AN 702: Interfacing a USB PHY to the Hard Processor System USB 2.0 OTG Controller

2017.09.22

an-702 Subscribe Send Feedback

The Arria® V Hard Processor System (HPS) and Cyclone® V HPS each provide two USB On-the-Go (OTG) controllers. Each USB 2.0 OTG controller supports a single USB port connected through a USB 2.0 Transceiver Macrocell Interface Plus (UTMI+) Low Pin Interface (ULPI) compliant PHY.

When interfacing your design to a USB PHY, it is important to do timing analysis to ensure that the interface between the USB controller and USB PHY works reliably across a range of process, voltage and temperature (PVT) variations.

Related Information

- **USB 2.0 OTG Controller**
  Chapter in the *Arria V Hard Processor System Technical Reference Manual*
- **USB 2.0 OTG Controller**
  Chapter in the *Cyclone V Device Hard Processor System Technical Reference Manual*

ULPI Signals

Table 1: Signals Included in the ULPI Interface

Signal	Description
CLK	Interface clock—All signals are synchronous to the clock.
DATA[7:0]	Data bus—Driven low by the controller during idle. The controller starts a transfer by sending a non-zero pattern. The PHY must assert DIR before using the data bus. Every time DIR toggles, DATA must be ignored for one clock cycle (the turnaround cycle).
DIR	Direction of the data bus—By default, DIR is low and the PHY listens for non-zero data from the controller. The PHY asserts DIR to get control of the data bus.
NXT	Next data—The PHY drives NXT high to throttle the data bus.
STP	Stop data—The controller drives STP high to signal the end of the data stream. The controller can also drive STP high to request data bus access from the PHY.

Related Information

- **Arria V Hard Processor System Technical Reference Manual**
  For more information on the ULPI interface
- **Cyclone V Device Hard Processor System Technical Reference Manual**
  For more information on the ULPI Interface
- **The Quartus II TimeQuest Timing Analyzer**

HPS USB Controller Timing Characteristics

The Arria V and Cyclone V Hard Processor System USB controllers have been characterized across a range of PVT variations. Detailed USB timing information appears in the Arria V and Cyclone V datasheets.

Related Information

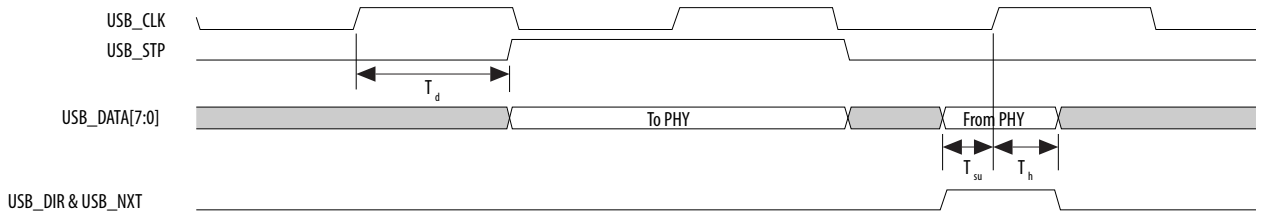
- **Cyclone V Device Datasheet**
- **Arria V GX, GT, SX, and ST Device Datasheet**

USB Controller Timing Requirements

Table 2: HPS USB Controller Media Access Controller (MAC) Timing Requirements

Symbol	Description	Min	Typ	Max	Units
T_{clk}	USB `CLK` clock period	-	16.67	-	ns
MAC T_d	`CLK` to `USB_STP/USB_DATA[7:0]` output delay	4.4	-	11.0	ns
MAC T_{su}	Setup time for `USB_DIR/USB_NXT/USB_DATA[7:0]`	2.0	-	-	ns
MAC T_h	Hold time for `USB_DIR/USB_NXT/USB_DATA[7:0]`	1.0	-	-	ns

USB Controller Setup and Hold Relationships



PHY Selection

The timing characteristic of ULPI PHYs vary across the spectrum of available devices.

Arria V and Cyclone V USB 2.0 OTG controllers support the following clock modes:

- Output clock mode—the PHY drives the clock to the controller
- Input clock mode—an external clock source on the board or a clock input sourced from the FPGA fabric drives the PHY.

All ULPI PHYs support output clock mode.

Newer ULPI PHYs support input clock mode. This mode compensates for timing mismatches between the PHY and the controller.

Related Information

- **Output Clock Mode** on page 7
- **Input Clock Mode** on page 8

USB PHY Timing Characteristics

Before selecting a USB PHY and clock mode of operation, you should perform a timing analysis of the USB controller and PHY.

If the USB PHY supports input and output clock modes with different timing characteristics, perform a timing analysis of both modes before making a selection. This document includes a detailed timing analysis example of a MicroChip USB3300 PHY in output clock mode.

Table 3: USB PHY Timing Characteristics

The PHY timing characteristics listed below are from the MicroChip USB3300 PHY that populates both the Arria V and Cyclone V SoC Development Boards. On the development board, this USB PHY operates in output clock mode.

Note: In the Arria V and Cyclone V SoC devices, the USB controller does not support PHYs using link power management (LPM) mode. It is recommended that designers use the MicroChip SB3300 PHY device that is verified on the development board.

Symbol	Description	Min	Typ	Max	Units
PHY T_{su}	PHY setup time for `USB_STP/USB_DATA[7:0]`	5.0	-	-	ns
PHY T_h	PHY hold time for `USB_STP/USB_DATA[7:0]`	0	-	-	ns
PHY T_d	Output delay for `USB_DIR/USB_NXT/USB_DATA[7:0]`	2.0	-	5.0	ns
ClkTrace T_d	Clock Trace delay	0.05	-	0.1	ns
DTrace T_d	Data Trace delay	0.05	-	0.1	ns
Clock T_u	Clock Source Uncertainty	-	0.3	-	ns

The clock source uncertainty (Clock T_u) parameter is a board-level guard band factor that models period uncertainty on the PHY's clock caused by clock source inaccuracies and jitter. You may modify this value based on your application. The following guidelines apply when using this parameter:

- For setup analysis, use Clock T_u to model period uncertainty in the launch-to-latch edge setup relationship.
- Do not use Clock T_u for hold analysis because the launch and latch edges are the same physical clock edge in time.

For this timing analysis example, assume the trace lengths of the `DATA[7:0]`, `STP` and `NXT` signals are matched when verifying if the USB timing is met off-chip.

Related Information

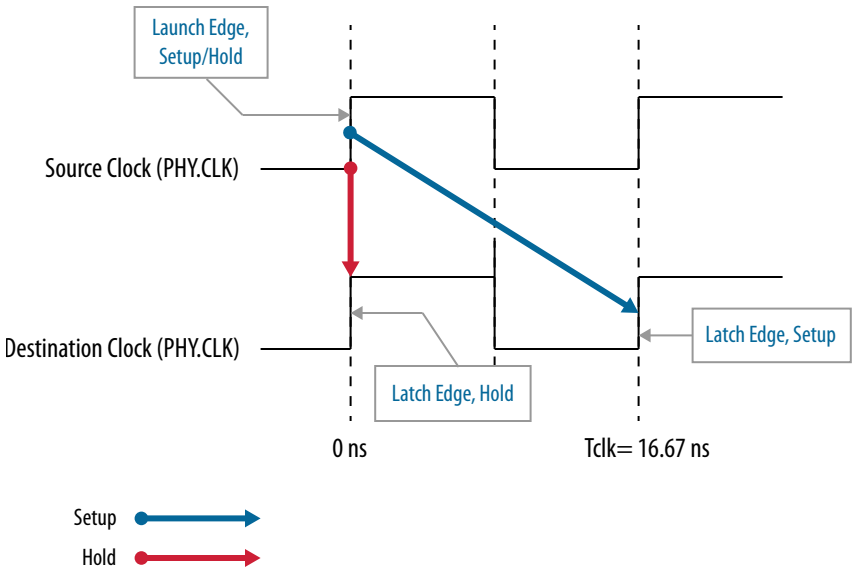
- **Arria V SoC Development Board Reference Manual**
  For more information about the Arria V SoC Development Board
- **Cyclone V SoC Development Board Reference Manual**
  For more information about the Cyclone V SoC Development Board

USB Setup and Hold Relationships

The diagram below shows the setup and hold relationships between the USB controller and PHY. The blue arrow represents the setup relationship. Data that launches from the rising edge of the PHY clock is latched on the rising edge of the next clock cycle.

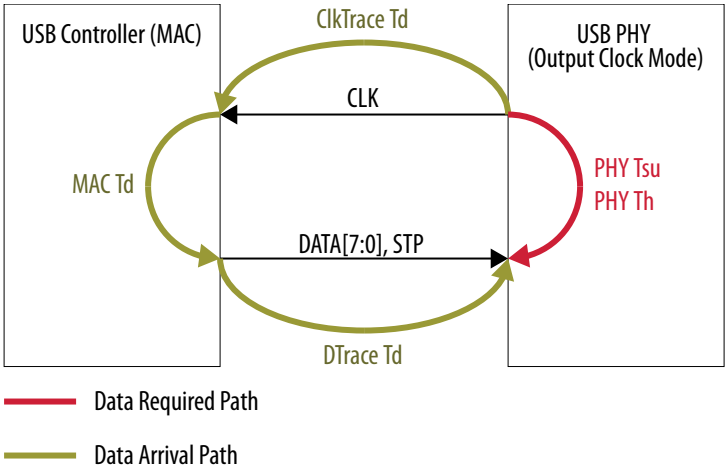
The red arrow represents the hold relationship. Data must be held at least until the rising clock edge of where the data is latched.

Figure 1: USB MAC Controller to PHY and USB PHY to MAC Controller Setup and Hold Relationship



USB Controller to PHY Setup and Hold Timing Arcs

Figure 2: USB Controller to PHY Setup and Hold Timing Arcs



USB Controller to PHY Setup Timing Analysis

To determine if your configuration meets setup timing requirements at the PHY, you must calculate the worst case setup time to verify that it falls within limits. To meet setup time requirements, the data arrival time must be less than or equal to the time at which data is required. The inequality below evaluates the data arrival and data required time using values in USB Controller and PHY timing characteristic **Table 2** and **Table 3**, respectively. By replacing each side of the inequality with the timing expressions that represent data arrival and data required time, you can verify if the setup timing requirements are met.

$$\text{Data Arrival} \leq \text{Data Required}$$

$$\text{Launch\_Edge} + \text{ClkTrace } T_d\text{\_max} + \text{MAC } T_d\text{\_max} + \text{DTrace } T_d\text{\_max} \leq (\text{Latch\_Edge} - \text{Clock } T_u) - \text{PHY } T_{su}$$

If you assume that Launch_Edge= 0 ns and Latch_Edge= T_{clk}, then the equation can be simplified:

$$\text{ClkTrace } T_d\text{\_max} + \text{MAC } T_d\text{\_max} + \text{DTrace } T_d\text{\_max} \leq (T_{clk} - \text{Clock } T_u) - \text{PHY } T_{su}$$

Isolate PHY T_{su} by moving parameter terms to one side of the inequality. Replace the parameters with specific timing characteristic values to determine if the worst case setup time for your configuration is greater than or equal to the minimum required setup time:

$$(T_{clk} - \text{Clock } T_u) - \text{ClkTrace } T_d\text{\_max} - \text{MAC } T_d\text{\_max} - \text{DTrace } T_d\text{\_max} \geq \text{PHY } T_{su}$$

$$(16.67 - 0.3) - 0.1 - 11.0 - 0.1 \geq 5.0$$

$$5.17 \text{ ns} \geq 5.0 \text{ ns}$$

USB Controller to PHY Hold Timing Analysis

To determine if your configuration meets hold timing requirements at the PHY, you must calculate the worst case hold time to verify that it falls within limits. To meet hold time requirements, the data must arrive and be held for longer than the data hold requirement. The inequality below evaluates the data arrival and data required time using values in USB Controller and PHY timing characteristic **Table 2** and

Table 3, respectively. By replacing each side of the inequality with the timing expressions that represent data arrival and data required time, you can verify if the hold timing requirements are met.

$$\text{Data Arrival} \geq \text{Data Required}$$

$$\text{Launch\_Edge} + \text{ClkTrace } T_d\text{\_min} + \text{MAC } T_d\text{\_min} + \text{DTrace } T_d\text{\_min} \geq \text{Latch\_Edge} + \text{PHY } T_h$$

If you assume that Launch_Edge= 0 ns and Latch_Edge= 0 ns, then the equation can be simplified and you can verify that the hold time is within limits:

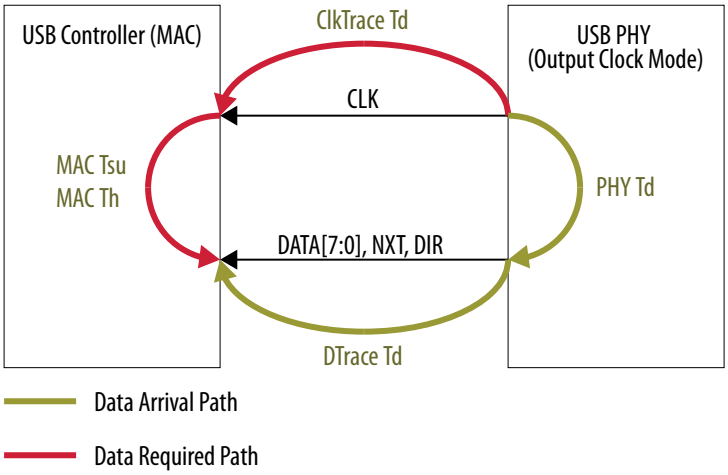
$$\text{ClkTrace } T_d\text{\_min} + \text{MAC } T_d\text{\_min} + \text{DTrace } T_d\text{\_min} \geq \text{PHY } T_h$$

$$0.05 + 4.4 + 0.05 \geq 0$$

$$4.5 \text{ ns} \geq 0 \text{ ns}$$

USB PHY to Controller Setup and Hold Timing Arcs

Figure 3: USB PHY to Controller Setup and Hold Timing Arcs



USB PHY to Controller Setup Timing Analysis

To determine if your configuration meets setup timing requirements at the USB Controller, you must calculate the worst case setup time to verify that it falls within limits. To meet setup time requirements, the data arrival time must be less than or equal to the time at which data is required. The inequality below evaluates the data arrival and data required time using values in USB Controller and PHY timing characteristic Table 2 and Table 3, respectively. By replacing each side of the inequality with the timing expressions that represent data arrival and data required time, you can verify if the setup timing requirements are met.

$$\text{Data Arrival} \leq \text{Data Required}$$

$$\text{Launch\_Edge} + \text{PHY } T_d\text{\_max} + \text{DTrace } T_d\text{\_max} \leq (\text{Latch\_Edge} - \text{Clock } T_u) + \text{ClkTrace } T_d\text{\_min} - \text{MAC } T_{su}$$

If you assume that the Launch_Edge = 0 ns and the Latch_Edge = T_{clk}, then the equation can be simplified:

$$PHY\ T_{d}\_max + DTrace\ T_{d}\_max \leq (T_{clk} - Clock\ T_{u}) + ClkTrace\ T_{d}\_min - MAC\ T_{su}$$

By moving terms to one side, you can determine if the worst case setup time for your configuration is greater than or equal to the minimum required setup time:

$$(T_{clk} - Clock\ T_{u}) + ClkTrace\ T_{d}\_min - PHY\ T_{d}\_max - DTrace\ T_{d}\_max \geq Mac\ T_{su}$$

$$(16.67 - 0.3) + 0.05 - 5.0 - 0.1 \geq 5.0$$

$$11.32\ ns \geq 5.0\ ns$$

USB PHY to Controller Hold Timing Analysis

To determine if your configuration meets hold timing requirements at the USB controller, you must calculate the worst case hold time to verify that it falls within limits. To meet hold time requirements, the data must arrive and be held for longer than the data hold requirement. The inequality below evaluates the data arrival and data required time using values in USB Controller and PHY timing characteristic **Table 2** and **Table 3**, respectively. By replacing each side of the inequality with the timing expressions that represent data arrival and data required time, you can verify if the hold timing requirements are met.

$$Data\ Arrival \geq Data\ Required$$

$$LaunchEdge + PHY\ Td\_min + DTrace\ Td\_min \geq LatchEdge + ClkTrace\ Td\_max + MAC\ Th$$

If you assume that the LaunchEdge = 0 ns and the LatchEdge = 0 ns, then the equation can be simplified and you can verify that the hold time is within limits:

$$PHY\ Td\_min + DTrace\ Td\_min - ClkTrace\ Td\_max \geq MAC\ Th$$

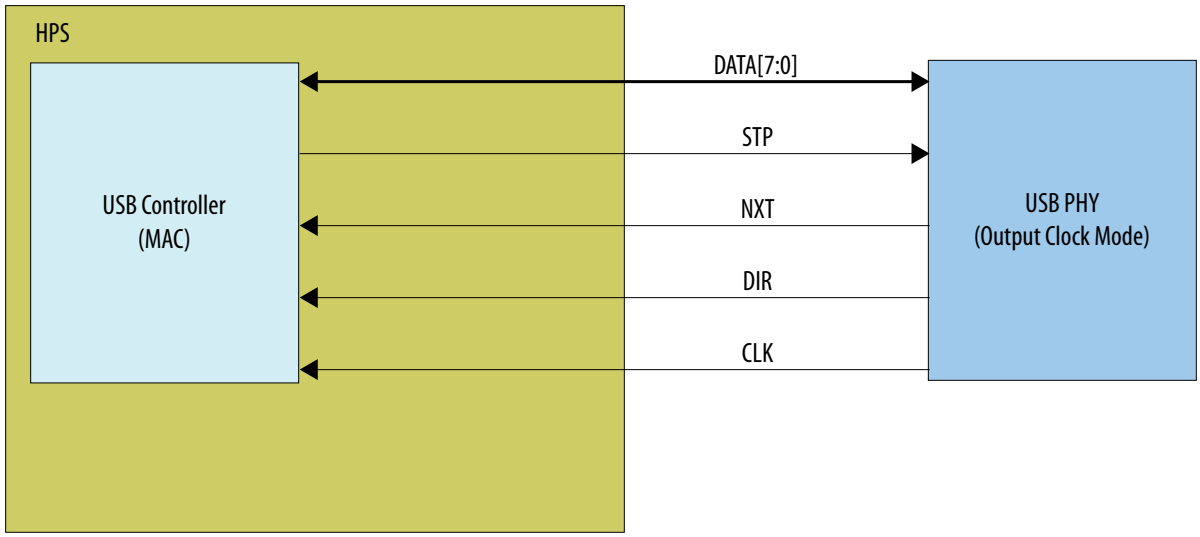
$$2.0 + 0.05 - 0.1 \geq 1.0$$

$$1.95\ ns \geq 1.0\ ns$$

Output Clock Mode

In output clock mode, the clock is generated by the USB PHY. All signals are synchronized to this clock. To use this mode of operation, you must configure the USB Controller PHY interface mode for "SDR with PHY clock output mode" in the **Peripheral Pins** tab of **HPS Parameters** window in Platform Designer (Standard). This mode of operation configures the USB Controller clock pin to operate in an input mode.

Figure 4: USB PHY in Output Clock Mode



Related Information

Configuring the HPS USB 2.0 OTG Controller on page 10
Refer to this section for more information on how to configure the USB interface in Platform Designer (Standard).

Input Clock Mode

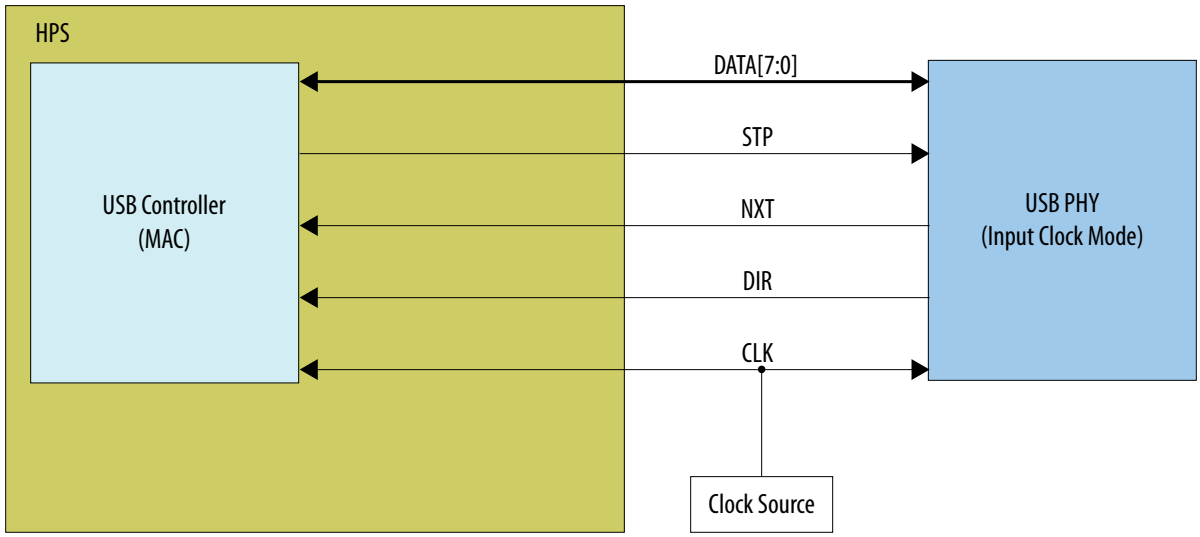
In input clock mode, the PHY receives a clock from an external source. All signals are synchronized to the clock. In this mode, a PLL in the FPGA or an external source generates the clock.

Note: For systems where the HPS must be operational before the FPGA fabric is configured, an external clock source should be used to drive the USB PHY clock. By using an external clock source, the FPGA fabric is not required to be configured before the HPS.

External Clock Source

Although the USB PHY is configured in input clock mode, the clock source is still driven into the USB controller as an input to the SoC device. As a result, you must configure the USB controller for "SDR with PHY clock output mode" in the **Peripheral Pins** tab of the **HPS Parameters** window of Platform Designer (Standard). This mode of operation configures the USB Controller clock pin to operate as an input.

Figure 5: USB PHY in Input Clock Mode with External Clock Source



Related Information

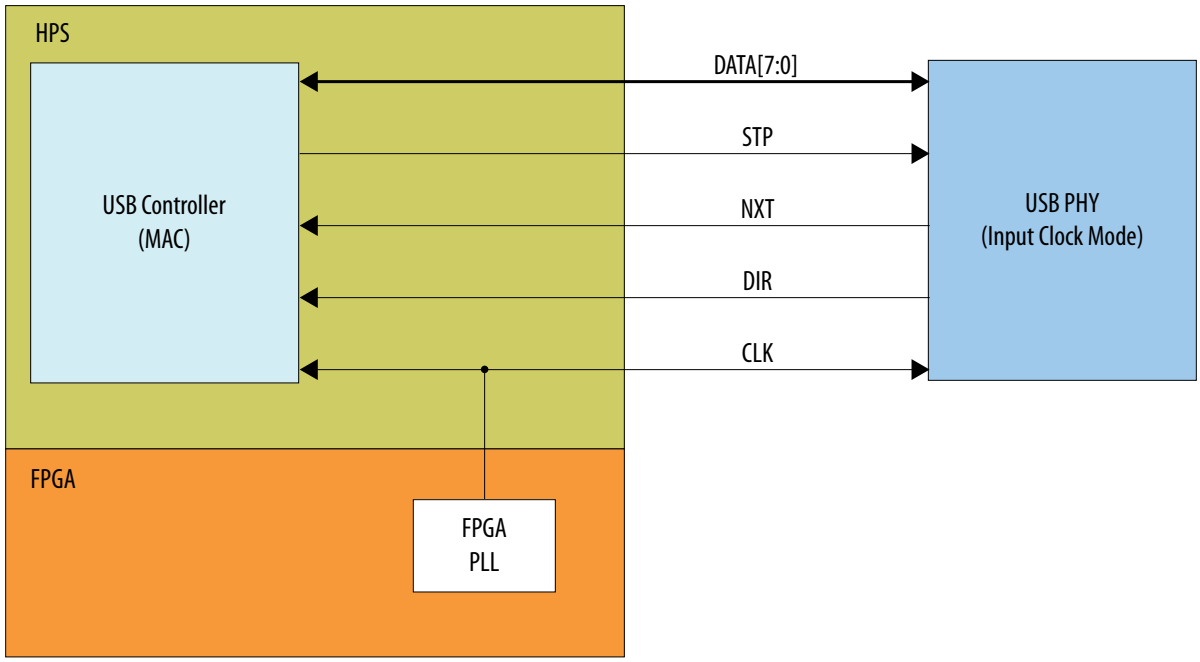
Configuring the HPS USB 2.0 OTG Controller on page 10
Refer to this section for more information on how to configure the USB interface in Platform Designer (Standard).

FPGA Clock Source

When the FPGA fabric drives the USB Controller clock output the USB interface requires the use of a loan I/O pin instead of the typical USB clock input pin.

User logic in the FPGA drives a clock signal, typically derived from a PLL, into the loan I/O assigned to the USB controller. This clock signal routes into the USB controller and externally to the USB PHY. This configuration provides a common clock source for both the USB controller and PHY much like when the PHY is configured for input clock mode with an external clock source. Because this mode of operation differs from the previous examples, you must configure the USB controller for **SDR with PHY clock input mode** in the **Peripheral Pins** tab of the **HPS Parameters** window of Platform Designer (Standard).

Figure 6: USB PHY in Input Clock Mode with FPGA PLL Clock Source



Note: When implementing input clock mode with an FPGA clock source, the FPGA must be configured prior to USB interface operation. This implementation can impact embedded software significantly and must be considered carefully before selecting the input clock mode with FPGA clock source. Using an external clock source (as shown in **Figure 5**) can accomplish the same objective without necessarily affecting embedded software.

Related Information

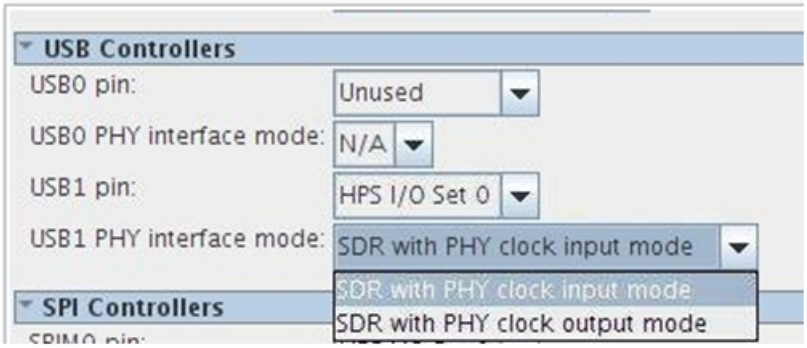
- **External Clock Source** on page 8
- **Configuring the HPS USB 2.0 OTG Controller** on page 10
  Refer to this section for more information on how to configure the USB interface in Platform Designer (Standard).

Implementing Input Clock Mode with FPGA Clock Source

The following example details how to interface the FPGA to the HPS USB Controller and the external USB PHY by using a Loan I/O in the HPS.

Configuring the HPS USB 2.0 OTG Controller

1. In the **Peripheral Pins** tab of the Hard Processor System parameter editor, select a USB controller by setting either **USB0 pin** or **USB1 pin** to one of the available HPS I/O pin sets.
2. Select the PHY interface mode in the corresponding list, **USB0 PHY interface mode** or **USB1 PHY interface mode**. Set the mode to **SDR with PHY clock input mode**.



Select the Controller Clock as Loan I/O

On the **Peripheral Pins** tab, scroll down to the **Peripherals Mux Table** to select the USB clock pin as loan I/O. This setting allows a clock from a PLL in the FPGA to connect to the USB controller.

mux_select_2	mux_select_3	* GPIO	* LoanIO
	EMAC0.TX_CLK (Set0)	GPIO00	LOANIO00
USB1.D0 (Set0)	EMAC0.TXD0 (Set0)	GPIO01	LOANIO01
USB1.D1 (Set0)	EMAC0.TXD1 (Set0)	GPIO02	LOANIO02
USB1.D2 (Set0)	EMAC0.TXD2 (Set0)	GPIO03	LOANIO03
USB1.D3 (Set0)	EMAC0.TXD3 (Set0)	GPIO04	LOANIO04
USB1.D4 (Set0)	EMAC0.RXD0 (Set0)	GPIO05	LOANIO05
USB1.D5 (Set0)	EMAC0.MDIO (Set0)	GPIO06	LOANIO06
USB1.D6 (Set0)	EMAC0.MDC (Set0)	GPIO07	LOANIO07
USB1.D7 (Set0)	EMAC0.RX_CTL (Set0)	GPIO08	LOANIO08
	EMAC0.TX_CTL (Set0)	GPIO09	LOANIO09
USB1.CLK (Set0)	EMAC0.RX_CLK (Set0)	GPIO10	LOANIO10
USB1.STP (Set0)	EMAC0.RXD1 (Set0)	GPIO11	LOANIO11
USB1.DIR (Set0)	EMAC0.RXD2 (Set0)	GPIO12	LOANIO12
USB1.NXT (Set0)	EMAC0.RXD3 (Set0)	GPIO13	LOANIO13

Refer to the following table for the appropriate loan I/O for each USB option.

Table 4: USB Loan I/O

USB I/O Pin Set	Loan I/O
USB0 I/O Set 0	LOANIO44
USB1 I/O Set 0	LOANIO10
USB1 I/O Set 1	LOANIO29

Connecting the Clock in the Top Level Design

The following example shows how to connect the FPGA clock to a Loan I/O when the USB PHY operates in input clock mode using an FPGA clock source from the SoC.

Add the following code snippets to your top-level design file, to connect the clock outputs to the loan I/O:

```
// top level module pin defines
// LOANIO10 = mac clock
```

```
inout  wire       LOANIO10,

// wire instances of the 3 loan IO buses from Platform Designer instance
wire [66:0] loan_out;
wire [66:0] loan_oe;

// this synthesis keep directive is required in
// order to connect PLL clock outputs to the Loan IO
wire        usb_mac_clk_from_pll    /* synthesis keep */;

// make assignment of the clocks to the appropriate loan IO
assign loan_out[10]   = usb_mac_clk_from_pll;
assign loan_oe[10]    = 1'b1;

// snippet of Qsys instantiation signal assignments
.hps_0_h2f_loan_io_in                 (),         // hps_0_h2f_loan_io.in
.hps_0_h2f_loan_io_out                (loan_out),    //                  .out
.hps_0_h2f_loan_io_oe                 (loan_oe),     //                  .oe
.hps_io_0_hps_io_gpio_inst_LOANIO10   (LOANIO10),    // hps_io_gpio_inst_LOANIO10
```

Revision History

Date	Version	Changes

September 2017	2017.09.22	• Modified *USB PHY Timing Characteristics* topic and added subtopics:
		  • *USB Setup and Hold Relationships*
		  • *USB Controller to PHY Setup and Hold Timing Arcs*
		  • *USB Controller to PHY Setup Timing Analysis*
		  • *USB Controller to PHY Hold Timing Analysis*
		  • *USB PHY to Controller Setup and Hold Timing Arcs*
		  • *USB PHY to Controller Setup Timing Analysis*
		  • *USB PHY to Controller Hold Timing Analysis*

July 2014	2014.07.21	• Corrected the `USB MAC Th` number value in the USB PHY to USB Controller Hold equation in the *USB PHY Timing Characteristics* section.
July 2014	2014.07.16	• Corrected the `ClkTrace Td_max` value in the USB PHY to USB Controller Setup equation in the *USB PHY Timing Characteristics* section. • Corrected the `USB PHY Td_min` value in the USB PHY to USB Controller Hold equation in the *USB PHY Timing Characteristics* section.

Date	Version	Changes
July 2014	2014.07.03	• Modified *Table 2: USB MAC Timing Requirements*. • Added *USB PHY Timing Characteristics* section. • Clarified *Output Clock Mode* section. • Modified *Input Clock Mode* section. • Removed *Two Clock Mode*, *Selecting the PHY Clock*, *Instantiating a PLL*, and *Constraining the Design* sections.