name	Affects	Depends	Relationship Description
Design Parameters				
G4	C_MPLB_DWIDTH	P4,P5,P14	-	Affects number of bits in PLB data bus.
G5	C_MPLB_AWIDTH	P3		Affects number of bits in PLB Master address bus.
G6	C_SPLB_AWIDTH	P35	-	Affects number of bits in PLB Slave address bus.
G7	C_SPLB_DWIDTH	P38,P39, P43	-	Affects number of bits in PLB Slave data bus.
G8	C_SPLB_MID_WIDTH	P42	G9	Affects the width of current master identifier signals and depends on $\log_2$(C_SPLB_NUM_MASTERS) with a minimum value of 1.
G9	C_SPLB_NUM_MASTERS	P46,P47, P54,P74	-	Affects the width of busy and error signals.
I/O Signals				
P3	M_ABUS[0:C_MPLB_ AWIDTH]	-	G5	Width varies depending on the size of the PLB master address bus.
P4	M_BE[0:C_MPLB_ DWIDTH/8 - 1]	-	G4	Width varies depending on the size of the PLB master data bus.
P5	M_wrDBus[0:C_MPLB_ DWIDTH - 1]	-	G4	Width varies depending on the size of the PLB master data bus.
P14	MPLB_MRdDBus[0:C_ MPLB_DWIDTH - 1]	-	G4	Width varies depending on the size of the PLB master data bus.
P35	SPLB_ABus[0:C_SPLB_ AWIDTH - 1]	-	G6	Width varies depending on the size of the PLB address bus.
P41	SPLB_BE[0:C_SPLB_ DWIDTH/8 - 1]	-	G7	Width varies depending on the size of the PLB data bus.
P42	SPLB_wrDBus[0:C_SPLB_ DWIDTH - 1]	-	G7	Width varies depending on the size of the PLB data bus.
P46	SPLB_masterID[0: C_SPLB_MIDWIDTH - 1]		G9	Width varies depending on the size of the PLB number of masters.
P59	Sl_rdDBus[0:C_SPLB_ DWIDTH - 1]	-	G7	Width varies depending on the size of the PLB data bus.
P62	Sl_MWrErr[0:C_SPLB_ NUM_MASTERS - 1]	-	G10	Width varies depending on the size of the PLB master ID width.

Table 3: **Parameter Port Dependencies *(Cont'd)***

Generic or Port	Name	Affects	Depends	Relationship Description
P63	SI_MRdErr[0:C_SPLB_NUM_MASTERS - 1]	-	G10	Width varies depending on the size of the PLB number of masters.
P70	SI_MBusy[0:C_SPLB_NUM_MASTERS - 1]	-	G10	Width varies depending on the size of the PLB number of masters.
P74	SI_MIRQ[0:C_SPLB_NUM_MASTERS - 1]	-	G10	Width varies depending on the size of the PLB number of masters.

Register Bit Ordering

All registers use big-endian bit ordering where bit-0 is Most Significant Bit (MSB) and bit-31 is LSB. Table 4 shows the bit ordering.

Table 4: **Register Bit Ordering**

0	1	2	. . .	29	30	31
MSB			. . .			LSB

Register Description

The memory map for the XPS USB 2.0 Device core is shown in Table 5 which includes endpoint configuration space (offset 0x0000), setup packet storage space (offset 0x0080), RAM for endpoint 0 buffer (offset 0x0088), register space for the USB registers (offset 0x0100), register space for the DMA registers (offset 0x200), and RAM for endpoint 1 - 7 buffers (offset 0x4000). Table 6 lists the mapping for the endpoint configuration space. All offsets are word offsets.

Table 5: **Register Address Map**(1),(2)

Register Name	Base Address + Offset (hex)	Reset Value (hex)	Access
Endpoint Configuration and Status Registers	C_BASEADDR + 0x0000	0x00000000	R/W
Setup Packet Storage Word 0(2)	C_BASEADDR + 0x0080	0x00000000	R
Setup Packet Storage Word 1(2)	C_BASEADDR + 0x0084	0x00000000	R
RAM for endpoint 0 buffer(5)	C_BASEADDR + 0x0088	0x00000000	R/W
USB Address Register	C_BASEADDR + 0x0100	0x00000000	R/W
Control Register	C_BASEADDR + 0x0104	0x00000000	R/W
Interrupt Status Register	C_BASEADDR + 0x0108	0x00000000	R
Frame Number Register	C_BASEADDR + 0x010C	0x00000000	R
Interrupt Enable Register	C_BASEADDR + 0x0110	0x00000000	R/W
Buffer Ready Register	C_BASEADDR + 0x0114	0x00000000	R/W
Test Mode Register	C_BASEADDR + 0x0118	0x00000000	R/W
Error Count Register(7)	C_BASEADDR + 0x011C	0x00000000	R
ULPI PHY Access Register	C_BASEADDR + 0x0120	0x00000000	R/W
DMA Software Reset Register(4)	C_BASEADDR + 0x0200	0x00000000	W
DMA Control Register(4)	C_BASEADDR + 0x0204	0x00000000	R/W
DMA Source Address Register(4)	C_BASEADDR + 0x0208	0x00000000	R/W

Table 5: **Register Address Map[1],[2] *(Cont'd)***

Register Name	Base Address + Offset (hex)	Reset Value (hex)	Access
DMA Destination Address Register[4]	C_BASEADDR + 0x020C	0x00000000	R/W
DMA Length Register[4]	C_BASEADDR + 0x0210	0x00000000	R/W
DMA Status Register[4]	C_BASEADDR + 0x0214	0x00000000	R/W
RAM for endpoint 1-7 buffers[6]	C_BASEADDR + 0x4000	0x00000000	R/W

Notes:

1. R/W - Read and Write access
2. R - Read Only access. Write into these registers does not have any effect.
3. W - Write Only access. Reading of these registers returns zero.
4. This Register is not included in the design if the parameter C_INCLUDE_DMA = 0.
5. RAM for endpoint 0 buffer should be 64 bytes, EpBase address (endpoint 0)+ 0x40 should not exceed 0x00FF.
6. RAM for endpoint 1 - 7 buffers range (C_BASEADDR + 0x4000) to (C_BASEADDR + 0x4000) + 0x1FFF. When the DMA logic is enabled, the user cannot access the RAM for endpoint 1-7 buffers through the PLB Slave interface; it is only accessible through the DMA controller.
7. This Register is not included in the design if the parameter C_INCLUDE_USBERR_LOGIC = 0. But SIE performs USB error checks as part of the USB 2.0 Specification and ignores the error packets if received.

Endpoint Configuration and Status Registers

The Endpoint Configuration and Status registers control the operational characteristics of each endpoint and reports its current condition. The total endpoint configuration register space is divided between the eight endpoints of the USB 2.0 Device as shown in Table 6.

Table 6: **Endpoint Configuration Registers** (*C_BASEADDR + Address Offset*)

Address Offset	Memory/Register Space
0x0000	Endpoint 0
0x0010	Endpoint 1
0x0020	Endpoint 2
0x0030	Endpoint 3
0x0040	Endpoint 4
0x0050	Endpoint 5
0x0060	Endpoint 6
0x0070	Endpoint 7

Each endpoint has four 32-bit registers that describe the behavior of the endpoint. These registers are located sequentially and arranged by endpoint number as shown in Table 7.

Table 7: **Endpoint Configuration Register**

Address Offset	Memory/Register Space
0x0000	Endpoint configuration and status register
0x0004	Reserved
0x0008	Buffer 0 count: 0 to 1024
0x000C	Buffer 1 count: 0 to 1024

The bit description for the Endpoint Configuration and Status Registers is given in Table 8. All the bits of this register can be modified by the firmware. Under normal operation, some of the bits are modified by the USB SIE itself and only their initial values need to be set by the firmware.

Table 8: **Endpoint Configuration and Status Register** *(C_BASEADDR + 0x0000)* (6),(7)

Bit(s)	Name	Access	Reset Value	Description
0	EpValid(1)	R/W	0	0 = disables the endpoint 1 = enables the endpoint
1	EpStall	R/W	0	0 = endpoint accepts INs and OUTs 1 = endpoint responds to the host only with a STALL
2	EpOutIn	R/W	0	0 = core receives data from the host (OUT from host) 1 = core sends data to the host (IN to host)
3	EpIso(2)	R/W	0	0 = not an isochronous endpoint 1 = is an isochronous endpoint
4	EpDataTgl(3)	R/W	0	0 = next DATA packet must be a DATA0 packet 1 = next DATA packet must be a DATA1 packet
5	EpBufSel(4)	R/W	0	0 = Buffer 0 is used 1 = Buffer 1 is used
6-16	EpPktSze(5)	R/W	0	Endpoint packet size
17-18	Reserved	NA	-	Reserved
19-31	EpBase(8)	R/W	0	Base offset of the buffers in the DPRAM

Notes:

1. EpValid is an MstEnbl bit for the respective endpoint.
2. EpIso enables endpoint as an isochronous endpoint.
3. SIE uses EpDataTgl to detect DATA PID toggle errors during the data transfers and its weak form of synchronization technique. This bit can be explicitly set in response to a firmware command.
4. Used to support ping-pong buffers during data transfers. If this bit is set, SIE toggles the buffer access during data transfers one at each time.
5. EpPktSze refers to maximum packet size limited to the respective endpoint.
6. The VALID, STALL, OUT_IN and ISO bits are set by the firmware to define how the endpoint operates. For example, to set up the endpoint to receive bulk OUT's from the host, set EpValid =1, EP_OUT_IN =0, EP_STALL =0, and EpIso =0.
7. The EpDataTgl and EpBufSel bits are modified by the SIE in response to the USB operations and only their initial values are set by the firmware.
8. EpBase is word addressable, the user should always right-shift the endpoint buffer address by 2 before writing EpBase.

Buffer Count Register0 (BCR0)

The Buffer 0 Count Registers shown in Table 9 indicate the size of data in bytes. If the endpoint is an OUT endpoint, then the SIE sets the value of this register at the end of a successful reception from the host. If the endpoint is an IN endpoint, the firmware sets the value of this register before transmission.

These registers are 32-bit wide and have R/W access.

Table 9: **Buffer Count Register0** *(C_BASEADDR + 0x0008)*

Bit(s)	Name	Access	Reset Value	Description
0-20	Reserved	NA	-	Reserved
21-31	OutInPktCnt	R/W	0	Packet count in the buffer

Buffer Count Register1 (BCR1)

The Buffer 1 Count Registers shown in Table 10 indicate the size of data in bytes. If the endpoint is an OUT endpoint, the SIE sets the value of this register at the end of a successful reception from the host. If the endpoint is an IN endpoint, the firmware sets the value of this register before transmission.

These registers are 32-bit wide and have R/W access.

Table 10: **Buffer Count Register1** *(C_BASEADDR + 0x000c)*

Bit(s)	Name	Access	Reset Value	Description
0-20	Reserved	NA	-	Reserved
21-31	OutInPktCnt	R/W	0	Packet count in the buffer

USB Address Register (UAR)

The USB Address register shown in Table 11 contains the host-assigned USB address of the device. There are 128 possible USB devices on the USB; therefore, the register takes values from 0 to 127. The lower seven bits of the register (6:0) are used to set the address. An address of 0 indicates that the device is un-enumerated. Address 0 is the default address of all USB devices at plug-in time and the address value on hardware reset. This register is 32-bits wide and has R/W access.

Table 11: **USB Address Register** *(C_BASEADDR + 0x0100)*

Bit(s)	Name	Access	Reset Value	Description
0-24	Reserved	NA	-	Reserved
25-31	USBAddr	R/W	0	Indicates the USB address of the device.

Control Register (CR)

As shown in Table 12, the MstRdy bit indicates SIE operation. When clear, the USB SIE is paused and does not respond to any USB activity. When set, the SIE operates normally. The RmteWkup bit initiates remote wake-up signalling to the host when the device has been suspended by the host. The RmteWkup bit should be set by the firmware only when the XPS USB 2.0 Device is in suspend mode. If the firmware sets the RmteWkup bit when XPS USB2 Device is not in suspend mode, the XPS USB2 Device does not issue a remote wake-up signalling to the host. The core generates a USB Resume interrupt after the successful completion of remote wake-up signalling with host and clears the RmteWkup bit.

Table 12: **Control Register** *(C_BASEADDR + 0x0104)*

Bit(s)	Name	Access	Reset Value	Description
0	MstRdy	R/W	0	0 = SIE is paused and does not respond to any USB activity 1 = SIE operates normally
1	RmteWkup(1)	R/W	0	0 = SIE does nothing 1 = SIE sends remote wake-up signalling to host
2-31	Reserved	NA	-	Reserved

Notes:

1. If the processor sets the RmteWkup bit when SIE in normal state, the XPS USB2 Device does not send the RmteWkup signalling to the host.

Interrupt Status Register (ISR)

The Interrupt Status Register (ISR) shown in Table 13 reports the status on the operation of the XPS USB2 Device core. Bits in this register get cleared as soon as they are read.

Table 13: **Interrupt Status Register** *(C_BASEADDR + 0x0108)*

Bit(s)	Name	Access	Reset Value	Description
0	Reserved	NA	-	Reserved
1	ULPI PHY Acc Comp	R	0	0 = ULPI PHY Access command completion not occurred 1 = ULPI PHY Access command completed
2	BitStuffErr(3)	R	0	0 = Bit Stuff error has not occurred 1 = Bit Stuff error has occurred
3	PIDErr(3)	R	0	0 = PID error has not occurred 1 = PID error has occurred
4	CRCErr(3)	R	0	0 = CRC error has not occurred 1 = CRC error has occurred
5	DMADne(1)	R	0	0 = DMA operation is not done 1 = DMA operation is done
6	DMAErr(1)	R	0	0 = DMA error has not occurred 1 = DMA error has occurred
7	USBRsm(2)	R	0	0 = Core has not received resume signalling from host 1 = Core received resume signalling from host
8	USBRst(2)	R	0	0 = Core has not received USB reset from host 1 = Core received USB reset from host
9	USBSpnd(2)	R	0	0 = Core not in suspend state 1 = Core in suspend state
10	USBDsc(2)	R	0	0 = Core connected to host 1 = Core is disconnected from host
11	FIFOBufRdy	R	0	0 = Endpoint 0 packet has not been received by core 1 = Endpoint 0 packet has been received by core
12	FIFOBufFree	R	0	0 = Endpoint 0 packet has not been transmitted by core 1 = Endpoint 0 packet has been transmitted by core
13	SETUPPkt	R	0	0 = Endpoint 0 Setup packet has not been received by core 1 = Endpoint 0 Setup packet has been received
14	SOFPkt	R	0	0 = Start of Frame packet has not been received by core 1 = Start of Frame packet has been received by core
15	HighSpd	R	0	0 = Core is running at Full-speed 1 = core is running at High-speed
16	Ep7ProcBuf1	R	0	0 = Endpoint 7, buffer 1 not processed 1 = Endpoint 7, buffer 1 processed
17	Ep6ProcBuf1	R	0	0 = Endpoint 6, buffer 1 not processed 1 = Endpoint 6, buffer 1 processed
18	Ep5ProcBuf1	R	0	0 = Endpoint 5, buffer 1 not processed 1 = Endpoint 5, buffer 1 processed
19	Ep4ProcBuf1	R	0	0 = Endpoint 4, buffer 1 not processed 1 = Endpoint 4, buffer 1 processed
20	Ep3ProcBuf1	R	0	0 = Endpoint 3, buffer 1 not processed 1 = Endpoint 3, buffer 1 processed

Table 13: **Interrupt Status Register** *(C_BASEADDR + 0x0108)* ***(Cont'd)***

Bit(s)	Name	Access	Reset Value	Description
21	Ep2ProcBuf1	R	0	0 = Endpoint 2, buffer 1 not processed 1 = Endpoint 2, buffer 1 processed
22	Ep1ProcBuf1	R	0	0 = Endpoint 1 buffer 1 not processed 1 = Endpoint 1, buffer 1 processed
23	Reserved	NA	-	Reserved
24	Ep7ProcBuf0	R	0	0 = Endpoint 7, buffer 0 not processed 1 = Endpoint 7, buffer 0 processed
25	Ep6ProcBuf0	R	0	0 = Endpoint 6, buffer 0 not processed 1 = Endpoint 6, buffer 0 processed
26	Ep5ProcBuf0	R	0	0 = Endpoint 5, buffer 0 not processed 1 = Endpoint 5, buffer 0 processed
27	Ep4ProcBuf0	R	0	0 = Endpoint 4, buffer 0 not processed 1 = Endpoint 4, buffer 0 processed
28	Ep3ProcBuf0	R	0	0 = Endpoint 3, buffer 0 not processed 1 = Endpoint 3, buffer 0 processed
29	Ep2ProcBuf0	R	0	0 = Endpoint 2, buffer 0 not processed 1 = Endpoint 2, buffer 0 processed
30	Ep1ProcBuf0	R	0	0 = Endpoint 1, buffer 0 not processed 1 = Endpoint 1, buffer 0 processed
31	Ep0ProcBuf0	R	0	0 = Endpoint 0, buffer 0 not processed 1 = Endpoint 0, buffer 0 processed

Notes:

1. This bit is undefined if the parameter C_INCLUDE_DMA = 0.
2. This bit indicates the current status of the XPS USB2 Device core.
3. This bit is undefined if the parameter C_INCLUDE_USBERR_LOGIC = 0.

The USB 2.0 Device has a single interrupt line (`USB_Irpt`) to indicate an interrupt. Interrupts are indicated by asserting the `USB_Irpt` signal (transition of the USB_Irpt from a logic 0 to a logic 1).

The Interrupt Enable Register (IER) allows specific bits of the Interrupt Status Register (ISR) to generate interrupts. The MstEnbl bit of this register allows all interrupts to be disabled simultaneously. The interrupt condition is cleared when the corresponding bit of the Interrupt Status Register (ISR) is cleared by writing a 1 to it. During power on, the `USB_Irpt` signal is driven low.

The following two conditions cause the `USB_Irpt` signal to be asserted:

- If a bit in the ISR is 1 and the corresponding bit in the IER is 1.
- Changing an IER bit from a 0 to 1 when the corresponding bit in the ISR is already 1.

Two conditions cause the `USB_Irpt` signal to be deasserted:

- Clearing a bit in the ISR, that is, by reading the ISR, provided that the corresponding bit in the IER is 1.
- Changing an IER bit from 1to 0, when the corresponding bit in the ISR is 1.

When both deassertion and assertion conditions occur simultaneously, the `USB_Irpt` signal is deasserted first, then is reasserted if the assertion condition remains TRUE.

Frame Number Register (FNR)

The Frame Number Register (FNR) shown in Table 14 is composed of two fields — Frame and Microframe. Frames are sent once every 1 ms and denote the beginning of a USB frame. All host scheduling starts at the start of Frame Time. The Microframe field is the result of additional Start of Frame tokens, sent once every 125 µs. When the USB is operated in the High Speed mode, this can generate a potentially high rate of interrupts. Therefore, the interrupt enable of Start of Frame should be used with caution.

Frame count values are of 11 bits and Microframe count values are of 3 bits.

Table 14: **Frame Number Register** *(C_BASEADDR + 0x010C)*

Bit(s)	Name	Access	Reset Value	Description
0-17	Reserved	NA	-	Reserved
18-28	FrmNum(10:0)	R	0	Frame numbers - 0 to 2047
29-31	uFrmNum(2:0)	R	0	Microframe numbers - 0 to 7

Interrupt Enable Register (IER)

The Interrupt Enable Register (IER) shown in Table 15 allows specific bits of the Interrupt Status Register (ISR) to generate interrupts. The MstEnbl bit of this register allows all interrupts to be disabled simultaneously. The interrupt condition is cleared when the corresponding bit of the Interrupt Status Register (ISR) is cleared. A specific bit of the IER can be cleared to prevent a long duration condition, such as USB Reset, from continuously generating an interrupt.

Table 15: **Interrupt Enable Register** *(C_BASEADDR + 0x0110)*

Bit(s)	Name	Access	Reset Value	Description
0	MstEnbl	R/W	0	0 = Disables the setting of all other interrupts 1 = Enables setting of all other interrupts
1	ULPIAccComp	R/W	0	0 = Disables ULPI access complete interrupt 1 = Enables ULPI access complete interrupt
2	BitStuffErrr(2)	R/W	0	0 = Disables Bit Stuff error interrupt 1 = Enable Bit Stuff error interrupt
3	PIDErr(2)	R/W	0	0 = Disables PID error interrupt 1 = Enables PID error interrupt
4	CRCErr(2)	R/W	0	0 = Disables CRC error interrupt 1 = Enables CRC error interrupt
5	DMADne(1)	R/W	0	0 = Disables DMA Done interrupt 1 = Enables DMA Done interrupt
6	DMAErr(1)	R/W	0	0 = Disables DMA Error interrupt 1 = Enables DMA Error interrupt
7	USBRsm	R/W	0	0 = Disables USB Resume interrupt 1 = Enables USB Resume interrupt
8	USBRst	R/W	0	0 = Disables USB Reset interrupt 1 = Enables USB Reset interrupt
9	USBSpnd	R/W	0	0 = Disables USB Suspend interrupt 1 = Enables USB Suspend interrupt
10	USBDsc	R/W	0	0 = Disables USB Disconnect interrupt 1 = Enables USB Disconnect interrupt
11	FIFOBufRdy	R/W	0	0 = Disables FIFIO Buf Rdy interrupt 1 = Enables FIFIO Buf Rdy interrupt

Table 15: **Interrupt Enable Register** *(C_BASEADDR + 0x0110)* ***(Cont'd)***

Bit(s)	Name	Access	Reset Value	Description
12	FIFOBufFree	R/W	0	0 = Disables FIFO Buf Free interrupt 1 = Enables FIFO Buf Free interrupt
13	SETUPPkt	R/W	0	0 = Disables Setup Packet received interrupt 1 = Enables Setup Packet received interrupt
14	SOFPkt	R/W	0	0 = Disables Start of Frame received interrupt 1 = Enables Start of Frame received interrupt
15	HighSpd	R/W	0	0 = Disables core operates in High Speed interrupt 1 = Enables core operates in High Speed interrupt
16	Ep7ProcBuf1	R/W	0	0 = Disables endpoint 7, buffer 1 processed interrupt 1 = Enables endpoint 7, buffer 1 processed interrupt
17	Ep6ProcBuf1	R/W	0	0 = Disables endpoint 6, buffer 1 processed interrupt 1 = Enables endpoint 6, buffer 1 processed interrupt
18	Ep5ProcBuf1	R/W	0	0 = Disables endpoint 5, buffer 1 processed interrupt 1 = Enables endpoint 5, buffer 1 processed interrupt
19	Ep4ProcBuf1	R/W	0	0 = Disables endpoint 4, buffer 1 processed interrupt 1 = Enables endpoint 4, buffer 1 processed interrupt
20	Ep3ProcBuf1	R/W	0	0 = Disables endpoint 3, buffer 1 processed interrupt 1 = Enables endpoint 3, buffer 1 processed interrupt
21	Ep2ProcBuf1	R/W	0	0 = Disables endpoint 2, buffer 1 processed interrupt 1 = Enables endpoint 2, buffer 1 processed interrupt
22	Ep1ProcBuf1	R/W	0	0 = Disables endpoint 1, buffer 1 processed interrupt 1 = Enables endpoint 1, buffer 1 processed interrupt
23	Reserved	NA	-	Reserved
24	Ep7ProcBuf0	R/W	0	0 = Disables endpoint 7, buffer 0 processed interrupt 1 = Enables endpoint 7, buffer 0 processed interrupt
25	Ep6ProcBuf0	R/W	0	0 = Disables endpoint 6, buffer 0 processed interrupt 1 = Enables endpoint 6, buffer 0 processed interrupt
26	Ep5ProcBuf0	R/W	0	0 = Disables endpoint 5, buffer 0 processed interrupt 1 = Enables endpoint 5, buffer 0 processed interrupt
27	Ep4ProcBuf0	R/W	0	0 = Disables endpoint 4, buffer 0 processed interrupt 1 = Enables endpoint 4, buffer 0 processed interrupt
28	Ep3ProcBuf0	R/W	0	0 = Disables endpoint 3, buffer 0 processed interrupt 1 = Enables endpoint 3, buffer 0 processed interrupt
29	Ep2ProcBuf0	R/W	0	0 = Disables endpoint 2, buffer 0 processed interrupt 1 = Enables endpoint 2, buffer 0 processed interrupt
30	Ep1ProcBuf0	R/W	0	0 = Disables endpoint 1, buffer 0 processed interrupt 1 = Enables endpoint 1, buffer 0 processed interrupt
31	Ep0ProcBuf0	R/W	0	0 = Disables endpoint 0, buffer 0 processed interrupt 1 = Enables endpoint 0, buffer 0 processed interrupt

Notes:

1. This bit is undefined if the parameter C_INCLUDE_DMA = 0.
2. This bit is undefined if the parameter C_INCLUDE_USBERR_LOGIC = 0.

Buffer Ready Register (BRR)

The Buffer Ready Register (BRR) has a buffer-ready bit corresponding to each buffer of each endpoint, as shown in Table 16. The firmware sets each bit when that buffer is ready for either USB IN or USB OUT traffic. Until that bit is set, an attempted IN or OUT to/from the buffer results in a NAK to the host. The ability of the buffer to handle an IN or OUT is determined by the EP_OUT_IN bit in the Configuration and Status register of the corresponding endpoint. Per the USB 2.0 Specification, endpoint 0 has only one buffer that handles IN or OUT.

Table 16: **Buffer Ready Register** *(C_BASEADDR + 0x0114)*

Bit(s)	Name	Access	Reset Value	Description
0-15	Reserved	NA	-	Reserved
16	Ep7Buf1Rdy	R/W	0	0 = Endpoint 7, buffer 1 is not ready for SIE transfer 1 = Endpoint 7, buffer 1 is ready for SIE transfer
17	Ep6Buf1Rdy	R/W	0	0 = Endpoint 7, buffer 1 is not ready for SIE transfer 1 = Endpoint 7, buffer 1 is ready for SIE transfer
18	Ep5Buf1Rdy	R/W	0	0 = Endpoint 7, buffer 1 is not ready for SIE transfer 1 = Endpoint 7, buffer 1 is ready for SIE transfer
19	Ep4Buf1Rdy	R/W	0	0 = Endpoint 7, buffer 1 is not ready for SIE transfer 1 = Endpoint 7, buffer 1 is ready for SIE transfer
20	Ep3Buf1Rdy	R/W	0	0 = Endpoint 7, buffer 1 is not ready for SIE transfer 1 = Endpoint 7, buffer 1 is ready for SIE transfer
21	Ep2Buf1Rdy	R/W	0	0 = Endpoint 7, buffer 1 is not ready for SIE transfer 1 = Endpoint 7, buffer 1 is ready for SIE transfer
22	Ep1Buf1Rdy	R/W	0	0 = Endpoint 7, buffer 1 is not ready for SIE transfer 1 = Endpoint 7, buffer 1 is ready for SIE transfer
23	Reserved	NA	-	Reserved
24	Ep7Buf0Rdy	R/W	0	0 = Endpoint 7, buffer 0 is not ready for SIE transfer 1 = Endpoint 7, buffer 0 is ready for SIE transfer
25	Ep6Buf0Rdy	R/W	0	0 = Endpoint 6, buffer 0 is not ready for SIE transfer 1 = Endpoint 6, buffer 0 is ready for SIE transfer
26	Ep5Buf0Rdy	R/W	0	0 = Endpoint 5, buffer 0 is not ready for SIE transfer 1 = Endpoint 5, buffer 0 is ready for SIE transfer
27	Ep4Buf0Rdy	R/W	0	0 = Endpoint 4, buffer 0 is not ready for SIE transfer 1 = Endpoint 4, buffer 0 is ready for SIE transfer
28	Ep3Buf0Rdy	R/W	0	0 = Endpoint 3, buffer 0 is not ready for SIE transfer 1 = Endpoint 3, buffer 0 is ready for SIE transfer
29	Ep2Buf0Rdy	R/W	0	0 = Endpoint 2, buffer 0 is not ready for SIE transfer 1 = Endpoint 2, buffer 0 is ready for SIE transfer
30	Ep1Buf0Rdy	R/W	0	0 = Endpoint 1, buffer 0 is not ready for SIE transfer 1 = Endpoint 1, buffer 0 is ready for SIE transfer
31	Ep0 Buffer Ready	R/W	0	0 = Endpoint 0, buffer 0 is not ready for SIE transfer 1 = Endpoint 0, buffer 0 is ready for SIE transfer

Test Mode Register (TMR)

The Test Mode Register (TMR) shown in Table 17 defines the different test modes in which the XPS USB 2.0 Device operates. The USB Implementers Forum, the organization that controls USB logo certification, requires all USB 2.0 devices that operate at High Speed to support these test modes.

Table 17: **Test Mode Register** *(C_BASEADDR + 0x0118)*

Bit(s)	Name	Access	Reset Value	Description
0-28	Reserved	NA	-	Reserved
29-31	Test Mode(2:0)	R/W	0	Value defines the test mode 0 - Normal Mode 1 - Test Mode J 2 - Test Mode K 3 - Test Mode NAK 4 - Test Mode Packet

The XPS USB 2.0 Device provides test mode support to facilitate compliance testing.

Four test modes are supported:

- **Test Mode J**: The core transmits a continuous chirp J and remains in this state until the time when it is reset.
- **Test Mode K**: The core transmits a continuous chirp K and remains in this state until the time when it is reset.
- **Test Mode NAK**: The core searches for any IN token with a valid crc5. If crc5 is valid, the core sends a NAK, otherwise it waits for the next valid IN token. The core remains in this state until it is reset.
- **Test Mode Packet:** As specified by the USB 2.0 Specification, the core transmits a test packet that is composed of a predefined sequence of bytes and is used for analog testing of the USB in the high speed mode. The packet data is loaded into a predefined sequence of locations in the DPRAM. This routine repeats continuously until the core is reset.

Error Count Register (ECR)

The Error Count Register (ECR) (ECR) is shown in Table 18. This register contains three counters each of 8-bit width. They are BitStuffErrCnt, PIDErrCnt, and CRCErrCnt. When USB Reset or read to this register is requested, all these counters are cleared and assigned to reset values. When PHY detects seven consecutive ones on the bus *(bit stuff error condition)*, SIE increases BitStuffErrCnt by one and BitStuffErr bit of Interrupt Status Register (ISR) is set. Whenever four PID check bits are not complement to their respective packet identifier bits while receiving the packet, SIE increases PIDErrCnt by one and PIDErr bit of Interrupt Status Register (ISR) is set. Whenever the received CRC does not match with the CRC calculated on the received packet (that is, for CRC5 while receiving token and for CRC16 while receiving data), SIE increases CRCErrCnt by one and CRCErr bit of Interrupt Status Register (ISR) is set.

Table 18: **Error Count Register** *(C_BASEADDR + 0x011C)* (1)(2)(3)

Bit(s)	Name	Access	Reset Value	Description
0-7	BitStuffErrCnt	R	0x0	Bit Stuff Error Counter
8-15	PIDErrCnt	R	0x0	PID Error Counter
16-23	CRCErrCnt	R	0x0	CRC Error Counter
24-31	Reserved	NA	-	Reserved

Notes:

1. This register is read-only.
2. When any of the counter reaches 255, it rolls back and start counting from 0.
3. If the user reads this register when the parameter C_INCLUDE_USBERR_LOGIC = 0, the core returns 0x0 as this register is not included in the design.

ULPI PHY Access Register (UPAR)

The ULPI PHY Access Register (UPAR) shown in Table 19 defines the type of access (Read or Write) on ULPI PHY registers.

Read: When type of access is configured as read, the user application:

- Writes the address of the PHY Register into PHYReg Addr and sets the WriteNotRead bit to 0.
- Core asserts the busy bit of the ULPI PHY Access Register (UPAR) until the successful read is performed on the respective PHY Register.
- SIE updates the PHY register read data into PHYRdWrData after the successful read onto the PHY register.
- Core clears the busy bit of ULPI PHY Access Register (UPAR) and sets the ULPIAccComp bit of the Interrupt Status Register (ISR).

Write: When type of access is configured as write, the user application:

- Writes the address of the PHY Register into PHYRegAddr and PHYRdWrData, and sets the WriteNotRead bit to 1.
- Core asserts the busy bit of the ULPI PHY Access Register (UPAR) until the successful write is performed on the respective PHY Register.
- Core clears the busy bit of ULPI PHY Access Register (UPAR) after the successful write onto the PHY register and sets ULPIAccComp bit of the Interrupt Status Register (ISR).

Table 19: **ULPI PHY Access Register** *(C_BASEADDR + 0x0120)* (1)(2)(3)

Bit(s)	Name	Access	Reset Value	Description
0	busy	R	0	0 = SIE is not busy 1 = SIE is busy
1-15	Reserved	NA	-	Reserved
16-23	PHYRdWrData	R/W	0x0	PHY Register Read or write Data
24	Reserved	NA	-	Reserved
25	WriteNotRead	R/W	0	0 = SIE Reads the PHY Register 1 = SIE Writes the PHY Register
31-26	PHYRegAddr	R/W	0x0	PHY Register Address

Notes:

1. SIE performs the register access onto the ULPI PHY register when ULPI Bus is IDLE (for example, when there is no data exchange between ULPI PHY and SIE). If the ULPI Bus is busy with ongoing data transfer, SIE waits for ULPI Bus IDLE to perform the ULPI PHY register access.
2. User application should not perform any writes onto the ULPI PHY access register when busy bit is set to 1. If it does so, these writes on the ULPI PHY Access Register are not successful.
3. ULPI PHY Register Access is not supported without assertion of the MstRdy bit of the Control Register (CR).

DMA Software Reset Register (DSRR)

The DMA Software Reset Register (DSRR) shown in Table 20 defines reset to the DMA modules by the XPS_USB2_Device core when C_INCLUDE_DMA set to 1.

Table 20: **DMA Software Reset Register** *(C_BASEADDR + 0x0200)*

Bit(s)	Name	Access	Reset Value	Description
0-31	RST	W	N/A	A write of 0x0000000A causes the reset to the DMA modules. A write of any other value has an undefined effect and returns a bus error.

DMA Control Register (DMACR)

The DMA Control Register (DMACR) shown in Table 21 defines the direction of the transfer as either Read or Write to DPRAM2. Endpoint Buffer Select bit of the DMA Control Register (DMACR) enables or disables the update of Buffer Ready status to SIE either by the Hardware or the Firmware. If Endpoint Buffer Select is set to 1, the DMA Controller updates Buffer Ready status to SIE based on bits [16:31] of the DMA Control Register (DMACR) at the end of a successful DMA transfer only.

Table 21: **DMA Control Register** *(C_BASEADDR + 0x0204)*

Bit(s)	Name	Access	Reset Value	Description
0	Direction(1) (2)	R/W	0	0 = Write data into DPRAM2 1 = Read data from DPRAM2
1	EpBufSel(3)	R/W	0	0 = Buffer Ready set by Firmware 1 = Buffer Ready set by DMA Controller
2-15	Reserved	NA	-	Reserved
16	Ep7Buf1Rdy	R/W	0	0 = Endpoint 7, buffer 1 is not ready for SIE transfer until the DMA operation completes 1 = Endpoint 7, buffer 1 is ready for SIE transfer when the DMA operation completes
17	Ep6Buf1Rdy	R/W	0	0 = Endpoint 6, buffer 1 is not ready for SIE transfer until the DMA operation completes 1 = Endpoint 6, buffer 1 is ready for SIE transfer when the DMA operation completes
18	Ep5Buf1Rdy	R/W	0	0 = Endpoint 5, buffer 1 is not ready for SIE transfer until the DMA operation completes 1 = Endpoint 5, buffer 1 is ready for SIE transfer when the DMA operation completes
19	Ep4Buf1Rdy	R/W	0	0 = Endpoint 4, buffer 1 is not ready for SIE transfer until the DMA operation completes 1 = Endpoint 4, buffer 1 is ready for SIE transfer when the DMA operation completes
20	Ep3Buf1Rdy	R/W	0	0 = Endpoint 3, buffer 1 is not ready for SIE transfer until the DMA operation completes 1 = Endpoint 3, buffer 1 is ready for SIE transfer when the DMA operation completes
21	Ep2Buf1Rdy	R/W	0	0 = Endpoint 2, buffer 1 is not ready for SIE transfer until the DMA operation completes 1 = Endpoint 2, buffer 1 is ready for SIE transfer when the DMA operation completes
22	Ep1Buf1Rdy	R/W	0	0 = Endpoint 1, buffer 1 is not ready for SIE transfer until the DMA operation completes 1 = Endpoint 1, buffer 1 is ready for SIE transfer when the DMA operation completes

Table 21: **DMA Control Register** *(C_BASEADDR + 0x0204)* ***(Cont'd)***

Bit(s)	Name	Access	Reset Value	Description
23	Reserved	NA	-	Reserved
24	Ep7Buf0Rdy	R/W	0	0 = Endpoint 7, buffer 0 is not ready for SIE transfer until the DMA operation completes 1 = Endpoint 7, buffer 0 is ready for SIE transfer when the DMA operation completes
25	Ep6Buf0Rdy	R/W	0	0 = Endpoint 6, buffer 0 is not ready for SIE transfer until the DMA operation completes 1 = Endpoint 6, buffer 0 is ready for SIE transfer when the DMA operation completes
26	Ep5Buf0Rdy	R/W	0	0 = Endpoint 5, buffer 0 is not ready for SIE transfer until the DMA operation completes 1 = Endpoint 5, buffer 0 is ready for SIE transfer when the DMA operation completes
27	Ep4Buf0Rdy	R/W	0	0 = Endpoint 4, buffer 0 is not ready for SIE transfer until the DMA operation completes 1 = Endpoint 4, buffer 0 is ready for SIE transfer when the DMA operation completes
28	Ep3Buf0Rdy	R/W	0	0 = Endpoint 3, buffer 0 is not ready for SIE transfer until the DMA operation completes 1 = Endpoint 3, buffer 0 is ready for SIE transfer when the DMA operation completes
29	Ep2Buf0Rdy	R/W	0	0 = Endpoint 2, buffer 0 is not ready for SIE transfer until the DMA operation completes 1 = Endpoint 2, buffer 0 is ready for SIE transfer when the DMA operation completes
30	Ep1Buf0Rdy	R/W	0	0 = Endpoint 1, buffer 0 is not ready for SIE transfer until the DMA operation completes 1 = Endpoint 1, buffer 0 is ready for SIE transfer when the DMA operation completes
31	Ep0Buf0Rdy	R/W	0	0 = Endpoint 0, buffer 0 is not ready for SIE transfer until the DMA operation completes 1 = Endpoint 0, buffer 0 is ready for SIE transfer when the DMA operation completes

Notes:

1. Write data into DPRAM means that the DMA controller writes data to DPRAM by reading the data from the external memory.
2. Read data from DPRAM means that the DMA controller reads data from DPRAM and writes the data to the external memory.
3. Only in DMA mode (C_INCLUDE_DMA = 1), can the Buffer Ready status can be updated, by either the Firmware or the Hardware, based on the Endpoint Buffer Select bit of the DMA Control register.

DMA Source Address Register (DSAR)

The DMA Source Address Register shown in Table 22 defines the source address for the current DMA transfer. When data is moved from the source address, this register updates to track the current source address. If the Direction bit of the DMA Control Register (DMACR) is set to 0, the DMA Source Address of the current DMA transfer should be the external memory address. If the Direction bit of the DMA Control Register (DMACR) is set to 1, the DMA Source Address of the current DMA transfer should be the DPRAM2 address.

Table 22: **DMA Source Address Register** *(C_BASEADDR + 0x0208)*

Bit(s)	Name	Access	Reset Value	Description
0-31	Srcaddr	R/W	0	Source Address for the current DMA transfer

DMA Destination Address Register (DDAR)

The DMA Destination Address Register shown in Table 23 defines the destination address for the current DMA transfer. When data is moved to the destination address, this register updates to track the current destination address. If the Direction bit of the DMA Control Register (DMACR) is set to 0, the DMA Destination Address of the current DMA transfer should be the DPRAM2 address. If the Direction bit of the DMA Control Register (DMACR) is set to 1, the DMA Destination Address of the current DMA transfer should be the external memory address.

Table 23: **DMA Destination Address Register** *(C_BASEADDR + 0x020C)*

Bit(s)	Name	Access	Reset Value	Description
0-31	DstAddr	R/W	0	Destination Address for the current DMA transfer

DMA Length Register (DMALR)

The DMA Length Register (DMALR) shown in Table 24 defines the total number of bytes to be transferred from the source address to destination address. This register is written only after configuring the DMA Control Register (DMACR), the DMA Source Address register, and the DMA Destination Address register, with their values and any other setup is complete. As bytes are successfully written to the destination, the DMA Length Register (DMALR) decreases to reflect the number of bytes remaining to be transferred. The DMA Length Register (DMALR) is zero after a successful DMA operation.

Table 24: **DMA Length Register** *(C_BASEADDR + 0x0210)*

Bit(s)	Name	Access	Reset Value	Description
0-31	DmaLen	R/W	0	Number of bytes to be transferred from source to destination

DMA Status Register (DMASR)

The DMA Status Register (DMASR) shown in Table 25 defines the status of the DMA controller as DMA Busy or DMA Error. When DMA operation is in progress, the DMA Busy is set to 1 until the DMA operation has finished. If the DMA operation encounters any error condition, the DMA Error is set to 1.

Table 25: **DMA Status Register** *(C_BASEADDR + 0x0214)*

Bit(s)	Name	Access	Reset Value	Description
0	DmaBusy	R	0	0 = DMA operation is not initiated 1 = DMA operation is in progress
1	DmaErr	R	0	0 = DMA Error has not occurred 1 = DMA Error has occurred
2-31	Reserved	NA	-	Reserved

Programming the Core

This section describes how to program the XPS USB 2.0 Device for various operations. For applications that use the Xilinx driver, users are expected to change the Vendor ID before using the XPS USB 2.0 Device.

Initialization Sequence

- At first, the XPS_USB2_Device core issues ULPI Reset to PHY during power-on reset. SIE extends this asynchronous ULPI Reset for around 2 to 4 µsec depending on the system clock frequency. The polarity of the ULPI Reset as active-High/Low is user-configurable with the parameter C_PHY_RESET_TYPE.
- Firmware enables the USB Disconnect, USB Suspend, and USB Reset bits of the Interrupt Enable Register (IER).
- Firmware enables SIE by setting the MstRdy bit of the Control Register (CR).
- When the MstRdy bit of the Control Register (CR) is set to 1 by Firmware, SIE issues a soft reset to PHY by writing 0x20 to PHY's Function Control Register to set the reset bit. PHY keeps SIE in busy by asserting ULPI DIR until it resets the internal state machine to IDLE state and updates the current state of the USB bus after the successful completion of the soft reset.
- SIE configures PHY to device mode by writing Pays OTG Register to 0x00, Function Control register to 0x41 (Sets TermSelect to 0b, XcvrSelect to 01b (HS) and, Opmode to 00b (Normal Operation)).
- If XPS_USB2_Device core was not connected to Host, the PHY updates Session End (VBUS not present) RXCMD to SIE. At that point, SIE waits for connection with Host.
- When XPS_USB2_Device core connects to Host, the PHY updates Vbus Valid (VBUS present) RXCMD to SIE.
- SIE pulls up the D+ line of the PHY by writing 0x45 (Sets TermSelect to 1b, XcvrSelect to 01b (FS), by writing Opmode to 00b (Normal Operation)) to the Function Control register of the PHY, and moves to Full-speed mode. SIE updates the USB Disconnect bit of the Interrupt Status Register (ISR) to 1.
- When PHY pulls up the D+ line, the host detects the XPS_USB2_Device connection and starts issuing a USB Reset to bring the device into the default unconfigured state.
- SIE detects USB Reset from the Host and updates the USB Reset bit of the Interrupt Status Register (ISR) to 1 until USB Reset signalling is finished between the Host and the XPS_USB2_Device.
- After the successful completion of the USB Reset by the Host, SIE updates the HSEn bit (0 - Full-speed, 1 - High-speed) and starts receiving Soft every 1 ms in Full-speed and every 125 µs in High-speed.
- XPS_USB2_Device responds to transfers from the Host based on the endpoint configuration and status registers programmed.

Operating in Interrupt Mode

If desired, configure the device to operate in the interrupt mode after enabling the desired interrupts in the Interrupt Enable Register (IER).

- Interrupts for USB Reset, USB Suspend, USB Resume (Remote-wakeup), and USB Disconnect can be enabled by writing to those specific bits of the IER.
- To generate an interrupt when the core is operating in High Speed, the High Speed bit of the IER should be set.
- To generate an interrupt when a start-up packet or SOF packet is received, those bits of the IER must be set.
- For endpoint 0, set the Fifo Buffer Ready and Fifo Buffer Free bits of the IER to generate interrupts when packets are received or transmitted respectively.
- For all other endpoints, set the respective Buffer Complete bit of the IER to generate interrupts when that endpoint buffer is complete.

Setting the USB 2.0 Device Address

Set the device to the unenumerated state by writing an address of 0 to the USB Address Register.

Configuring an Endpoint

Program the individual Endpoint Configuration and Status Registers to configure the respective endpoints:

- A specific endpoint can be enabled by writing a 1 to the EpValid bit of the endpoint's configuration and status register.
- The direction of an endpoint can be set to IN by writing a 1, or to OUT by writing 0 to the EP_OUT_IN bit of the register.
- The endpoint can be configured as an isochronous endpoint by writing a 1, or as a bulk endpoint by writing 0 to the EpIso bit of the register.
- The packet size for the endpoint can be set by writing to the EpPktSze bits of the register.
- The base offset of the endpoint buffers in the DPRAM can be set by writing to the EpBase bits of the register.

Handling a Control Packet

1. Wait for the SETUPPkt interrupt to detect the reception of the setup packet.
2. Read the setup packet from the buffer location of Endpoint 0 in the DPRAM, which causes the SETUPPkt bit of the Interrupt Status Register (ISR) to be cleared.
3. Process the received Chapter 9 command (as detailed in the USB 2.0 Specification) and prepare a buffer for a response that must be sent for the subsequent IN packet.

Handling Bulk/Isochronous IN Transactions

For a Bulk or Isochronous IN transaction in No DMA Mode, perform the following steps:

1. Write the IN packet into the selected endpoint buffer location in the DPRAM.
2. Write the packet count into the specific endpoint buffer's Buffer Count Register.
3. Set the Buffer Ready bit for the selected endpoint buffer in the Buffer Ready Register (BRR).
4. Wait for the Buffer Ready interrupt of the selected endpoint buffer in the Buffer Ready Register (BRR) (this bit must be cleared) to ensure that the transmitted data has been received by the host and the buffer is available for the next write.

For a Bulk or Isochronous IN transaction in DMA-Mode, perform the following steps:

1. Write the packet count into the specific endpoint buffer's Buffer Count Register.
2. Configure DMA by writing into the Direction bit of DMA Control Register (DMACR) to 0, DMA Destination Address (same as EpBase of the endpoint configuration register of the respective endpoint), DMA Destination Address and DMA Length registers.
3. DMA generates DMA Done interrupt after successfully writing the configured length of data into the DPRAM.
4. Set the Buffer Ready bit for the selected endpoint buffer in the Buffer Ready Register (BRR).
5. Wait for the Buffer Complete interrupt of the selected endpoint buffer in the Interrupt Status Register (ISR) to ensure that the transmitted data has been received by the host and the buffer is available for the next write.

Handling Bulk/Isochronous OUT Transactions

When the host sends an OUT packet to an endpoint buffer on the device while the buffer is in use, the device core sends a NAK to the host. After the buffer is free, the packet is written into the buffer and an ACK is automatically sent to the host.

On reception of the OUT packet in No DMA Mode, perform the following steps:

1. Poll the Buffer Complete bit of the selected endpoint buffer in the Interrupt Status Register (ISR) to detect the reception of a packet.
2. Check that the received packet count matches with the specified packet count in the Buffer Count Register.
3. Read the OUT packet from the selected endpoint buffer location in the DPRAM.
4. Set the Buffer Ready bits of the selected endpoint buffer to prepare it for the next transaction.

On reception of the OUT packet in DMA-Mode, perform the following steps:

1. Poll the Buffer Complete bit of the selected endpoint buffer in the Interrupt Status Register (ISR) to detect the reception of a packet.
2. Check the received packet count matches with the specified packet count in the Buffer Count Register.
3. Read the OUT packet by configure DMA by writing into Direction bit of DMA Control Register (DMACR) to 1, DMA Source Address (same as EpBase of the endpoint configuration register of the respective endpoint), DMA Destination Address, and DMA Length registers.
4. DMA generates DMA Done interrupt after successfully reading the configured length of data into the DPRAM.
5. Set the Buffer Ready bits of the selected endpoint buffer to prepare it for the next transaction.

USB Reset Signalling

When the host wants to start communicating with a device, it starts by applying a 'Reset' condition which sets the device to its default unconfigured state. This 'Reset' should not be confused with a microcontroller power-on type reset. It is a USB protocol reset to ensure that the device USB signalling starts from a known state. A high-speed capable XPS USB 2.0 Device can be reset while the device is in the Powered, Default, Address, Configured or Suspended states. XPS USB 2.0 Device can be successfully reset by any host (even USB1.x host).

The Host performs USB Reset Signalling High-speed capable XPS USB 2.0 Device as follows:

1. Host drives SE0, the start of SE0 is referred to as time T0.
2. The SIE detects assertion of SE0.
    a. If SIE detects SE0 from USB Suspend state, then SIE begins the high-speed handshake detection after the detection of SE0 for no less than 2.5 µs.
    b. If SIE detects SE0 from a non-suspended full-speed mode, the SIE begins a high-speed handshake detection after the detection of SE0 for no less than 2.5 µs and no more than 3.0 ms.
    c. If SIE detects SE0 from a non-suspended high-speed mode, the SIE waits for 3.0 ms before reverting to Full-speed mode. After reverting to Full-speed, SIE checks the line state update from PHY for SE0 (to check whether the Host issued a suspend or USB Reset), no less than 1 ms from the point SIE reverted to Full-speed. If SIE detects SE0 by the RXCMD updating from PHY, SIE begins the high-speed handshake detection.
3. Because the XPS USB 2.0 Device is a high-speed capable device, SIE follows the High-speed Handshake Detection protocol.

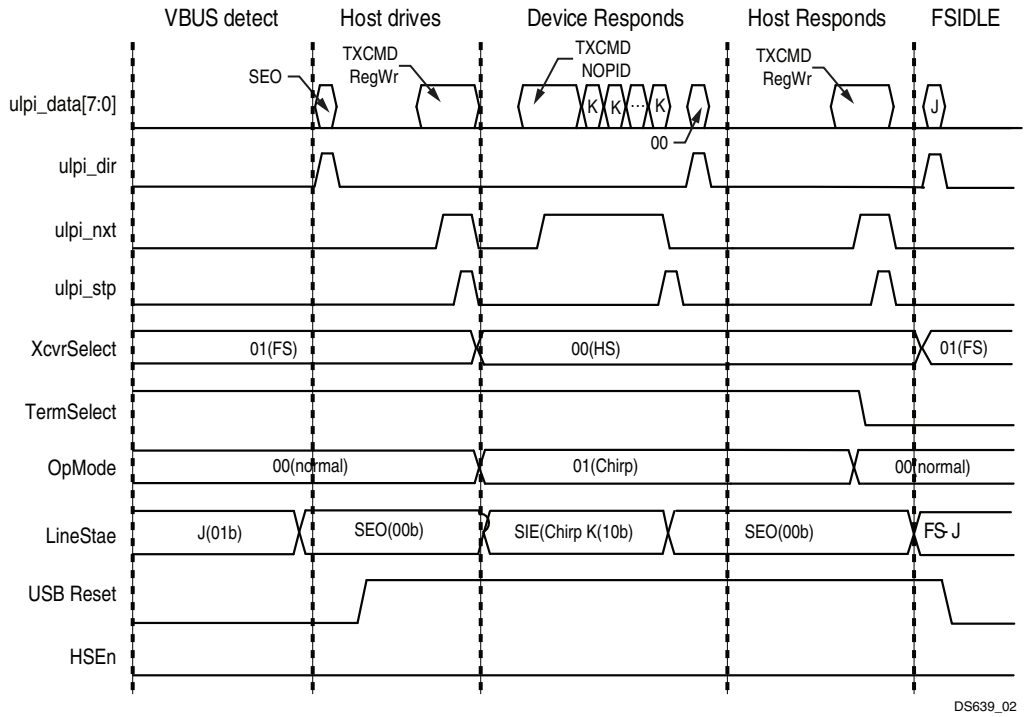

Figure 2: USB Reset Signalling with USB1.x Host - Full Speed

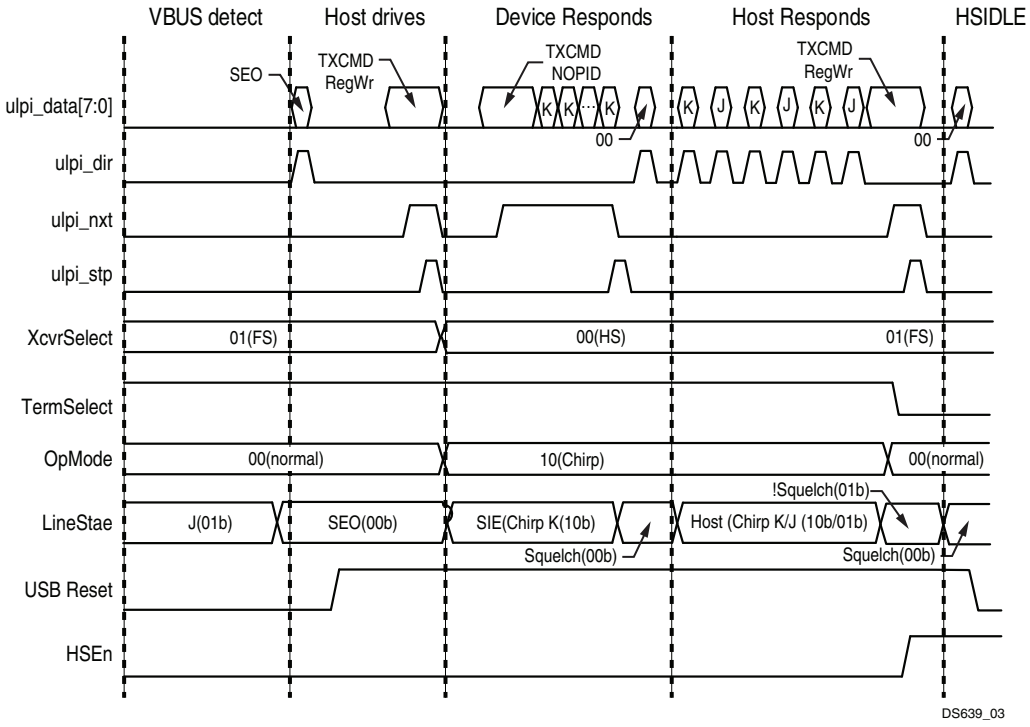

Figure 3: **USB Reset Signalling with USB 2.0 Host - High Speed Handshaking**

High-Speed Handshake Detection Protocol

1. SIE writes 0x54 (sets XcvrSelect to 00b, TermSelect to 1b and Opmode to 10b (Chirping)) to the Function Control register of the PHY.
2. Immediately after the preceding write, SIE puts TXCMD as 0x40 (NOPID) and starts transmitting a chirpK for 2 ms.
3. After 2 ms of ChirpK transmission, PHY updates RXCMD with line state as SE0.
4. SIE detects SE0 and waits for 100 µs to look for high-speed Host chirp, for example, an alternating sequence of ChirpKs and ChirpJs.
5. If SIE does not observe any high-speed Host chirp, the SIE configures PHY to Full-speed mode by writing 0x45 (sets XcvrSelect to 01b, TermSelect to, 1b, and Opmode to 00b (Normal Operation)) to the Function Control register of PHY. SIE is now in Full-speed mode and waits for Full-speed USB traffic from Host.
6. If SIE observes the high-speed Host chirp, the SIE checks for a minimum of chirp K-J-K-J-K-J.
7. SIE also checks each individual chirpK and chirpJ for at least 2.5 µs.
8. After detecting the minimum sequence, SIE configures PHY to high-speed mode by writing 0x40 (Sets TermSelect to 0b, XcvrSelect to 00b (HS) and Opmode to 00b (Normal Operation)) to the Function Control register of PHY and updates the HSEn bit of the Interrupt Status Register (ISR) to 1.
9. SIE is now in High-speed mode and sees line state as !squelch (linestate != 00) on the RXCMD update.
10. If SIE observes squelch as line state update from PHY (for example, RXCMD[1:0]), the SIE treats that Host has completed the chirp and stops updating the status as USB Reset in the Interrupt Status Register (ISR).
11. Now SIE waits for High-speed USB traffic from Host.

Suspend Signalling

When the Host does not want to continue any further communication with the connected device, it keeps the XPS USB 2.0 Device in the suspend mode by ceasing all of its transmissions (including SOFs). The XPS USB 2.0 Device recognizes the suspend condition in high-speed mode as follows:

1. Initially, Host and the XPS USB 2.0 Device are either in Full-speed or High-speed mode.
2. When SIE sees no bus activity for 3 ms, it enters Full-speed mode by writing 0x45 (Sets TermSelect to 1b, XcvrSelect to 01b (FS), and Opmode to 00b (Normal Operation)) to the Function Control register.
3. After 1 ms, SIE samples the line state from the RXCMD update of PHY.
4. If the line state is Full-speed J, SIE enters into the Suspend state after 7 ms and updates the status as suspended by asserting the USB Suspend bit of the Interrupt Status Register (ISR).
5. SIE is now SIE in the Suspended state.
6. Resume or USB Reset can bring SIE out from suspend state.

Resume Signalling

When XPS USB 2.0 Device in the suspend state, the host wakes up the device with a resume signalling. The XPS USB 2.0 Device detects Resume signalling as listed in the following steps:

1. When both Host and XPS USB 2.0 Device are in Low Power Mode
2. If SIE observes Full-speed K as line state (that is, RXCMD[1:0]) update from PHY, then SIE treats it as Resume signalling from PHY and update the status as Resume by setting USB Resume bit of the Interrupt Status Register (ISR).
3. SIE waits to detect the End of Resume i.e SE0 as line state update from PHY
4. If SIE detects SE0 line state update from PHY:
   a. SIE moves to Full-speed if SIE was in Full-speed prior to entering into suspend state.

b. SIE moves to High-speed by writing 0x40 (Sets TermSelect to 0b, XcvrSelect to 00b (HS), and Opmode to 00b (Normal Operation)) to the Function Control register of the PHY if SIE was in High-speed prior to entering into the suspend state.

5. SIE updates the HSEn bit of Interrupt Status Register (ISR) accordingly.

The Suspend signalling followed by Resuming signalling from Host is shown in Figure 4 (Full-speed) and Figure 5 (High-speed).

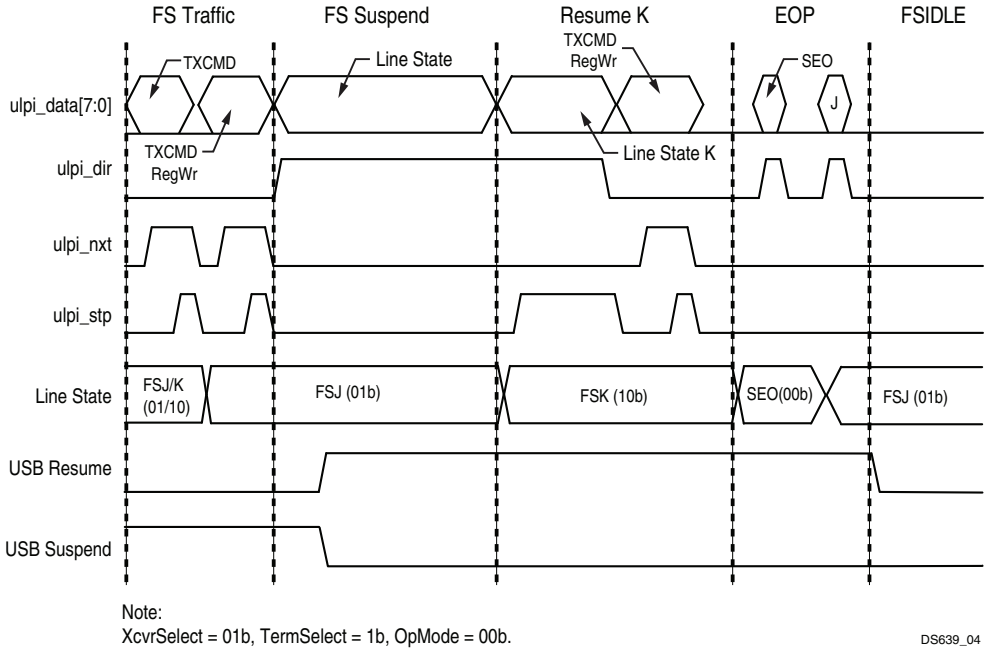

Figure 4: **Suspend Signalling Followed with Resume Signalling from Host - Full Speed**

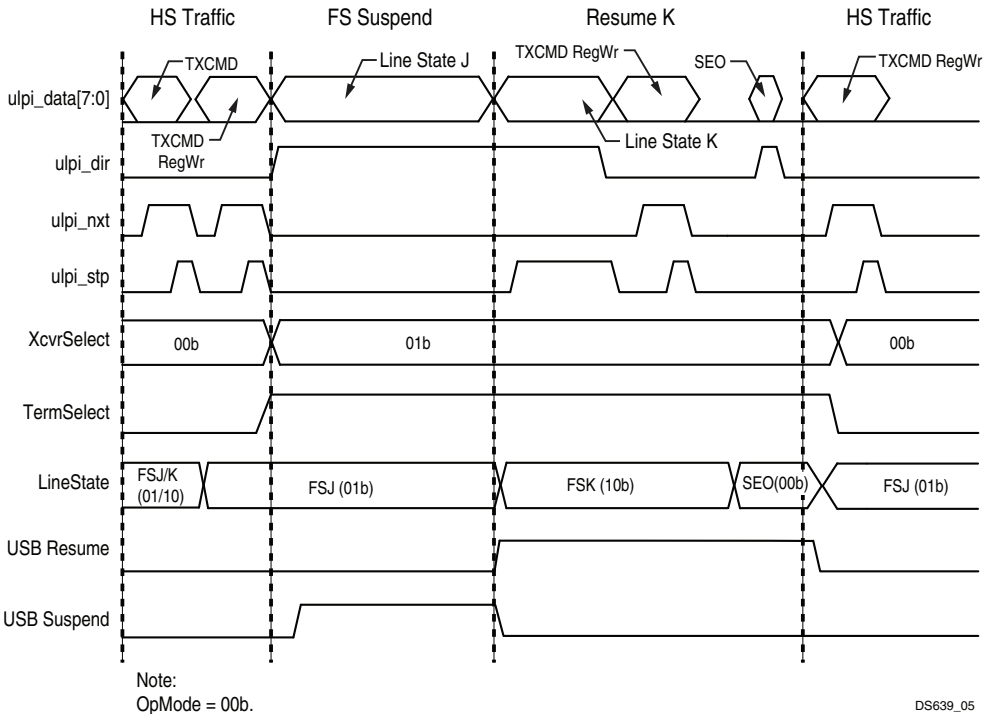

Figure 5: **Suspend Signalling Followed with Resume Signalling from Host - High Speed**

Remote Wakeup Signalling

When the XPS USB 2.0 Device is suspended by the host, it supports the Remote Wakeup feature and initiates a Resume itself. The XPS USB 2.0 Device initiates the Remote Wakeup signalling as follows. The Suspend signalling is followed by Remote wakeup signalling from the XPS USB 2.0 Device, followed by a Resume signalling from the Host as shown in Figure 6 (Full-speed) and Figure 7 (High-speed):

1. When both the Host and the XPS USB 2.0 Device are in Low Power Mode.
2. If RmteWkup bit of the Control Register (CR) is set to 1, then SIE begins Remote Wake-up signalling by writing 0x54 (sets XcvrSelect to 00b, TermSelect to 1b, and Opmode to 10b (Chirping)) to the Function Control register of the PHY.
3. Immediately after the write in Step 2, SIE puts TXCMD as 0x40 (NOPID) following a Full-speed K for 3 ms and updates the status as Resume by setting the USB Resume bit of the Interrupt Status Register (ISR).
4. The Host takes over driving the Resume K within 1 ms after detecting the Remote Wakeup from SIE.
5. SIE wait to detect the End of Resume, for example, SE0 as line state update from PHY.
6. If SIE detects SE0 line state update from PHY,
   a. SIE moves to Full-speed by writing 0x45 (Sets TermSelect to 1b, XcvrSelect to 01b (FS), and Opmode to 00b (Normal Operation)) to the Function Control register of the PHY, if SIE was in Full-speed prior to entering into the suspend state.
   b. SIE moves to High-speed by writing 0x40 (Sets TermSelect to 0b, XcvrSelect to 00b (HS) and Opmode to 00b (Normal Operation)) to the Function Control register of the PHY, if SIE was in High-speed prior to entering into the suspend state.
7. SIE updates the HSEn bit of Interrupt Status Register (ISR) accordingly.

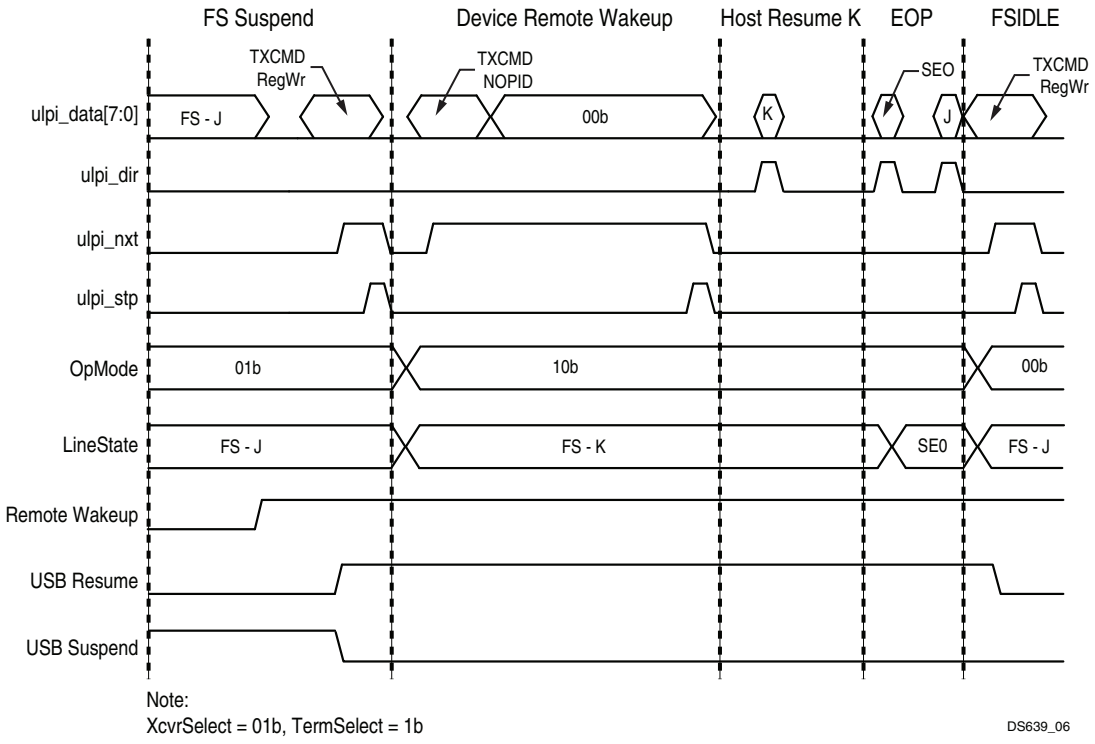

Figure 6: **Suspend Signalling Followed by Remote Wakeup Signalling from SIE - Full Speed**

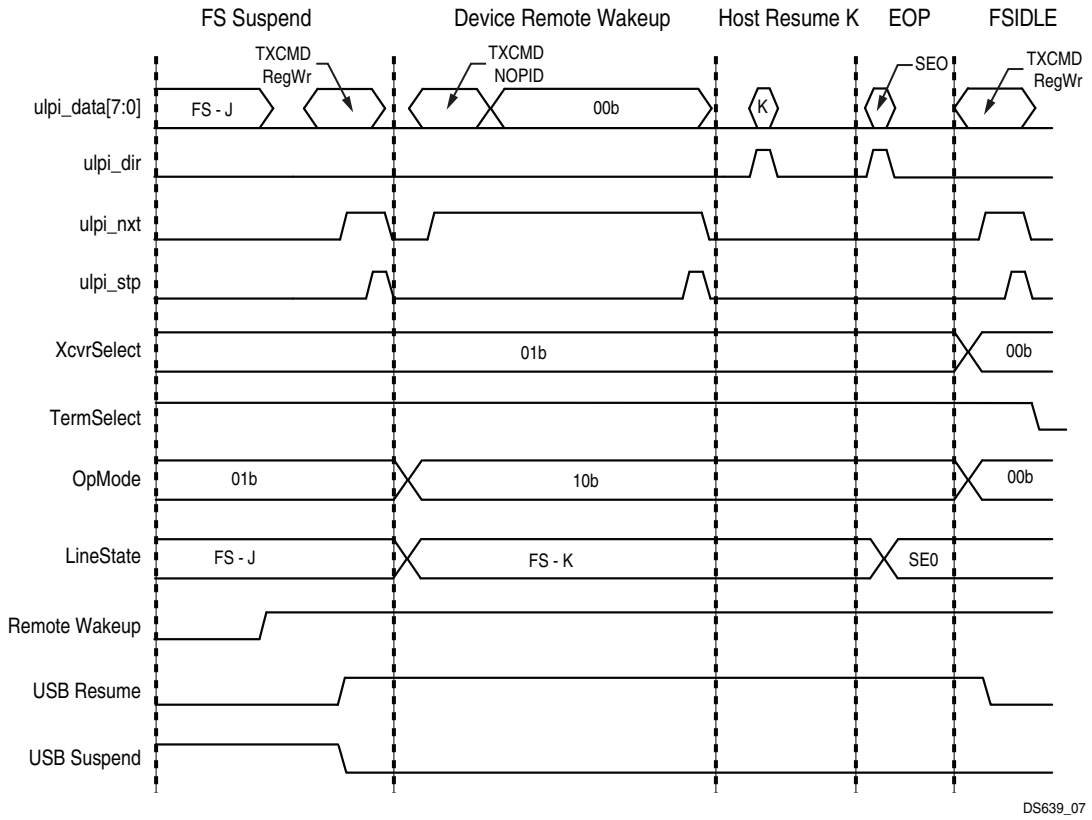

Figure 7: **Suspend Signalling Followed by Remote Wakeup Signalling from SIE - High Speed**

Test Mode Operation

The default mode of operation for the USB 2.0 Device is the normal mode, for which all the bits of the Test Mode Register (TMR) are set to 0. To put the core in Test Mode operation, program the bits [29:31] of the Test Mode Register (TMR) for different test modes:

- To put the core in Test mode J, program TMR[29:31] = 001. In this mode, Chirp J sequences must be seen on the bus.
- To put the core in Test mode K, program TMR[29:31] = 010. In this mode, Chirp K sequences must be seen on the bus.
- To put the core in Test mode NAK, program TMR[29:31] = 011. In this mode, NAK must be seen on the bus.
- To put the core in Test mode packet, program TMR[29:31] = 100. In this mode, the test packet specified by the USB 2.0 Specification must be seen on the bus.

Timing Diagrams

The XPS USB 2.0 Device Intellectual Property (IP) supports the following modes of transfer on the PLB:

- Single read
- Single write
- Burst Read
- Burst Write

Single Read

When a transfer is enabled (PLB_PAValid = 1), the controller compares the incoming address (PLB_ABus) with the base address. If they match, it latches all the PLB inputs (as they are not available after Sl_addrAck), decodes the latched address, and returns the corresponding read data on the Sl_rdDBus. Sl_rdDAck qualifies the read data and Sl_rdComp implies the completion of the read transfer.

Sl_addrAck, Sl_rdDAck, and Sl_rdComp must be asserted within 16 PLB clock cycles after the assertion of PLB_PAValid (including the clock cycle where the PLB_PAValid is asserted). There must be a minimum difference of two clock cycles between the assertion of Sl_addrAck and Sl_rdDAck.

The timing diagram for a single read transaction is shown in Figure 8.

For single read transactions:

- Reads from address locations that are defined as reserved return all 0s on the Sl_rdDBus bus.
- Reads from write only address locations return all 0s on the Sl_rdDBus bus.

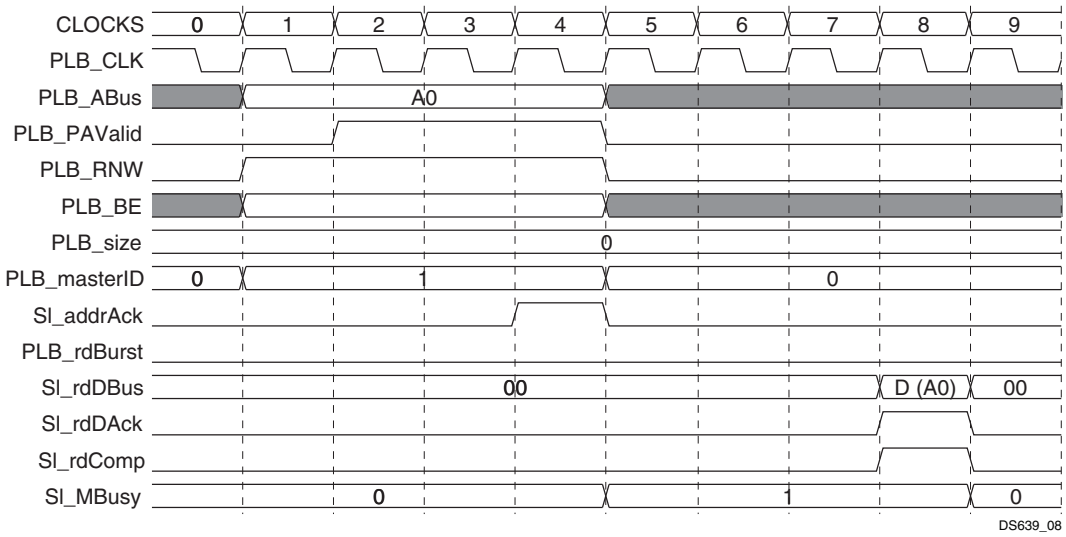

Figure 8: **PLB Single Read Transaction Timing Diagram**

Single Write

When a transfer is enabled (PLB_PAValid = 1), the controller compares the incoming address (PLB_ABus) with the base address. If they match, it latches all the PLB inputs (as they are not available after Sl_addrAck), decodes the latched address and performs the corresponding write operation. The Sl_wrComp indicates the completion of the write operation, while Sl_wrDAck indicates that the data on PLB_wrDBus is written to the addressed location.

Sl_addrAck, Sl_wrDAck, and Sl_wrComp must be asserted within 16 PLB clock cycles after the assertion of PLB_PAValid (including the clock cycle where the PLB_PAValid is asserted).

The timing diagram for a single write transaction is shown in Figure 9.

For single write transactions:

- Writes to address locations and bits that are defined as reserved does not have any effect.
- Writes to address locations defined as read only does not have any effect.

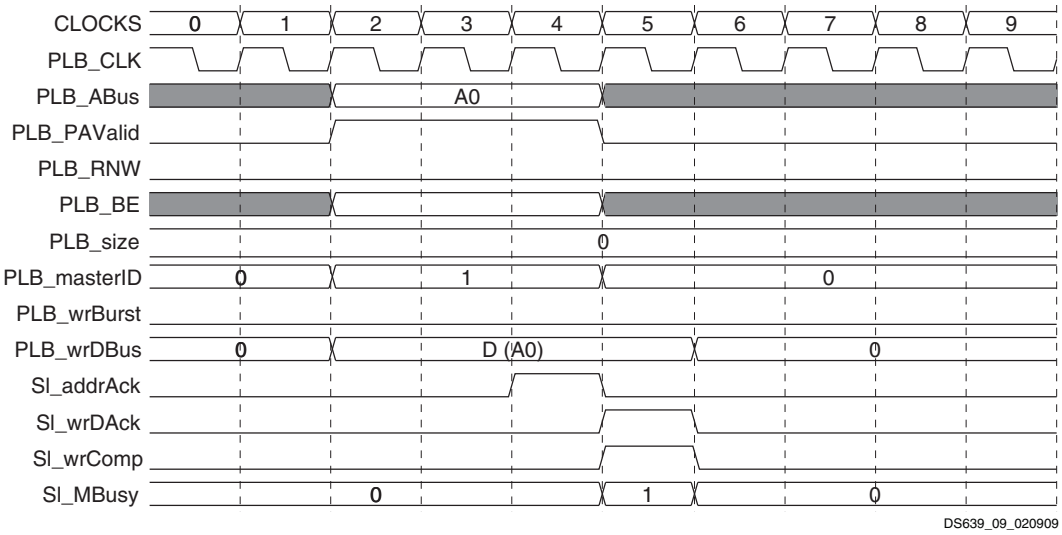

Figure 9: **PLB Single Write Transaction Timing Diagram**

Burst Read

Figure 10 and Figure 11 illustrates the Burst Read behavior of the XPS USB 2.0 Device through the PLB Slave and PLB Master Interfaces respectively.

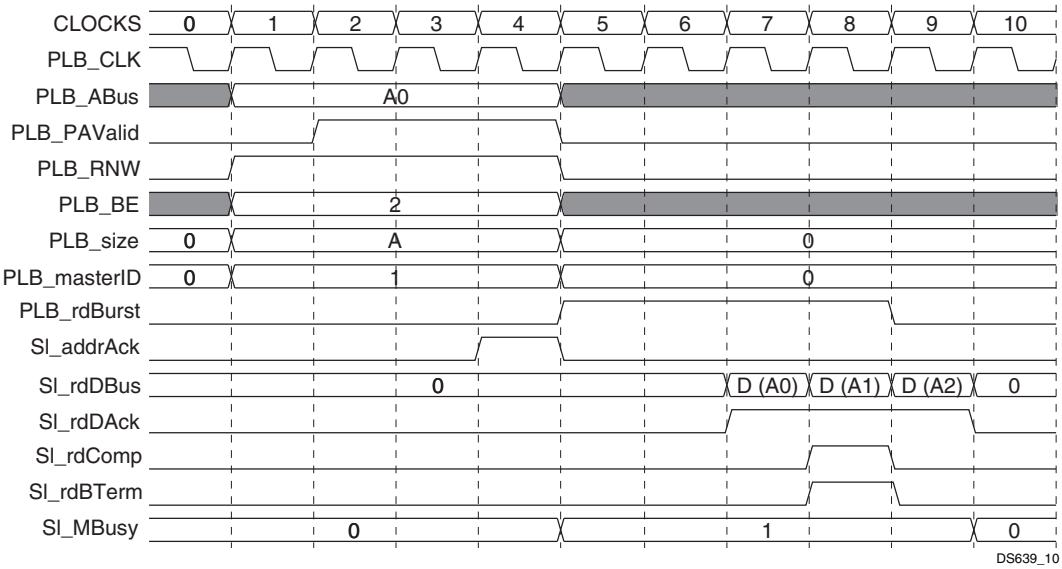

Figure 10: **PLB Slave Burst Read Transaction Timing Diagram**

Figure 11: **PLB Master Burst Read Transaction Timing Diagram**

Burst Write

Figure 12 and Figure 13 illustrates the Burst Write behavior of the XPS USB2 Device through PLB Slave and PLB Master Interfaces, respectively.

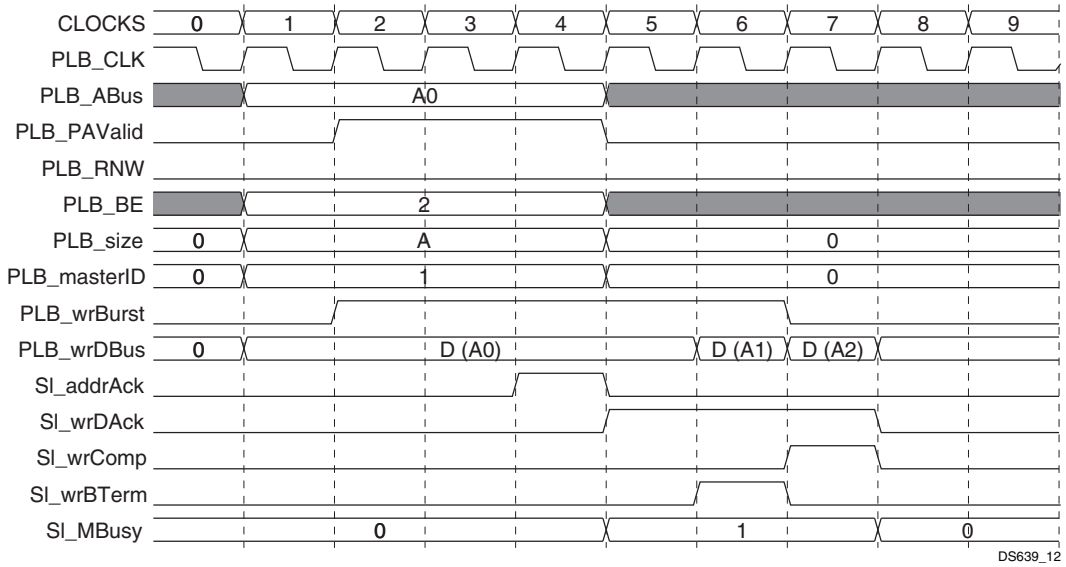

Figure 12: **PLB Slave Burst Write Transaction Timing Diagram**

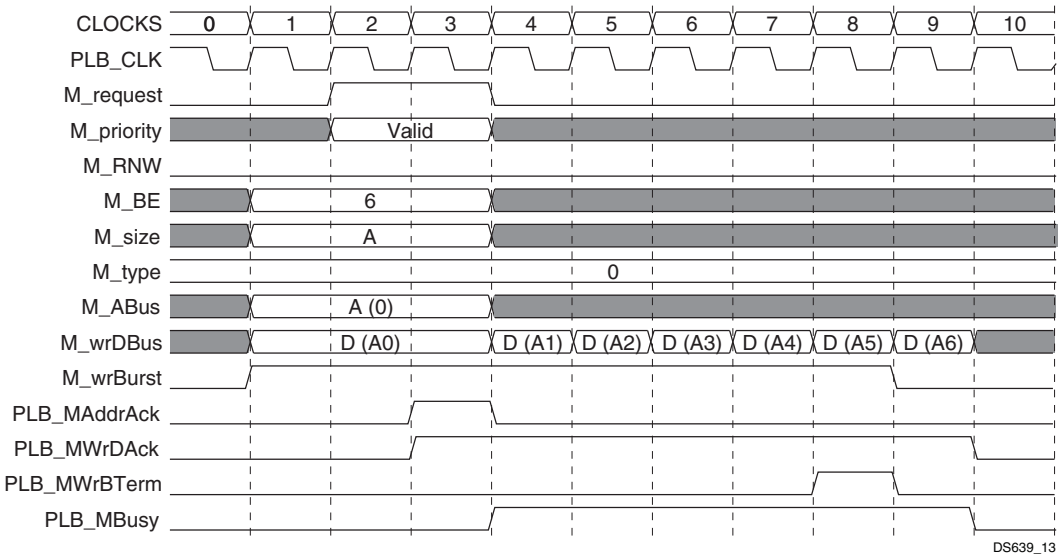

Figure 13: **PLB Master Burst write Transaction Timing Diagram**

Clocking and Reset

Clocking

The clock to the USB 2.0 Device runs at 480 MHz when operating at high speed. Because this frequency is too high for the SIE clock, as well as for the FPGA, the PHY interfaces with the SIE and generates a 60 MHz clock.

The XPS USB2 Device uses two clocks:

- **PLB CLK**: The PLB Bus interface, Port B of the DPRAM, and the PLB registers use this clock. The minimum PLB CLK frequency needed to achieve the maximum performance of 480 MHz is 60 MHz.
- **ULPI CLK**: The SIE interface and Port A of the DPRAM operate using this clock. The ULPI clock is generated by the PHY and has a fixed frequency of 60 MHz.

Reset

The XPS USB2 Device core is reset using the `PLB_Rst` signal which resets all devices that are connected to the PLB Bus in the processor system. The minimum duration of the reset pulse required to reset the logic on the SIE side is 1 ULPI clock period.

Design Constraints

Location Constraints

The ULPI pins of the core should be connected to the corresponding pins of the ULPI PHY.

Timing Constraints

The core has two different clock domains: PLB_CLK and ULPI_Clock. The following constraints can be used with the XPS USB2 Device.

Period Constraints for Clock Nets

Note: The following constraints are automatically added to the system with the core's tcl script. If the SPLB_Clk and MPLB_Clk clocks have a different frequency when C_INCLUDE_DMA is enabled, then similar cross-clock domain constraints between ULPI <-> MPLB, ULPI <-> MPLB, and SPLB <-> MPLB domains are added automatically by the tcl script. The user can overwrite these constraints by explicitly declaring the constraints in the UCF.

ULPI_CLK

The ULPI clock input is generated through the ULPI PHY and has a fixed frequency of 60 MHz.

```
# Set the ULPI_CLK constraints
NET "ULPI_CLK" TNM_NET = "ULPI_CLK";
TIMESPEC "TS_ULPI_CLK" = PERIOD "ULPI_CLK" 16667 ps HIGH 50%;
```

PLB_CLK

The clock provided to PLB_CLK must be constrained for a clock frequency of 60 MHz - 125 MHz.

```
# Set the PLB_CLK constraints; This can be relaxed based on the actual frequency
NET "PLB_CLK" TNM_NET = "PLB_CLK";
TIMESPEC "TS_PLB_CLK" = PERIOD "PLB_CLK" 10 ns HIGH 50%;
```

ULPI Interface Constraints

```
# Set the OFFSET IN delay as 8.50 ns with respect to ULPI_CLK
OFFSET = IN 8.50 ns VALID 11.50 ns BEFORE ULPI_CLK RISING;
# Set the OFFSET OUT delay as 10.50 ns with respect to ULPI_CLK
OFFSET = OUT 10.50 ns AFTER ULPI_CLK RISING;
# Set MAX DELAY constraint on ULPI_Dir pin
NET "ULPI_Dir" MAXDELAY=4.5 ns;
# Cross clock domain timing Constraints between ULPI_Clk and SPLB_Clk
#DMA is included and both slave and master plb clock frequencies are EQUAL
NET "ULPI_Clock" TNM_NET = "ulpi_0_clock_net";
NET "SPLB_Clk" TNM_NET = "splb_0_clock_net";
TIMEGRP "ulpi_0_clock_grp" = "ulpi_0_clock_net";
TIMEGRP "splb_0_clock_grp" = "splb_0_clock_net";
TIMESPEC TS_splb_0_to_ulpi_0_clk = FROM "splb_0_clock_grp" TO "ulpi_0_clock_grp" 32.2 ns
DATAPATHONLY; # (2 * ULPI Clock period - 1)
TIMESPEC TS_ulpi_0_to_splb_0_clk = FROM "ulpi_0_clock_grp" TO "splb_0_clock_grp" 19.0 ns
DATAPATHONLY; #(2 * SAXI Clock period - 1)
```

Design Implementation

Target Technology

The target technology is an FPGA listed in the Supported Device Families field of the LogiCORE IP Facts Table. The device used must have the following attributes:

- Large enough to accommodate the core
- Contain a sufficient number of Input Output Blocks (IOBs)/block RAM

Device Utilization and Performance Benchmarks

The XPS USB 2.0 Device core resource utilization for various parameter combinations measured with the Virtex®-6 FPGA as the target device are detailed in Table 26.

Table 26: **Performance and Resource Utilization Benchmarks on the Virtex-6 FPGA (xc6vlx130t-1-ff1156)**

Parameter Values		Device Resources			Performance
C_INLUDE_USBERR_LOGIC	C_INCLUDE_DMA	Slices	Slice Flip-Flops	LUTs	f_{Max} (MHz)
0	0	734	720	2,201	160
0	1	902	1,163	2,464	160
1	0	713	767	2,167	160
1	1	814	1,208	2,451	160

The XPS USB 2.0 Device core resource utilization for various parameter combinations measured with the Spartan®-6 FPGA as the target device are detailed in Table 27.

Table 27: **Performance and Resource Utilization Benchmarks on the Spartan-6 FPGA(xc6slx150t-2-fgg900)**

Parameter Values		Device Resources			Performance
C_INLUDE_USBERR_LOGIC	C_INCLUDE_DMA	Slices	Slice Flip-Flops	LUTs	f_{Max} (MHz)
0	0	715	725	2,103	100
0	1	886	1,165	2,555	100
1	0	737	774	2,103	100
1	1	979	1,226	2,685	100

Because the XPS USB 2.0 Device core can be used with other design modules in the FPGA, the utilization and timing numbers reported in this section are estimates only. When this core is combined with other designs in the system, the utilization of FPGA resources and timing of the XPS USB 2.0 Device core varies from the results reported here.

The XPS USB 2.0 Device core resource utilization for various parameter combinations measured with the Virtex-5 FPGA as the target device are detailed in Table 28.

Table 28: **Performance and Resource Utilization Benchmarks on the Virtex-5 FPGA (xc5vlx30-1-ff676)**

Parameter Values		Device Resources			Performance
C_INLUDE_USBERR_LOGIC	C_INCLUDE_DMA	Slices	Slice Flip-Flops	LUTs	f_{Max} (MHz)
0	0	862	779	2,075	150
0	1	1,172	1,249	2,622	150
1	0	941	826	1,994	150
1	1	1,177	1,296	2,714	150

The XPS USB2 Device core resource utilization for various parameter combinations measured with the Virtex-4 FPGA as the target device are detailed in Table 29.

Table 29: **Performance and Resource Utilization Benchmarks on the Virtex-4 FPGA (xc4vfx100-10-ff1152)**

Parameter Values		Device Resources			Performance
C_INLUDE_USBERR_LOGIC	C_INCLUDE_DMA	Slices	Slice Flip-Flops	LUTs	f_{Max} (MHz)
0	0	1,849	795	2,798	125
0	1	2,555	1,267	3,622	125
1	0	1,999	826	2,883	125
1	1	2,515	1,297	3,660	125

The XPS USB2 Device core resource utilization for various parameter combinations measured with the Spartan-3 FPGA as the target device are detailed in Table 30.

Table 30: **Performance and Resource Utilization Benchmarks on the Spartan-3 FPGA (xc3s1600e-4-fg484)**

Parameter Values		Device Resources			Performance
C_INLUDE_USBERR_LOGIC	C_INCLUDE_DMA	Slices	Slice Flip-Flops	LUTs	f_{Max} (MHz)
0	0	1,709	779	2,775	100
0	1	2,421	1,250	3,602	100
1	0	1,907	828	2,905	100
1	1	2,573	1,303	3,712	100

System Performance

To measure the system performance (F_{MAX}) of the XPS USB Device core, it was added as the Device Under Test (DUT) to a Virtex-6 FPGA system as shown in Figure 14, and a Spartan-6 FPGA system as shown in Figure 15, a Virtex-5 FPGA system as shown in Figure 16, Virtex-4 FPGA system as shown in Figure 17, and a Spartan-3A FPGA system as shown in Figure 18.

Because the XPS USB Device core is used with other design modules in the FPGA, the utilization and timing numbers reported in this section are estimates only. When the XPS USB Device core is combined with other designs in the system, the utilization of FPGA resources and timing vary from the results reported here.

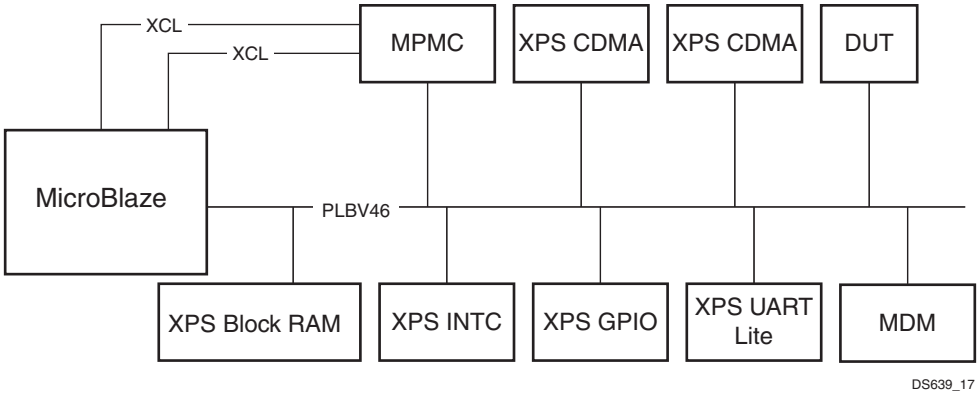

Figure 14: **Virtex-6 FPGA System with the XPS USB2 Device as the DUT**

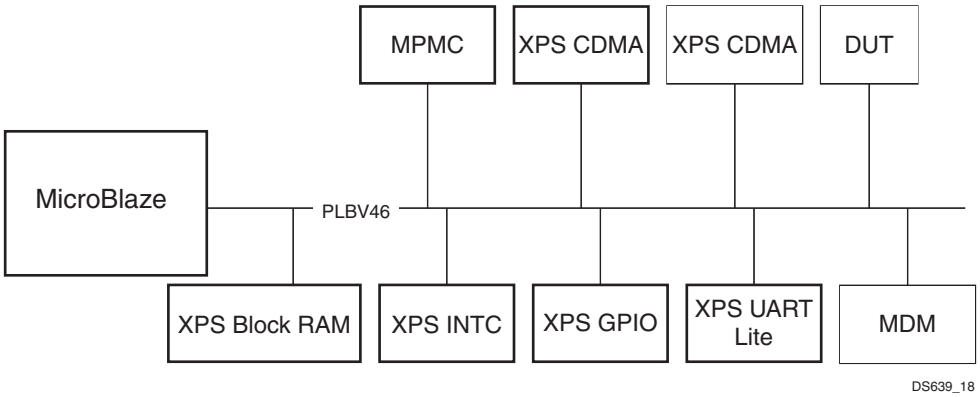

Figure 15: **Spartan-6 FPGA System with the XPS USB2 Device as the DUT**

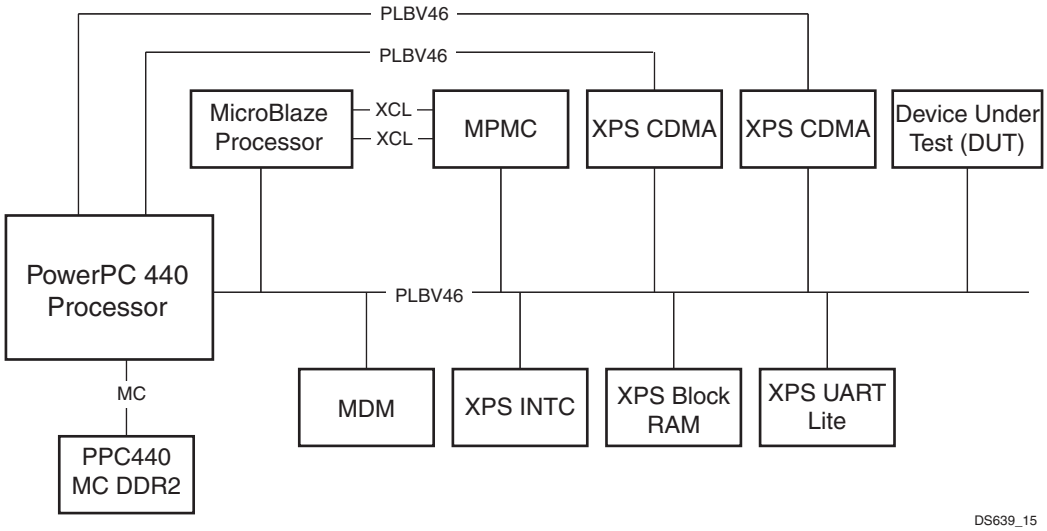

Figure 16: **Virtex-5 LX FPGA System with the XPS USB2 Device as the DUT**

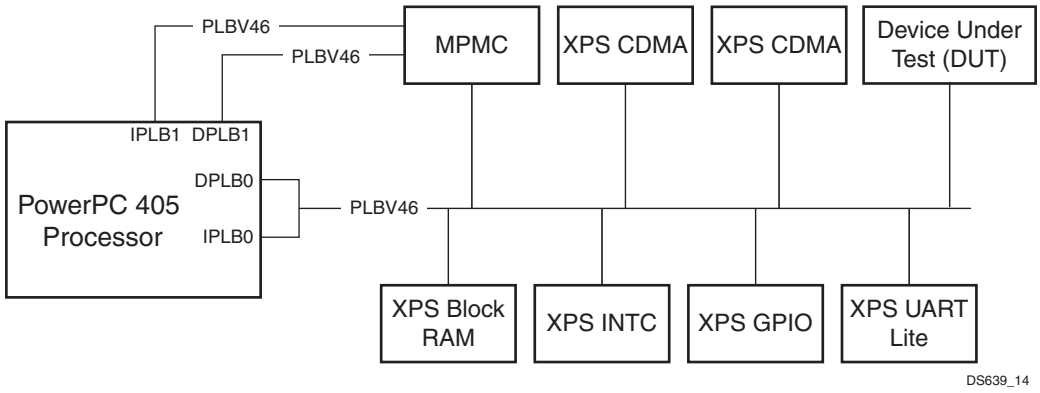

Figure 17: **Virtex-4 FX FPGA System with the XPS USB2 Device as the DUT**

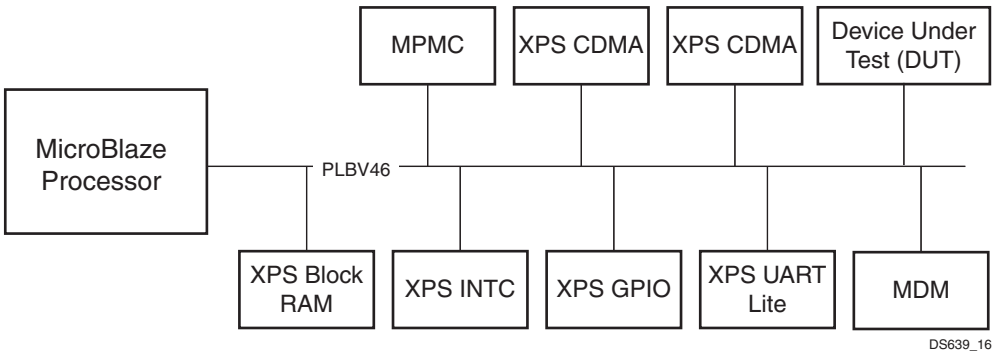

Figure 18: **Spartan-3A FPGA System with the XPS USB2 Device as the DUT**

The target FPGA was then filled with logic to drive the Lookup Table (LUT) and block RAM 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 31.

Table 31: **XPS Central DMA System Performance**

Target FPGA	Estimated F_{MAX} (MHz)
V6LX130t -1	150
S6LX45t -2	100
V5LXT50 -1	125
V4FX60 -10	100
S3A700 -4	90

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

Throughput

To measure the system throughput of the XPS USB Device core, it was used on a Virtex-5 FPGA system similar to Figure 16 and tested on a Xilinx ML507 board.

For this test, the firmware running on the ML507 evaluation board behaves as a mass storage device. A Windows application running on a Windows Desktop PC sets up the mass storage device for the throughput test on its BULK IN endpoint. The windows application has an option of selecting the amount of traffic being sent on the bus for the throughput test. The selection options were 1MB/10MB/1000MB. When the option was selected, the Windows application sets up the mass storage device to send the selected amount of data. Upon completion of the data transfer, the result was displayed on the Windows screen in megabytes per second (Mb/s). The throughput was measured using the amount of data sent over the bus and the time taken for transmission. A LeCroy USB analyzer was used to capture the transactions on the bus. The throughput results (See Table 32) were further provided based on the number of USB data packets received between two SOF packets. The throughput numbers are shown in the Table 32.

Table 32: **XPS Central DMA System Performance**

Throughput in the Functional Simulations	Throughput on the ML Board
12 packets per micro frame = 49,152,000 B/s	11 packets per micro frame = 45,056,000 B/s

Note: In BULK IN mode, the performance of the device depends on the number of data transfer packets that the Host application can initiate between the two SOFs.

Support

Xilinx provides technical 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*.

Ordering Information

This Xilinx LogiCORE IP module is provided under the terms of the Xilinx Core Site License. The core is generated using the Xilinx Integrated Software Environment (ISE®) Embedded Edition software (EDK). For full access to all core functionality in simulation and in hardware, you must purchase a license for the core. Contact your local Xilinx sales representative for information on pricing and availability of Xilinx LogiCORE IP.

For more information, visit the XPS USB2 Device product web page.

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 modules and software, contact your local Xilinx sales representative.

This Embedded IP module is provided under the terms of the Xilinx Core Site License. A free evaluation version of the core is included with the ISE Design Suite Embedded Edition software. Use the Xilinx Platform Studio (XPS) application included with the Embedded Edition software to instantiate and use this core.

For full access to all core functionality in simulation and in hardware, you must purchase a license for the core. Contact your local Xilinx sales representative for information on pricing and availability of Xilinx LogiCORE IP.

Reference Documents

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

For a glossary of technical terms used in Xilinx documentation, see: www.xilinx.com/company/terms.htm.

1. IBM CoreConnect 128-Bit Processor Local Bus, Architectural Specification (v4.6).
2. Universal Serial Bus Specification, Revision 2.0
3. UTMI+ Low Pin Interface (ULPI) Specification, Revision 1.1

Revision History

Date	Version	Revision
10/30/07	1.0	Xilinx Initial Release
04/21/08	1.1	Added Automotive Spartan-3E, Automotive Spartan-3A, Automotive Spartan-3, and Automotive Spartan-3A DSP support.
07/10/08	1.2	Figure 1 updated with DMA controller & PLB Master interface support. DMA Registers description added under XPS USB2 Device Register Description. Figure 10 and Figure 11 added.
01/02/09	1.3	Added Figure 2, Figure 3, Figure 4, Figure 5, Figure 6, Figure 7, Figure 10, Figure 11, Figure 12, and Figure 13 to explain the features Suspend, Resume, Reset and Remote Wake-up signalling briefly. Updated Figure 8 and Figure 9. Updated Performance and resource utilization Benchmarks on Spartan3, Virtex-4 and Virtex-5 families respectively.
4/24/09	1.4	Replaced references to supported device families and tool name(s) with hyperlink to PDF file; added System Performance and Throughput sections and associated figures.
7/20/09	1.5	Updated Performance and resource utilization Benchmarks on Spartan-6 and Virtex-6 families respectively
10/22/09	1.6	Added Error Count Register (ECR) to count different USB Errors occur at USB interface. Updated Interrupt Status Register (ISR) with BitStuffErr, PID error, CRC error bits.
02/04/10	1.7	Updated the resource utilization tables for spartan6 and virtex6 families.
04/05/10	1.8	Added the ULPI Interface Constraints section and updated the interface constraints.
07/23/10	1.9	Created v3.01a for 12.2 release; converted to current DS template; updated images to graphic standard; added ordering information.
9/12/10	2.0	Created v4.00a for the 12.3 release: added C_INCLUDE_USBERR_LOGIC and C_PHY_RESET_TYPE parameter to the data sheet and updated the resource utilization tables accordingly.
12/14/10	3.0	Created v5.00a for the 12.4 release.
3/1/11	3.1	Updated to v6.00a for the 13.1 release.
10/19/11	3.2	Summary of core changes • Updated to v7.00.a for the 13.3 release • Added ULPI/UPAR register access to the XPS core to control the PHY Summary of documentation changes • Added List of Acronyms. Spelled out acronym for first occurrence • Updated Notice of Disclaimer and copyright notice • Updated IP Facts table • Updated links in Ordering and Licensing Information section • Reorganized device information throughout so that newer devices are listed first • Changed all “BRAM” to “block RAM”
10/16/12	3.4	Interrupt Status Register, bit 12 updated; note in ULPI PHY Access Register updated. ISE release 14.3.

Notice of Disclaimer

The information disclosed to you hereunder (the "Materials") is provided solely for the selection and use of Xilinx products. To the maximum extent permitted by applicable law: (1) Materials are made available "AS IS" and with all faults, Xilinx hereby DISCLAIMS ALL WARRANTIES AND CONDITIONS, EXPRESS, IMPLIED, OR STATUTORY, INCLUDING BUT NOT LIMITED TO WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, OR FITNESS FOR ANY PARTICULAR PURPOSE; and (2) Xilinx shall not be liable (whether in contract or tort, including negligence, or under any other theory of liability) for any loss or damage of any kind or nature related to, arising under, or in connection with, the Materials (including your use of the Materials), including for any direct, indirect, special, incidental, or consequential loss or damage (including loss of data, profits, goodwill, or any type of loss or damage suffered as a result of any action brought by a third party) even if such damage or loss was reasonably foreseeable or Xilinx had been advised of the possibility of the same. Xilinx assumes no obligation to correct any errors contained in the Materials or to notify you of updates to the Materials or to product specifications. You may not reproduce, modify, distribute, or publicly display the Materials without prior written consent. Certain products are subject to the terms and conditions of the Limited Warranties which can be viewed at http://www.xilinx.com/warranty.htm; IP cores may be subject to warranty and support terms contained in a license issued to you by Xilinx. Xilinx products are not designed or intended to be fail-safe or for use in any application requiring fail-safe performance; you assume sole risk and liability for use of Xilinx products in Critical Applications: http://www.xilinx.com/warranty.htm#critapps.