USB3.0 (Host) IP Protocol & Link Layer Core

Design Gateway Co.,Ltd
54 BB Building 13th Fl., Room No.1302 Sukhumvit 21 Rd. (Asoke), Klongtoey-Nua, Wattana, Bangkok 10110
Phone: (+66) 02-261-2277
Fax: (+66) 02-261-2290
E-mail: sales@design-gateway.com
URL: www.design-gateway.com

Core Facts

Provided with Core	
Documentation	Reference design manual
Design File Formats	Encrypted hdl File
Instantiation Templates	VHDL
Reference Designs & Application Notes	QuartusII Project, See Reference Design Manual
Additional Items	Demo on Altera Development Kit (Requires AB08-USB3HSMC)
Support	
Support Provided by Design Gateway Co., Ltd.	

Features

- Compliant with USB3.0 specification Revision1.0.
- Support SuperSpeed (5Gbps)
- USB3.0 Host Controller
- Implement link layer and protocol layer
- Physical layer interfaces to PHY chip by TI (TUSB1310A)
- IP core clocks are adjustable (250MHz for PIPE I/F, more than 125MHz for internal)
- Support 16bit PIPE interface
- Support IN/OUT root point up to 15 points
  - 1 point for control
  - 7 points each for IN/OUT
- Support All transmission taps (Control, Bulk, Isochronous and Interruput transmission)
- Simple transaction interface with Host processor or DMA interface
- Free demo SOF file for evaluation on Altera board.
- Reference design available on CycloneIV GX, CycloneVE, ArriaV GX Development Kit with Design GateWay AB08-USB3HSMC Card

Table 1: Example Implementation Statistics (Control x1, IN x2, OUT x2)

Family	Example Device	Fmax (MHz)	Combinational ALUTs[1]/ Logic Elements	Registers[1]	Pin[2]	Block Memori bit	PLL	Design Tools
CycloneIV GX	EP4CGX150DF31C7	133	8,778	3,885	70	135,168	2	QuartusII 13.1
Arria II GX	EP2AGX125EF35C4	158	5,751	3,885	70	135,168	2	QuartusII 10.1
CycloneV E	5CEFA7F31I7	140	5,816	4,417	70	135,168	2	QuartusII 14.0
ArriaV GX	5AGXFB3H4F35C4	142	5,816	4,305	70	135,168	2	QuartusII 14.0

Notes:
1) Actual slice count dependent on percentage of unrelated logic – see Mapping Report File for details
2) Assuming all core I/Os, I/Os for TI_PHY and clocks are routed off-chip and internal signals are enclosed by FF.
3) The number of end port of the core is variable. In case of Bulk transmission, FIFO size ican be reduced to one

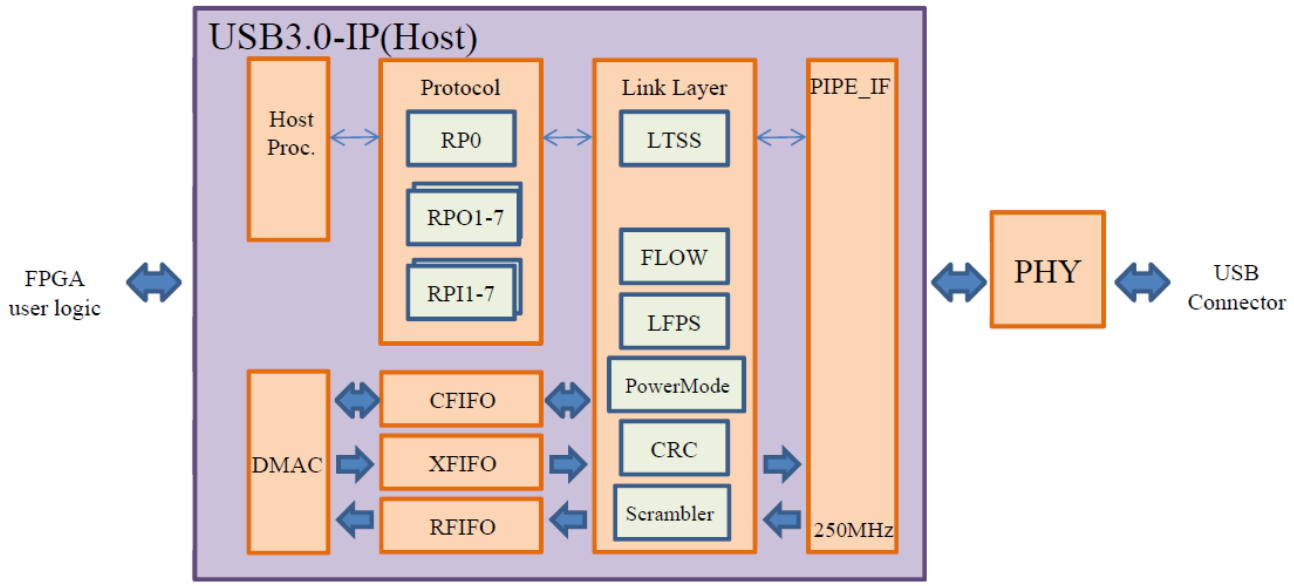


Figure 1: USB3.0 (Host) IP Block Diagram

Applications

USB3.0 (Host) IP Core is ideal for use in a USB3.0 supported device system which require high bandwidth up to 5.0Gbps. This IP Core will process almost all USB3.0 protocols (some part of chapter 6, chapter 7 and 8 of the USB3.0 specification) by hardware. It achieves processing by low-end CPU. By setting parameter, this IP Core flexibly supports both a device achieved by minimum root point such as mass storage class and a high-end device which need multiple root points. This IP Core is optimized for saving FPGA internal logic resource by eliminating legacy USB protocol of USB2.0 (480Mbps) or earlier, so that it provides most cost-effective solution for 5Gbps super speed implementation.

General Description

The USB3.0 (Host) IP Core implements link layer and protocol layer. Only setting data address on memory prepared by Host processor, transmission length and other parameter to the register in the IP Core, IP Core will process dividing into packets, adding CRC & Scramble and flow control on USB bus, and return processing result to the register. Data receive process is also same flow.

Host interface of the IP Core consist of simple register access interface, provide simple DMA access interface for memory and able to connect to NiosII or uniphy_ddr3.

For connection with PHY chip, it is compliant with standard PIPE interface. So it is just able to connect through Flip-Flop for timing adjustment with FPGA port.

Internal clock in the IP Core is more than 125MHz (125M x 4bytes = 500MB/s) and clock for connection with PHY chip is fixed to 250MHz (500MHz x 2 bytes). However Host interface and DMA interface are able to connect by low frequency clock using necessary synchronization.

Functional Description

The USB3.0 (Host) IP Core is structured by 3 blocks.

Protocol Layer

Protocol Layer manages data on memory assigned by register from Host processor, and divides to USB3.0 packet and sends to Link layer. Receiving is opposite process. This layer manages End-to-End sequence number with host (host bus adaptor auch as PC) and credit process. By software on Host processor, one root point can control several devices or end points.
Note: In this document, the control circuit of host side is called root point.

- **RP0(Root point 0)**
  Process control transmession specified by USB3.0.
  It includes setup packet receiving, data transmission for control (In/Out) and status transmission.
- **RPO(Root point Out 1~7)**
  Process BULK-OUT transmission, INTERRUPT-OUT transmission and ISOCHRONOUS-OUT transmission.
- **RPI(Root point In 1~7)**
  Process BULK-IN transmission, INTERRUPT- IN transmission and ISOCHRONOUS-IN transmission.
- **MPP(Multi-purpose point)**
  It is used for transmission/receiving process of Port-Capabilities / Port-Configuration/Response after changing to U0 status and used for transmission/receiving process of Isochronous Time Stamp(ITS) and Link Management Packet(LMP).
- **FIFO**
  CFIFO is data sending FIFO for EP0. XFIFO is data transmission FIFO (up to 2pcs/automatic assignment) which is shared at RPI. RFIFO is data receiving FIFO (up to 2pcs/automatic assignment) which is shared at RPO.
- **DMAC, Arbiters**
  DMA request from each root points and packet send/receive request are adjusted and sequencially processed in Protocol Layer.

Link Layer

Link Layer adds CRC (CRC-5 or16 or 32) to packet from Protocol Layer, scrambles it and send it with 8B(x4) symbol to PIPE_IF. Sending and receiving between links and flow management are also processed by this layer.

- **LSSSM block**

  Manage link status specified by USB3.0, initialize and do sequential processing of power status.

- **FLOW block**
  Flow control between links. Manage transmission status (normal or abnormal), retry and credit process between links.
- **LFPS block**

  Send and receive LFPS(Low Frequecncy Periodic Signaling) when initialization or returning from power mode.

- **Power Mode block**

  Process send / receive Link Command when switching to power mode.

- **Transmission/Receiving process block**

  Add/check CRC, add/cancel scramble and add/check Link Control word.

PIPE Interface

PIPE interface sends/receives data and signal from/to Link Layer to/from PHY, synchronized with PIPE clock.

- **Transmission block**

  It converts 8B(x4) symbol to 8B(x2) symbol, executes Elastic process for the symbol and adds SKIP order set. (In case that internal is 125MHz operation, transceiver side also need Elastic procass because of frequency differential with PHY clock.)

- **Receiving block**
  It is reverse process of Transmission block.
- **Control block**

  Send necessary signal synchronized with PIPE clock, according to state transition of LinkLayer. And It sends signal from PHY to Link Layer synchronized with internal clock.

FPGA Controller

Normally host processor which executes application software is used as FPGA internal controller, and it manages control which is upper than USB device framework (Specification chapter 9 and after) by register access of USB 3.0 Device IP Core. System controller consists of Host processor, DMA interface, memory and so on.

USB3.0 PHY

Use external chip supported USB3.0, such as TUSB1310A by TI which has PIPE_IF.

Core I/O Signals

Core I/O signals are not fixed with specified device and any pins to able to connect to user logic flexibly.All core I/O signals are shown as following table 2.
Logic of these signals are active high when not specified.

Table 2: Core I/O Signals.

Signal	Signal Direction	Description
Common Interface Signal		
RST_N	In	Reset USB3.0 (device) IP core. Active low.
CLK	In	IP Core operating frequency output（125MHz）.
PCLK	In	PIPE clock (250MHz). Input same frequency (phase is able to adjust by DCM) with PIPE clock generated by PHY chip. Able to switch to the constant operating clocks (such as CLK) during PCLK stop.
I_RPO_ENB[7:1]	In	Define implementation/un-implementation of Root Point Out(RPO). Able to implement up to 7 points. [1] RPO1, [7] RPO7. 1=Implement
I_RPI_ENB[7:1]	In	Define implementation/un-implementation of Root Point In(RPI). Able to implement up to 7 points. [1] RPI1, [7] RPI7. 1=Implement

Signal	Signal Direction	Description
PIPE Interface Signal		
O_PIPE_READY	Out	PIPE clock (PCLK) is operating. After changing to other state except P3, detect rising of I_PHY_STATUS_ASYN and assert.
O_RX_TERMINATION	Out	Control RX termination existence (yes or no). 1 = yes, 0 = no. Change asynchronizing with PCLK (change even though during PCLK stop)
O_TX_DETRX_ASYN	Out	Control detecting RX termination existence of opposite side. Change asynchronizing with PCLK, Use it at near P3 state(O_PIPE_READY negate).
O_TX_EIDLE_ASYN	Out	Control TX signal(SSTXP/SSTXN) to electrical idle state. Change asynchronizing with PCLK, Use it at near P3 state(O_PIPE_READY negate).
O_PWR_DOWN_ASYN[1:0]	Out	Control power state of PHY. 00:P0, 01:P2, 10:P3, 11:P3 Change asynchronizing with PCLK, Use it at near P3 state(O_PIPE_READY negate).
O_TX_DETRX_LPBK	Out	Control detecting RX termination existence of opposite side. Change synchronizing with PCLK, Use it at except near P3 state(O_PIPE_READY assert).
O_TX_ELECIDLE	Out	Control TX signal(SSTXP/SSTXN) to electrical idle state. Change synchronizing with PCLK, Use it at except near P3 state(O_PIPE_READY assert).
O_POWER_DOWN [1:0]	Out	Control power state of PHY. 00:P0, 01:P2, 10:P3, 11:P3 Change synchronizing with PCLK, Use it at except near P3 state(O_PIPE_READY assert).
I_PWRPRESENT	In	Input the state of power supplying to VBUS. Asynchronizing with PCLK.
I_RX_ELECIDLE	In	Input electrical idle state of RX signal (SSRXP/SSRXN). Asynchronizing with PCLK.
I_PHY_STATUS_ASYN	In	Input the signal which control to input PHY status. Asynchronizing with PCLK. After changing from P3 state, it is used for detecting PCLK operation start.
I_RX_STATUS011_ASYN	In	Input detected result of RX termination existence of opposite side. Asynchronizing with PCLK. 1 = yes.
I_PHY_STATUS	In	Input the signal which control to input PHY status.Synchronizing with PCLK. During O_PIPE_READY is asserted, it displays detecting completion of RX terminal existence.
I_RX_STATUS011	In	Input detected result of RX termination existence of opposite side.Synchronizing with PCLK. Valid when I_PHY_STATUS = 1. 1=yes.
O_TX_DATAK[1:0]	Out	TX symbol (8B code). K code (1) or D code (0). Synchronizing with PCLK.
O_TX_DATA[15:0]	Out	Twice of TX symbol (8B code). Synchronizing with PCLK.
I_RX_VALID	In	The timing which RX symbol is valid. 1 = valid. Synchronizing with PCLK.
I_RX_DATAK[1:0]	In	RXsymbol (8B code). K code (1) or D code (0). Synchronizing with PCLK.
I_RX_DATA[15:0]	In	Twice of RX symbol (8B code). Synchronizing with PCLK.

Signal	Signal Direction	Description
Host Register Interface Signal		
I_LINK_REG_RE[15:0]	In	Read signal of control register of link layer. Not all 16 lines are implemented. 1 line is 4bytes. Assert 1 cycle synchronized with CLK.
I_LINK_REG_WE[15:0]	In	Write signal of control register of link layer. Not all 16 lines are implemented. 1 line is 4bytes. Assert 1 cycle synchronized with CLK.
I_PRTE_REG_RE[15:0]	In	Read signal of control register of protocol layer. Not all 16 lines are implemented. 1 line is 4bytes. Assert 1 cycle synchronized with CLK.
I_PRTE_REG_WE[15:0]	In	Write signal of control register of protocol layer. Not all 16 lines are implemented. 1 line is 4bytes. Assert 1 cycle synchronized with CLK.
I_XPP_REG_RE[511:0]	In	Read signal of control register of each root point. Each root point has 32 lines, however not all 32 lines are implemented. 1 line is 4bytes. Assert 1 cycle synchronized with CLK.
I_XPP_REG_WE[511:0]	In	Write signal of control register of each root point. Each root point has 32 lines, however not all 32 lines are implemented. 1 line is 4bytes. Assert 1 cycle synchronized with CLK.
O_RP_REG_RD[31:0]	Out	Read data of all control register. Valid at the next cycle when any _RE is asserted.
I_RP_REG_WD0[31:0]	In	Input write data to control register. Valid at the timing when any _WE is asserted.
I_RP_REG_WD1[31:0]	In	Input write data to control register. Valid at the timing when any _WE is asserted. In case of using RPI4~7, RPO4~7, you have to connect to the signal which is same logic with I_RP_REG_WD0. Connecting to different FF is better for load balancing.
O_RP_IRQ	Out	Interrupt signal from USB core. 1 = Interrupt. Level signal.
O_EXT_CNTL [3:0]	Out	USB core external control signal.Control register of link layer can switch ON/OFF.
O_LANE_POL	Out	RX Lane Porarity Inversion. 1: Invert , 0: Not Invert.
I_USB20_RESET	In	Input reset from USB2.0. 1 = reset. In case of no USB2.0 core, fix to 0.

Signal	Signal Direction	Description
DMA Access Interface Signal		
I_DMAC_IDLE	In	Next DMA access start enables.
O_DMAC_REQ	Out	DMA access start. Synchronize with CLK. Assert for 1cycle.
O_DMAC_ADR [31:0]	Out	DMA start address. Valid at O_DMAC_REQ asserted.
O_DMAC_U2M	Out	DMA direction. Valid at O_DMAC_REQ asserted. 0: Memory to USB, 1: USB to Memory.
O_DMAC_LEN[8:0]	Out	DMA length. Valid at O_DMAC_REQ asserted. 0x100: 256words(4K bytes), 0x001: 1word(4bytes)
O_DMAC_BE [3:0]	Out	Valid / Invalid of each bytes.
O_DMAC_DONE	Out	Data transmission completed. Assert 1 cycle after several cycle after sending end data completed.
I_DMAC_M2U_VLD	In	Valid timing of I_DMAC_M2U_DATA in case of DMA from Memory to USB.
I_DMAC_M2U_DATA[31:0] or [63:0]	In	Input data from memory in case of DMA from Memory to USB. Data width can adjust by “DMA64_MODE” paramerter ot top module. 4(or 8) bytes transmission complete (no wait) in the timing I_DMAC_M2U_VLD=1’b1.
I_DMAC_U2M_WAIT	In	Timing which memory cannot accept data in case of DMA from USB to Memory.
O_DMAC_U2M_OUT	Out	Valid timing of O_DMAC_U2M_DATA in case of DMA from USB to Memory.
O_DMAC_U2M_DATA[31:0] or [63:0]	Out	Output data to memory in case of DMA from USB to Memory. Data width can adjust by “DMA64_MODE” paramerter ot top module. 4(or 8) bytes transmission complete in the timing O_DMAC_U2M_OUT=1’b1, I_DMAC_U2M_WAIT=1’b0

Notes of PIPE interface

This is general PIPE interface, but must be careful to synchronous/asynchronous of signal when it connects with external PHY chip (see figure 2).
Input signals have signal which references at PIPE clock (PCLK) from PHY is stopping. These signals is direct to pin or the re-synchronized signal (with _ASYN) by FF operated by core internal clock. In case that there is reference signal synchronized PIPE clock during PIPE clock is operating, it must be separate to 2 lines.
Some output signals are also changed even though PIPE clock (PCLK) is stopping. These signals is direct to pin or output with re-synchronized (with _ASYN) by FF operated by core internal clock. In case that there is the output signal synchronized PIPE clock during PIPE clock is operating, multiplex in front of a pin or output by synchronized with FF operated by switching clock.

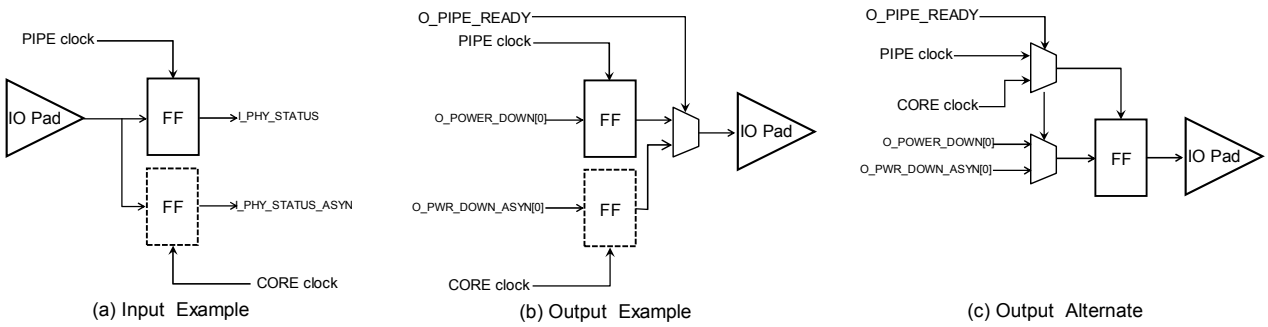


Figure 2: PIPE interface connection example

Timing Diagram of Host register interface

Register access is shown at Figure 3. 1cycle asserting I_xxx_REG_RE to output read value of the register to O_RP_REG_RD. And write when write value is input to I_RP_REG_WD with 1 cycle asserting I_xxx_REG_WE.

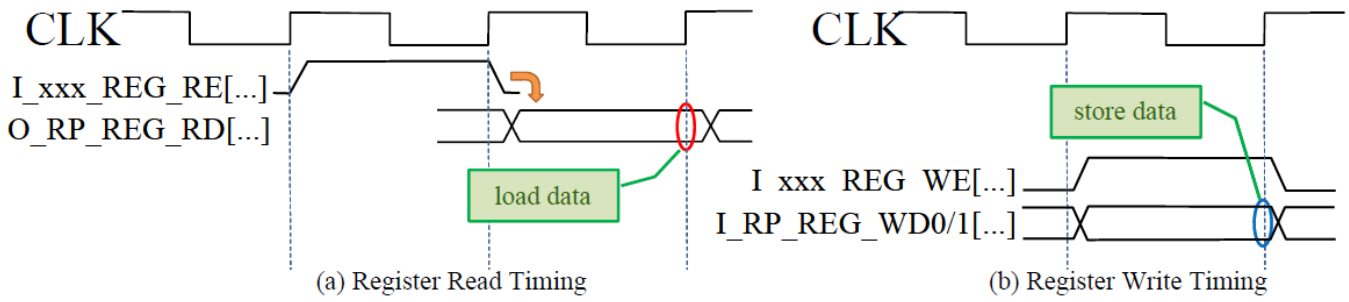
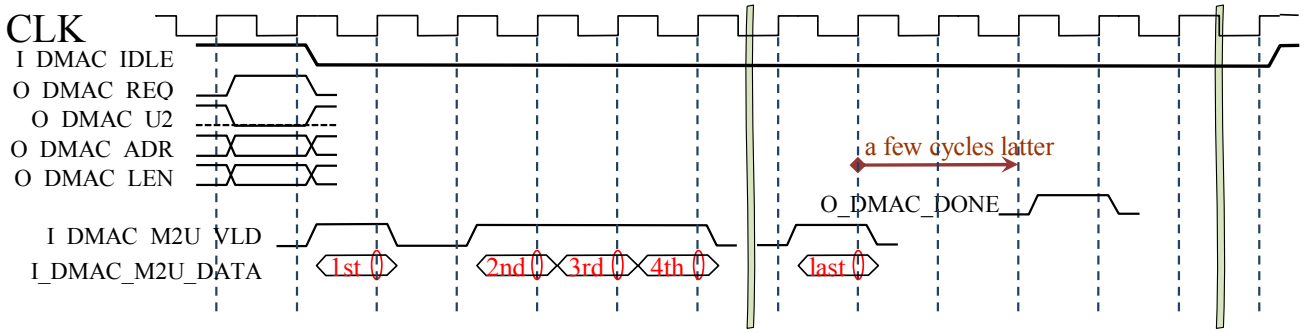


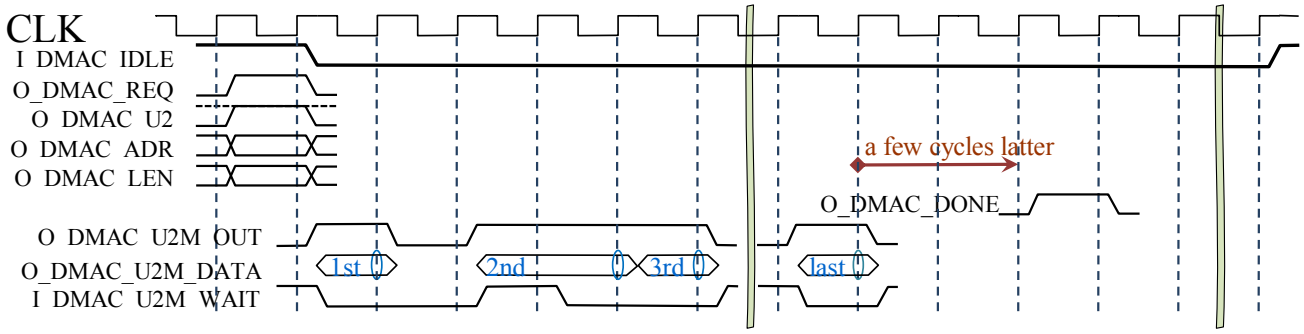
Figure 3 : Signal waveform of host register interface

Timing Diagram of DMA access interface

When I_DMAC_IDLE is asserted, O_DMAC_REQ is also asserted for 1 cycle as shown at Figure 4. In the same time, access address, direction and length are specified.
Data is transferred with flow control by VLD or WAIT signal depend on direction.
When the transmission completed and after several cycles, O_DMAC_DONE is asserted for 1 cycle then DMA access completed.



(a) DMA Read Timing(Memory to USB)



(b) DMA Write Timing (USB to Memory)

Figure 4 : Signal waveform of DMA access interface

IP Core control register

IP Core Control register which Host processor can access are shown as following Table 4.

Table 4: IP Core control register

Register Name	R/W	Offset	Description
Link Layer Register [ADDRESS=C_BASEADDR+LINK_BASE+OFFSET]			
LINK_CNFG	R/W	0h	Link Layer Configuration Register
	USB_VALID	[0]	Enable USB bus (LTSSM) state transition except to SS_Disable (Operation enable state)
	PMD_ENB	[1]	Enable to receive Power Mode transition request from Host.
	SCRB_DIS	[2]	Disable not scramble. For debugging.
	EXT_CNT	[5:4]	ON/OFF O_EXT_CNTL [1:0] (refering I/O signal).
	U2_INACT_CNT	[15:8]	Set the count time (U2_Inactive_timer) transiting from U1 to U2. Value=Actual time/(USB_CLK× 0x10000)
LINK_CNTL	W	1h	Link Layer Control Register. Instructed if “1” is written. Set 1-bit only(one-hot) at same time. No described bits should be written “0”
	GO_RXDET	[0]	Make USB bus state (LTSSM) transit from SS_Disable to Rx_Detect_Active (Operation start). LINK_CNFG[USB_VALID] must be ON before.
	GO_RCOV	[3]	Transit from Active_U0 state to Recovery state.
	GO_BH_PORT_RESET	[7]	Generate BH_PORT_RESET(WamReset).
	GO_PMD_NUM	[9:8]	Set Power Mode level for transiting by GO_PMD. 01:U1, 10:U2, 11;U3
	GO_PMD	[10]	Request transition to Power Mode. Success or Not are depend on Host side status.
	GO_U0	[11]	Order to return from Power Mode (Active_U1/U2/U3).
LINK_IRQE	R/W	2h	Link Layer Interrupt Register.
	IRQ	[0]	Interruption requesting. Read Only.
	ENB	[1]	When IRQ=1, Interrupt signal to external (referring O_EP_IRQ. I/O signal) is ON.
LINK_LTSSM	R	3h	Link Layer Status Register.
	RCOV	[6:0]	The reason why Link Layer transit to Recovery state. Kept during LINK_LTSSM[7] is ON.
	IRQ_FACTOR	[15:7]	Cause of Link Layer interruption. Status of Link Layer is possible to change without command from CPU. When any bit is ON, LINK_IRQE[IRQ] is also ON. Clear by 1 write to the bit.
	LTSSM	[28:24]	Current state of Link Layer. The states are compliant with USB3.0 specification. Code allocation , which is implementation specific, is not disclosed in here.
	VBUS_OFF	[31]	Current VBUS is OFF.

Register Name	R/W	Offset	Description
Protocol Layer Register [ADDRESS=C_BASEADDR+PRTE_BASE+OFFSET]			
PRTE_CNFG	R/W	0h	Protocol Layer Configuration Register
PRTE_CNTL	W	1h	Protocol Layer control register. Instructed if “1” is written. Set 1-bit only(one-hot) at same time. No described bits should be written “0”
ARBT_RESET		[0]	Arbiter reset of Protocol Layer. For debugging.
PRTE_IRQE	R/W	2h	Protocol Layer Interrupt Enable Register.
RP0		[0]	Enable interruption from RP0.
RRI		[7:1]	Enable interruption from RPI1～7. [1]: RPI1, [7]:RPI7
MPP		[16]	Enable interruption from MPP.
RPO		[23:17]	Enable interruption from RPO1～7. [17]:RPO1, [23]RPO7
PRTE_IRQ	R	3h	Protocol Layer interrupt Register.
RP0		[0]	Under requesting interruption from RP0. When applicable IRQE bit is 1, interrupt signal for external will be ON.
RPI		[7:1]	Under requesting interruption from RPI1～7. When applicable IRQE bit is 1, interrupt signal for external will be ON. [1]: RPI1, [7]RPI7
MPP		[16]	Under requesting interruption from MPP. When applicable IRQE bit is 1, interrupt signal for external will be ON.
RPO		[23:17]	Under requesting interruption from RPO1～7. When applicable IRQE bit is 1, interrupt signal for external will be ON. [17]: RPO1, [23]RPO7
DMA_BASE	R/W	4h	DMA Base Register.
BASE		[31:28]	Upper bit of memory address for DMA.

Register Name	R/W	Offset	Description
Poute Point Zero (RP0) Register [ADDRESS=C_BASEADDR+XP_BASE+(0x80×0)+0x000+OFFSET]			
RP0_CNFG	R/W	00h	RP0 Configuration Register.
VALID		[0]	RP operation enables. Must setup transmission existence, data length and direction before operation. After sending SetupDP packet, transit to each state following response from device side.
RP0_CNTL	R/W	01h	RP0 control register. Show status and update all bits of [31:16]. Requesting interrupt if any bit in [13:0] is ON to requesting interruption. [23:16] show detected events. [31:24] show transitional events.
RP0_CLR	W	02h	RP0 control register clear. Clear applicable bit by 1 writing.
RP0_SET	W	03h	RP0 control register set. Set applicable bit by 1 writing.
COMP		[0]	COMP state (transmission completed). Return to IDLE when clearing.
SETUP_ERDY		[1]	ACK(NumP=0) is returned for SetupDP. Or NRDY is returned for data transmission (IN or OUT). Sending ERDY from Device side to transit to WAIT_INOUT or WAIT_STT(STATUS sending). Transit to COMP state when clearing. (Normally automatically)
WAIT_INOUT		[2]	Waiting software restart before data transmission. Clearing it to start data transmission. (Some devices cannot response if it responses too fast. Software side needs waiting a while.)
WAIT_STT		[3]	Waiting software restart before STATUS sending. Clearing it to start STATUS sending. (Some devices cannot response if it responses too fast. Software side needs waiting a while.)
WAIT_ERDY		[10]	NRDY is returned for STATUS sending. Sending ERDY from Device side to transit to WAIT_STT (STATUS sending). Transit to COMP state when clearing. (Normally automatically)
WAIT_RECV		[14]	Waiting for any packet. Valid when BUSY is ON.
BUSY		[15]	It is ON except (IDLE state or [13:0] state). When WAIT_RECV is ON, Clearing this bit to return to IDLE. And return to IDLE state when both this bit and [14] are cleared at same time.
ERROR		[16]	Receive STALL packet or detect some error.
OVER		[17]	Traffic of Device is larger than specified in RP0_DLEN. (Host side is fewer.)
UNDER		[18]	Traffic of Device is fewer than specified in RP0_DLEN. (Host side is more.)
RETRY		[24]	Error exists in received DP packet. Or retry returns for DP sending (OUT operation or SetupDP sending).
RECV_STALL		[25]	Receive STALL packet.
RECV_NRDY		[26]	Receive NRDY packet.
RECV_ERDY		[27]	Receive ERDY packet.
RP0_DLEN	R/W	04h	RP0 data length setting register.
DLEN		[20:0]	Length of actual transferred data.
OUTRDY		[30]	OUT transmission (from Host to Device) ready.
INRDY		[31]	IN transmission (from Device to Host) ready.
RP0_PLEN	R	05h	RP0 data length result register
PLEN		[20:0]	Length of actual transferred data.
OUTRDY		[30]	OUT transmission (from Host to Device) ready.（copy of DLEN register）
INRDY		[31]	IN transmission (from Device to Host) ready.（copy of DLEN register）
RP0_BFFR	R/W	06h	RP0 data length buffer (memory address) register
BADDR		[27:8]	Start address of the memory for data input/output.
RP0_SEQN	R/W	07h	RP0 sequence number register
SEQN		[4:0]	Current sequence number of packet. Writable, but normally no need.
RP0_DEST	R/W	08h	RP0 destination setting register
DEV		[10:4]	Device address.
RUT		[19:12]	Root String of hub. LSB 4bits each. (Refer USB3.0 specification chapter 8.9)

RP0_SETUS0	R/W	09h	RP0 setup data register 0
RP0_SETUS1	R/W	0Ah	RP0 setup data register 1
Setup0/1		[31:0]	Sending contents by SetupDP packet.

Register Name	R/W	Offset	Description
Root Point Out (RPO) Register			[ADDRESS=C_BASEADDR+XP_BASE+(0x80×N)+0x000+OFFSET]
RPO_CNFG	R/W	00h	RPO Configuration Register.
VALID		[0]	RP operation valid. Must set data length before operation.
ISOCHR		[1]	Isochronous mode transmission
BURST		[18:16]	Set burst length. 4~1.
FIFO_REQ		[30]	Arbitration request of sending FIFO of RPO. For debugging. Read only.
FIFO_LOC		[31]	The mode which RPO occupies FIFO. For debugging.
RPO_CNTL	R/W	01h	RPO control register. Show status and update all bits of [31:16]. Requesting interrupt if any bit in [13:0] is ON to request interruption. [23:16] show detected events. [31:24] show transitional events.
RPO_CLR	W	02h	RPO control register clear. Clear applicable bit by 1 writing.
RPO_SET	W	03h	RPO control register set. Set applicable bit by 1 writing.
COMP		[0]	COMP state (transmission completed). Clear it to return to IDLE or transit to COMP_ERDY.
COMP_ERDY		[8]	The end of ACK after transmission is NumP=0. When during FLOW control state and ERDY is received form Device, transit to IDLE and enable next transmission. In case of no EDRY receiving from Device, transit to IDLE for next transmission by software timeout (because some device do unnecessary FLOW control). This state does not have interrupt.
WAIT_ERDY		[10]	NRDY is returned for data transmission. After receiving ERDY from Device and clearing, data transmission will start again. In case of clear before EDRY receiving, keep the state. Clearing VAILD to transit to IDLE.
WAIT_RECV		[14]	Waiting for any packet. Valid when BUSY is ON.
BUSY		[15]	It is ON except (IDLE state or [13:0] state). When WAIT_RECV is ON, Clearing this bit to return to IDLE. And return to IDLE state when both this bit and [14] are cleared at same time.
ERROR		[16]	STALL packet is received or some abnormal is detected.
OVER		[17]	Traffic of Device is larger (Host side is fewer). Isochronous mode only.
FLOW		[19]	Under flow control.
RETRY		[24]	Retry is returned after DP sending.
RECV_STALL		[25]	STALL packet is received.
RECV_NRDY		[26]	NRDY packet is received.
RECV_ERDY		[27]	ERDY packet is received.
RECV_ACK0		[28]	ACK with NumP=0 is received.
RPO_DLEN	R/W	04h	RPO data length setting register
DLEN		[20:0]	Data length.
ZERO		[31]	In case data length is integral multiples of 1KB, send 0 byte length data packet.
RPO_PLEN	R	05h	RPO data length result register
PLEN		[20:0]	Actual transferred data length. Except 0 means a data is received.
ZERO		[31]	In case data length is integral multiples of 1KB, send 0 byte length data packet.
RPO_BFFR	R/W	06h	RPO data length buffer (memory address) register
BADDR		[27:8]	Start address of the memory for data input/output.
RPO_DEST	R/W	08h	RPO destination setting register
EPN		[3:0]	End Point number (EPO)

	DEV	[10:4]	Device address.
	RUT	[19:12]	Root String of hub. LSB 4bits each. (Refer USB3.0 specification chapter 8.9)
EPO_SEQN	R/W	07h	RPO sequence number register
	COMP_SEQN	[4:0]	Sequence number of transferred packet (ACK sent).
	BFR_SEQN	[12:8]	Sequence number of stored packet in buffer (memory).
	TRN_SEQN	[20:16]	Sequence number of sent packet.
	NUM_SEQN	[28:24]	Sequence number of packet which credit is assigned by ACK.

Register Name	R/W	Offset	Description
Root Point In (RPI) Register [ADDRESS=C_BASEADDR+XP_BASE+(0x80×N)+0x000+OFFSET]			
RPI_CNFG	R/W	00h	RPI Configuration Register.
	VALID	[0]	RP operation valid. Must set data length before operation.
	ISOCHR	[1]	Isochronous mode transmission.
	SW_WAIT_ERDY	[2]	Interrupt happen at WAIT_EDRY state and software handle it. 0: OFF(Not happen), 1:ON(happen)
	BURST	[18:16]	Set burst length. Set 4~1.
	FIFO_REQ	[30]	Arbitration request of sending FIFO of RPI. For debugging. Read only.
	FIFO_LOC	[31]	The mode which RPI occupies FIFO. For debugging.
RPI_CNTL	R/W	01h	RPI control register. Show status and update all bits of [31:16]. Requesting interrupt if any bit in [13:0] is ON to request interruption. [23:16] show detected events. [31:24] show transitional events.
RPI_CLR	W	02h	RPI control register clear. Clear applicable bit by 1 writing.
RPI_SET	W	03h	RPI control register set. Set applicable bit by 1 writing.
	COMP	[0]	COMP state (transmission completed). Clear it to return to IDLE or transit to COMP_ERDY.
	COMP_ERDY	[8]	The end of DP after transmission is EOB=1. When during FLOW control state and ERDY is received form Device, transit to IDLE and enable next transmission. In case of no EDRY receiving from Device, transit to IDLE for next transmission by software timeout (because some device do unnecessary FLOW control). This state does not have interrupt.
	WAIT_ERDY	[10]	NRDY is received for data transmission or received DP is EOB=1. After receiving ERDY from Device and clearing, data transmission will start again. In case of clear before EDRY receiving, keep the state. Clearing VAILD to transit to IDLE.
	WAIT_RECV	[14]	Waiting for any packet. Valid when BUSY is ON.
	BUSY	[15]	It is ON except (IDLE state or [13:0] state). When WAIT_RECV is ON, Clearing this bit to return to IDLE. And return to IDLE state when both this bit and [14] are cleared at same time.
	ERROR	[16]	STALL packet is received or some abnormal is detected.
	OVER	[17]	Traffic of Device is larger than DLEN (Host side is fewer).
	UNDER	[18]	Traffic of Device is fewer than DLEN. (Host side is more.)
	FLOW	[19]	Under flow control.
	FLOW2	[20]	Under flow control.
	RETRY	[24]	Under retrying.
	RECV_STALL	[25]	STALL packet is received.
	RECV_NRDY	[26]	NRDY packet is received.
	RECV_ERDY	[27]	ERDY packet is received.
RPI_DLEN	R/W	04h	RPI data lendth setting register
	DLEN	[20:0]	Data length.
RPI_PLEN	R	05h	RPI data length result register
	PLEN	[20:0]	Actual transferred data length. Except 0 means a data is sent.
RPI_BFFR	R/W	06h	RPI data length buffer (memory address) register
	BADDR	[27:8]	Start address of the memory for data input/output.
RPI_DEST	R/W	08h	RPI destination setting register
	EPN	[3:0]	End Point number. (EPI)
	DEV	[10:4]	Device address.
	RUT	[19:12]	Root String of hub. LSB 4bits each. (Refer USB3.0 specification chapter 8.9)
RPI_SEQN	R/W	07h	RPI sequence number register

	COMP_SEQN	[4:0]	Sequence number of transferred packet (ACK received).
	BFR_SEQN	[12:8]	Sequence number of taken packet from buffer (memory).
	RCV_SEQN	[20:16]	Sequence number of received packet.

Register Name	R/W	Offset	Description
Multi Purpose Point (MPP) Register [ADDRESS=C_BASEADDR+XP_BASE+(0x80×0)+0x400+OFFSET]			
MPP_CNFG	R/W	00h	MPP Configuration Register.
TRNS		[0]	Send packet from MPP. It is cleared when completed. When being Active_U0, send Port Capabilities, Port Configuration Response with automatically ON/OFF.
MPP_THD0	R/W	04h	MPP sending packet 0
MPP_THD1	R/W	05h	MPP sending packet 1
THD0/1		[31:0]	Data of sending packet
MPP_RHD0	R/W	06h	MPP receiving packet 0.
MPP_RHD1	R/W	07h	MPP receiving packet 1.
RHD0/1		[31:0]	Data of receiving packet
MPP_RCVD	R/W	08h	EPI sequence number register
MISC_RCNT		[3:0]	When Receiving packet, it will count up +1. After 0xF, return to 0x0. Writable.
MISC_RCVD_H		[7]	Receive any packet. ON: interruption request. Writable
PCFG_RCVD		[30]	After Active_U0, receive Port Configuration.
PCAP_RCVD		[31]	After Active_U0, receive Port Capabilities.

Register Map

Core register location connecting I_xxx_REG_RE/WE[..] in sequence of address is shown as figure 5.

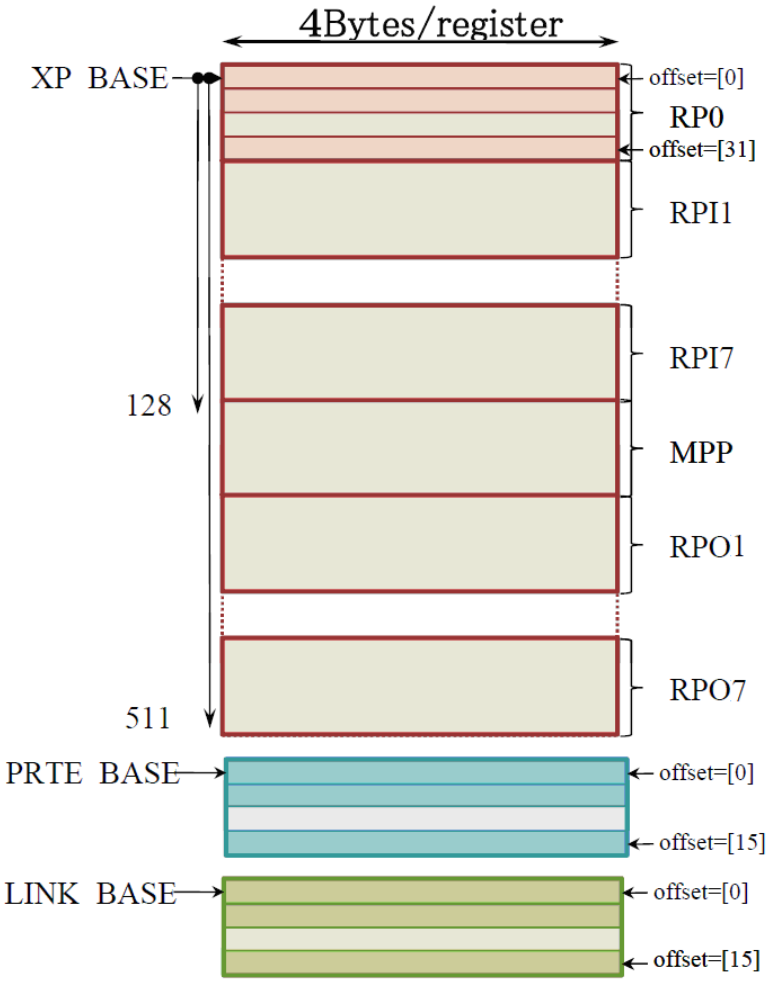


Figure 5 : Address map of IP-Core internal register

IP Core control step

This is the summary of the IP Core control step.

USB bus initialization

When device is connected with USB bus, bus operation is started. The step until U0 state is as following.

① Check ON of VBUS by control register (such as LINK_CNFG[EXT_CNT])

② Operation ready by LINK_CNFG[USB_VALID].

③ Order state transition to Rx_Detect_Active state by LINK_CNTL[GO_RXDET]

④ Automatic link if device side is also operation ready

If it is Active_U0 state by LINK_LTSSM[LTSSM], the linking is successful. If it returns to SS_disable, go to② and retry.

Until USB device configured

The step until USB device configured state by control transmission by RP0 after linking is successful, is as following.

① RP0_CNFG[VALID] makes RP0 operation ready

- The first SetupDP will be SET_ADDRESS, so the address is 0 by RP0_DEST[DEV]
- Send SetupDP packet to receive ACK. SET_ADDRESS is no data transmission to be WAIT_STT state
- Send STATUS packet and send final ACK packet (RESULT)

② Set address by RP0_DEST[DEV]. Device becomes Address state

③ After that, operate control transmission such as SetupDP packet several times

- In case that control transmission has data transmission, keep receiving buffer at memory (IN transmission) or set sending data to memory (OUT transmission) and order to RP0_BFFR and RP0_DLEN
- Send SetupDP packet to receive ACK. In case with data transmission (IN/OUT), it becomes WAIT_INOUT state
- After complete the transmission, send STATUS packet. After that, same as no data transmission

④ Finally send SET_CONFIGURATION to become Configured state

BULK_IN, BULK_OUT transmission of USB device

This is the transmission by using BULK_IN(RPI), BULK_OUT(RPO) after device is configured state. Each transmission are differenet for every device class. Following step is for BULK_IN, but the step for BULK_OUT is also nearly same.

① Set VALID to ON and set burst length by RPO_CNFG

② Keep receiving buffer in memory and order ot RPO_BFFR and RPO_DLEN
Set estimation data length (or more) to DLEN

③ If something are received, PLEN shows receiving data length. So check receiving buffer. Clear COMP state and return to ②.

Core verification method

USB3.0 (Host) IP Core logic can be verified by Altera evaluation board together with AB08-USB3HSMC adapter board that enables real operation check. Please ask AB08-USB3HSMC adapter board availability to Design Gateway.

Recommended design skill

To implement this IP Core to user circuit, technical skill about Qsys and NiosII are required. And general knowledge about high-speed interface standard are also recommended. Moreover knowledge of the protocol specification of USB3.0 standard for hardware debugging and USB device specification for software development and debugging are required.

Ordering Information

This product is available directly from Design Gateway. Please contact Design Gateway for pricing and additional information about this product using the contact information on the front page of this datasheet.

Revision History

Revision	Date	Description
1.0	10-Feb-2015	Release 1st English version
1.1E	09-Mar-2015	Fixed and updated some description
1.2E	22-Apr-2015	Added some feature in IP-core register
1.3E	13-May-2015	Add CycloneV/ArriaV device support
1.4E	26-May-2015	Fixed bit position of WAIT_ERDY [2]->[10], and COMP_ERDY [1] -> [8] at RPO/RPI