Datasheet

USB 2.0 Total IP Solution

Arasan Chip Systems Inc.
2010 North First Street, Suite #510, San Jose, CA 95131
Ph: 408-282-1600
Fax: 408-282-7800
www.arasan.com

Disclaimer

This document is written in good faith with the intent to assist the readers in the use of the product. Circuit diagrams and other information relating to Arasan Chip Systems' products are included as a means of illustrating typical applications. Although the information has been checked and is believed to be accurate, no responsibility is assumed for inaccuracies. Information contained in this document is subject to continuous improvement and development.

Arasan Chip Systems' products are not designed, intended, authorized or warranted for use in any life support or other application where product failure could cause or contribute to personal injury or severe property damage. Any and all such uses without prior written approval of an Officer of Arasan Chip Systems Inc. will be fully at the risk of the customer.

Arasan Chip Systems Inc. disclaims and excludes any and all warranties, including, without limitation, any and all implied warranties of merchantability, fitness for a particular purpose, title, and infringement and the like, and any and all warranties arising from any course or dealing or usage of trade.

This document may not be copied, reproduced, or transmitted to others in any manner. Nor may any use of information in this document be made, except for the specific purposes for which it is transmitted to the recipient, without the prior written consent of Arasan Chip Systems Inc. This specification is subject to change at any time without notice. Arasan Chip Systems Inc. is not responsible for any errors contained herein.

In no event shall Arasan Chip Systems Inc. be liable for any direct, indirect, incidental, special, punitive, or consequential damages; or for loss of data, profits, savings or revenues of any kind; regardless of the form of action, whether based on contract; tort; negligence of Arasan Chip Systems Inc or others; strict liability; breach of warranty; or otherwise; whether or not any remedy of buyers is held to have failed of its essential purpose, and whether or not Arasan Chip Systems Inc. has been advised of the possibility of such damages.

Restricted Rights

Use, duplication, or disclosure by the Government is subject to restrictions as set forth in FAR52.227-14 and DFAR252.227-7013 et seq. or its successor.

Copyright Notice

No part of this specification may be reproduced in any form or means, without the prior written consent of Arasan Chip Systems, Inc.

Questions or comments may be directed to:

Arasan Chip Systems Inc.
2010 North First Street, Suite 510
San Jose, CA 95131
Ph: 408-282-1600
Fax: 408-282-7800
Email: sales@arasan.com

Contents

Tables

Figures

1 Introduction

Universal Serial Bus (USB) is an industry standard that defines the cables, connectors and communications protocols used in a bus for connection, communication, and power supply between computers and electronic devices. USB standardizes the connection of computer peripherals (including keyboards, pointing devices, digital cameras, printers, portable media players, disk drives and network adapters) to personal computers, both to communicate and to supply electric power. It has become commonplace on other devices, such as smartphones, PDAs and video game consoles.

1.1 Arasan's Contribution to USB

Arasan Chip Systems provides a complete suite of USB-compliant IP including low speed, high speed and super speed USB products. Arasan became a member if the USB-IF standards body in 1996 and delivered its first USB 1.0 IP product in that year. The offering expanded to USB 2.0 products in 2000 and USB 3.0 products in 2009. Arasan's customer base includes many major systems and SoC companies in variety of industries.

Arasan is the only company to offer a USB Total IP Solution. In addition to offering a complete suite of digital IP for USB, Arasan's analog IP team onlhas also developed the USB 2.0 PHY and USB HSIC PHY. As part of its end-to-end solution approach to IP, Arasan also offers USB 3.0 software stacks and drivers which are backward compatible with 2.0 specifications.

Arasan's active involvement and contribution to the relevant standards bodies, lead to deep domain expertise, which in turn results in early availability of high quality standards compliant IP for our customers.

1.2 Arasan's Total IP Solution

Arasan provides a Total IP Solution, which encompasses all aspects of IP development and integration, including analog and digital IP cores, verification IP, software stacks & drivers, and hardware validation platforms. Benefits of Total IP Solution:

- Seamless integration from PHY to Software
- Assured compliance across all components
- Single point of support
- Easiest acquisition process (one licensing source)
- Lowest overall cost including cost of integration
- Lowest risk for fast time to market

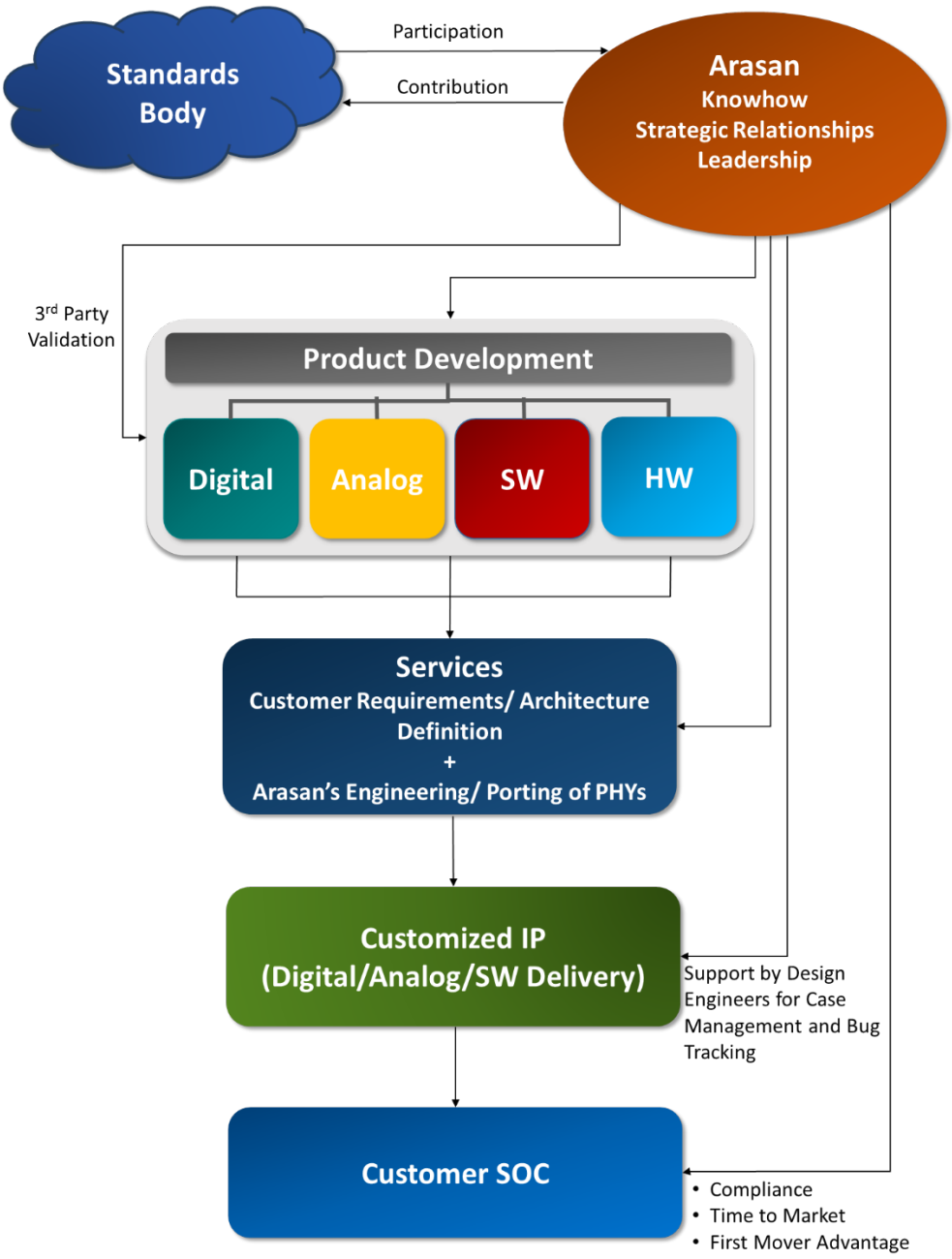

Figure 1: Arasan's Total IP Solution

2 USB 2.0 Host IP

2.1 Overview

The Arasan USB 2.0 Host IP is an USB 2.0 specification compliant host IP core with an optional AHB, PCI, or custom host interface. The USB 2.0 Host IP supports

- 480 Mbit/s in High Speed (HS) mode
- 12 Mbit/s in Full Speed (FS) mode
- 1.5 Mbit/s in Low Speed (LS) mode

The Arasan's USB 2.0 Host controller is designed for flexibility and ease of use and facilitates implementation of a wide variety of applications with fast turnaround time. This design is technology independent and migrating to any technology is fast and simple. This EHC can be easily interfaced to standard buses such as ARM, S-Bus and so on.

The IP consists of an Enhanced Host Controller Interface (EHCI) and a companion Open Host Controller Interface (OHCI). The EHCI processor handles HS transactions and is the default owner of the root hub that connects to the downstream ports. In a downstream data transfer, the EHCI sends data to the Host Parallel Interface Engine (HPIE) for encoding and CRC appending. Data received by the USB 2.0 Root Hub is forwarded to the downstream ports. Similarly, FS and LS transactions are handled by the OHCI, Host Serial Interface Engine (HSIE), and USB 1.1 Root Hub.

The Root Hub performs multiplexing and forwarding of packets between the downstream ports and USB 2.0/1.1 Root Hubs. Up to 8 downstream ports can be connected to the USB 2.0 Host IP core. With the addition of an optional ULPI Wrappers, the Arasan USB 2.0 Host IP core can be connected directly to a 16-bit standard UTMI+ or 8-bit ULPI transceiver.

2.2 Features

- Compliant with the following specification versions:
  - USB specification revision 2.0 EHCI specification revision 1.0

- USB 2.0 Host:
  - Supports up to 127 devices and 8 downstream ports
  - OHCI companion processor for USB 1.1 transfers
  - 16-bit UTMI+ and 8-bit ULPI interfaces
  - Direct addressing all IP core registers from AHB, PCI, or custom bus
  - DMA controller supports high-speed data transfers between USB Host IP and host bus
  - Supports low, full and high speed devices
  - Technology independent
  - Integrated root hub with up to 3 ports

- Host Interface:
  - 8, 16, or 32-bit host bus
  - Optional 133 MHz AHB Rev. 2.0 master/slave interface
  - Optional 33 MHz PCI Rev. 2.2 master/target interface
  - Optional custom bus interface

2.3 Architecture

2.3.1 Functional Block Diagram

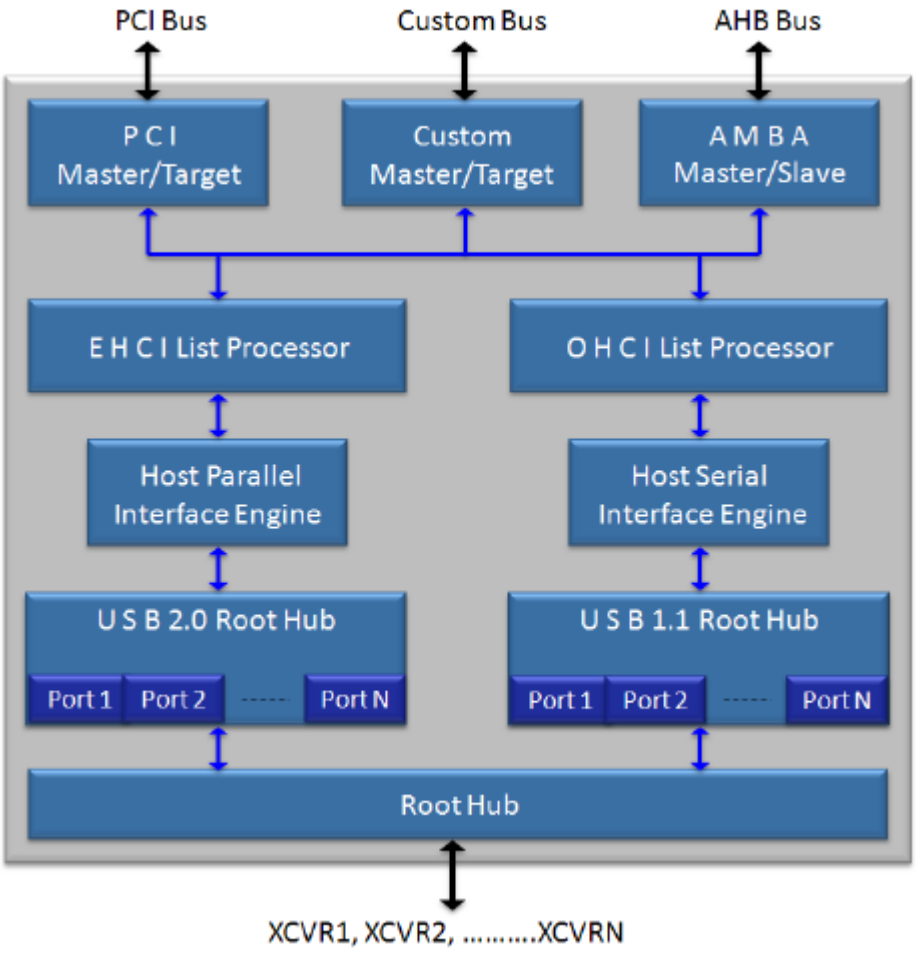

Figure 2: USB Host IP Core Functional Block Diagram

2.3.2 Functional Block Diagram Description

2.3.2.1 PCI Master/Slave Function

The PCI Master/Slave function is a bus interface unit that performs all data transfers necessary for the back-end core to access the system memory as well as the data transfers necessary for the bus master to access the PCI BUS. The target functionality supports memory read/write and configuration read/write commands. It has two functions. Function zero corresponds to USB 1.1 Host controller and Function two to USB 2.0 Host controller. The master supports memory read and memory write burst and single data phase transactions. The main functions are reading and writing EHCI specific data structures and data transfer from the system memory to the device and vice-versa. An optional PCI version 2.2 master/target interfaces allows a PCI host to access all registers in the USB 2.0 Host IP. The PCI interface supports programmable I/O and DMA data transfer methods. In programmable I/O method, the PCI host driver transfers data using the Buffer Data Port register. In DMA method, the DMA controller is the bus master during the data transfer. The PCI master/target interface also supports Power management conforming to the PCI power management interface specification.

2.3.2.2 Custom Interface (Optional)

Custom bus with special requirements such as a bus width from 4-bit to 64-bit can be provided by Arasan Chip Systems. The custom bus can be used to support processors such as 8051. A DMA controller can also be included as an optional module. Host processor connected to the custom bus has direct access to all registers in the USB 2.0 Host IP. Configuration and operation of the USB host core can be controlled by the host processor.

2.3.2.3 AHB Interface (Optional)

The AHB master/slave interface provides a high-speed connection between the AHB host and USB 2.0 host controller. When operating as an AHB slave, the DMA controller can be used to manage the high-speed data transfers. The AHB host has direct access to all registers in the USB 2.0 Host IP. Configuration and operations of the USB host can be controlled by an AHB host through the AHB interface.

2.3.2.4 UTMI+ Interface

The Arasan USB 2.0 Host IP core implements a 16-bit UTMI+ compliant module that provides a seamless interface to standard interface components such as the Philips USB 2.0 UTMI+ transceiver.

2.3.2.5 ULPI Interface

The Arasan USB 2.0 Host IP core provides an optional UTMI+ to ULPI wrapper. Instead of using a high pin count UTMI+ interface, user may choose to use an ULPI interface to reduce the pin count to 12.

2.3.2.6 EHCI List Processor

This block takes care of the USB transaction with high speed devices, which are USB 2.0 compliant. It processes the EHCI version 2.0 specific data structures.

2.3.2.7 OHCI List Processor

This block takes care of the USB transaction with full and low speed devices, which are specification 1.1 compliant. This block only processes USB 1.1 compliant data structures.

2.3.2.8 Host Parallel Interface Engine (HPIE)

This parallel interface engine decodes the packet from the device and also does CRC checking Supports 127 devices bridging the core to other system buses like ARM, Motorola, and Sun and so on. Supports Control, Bulk, and Isochronous and interrupt data transfer types.

2.3.2.9 Host Serial Interface Engine (HSIE)

This is the host serial interface engine for USB1.1 OHCI List processor engine. It does NRZI encoding / decoding, bitstuffing / debitstuffing, CRC checking / generation, packet decoding and serial to parallel conversions and vice-versa.

2.3.2.10 USB 2.0 Root Hub

This block takes care of USB 2.0 device specific connect/disconnect, power management, suspend and resume signaling.

2.3.2.11 USB 1.1 Root Hub

This block takes care of USB 1.1 device specific connect/disconnect, power management, suspend and resume signaling.

2.3.2.12 Root Hub

Depending on device speed this block switches different downstream devices to either USB 2.0 or USB 1.1 root hubs.

2.4 PIN Diagram

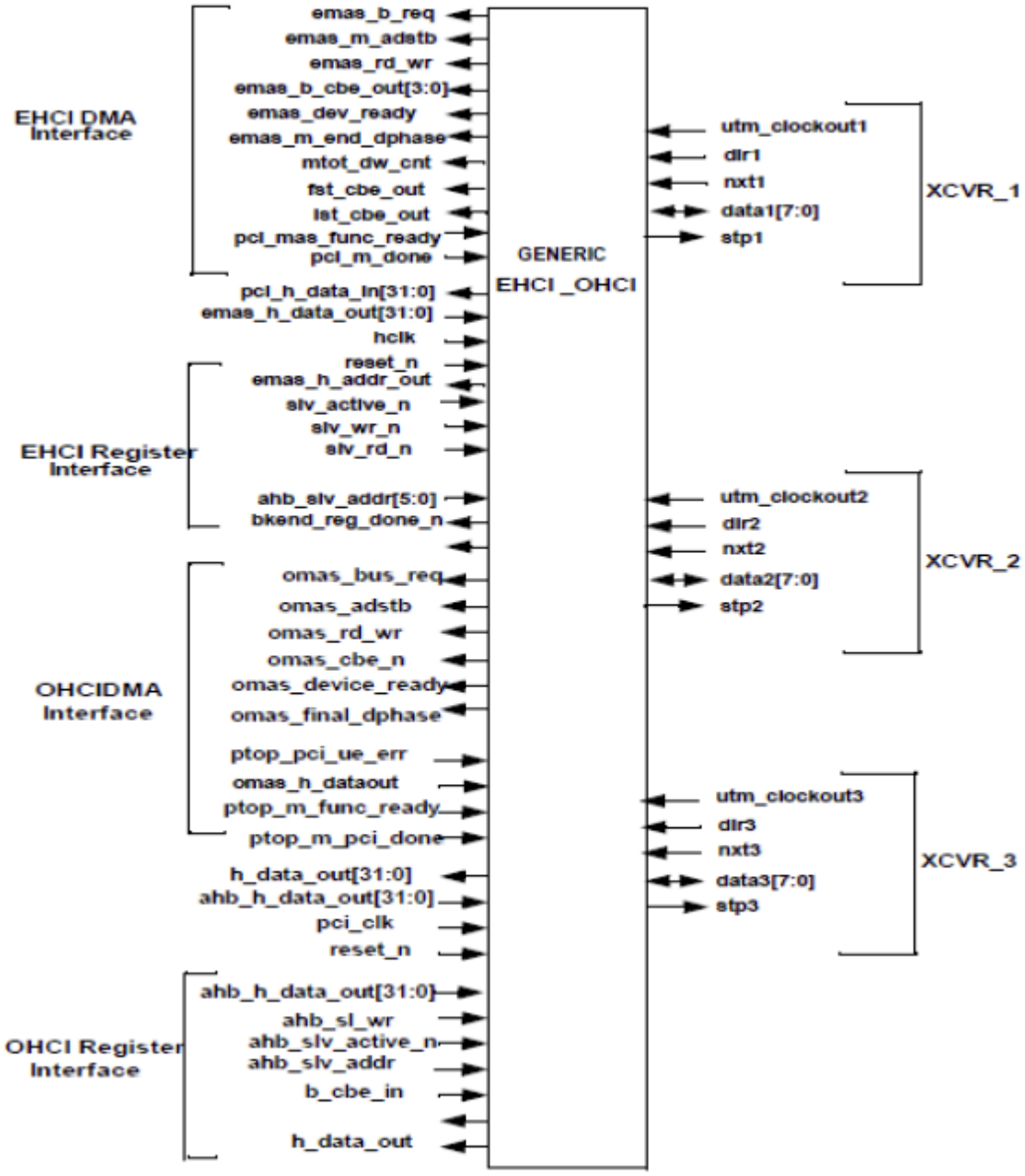

Figure 3 : USB Host IP Pinout Diagram

2.5 Signal Interfaces

Table 1: EHCI DMA Interface Signals

Pin	Direction	Description
data_in [31:0]	OUT	Register write data to EHCI
data_out [31:0]	IN	Register read data from EHCI
fst_cbe_out[3:0]	OUT	This pin indicates the first valid byte
lst_cbe_out[3:0]	OUT	This pin indicates the last valid byte
emas_b_req	OUT	EHCI request to system master core to perform DMA
em_m_adstb	OUT	EHCI indicates the presence of DMA address on data lines
emas_rd_wr	OUT	Backend indicates to system master core to do a read or write operation 0 - Read. 1 - Write
emas_b_cbe_out [3:0]	OUT	EHCI provides byte enables for write transfer. Has same timing as write data
emas_device_ready	OUT	EHCI indicates that it is ready for DMA transfer. It should not be deasserted before the DMA completes
emas_m_end_dphase	OUT	Asserted by EHCI for last beat
pci_m_func_rdy	OUT	Asserted by the system core whenever there is a successful transfer
mtot_dword_cnt	OUT	Number of Dwords transferred
fst_cbe_out	OUT	First valid byte
lst_cbe_out	OUT	Last valid byte
emas_h_dataout	OUT	Output data from EHCI
pci_h_data_in	IN	Input data to EHCI
pci_m_done	OUT	Asserted by the system core when the last byte for current DMA transfers
hclk	IN	133mhz clock frequency
hreset_n	IN	Active low reset signal
utm_clockout1	IN	Interface clock. 60 MHZ Input clock from ULPI PHY
dir1	IN	Input signal from ULPI PHY, controls the direction of the data bus. The PHY pulls dir high whenever the interface cannot accept data from the Link
nxt1	IN	Input signal from the PHY used to throttle all data types, except register read data and the RX CMD. Identical to RX valid during USB receive, and TX Ready during USB transmit. The PHY also asserts nxt and dir simultaneously to indicate USB receive activity (RXActive), if dir was previously low. The PHY is not allowed to assert nxt during the first cycle of the TX CMD driven by the Link
stp1	OUT	Output to ULPI PHY. The Link must assert stp to signal the end of a USB transmit packet or a register write operation, and optionally to stop any receive. The step

		signal must be asserted in the cycle after the last data byte is presented on the bus
data1[7:0]	INOUT	8 bit data bus. Driven to 00h by the Link when the ULPI bus is idle. The signals in this bus are synchronized with positive edge of clk60
utm_clockout2.	IN	Interface clock. 60 MHZ Input clock from ULPI PHY
dir2	IN	Input signal from ULPI PHY, controls the direction of the data bus. The PHY pulls dir high whenever the interface cannot accept data from the Link
nxt2	IN	Input signal from the PHY used to throttle all data types, except register read data and the RX CMD. Identical to RX valid during USB receive, and TX Ready during USB transmit. The PHY also asserts nxt and dir simultaneously to indicate USB receive activity (RXActive), if dir was previously low. The PHY is not allowed to assert nxt during the first cycle of the TX CMD driven by the Link
stp2	OUT	Output to ULPI PHY. The Link must assert stp to signal the end of a USB transmit packet or a register write operation, and optionally to stop any receive. The stp signal must be asserted in the cycle after the last data byte is presented on the bus
data2[7:0]	INOUT	8 bit data bus. Driven to 00h by the Link when the ULPI bus is idle. The signals in this bus are synchronized with positive edge of clk60
utm_clockout3	IN	Interface clock. 60 MHZ Input clock from ULPI PHY
dir3	IN	Input signal from ULPI PHY, controls the direction of the data bus. The PHY pulls dir high whenever the interface cannot accept data from the Link
nxt3	IN	Input signal from the PHY used to throttle all data types, except register read data and the RX CMD. Identical to RX valid during USB receive, and TX Ready during USB transmit. The PHY also asserts nxt and dir simultaneously to indicate USB receive activity (RXActive), if dir was previously low. The PHY is not allowed to assert nxt during the first cycle of the TX CMD driven by the Link
stp3	OUT	Output to ULPI PHY. The Link must assert stp to signal the end of a USB transmit packet or a register write operation, and optionally to stop any receive. The stp signal must be asserted in the cycle after the last data byte is presented on the bus
data3[7:0]	INOUT	8 bit data bus. Driven to 00h by the Link when the ULPI bus is idle. The signals in this bus are synchronized with positive edge of clk60

Table 2: OHCI DMA Interface Signals

Pin	Direction	Description
ahb_slv_active_n	IN	Indicates to the OHCI, that an external master accesses this Slave. Asserted till the transfer completes
ahb_slv_wr	IN	Register write to the OHCI target
ahb_slv_addr	IN	Address of the target register
h_data_out [31:0]	OUT	Register read data from OHCI
ahb_h_data_out [31:0]	IN	Register write data to OHCI
b_cbe_in	IN	Command byte enable from master
omas_bus_req	OUT	OHCI request to system master core to perform DMA
om_adstb_n	OUT	OHCI indicates the presence of DMA address on data lines
omas_rd_wr .	OUT	Back end indicates to system master core to do a read or write operation 0 - Read. 1 - Write
omas_cbe_n [3:0]	OUT	OHCI provides byte enables for write transfer. Has same timing as write data
omas_device_ready	OUT	OHCI indicates that it is ready for DMA transfer. It should not be deasserted before the DMA completes
omas_final_dphase	OUT	Asserted by OHCI for last beat
ptop_m_func_rdy	OUT	Asserted by the system core whenever there is a successful transfer
ptop_m_pci_done	OUT	Asserted by the system core when the last beat for current DMA ends
clk	IN	Operating clock frequency
clk_out	IN	USB interface clock frequency
reset_n	IN	Active low reset signal

2.6 SoC Level Integration

2.6.1 Verification Environment

The Verification Environment used to verify the EHCI 2.0 DUT. This Verification Environment includes test suites to provide a complete verification solution for functional verification of EHCI 2.0 DUT. This Verification Environment consists of:

The EHCI is tested with PCI models and Device model. The PCI model has an address constraint that the address range for data, Qhead, QTd, IsoTd and SiTd are to be separate. This is to easily identify the memory dumping once a testcase's simulation is completed.

When a transaction is to be started by the Enhanced Host controller, the PCI model has to load the memory with data structure with Qhead and QTd in the corresponding locations.

The EHCI reads the data structure one by one and performs the USB transaction and updates the data structures in the same unique locations. The updated data structure as such is dumped in the file “ds.dmp”.

The Verification Environment includes exhaustive built-in tests to verify the various functionalities of the EHCI 2.0 DUT.

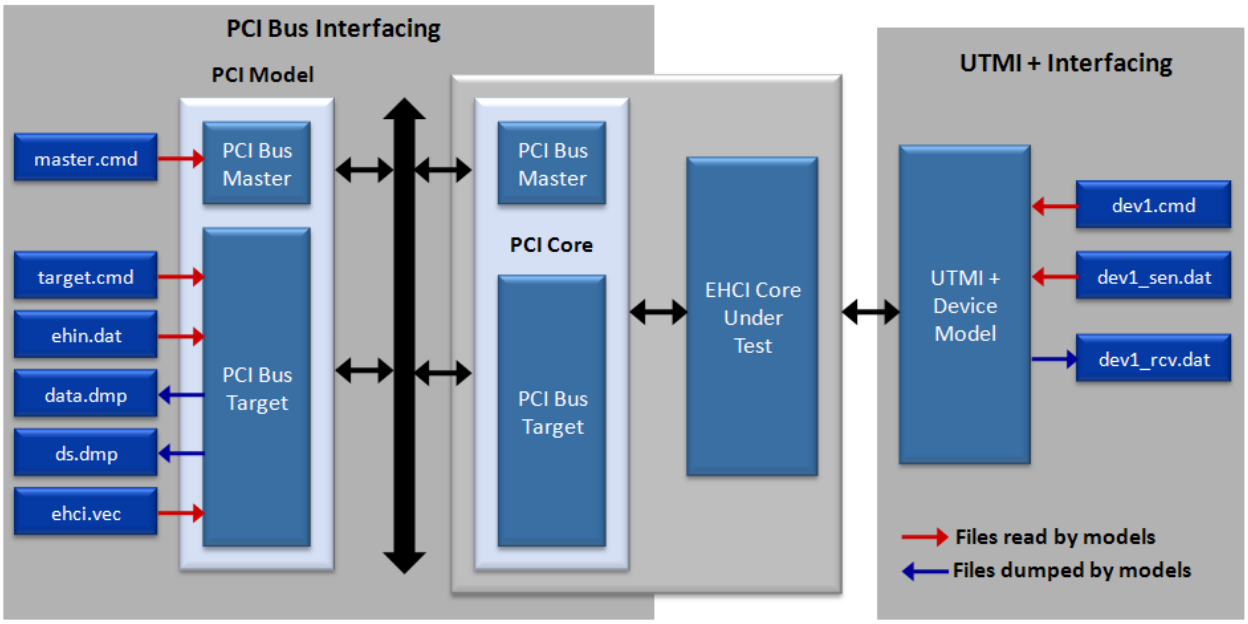

Figure 4: Verification Environment

2.6.2 IP Deliverables

The IP package consists of the following:

- RMM-compliant synthesizable RTL design in Verilog
- Easy-to-use test environment
- Synthesis scripts
- Technical documents
- Simulation scripts

3 USB 2.0 Hub IP

3.1 Overview

The Universal Serial Bus (USB) is a type of serial bus that enables transfer of data between a host computer and various types of peripheral devices. The USB host has point-to-point connections with USB devices via a tiered star topology, in which each star contains a device called a hub. The USB enables connection of up to 127 devices via this tiered star topology. In addition, devices can be connected or disconnected while the system is still operating. All the above things are made possible via the USB2.0 Hub.

The Arasan USB 2.0 Hub IP core is an USB 2.0 specification compliant hub core that supports 480 Mbit/s in High Speed (HS) mode, 12 Mbit/s in Full Speed (FS) mode, and 1.5 Mbit/s in Low Speed (LS) mode. The Arasan USB 2.0 Hub IP core consists of the Hub Controller, Hub Repeater, Transaction Translators, Routing Logic, and Downstream Ports. The major functions of the hub being device connection/disconnection detection, establishing upstream and downstream connectivity, enforcement of specific USB defined End-Of- Frame points over the bus and suspend/resume handling.

The USB 2.0 Hub core consists of Hub Repeater, Hub Controller and Transaction Translator. The Hub Controller provides the mechanism for host to hub communication. Hub specific status and control commands permit the host to configure a hub and to monitor and control its transaction translator and individual downstream ports.

The Hub Controller controls the operation of the USB hub by interpreting both the USB commands and Hub class specific commands. High-speed packets originated from the root hub are forwarded by the Hub Repeater to the HS Downstream Ports through the Routing Logic. Full-speed and low-speed packets scheduled by the host system software as split transactions are forwarded to the Transaction Translators.

The Transaction Translators handle split transactions that convey isochronous, interrupt, control, and bulk transfers across the high-speed bus to and from the full-speed and low-speed devices that attached to the hub. The Transaction Translators also perform CRC on incoming packets from the Hub Repeater or Routing Logic. The Routing Logic connects the Hub Repeater to the Downstream Ports in a high-speed transfer, and it connects the Transaction Translators to the Downstream Ports in a full-speed or low-speed transfer. The number of Downstream Ports is scalable.

3.2 Features

- High speed support: 480 Mbit/s
- Full speed support: 12 Mbit/s
- USB 2.0 Compliant
- High/Full speed support using 8/16 bit UTMI/ULPI interface
- UTMI Interface Clock: 30/60 MHz

- USB Suspend/Resume support
- Supports all hub specific requests
- UTMI and UTMI+ Txcvr compatible upstream and downstream support
- Technology and process independent
- Supports Remote wakeup features
- Parametrizable number of downstream ports
- Shared TT for the downstream ports for HS/FS/LS
- Supports suspend/resume for power management
- HS repeater for the downstream HS device
- FS/LS repeater for the downstream FS/LS device when Hub is connected to FS host upstream
- Downstream device connect / disconnect detection

3.3 Architecture

3.3.1 Functional Block Diagram

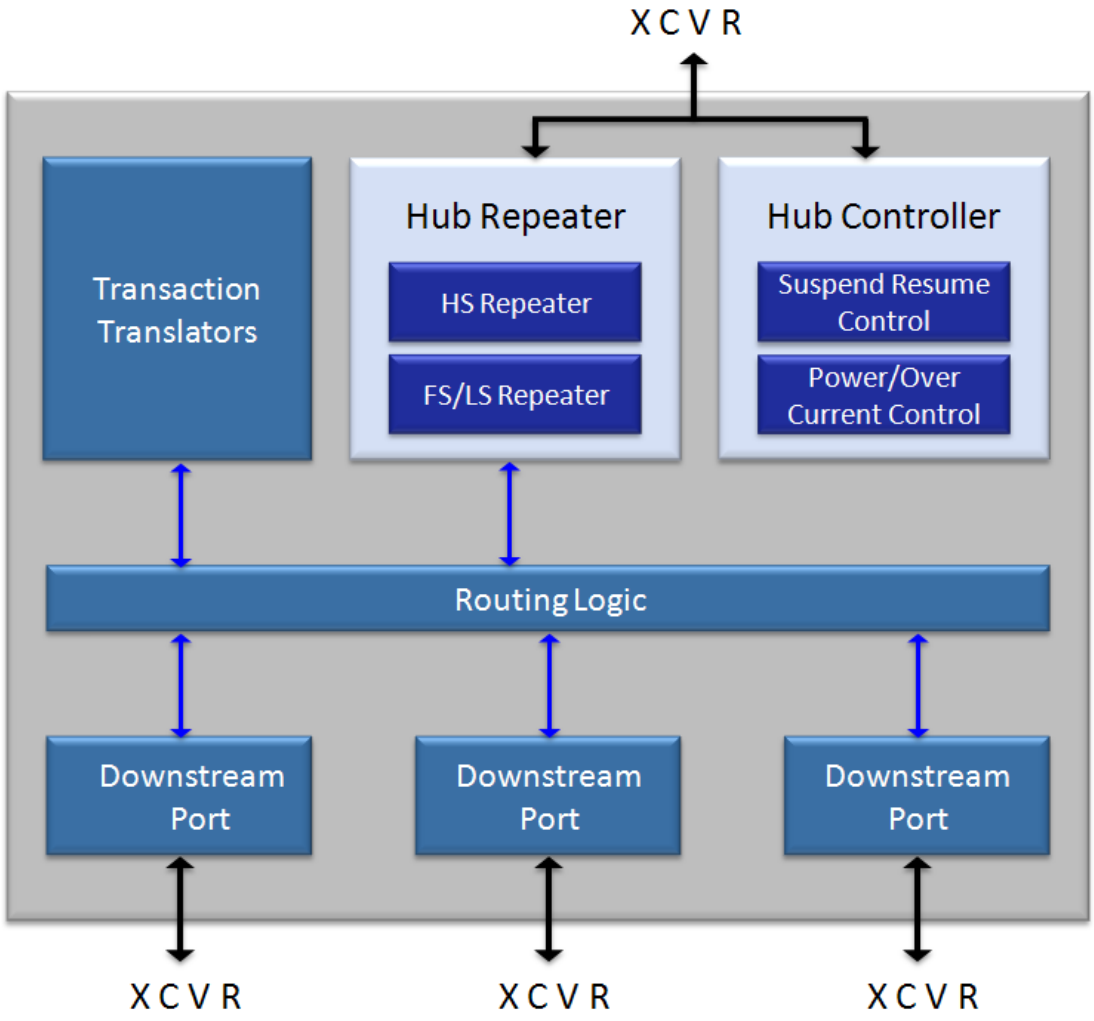

Figure 5: USB 2.0 HUB IP Core Functional Block Diagram

3.3.2 Functional Block Diagram Description

3.3.2.1 Transaction Translators

This block isolates the high-speed bus from the full-speed bus using data buffers. Split transaction wrappers of the transactions meant for the full speed bus are removed and both the token and data packet for the full speed bus are stored inside the data buffer. The data buffer houses the status information from the full speed bus.

3.3.2.2 Hub Repeater

The hub repeater is a protocol controlled switch between the upstream and downstream ports. Hub repeaters also have hardware support for reset and suspend/resume handling. It also supports bus fault detection and recovery.

3.3.2.3 HS Repeater

This block handles the broadcast of upstream data to downstream ports and also routes the downstream connectivity to the upstream. The repeater state machine block operating at 30Mhz (from the external clock) controls the repeater data path between the upstream ports and the downstream ports.

3.3.2.4 FS/LS Repeater

This block handles the broadcast of upstream data to downstream ports and also routes the downstream connectivity to the upstream. The repeater state machine block operation at 60Mhz (from the external clock) controls the repeater data path between the upstream ports and the downstream ports.

3.3.2.5 Hub Controller

The Controller provides the communication to/from the host. The information like Hub-specific status and control commands permit the host to configure a hub and to monitor and control its ports.

3.3.2.6 Suspend Resume Control

The Hub Repeater is a protocol-controlled switch between the upstream port and downstream ports. It has the support for reset and suspend/resume signaling. Hub support's suspend and resume both as a USB device and in terms of propagating suspend and resume signaling. Hub supports both global and selective suspend and resume. Global suspend/resume refers to the entire bus being suspended or resumed without affecting any hub's downstream facing port states; selective suspend/resume refers to a downstream facing port of a hub being suspended or resumed without affecting the hub state.

3.3.2.7 Power/Over Current Control

The power management system may transition a device to the suspended state or power-off the device in order to control and conserve power. Power switch control and the over current protection of the downstream ports are implemented. The architecture supports individual port power switching and Gang mode.

3.3.2.8 Routing Logic

The operating speed of a device attached on a downstream facing port determines whether the Routing Logic connects a port to the Transaction Translator or Hub Repeater sections. When the hub upstream facing port is attached to an environment that is operating at high-speed, the full-

/low-speed hub repeater is not operational. Hence when a high-speed device is attached on downstream facing port, the routing logic connects the port to the hub repeater and the hub repeater operates as a high-speed repeater. When a full-/low-speed device is attached on a downstream facing port, the routing logic connects the port to the transaction translator.

3.3.2.9 Downstream Port

This block implements the downstream ports state machine. It handles/reports the status of the downstream ports to the control endpoint and status change endpoint. This block operates at 30Mhz with serial data interface to the downstream ports.

3.4 Signal Interfaces

Table 3: HUB2.0 Upstream Port PIN Description

Pin	Direction	Description
pwr_on_rst	Input	Power on reset signal (asynchronous)
utm_clkout	Input	Clock. This output is used for clocking receive and transmit parallel data. 30 MHz HS/FS, with 16-bit interface
utm_rxvalid	Input	Receive Data Valid. Indicates that the utm_data bus has valid data. The Receive Data Holding Register is full and ready to be unloaded. The SIE is expected to latch the utm_data out bus on the clock edge
utm_rxactive	Input	Receive Active. Indicates that the receive state machine has detected SYNC and is active. utm_rxactive is negated after a Bit Stuff Error or an EOP is detected
utm_rxerror	Input	Receive Error. 0 Indicates no error. 1 Indicates that a receive error has been detected
utm_txready	Input	Transmit Data Ready. If is asserted, the SIE must always have data available for clocking in to the TX Holding Register on the rising edge of CLK.
linestate[1:0]	Input	Line State. These signals reflect the current state of the single ended receivers. They are combinatorial until a usable CLK is available then they are synchronized to CLK. They directly reflect the current state of the DP (Linestate[0]) and DM.(LineState[1]) signals: DM DP Description 0 0 0 : SE0 0 1 1 : 'J' State 1 0 2 : 'K' State 1 1 3 : SE1
utm_data16_8	Input	Data Bus 16 - 8. Selects between 8 and 16 bit data transfers. 1 : 16-bit data path operation enabled. DataIn(8-15), DataOut(8-15), TXValidH, and RXValidH

		operational. CLK = 30 MHz. 0 : 8-bit data path operation enabled. DataIn(8-15),DataOut(8-15), TXValidH, and RXValidH undefined. CLK = 60 MHz.
utm_xcvrsel[1:0]	Output	Transceiver Select. This signal selects between the FS and HS transceivers: 0: HS transceiver enabled 1: FS transceiver enabled This signal is not provided in FS Only and LS Only transceiver implementations
utm_termsel	Output	Termination Select. This signal selects between the FS and HS terminations: 0: HS termination enabled 1: FS termination enabled This signal is not provided in FS Only and LS Only transceiver implementations
utm_reset	Output	Reset all state machine in UTM
utm_suspendm	Output	Suspend. Places the Macrocell in a mode that draws minimal power from supplies. Shuts down all blocks not necessary for Suspend/ Resume operation. While suspended, utm_termsel must always be in FS mode to ensure that the 1.5K pull_up on DP remains powered. 0: Macrocell circuitry drawing suspend current 1: Macrocell circuitry drawing normal current
utm_opmode[1:0]	Output	Operational Mode. These signals select between various operational modes: [1] [0] Description 0 0 0: Normal Operation 0 1 1: Non-Driving 1 0 2: Disable Bit Stuffing and NRZI encoding 1 1 3: Reserved
utm_txvalid	Output	Transmit Valid. Indicates that the DataIn bus is valid. The assertion of Transmit Valid initiates SYNC on the USB. The negation of Transmit Valid initiates EOP on the USB
utm_data[15:0]	Input/Output	Utm_databus. This is 16-bit parallel in/out data. Value of this data bus varied accordingly the signal Utm_databus16_8. If the value of this signal is one means the utm_data will send the 16-bits of data, otherwise it sends 8-bits of data
utm_validh	Output	validh. This signal that the higher order of the data_bus is valid or not. When Databus16_8 is zero then the value of this signal is undefined. The status of the lower order byte is determined by Txvalid and Rxvalid
rx_dp_up	Input	Rx_DP. This is single ended receive data,positive terminal. The data is only valid if FsLsSerialMode is set to 1b

rx_dm_up	Input	Rx_DM. This is single ended receive data,negative terminal The data is only valid if FsLsSerialMode is set to 1b
rx_rcv_dp	Input	Rx_RCV. Receive data. The data is only valid if FsLsSerialMode is set to 1b
tx_enable_n_up	Output	Tx_Enable_N. This is the active low output enable signal
tx_dat_up	Output	Tx_DAT. Differential data at D+/D- output
tx_se0_up	Output	Tx_Se0. Force single ended zero
fslsserialmode_ up	Input	FsLsSerialMode. This signal is zero means FS and LS packets are sent using the parallel interface. One means FS and LS packets are sent using the serial interface

Active low signals have been indicated with a suffix of '_n'.

NOTE: The above indicated signals are for the upstream port of the USB Hub. These same signal set is used for downstream port of this USB Hub. In addition the following signals power and over current protection.

Table 4: HUB2.0 Downstream Port PIN Description

Pin	Direction	Description
pwr_on_rst	Input	Power on reset signal (asynchronous)
utm_clkout	Input	Clock. This output is used for clocking receive and transmit parallel data. 30MHz HS/FS, with 16-bit interface
utm_rxvalid	Input	Receive Data Valid. Indicates that the utm_data bus has valid data. The Receive Data Holding Register is full and ready to be unloaded. The SIE is expected to latch the utm_data out bus on the clock edge
utm_rxactive	Input	Receive Active. Indicates that the receive state machine has detected SYNC and is active. utm_rxactive is negated after a Bit Stuff Error or an EOP is detected
utm_txready	Input	Transmit Data Ready. If is asserted, the SIE must always have data available for clocking in to the TX Holding Register on the rising edge of CLK.
linestate[1:0]	Input	Line State. These signals reflect the current state of the single ended receivers. They are combinatorial until a usable CLK is available then they are synchronized to CLK. They directly reflect the current state of the DP (Linestate[0]) and DM.(LineState[1]) signals: DM DP Description 0 0 0 : SE0 0 1 1 : 'J' State

		1 0 2 : 'K' State 1 1 3 : SE1
utm_data16_8	Input	Data Bus 16 - 8. Selects between 8 and 16 bit data transfers. 1 : 16-bit data path operation enabled. DataIn(8-15), DataOut(8-15), TXValidH, and RXValidH operational. CLK = 30 MHz. 0 : 8-bit data path operation enabled. DataIn(8-15),DataOut(8-15), TXValidH, and RXValidH unde-fined.CLK = 60 MHz.
utm_xcvrsel[1:0]	Output	Transceiver Select. This signal selects between the FS and HS transceivers: 0: HS transceiver enabled 1: FS transceiver enabled This signal is not provided in FS Only and LS Only transceiver implementations
utm_termsel	Output	Termination Select. This signal selects between the FS and HS terminations: 0: HS termination enabled 1: FS termination enabled This signal is not provided in FS Only and LS Only transceiver implementations.
utm_reset	Output	Reset all statemachine in UTM.
utm_suspendm	Output	Suspend. Places the Macrocell in a mode that draws minimal power from supplies. Shuts down all blocks not necessary for Suspend/ Resume operation. While suspended, utm_termsel must always be in FS mode to ensure that the 1.5K pull_up on DP remains powered. 0: Macrocell circuitry drawing suspend current 1: Macrocell circuitry drawing normal current
utm_opmode[1: 0]	Output	Operational Mode. These signals select between various operational modes: [1] [0] Description 0 0 0: Normal Operation 0 1 1: Non-Driving 1 0 2: Disable Bit Stuffing and NRZI encoding 1 1 3: Reserved.
utm_txvalid	Output	Transmit Valid. Indicates that the DataIn bus is valid. The assertion of Transmit Valid initiates SYNC on the USB. The negation of Transmit Valid initiates EOP on the USB.
utm_data[15:0]	Input/Output	Utm_databus. This is 16-bit parallel in/out data. Value of this data bus varied accordingly the signal Utm_databus16_8. If the value of this signal is one means the utm_data will send the 16-bits of data, otherwise it sends 8-bits of data.

utm_validh	Output	validh. This signal that the higher order of the data_bus is valid or not. When Databus16_8 is zero then the value of this signal is undefined. The status of the lower order byte is determined by Txvalid and Rxvalid.
rx_dp_up	Input	Rx_DP. This is single ended receive data,positive terminal. The data is only valid if FsLsSerialMode is set to 1b
rx_dm_up	Input	Rx_DM. This is single ended receive data,negative terminal The data is only valid if FsLsSerialMode is set to 1b
rx_rcv_dp	Input	Rx_RCV. Receive data.The data is only valid if FsLsSerialMode is set to 1b
tx_enable_n_up	Output	Tx_Enable_N. This is the active low output enable signal
tx_dat_up	Output	Tx_DAT. Differential data at D+/D- output
tx_se0_up	Output	Tx_Se0. Force single ended zero
fslsserialmode_ up	Input	FsLsSerialMode. This signal is zero means FS and LS packets are sent using the parallel interface. One means FS and LS packets are sent using the serial interface
turn_power_on	Output	This signal is used for the external power supply to provide power to the downstream port
over_curr_indic	Input	This signal is asserted if an over current was detected

3.5 SoC Level Integration

3.5.1 Verification Environment

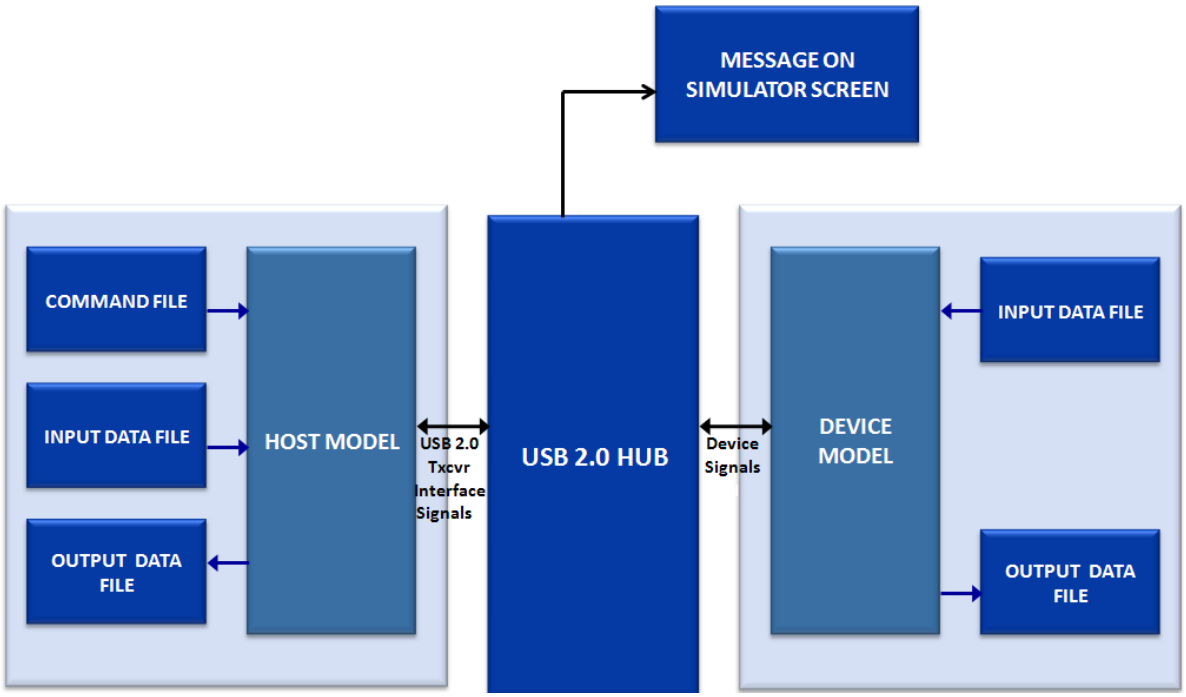

Figure 6: RTL Verification Environment

3.5.2 IP Deliverables

The IP package consists of the following:

- RMM-compliant synthesizable RTL design in Verilog
- Easy-to-use test environment
- Synthesis scripts
- Technical documents
- Simulation scripts

4 USB 2.0 Device IP

4.1 Overview

The USB 2.0 Device IP core is enables designers in the PC, mobile, consumer and communication markets to bring significant power and performance enhancements to the popular USB standard while offering compatibility with billions of USB-enabled devices currently in the market.

The Arasan USB 2.0 compliant Device core is available with an AHB/AXI, OCP or custom system bus interface. The USB2.0 device core supports 480 Mbits/s in High Speed (HS) mode and 12 Mbits/s in Full Speed (FS) mode of operation. Arasan provides designers with a comprehensive, silicon-proven configurable digital USB 2.0 Device solution that conforms to the USB 2.0 specification. It is designed to seamlessly integrate into any SoC design for an easy and cost effective solution. The Arasan USB 2.0 IP core supports up to 30 configurable IN/OUT non-control endpoints.

Each non-control endpoint has a controller for supporting interrupt, bulk and isochronous transfers. The dedicated control endpoint 0 handles USB defined command structure for Device Control. The USB 2.0 Device IP includes a multi-channel DMA that can be configured to access any endpoint through registers. Optionally, it can interface with an external DMA controller. The Device IP core provides an UTMI+/ULPI interface that allows connection to any USB 2.0 transceiver module.

4.2 Features

- High speed support: 480 Mbit/s
- Full speed support: 12 Mbit/s
- USB 2.0 Compliant
- High/Full speed support using 8/16 bit UTMI+/ULPI interface
- Master DMA implementation for each endpoint
- Optional PIO Mode for each endpoint (can be used for Interrupt endpoints)
- System bus Master/Target clock
- UTMI+ Interface Clock: 30/60 MHz
- Endpoint Configuration
- Configurable up to 15 Tx and Rx endpoints
- Configuration options: Bulk, control, isochronous, interrupt
- Dedicated control endpoint zero
- Configurable dual port RAM shared between endpoints
- USB Suspend/Resume support
- 32/64 bit AXI, AHB or OCP bus interfaces
- Interfaces between USB 2.0 bus and the User bus.
- Supports 16-bit UTMI+ /8-bit ULPI Interface to USB2.0 Transceiver.
- Support direct register addressing for all registers from the User bus.
- Software control for device remote-wakeup.
- User bus facilitates connection to common microcontrollers and DMA controllers.

- Three different modes of operation of an in-endpoint (Auto validation mode, manual validation mode, Fly mode)
- Supports standard 16-bit transfers on the User bus
- Supports Endpoint Maximum Packet Size up to 1024 bytes.

4.3 Architecture

4.3.1 Functional Block Diagram

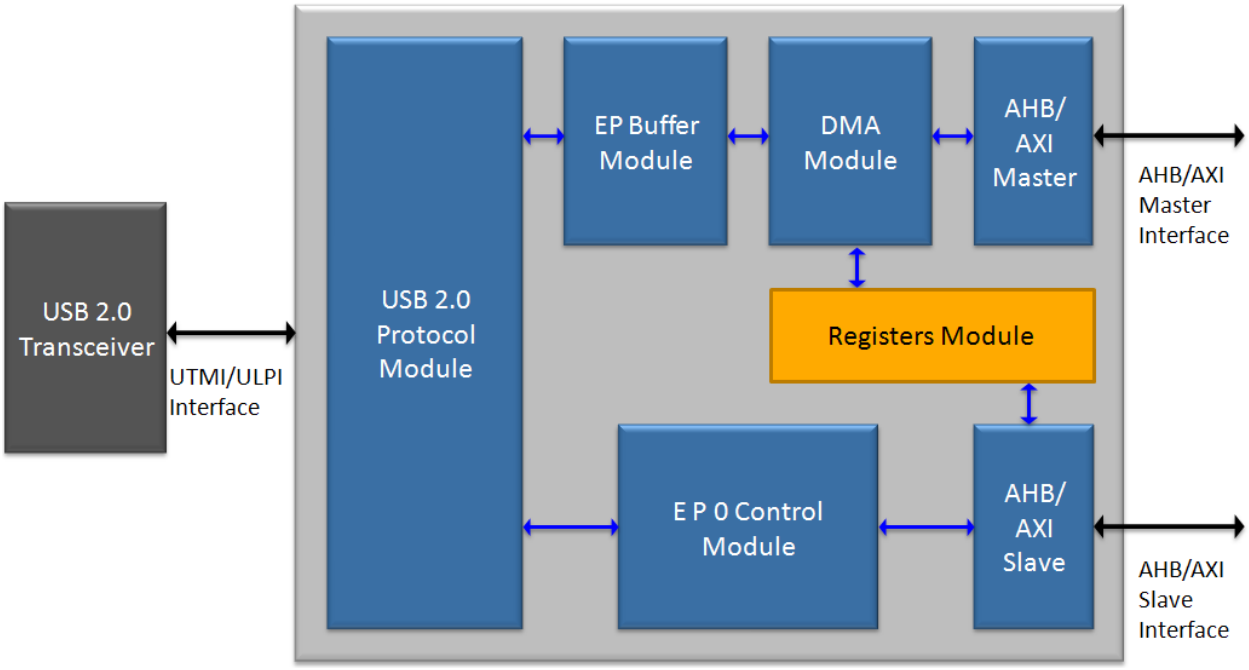

Figure 7: USB 2.0 Device IP Core Block Diagram

4.3.2 Functional Block Diagram Description

4.3.2.1 USB 2.0 Protocol Module

The parallel interface engine supports all USB protocols. The reset, suspend and resume sequence are taken care by this block. The PID decoding and PID encoding for the data packets and handshake are done in this block. The transmission as well as reception of packets is handled by this block. The CRC generation and checking for the transmission and reception of data packets are done by this module.

4.3.2.2 EP Buffer Module

Endpoint RAM operation is controlled by this Endpoint Buffer block. This block generates the read and writes control signals to access and dump the data inside the RAM. The read/ write pointers access is shared by USB side as well as backend side.

4.3.2.3 DMA Module

The DMA read and write operation control is taken care of this block. Read/Write operation is based on the internal FIFO status. Whenever the OUT endpoint FIFO data is available (Max packet size or short packet), this block will initiate the dma_req to the external interface to perform the DMA Write to the external memory. For IN endpoint whenever the endpoint FIFO is empty the dma_req signal will be initiated to perform DMA read from the external memory. This block uses clk30 for its operation.

4.3.2.4 Registers Module

This block holds all the internal posts registers namely Main Control, USB Control, Control Endpoint, Non-Control Endpoint and DMA registers. The register configuration is handled inside the block and also provides control signals to all other blocks. This block uses transceiver clk30 for its operation.

4.3.2.5 AHB/AXI Master Interface

This block performs AHB Master Function with respect to the Backend Interface. Whenever the device wants to perform a data transfer to/from system memory to the USB, it acts as a master.

4.3.2.6 AHB/AXI Slave Interface

This module provides the interface to access the function registers. With this interface, the configuration of End points and the handling of standard device requests of the Device controller take place through AHB/AXI bus.

4.3.2.7 EP0 Controller Module

Endpoint zero has special significance in a USB system. It is a CONTROL endpoint, and it is required by every USB device. The USB host uses special SETUP tokens to signal transfers that deal with device control; only CONTROL endpoints accept these special tokens. The USB host sends a suite of standard device requests over endpoint zero.

4.4 Pin Details

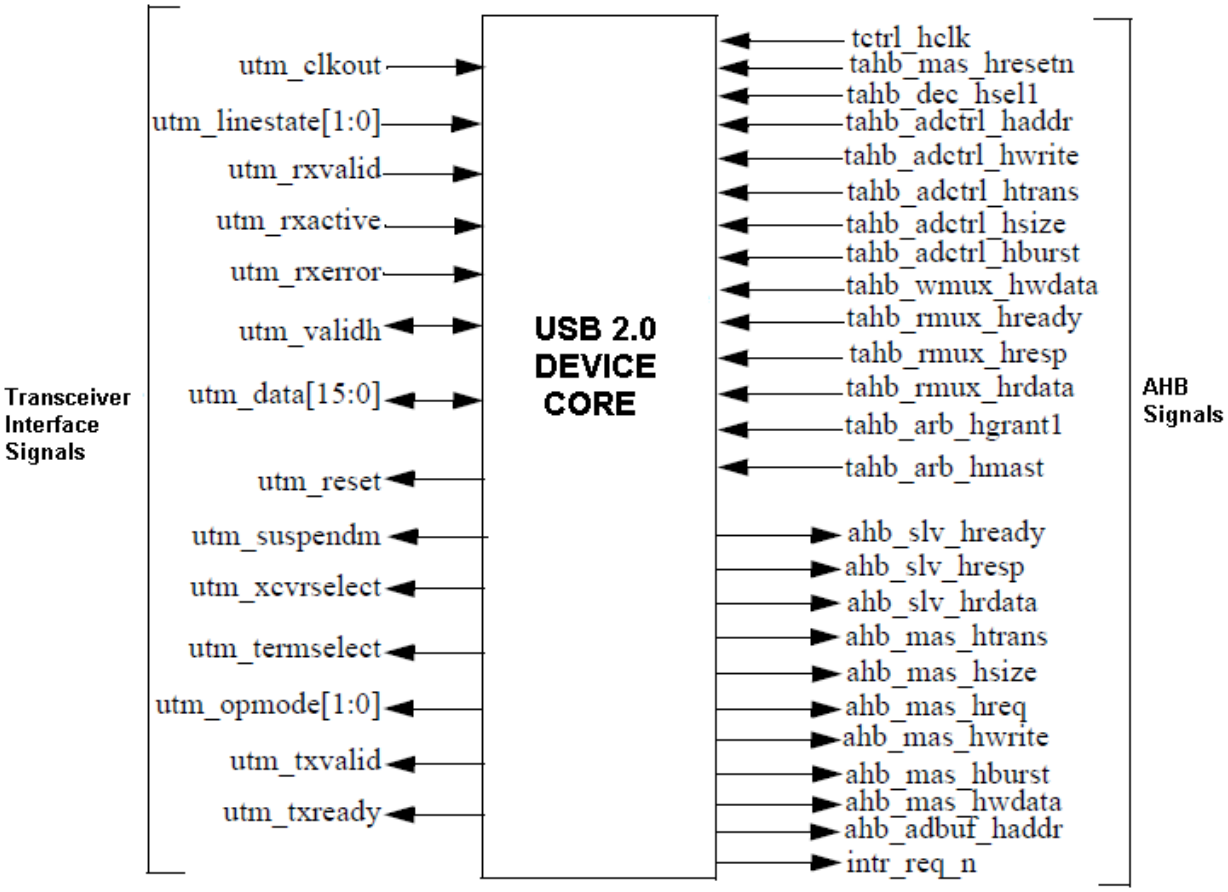

Figure 8: Pinout Diagram

4.5 Signal Interfaces

The USB 2.0 IP Core has the following interfaces:

- UTMI+/ULPI Interface
- AHB Interface

4.5.1 UTMI+/ULPI Interface

The high speed and full speed support in the USB end is provided by the UTMI+/ULPI interface. An 8 bit UTMI+ interface with 60 MHz is preferred, since the migration to ULPI is easy. Additionally, a UTMI+ to ULPI wrapper is provided as an option, in order to support low pin count external USB 2.0 PHY. Also, OTG support can be added for USB 2.0 using this interface.

Table 5: UTMI+ Interface Signals

Pin	Direction	Description
utm_clkout	IN	All the outputs get asserted synchronous to the rising edge of this clock only.
utm_rxvalid	IN	Signal indicating that the lower byte of data bus from the transceiver contains valid data.
utm_rxactive	IN	Signal indicating that SYNC has been detected and the packet bytes will follow. It gets deasserted after an EOP is detected
utm_rxerror	IN	Signal indicating that there is packet error in the received packet.
utm_txready	IN	Signal indicating that the transceiver has clocked the data from the bus and is ready for the next transfer on the bus. Used as handshake signal for utm_txvalid from device.
utm_linestate	IN	These signals reflect the current state of the USB bus
utm_validh	INOUT	When asserted, it means that the data[15:0] are valid
utm_reset	OUT	Signal used to reset all state machines of the transceiver.
utm_xcvrselect	OUT	This signal selects between the FS and HS transceivers. This signal along with the signal utm_termselect is used to indicate the speed of device operation
utm_termselect	OUT	This signal selects between the FS and HS terminations. This signal along with the signal utm_xcvrselect is used to indicate the speed of device operation.
utm_opmode	OUT	These signals are used by the transceiver to select between different operating modes as follows 00 - Normal Operation 01 - Non - driving 10 - Disable Bit - Stuffing and NRZI encoding 11 - Reserved
utm_databus16_8	OUT	Signal used to indicate 8-bit or 16-bit mode of operation to the transceiver. Tied to '1' to support 16-bit mode of operation
utm_suspendm	OUT	Signal used to place the transceiver in a suspend mode
utm_txvalid	OUT	Signal to indicate a packet transmission is to be initiated and the lower byte of data bus has valid data.

Table 6: AHB Interface Signals

Pin	Direction	Description
tctrl_hclk	IN	All outputs are synchronous to the rising edge of this clock. Inputs are sampled at the positive going edge of this clock.
tahb_mas_hresetn	IN	This is the system reset to be given to the core. It is an active low signal.
tahb_dec_hsel1	IN	This signal indicates that the current transfer is intended for the selected slave.
tahb_adctrl_haddr	IN	The 32-bit system address bus
tahb_adctrl_hwrite	IN	When HIGH this signal indicates a write transfer and when LOW a read transfer
tahb_adctrl_htrans[1:0]	IN	2-bit encoded signal that indicates to the slave about the type of transfer initiated by the master. The encoding is similar to the one specified in the AMBA-AHB specification. It is as follows:- 00b IDLE 01b BUSY 10b NONSEQ 11b SEQ
ahb_adctrl_hsize [2:0]	IN	This input indicates the size of the transfer. In other words, this indicates the no. of valid bytes in the data bus.
tahb_adctrl_hburst [2:0]	IN	This 3-bit encoded input provides the slave, the information on whether the transfer initiated is a burst and the kind of burst transfer
tahb_wmux_hwdata[31:0]	IN	This 32-bit input is the wdata bus using which the master writes data into the target register/memory
tahb_rmux_hready	IN	Active high signal indicating the readiness of the selected slave to complete the current transfer
tahb_rmux_hresp [1:0]	IN	2-bit encoded input that provides the information about the status of the current transfer
tahb_rmux_hrdata [31:0]	IN	32-bit data bus that provides the read data from the selected slave
tahb_arb_hgrant1	IN	Active high signal indicating that the core master has been granted use of the bus
tahb_arb_hmast[3:0]	IN	It is the Master number which is currently executing the transactions
ahb_slv_hready	OUT	When high indicates that the slave is ready to complete the current transfer. The slave shall de-assert this signal, in order to insert wait states on the bus
ahb_slv_hresp[1:0]	OUT	This 2-bit output bus is used by the slave to provide the status of the transfer. The encoding is as follows 00b OKAY

		01b ERROR 10b RETRY 11b SPLIT
ahb_slv_hrdata[31:0]	OUT	32-bit output data bus used by the slave to return data on master reads
ahb_mas_htrans[1:0]	OUT	A 2-bit encoded signal describing the type of transfer in progress viz., nonseq, seq, idle or busy. This is needed to check whether the address buffer is required to increment to the next sequential address or not
ahb_mas_hsize[2:0]	OUT	This input indicates the size of the transfer. In other words, this indicates the no. of valid bytes in the data bus
ahb_mas_hreq	OUT	Active high signal indicating to the arbiter that this master(core), needs use of the bus
ahb_mas_hwrite	OUT	Active high signal indicating that the master intends to do a write transfer
ahb_mas_hburst [2:0]	OUT	This 3-bit encoded input provides the slave, the information on whether the transfer initiated is a burst and the kind of burst transfer
ahb_mas_hwdata [31:0]	OUT	This 32-bit input is the wdata bus using which the master writes data into the target register/memory
ahb_adbuf_haddr [31:0]	OUT	32-bit address bus that is to drive the HADDR bus of the AHB
intr_req_n	OUT	This bit indicates the interrupt from the backend

4.6 SoC Level Integration

4.6.1 Verification Environment

The Verification Environment used to verify the USB2.0 DUT. This Verification Environment includes test suites to provide a complete verification solution for functional verification of USB2.0 DUT. This Verification Environment consists of:

- USB 2.0 BFM
- AHB Interface

The Verification Environment includes exhaustive built-in tests to verify the various functionalities of the USB2.0 DUT

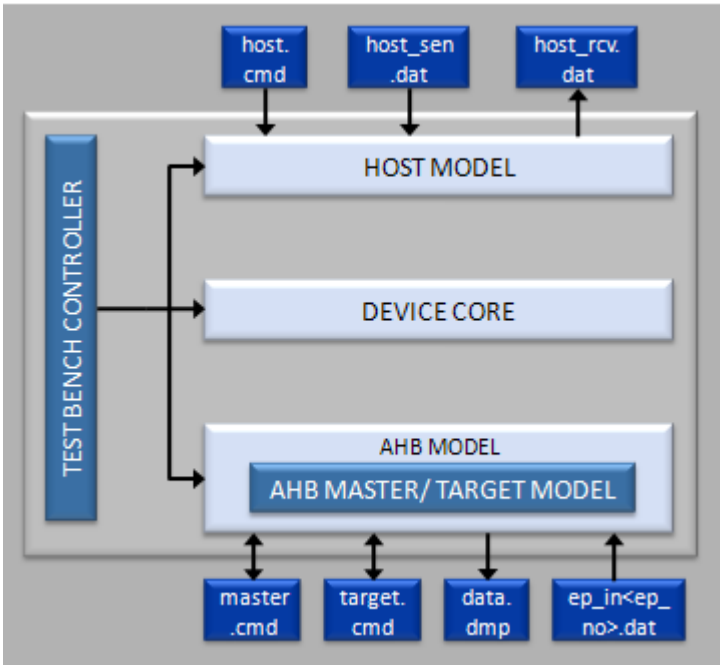

Figure 9: USB 2.0 Verification Environment

4.6.2 IP Deliverables

- RMM-compliant synthesizable RTL design in Verilog
- Easy-to-use test environment
- Synthesis scripts
- Technical documents
- Simulation scripts

5 USB 2.0 OTG IP

5.1 Overview

The Arasan USB 2.0 OTG IP Core is compliant with the OTG Supplement Rev. 1.0a. The USB 2.0 OTG core supports Host Controller, Device Controller, and OTG functionality. When operating as a host (or A-device), it supports 480 Mbit/s in High Speed (HS) mode, 12 Mbit/s in Full Speed (FS) mode, and 1.5 Mbit/s in Low Speed (LS) mode. When operating as a peripheral (or B-device), it supports HS and FS modes. Optional AHB, PCI, and Custom buses are available to provide high-speed connection to the USB interface.

Session Request Protocol (SRP) and Host Negotiation Protocol (HNP) are managed by the SRP/ HNP Control Logic. SRP allows a B-device to request an A-device to turn on Vbus to start a session, while HNP allows two connected dual-role devices to change roles and eliminates the needs for the user to switch cable connections. The Vbus Control Circuit supports the generation of data-line pulsing and Vbus pulsing methods when initiating the SRP as a B-device, the detection of both pulsing methods when acting as an A-device, and the sourcing of a minimum of 8 mA on Vbus. The Vbus Control Circuit also handles the pull-up and pull-down connections to D+ and during host/device role switching. The SRP/HNP Logic and Vbus Control Circuit control the operating mode of an USB port as either a host or peripheral. The Arasan USB 2.0 OTG port requires an external USB 2.0 transceiver with a standard UTMI interface.

The Arasan USB 2.0 OTG IP Core offers a high level of flexibility by allowing designers to implement multiple USB 2.0 OTG ports with a wide selection of processor interfaces. AHB, PCI, and Custom buses are available to provide a high-speed connection to the USB interface. The Arasan USB 2.0 OTG IP Core is augmented by the availability of the Arasan OTG Software Stack that supports host driver, device driver, class of devices, and Linux operating system. The Arasan USB 2.0 OTG IP Core is an RTL design in Verilog and VHDL that implements an USB 2.0 OTG controller on an ASIC or FPGA.

5.2 Features

- Compliant with OTG Supplement Rev. 1.0a
- USB 2.0 Compliant
- Supports 480 Mbit/s (HS), 12 Mbit/s (FS), and 1.5 Mbit/s (LS) as a host or A-device
- Supports 12 Mbit/s (FS), and 1.5 Mbit/s (LS) as a peripheral or B-device
- Supports Session Request Protocol (SRP) and Host Negotiation Protocol (HNP)
- Only one OTG port is implemented
- Supports data-line pulsing and Vbus pulsing as A-device or B-device
- High/Full speed support using 8/16 bit UTMI/ULPI interface
- USB transaction protocols are handled by hardware
- Minimum 8 mA Vbus and over 500 ma sourcing to peripherals with 5 V external power supply
- Direct addressing of all IP core registers from PCI, AHB, or Custom bus

- Suspend/resume support for power management
- Optional 200 MHz 32-bit AHB bus. Optional 33 MHz PCI Rev. 2.2 bus interface also available.
- Master DMA implementation for each endpoint
- Optional PIO Mode for each endpoint (can be used for Interrupt endpoints)
- System bus Master/Target clock
- UTMI Interface Clock: 30/60 MHz
- Endpoint Configuration
- Configurable up to 15 Tx and Rx endpoints
- Configuration options: Bulk, control, isochronous, interrupt
- Dedicated control endpoint zero
- Configurable dual port RAM shared between endpoints

5.3 Architecture

5.3.1 Functional Block Diagram

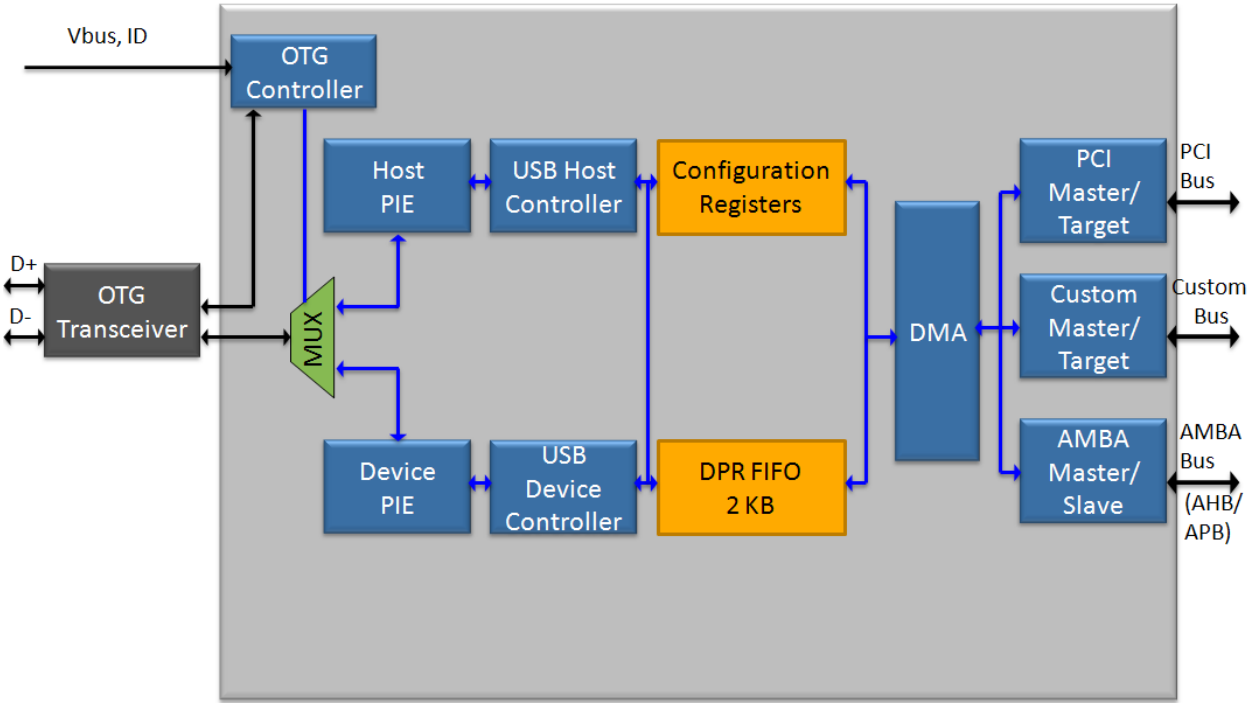

Figure 10: USB 2.0 OTG IP Core Functional Block Diagram

5.3.2 Functional Block Diagram Description

5.3.2.1 OTG Controller

The Vbus Control Circuit controls and monitors the voltage levels of Vbus, D+, and D- during SRP and HNP transactions. The SRP/HNP Logic of the OTG Controller consists of state machines that perform the SRP and HNP protocols as well as the configuration of an OTG port as a host or device. OTG descriptors are provided in the SRP/HNP Logic during enumeration.

5.3.2.2 Host PIE

The host parallel interface engine supports all USB protocols. This block takes care of Reset, Suspend and Resume sequence. The PID decoding and PID encoding for the data packets and handshake are done in this block. This block handles transmission as well as reception of packets. The CRC generation and checking for the transmission and reception of data packets are done here.

5.3.2.3 Device PIE

The device parallel interface engine supports all USB protocols. This block takes care of Reset, Suspend and Resume sequence. The PID decoding and PID encoding for the data packets and handshake are done in this block. This block handles transmission as well as reception of packets. The CRC generation and checking for the transmission and reception of data packets are done here. The time out is also checked for the packet reception.

5.3.2.4 USB Host Controller

Host controller operates with AHB as its back end. In this mode the on chip transceiver is used in UTMI+ mode. The Host Controller handles Device connect and Disconnect detection, Error detection, Data Toggling, Interrupt generation, Suspend and Resume detection. It also supports Host Negotiation protocol.

5.3.2.5 USB Device Controller

Device controller communicates with ARM CPU through an AHB interface. This interface supports the configuration of End points and handles the standard device requests of the Device controller. It also supports session request protocol which allows the B-device to request the A-device to turn on VBUS and start a session.

5.3.2.6 Configuration Registers

The Embedded 2.0 Host Controller has memory mapped register set called the Embedded Registers. These registers are dedicated to implement Host Controller functionality with respect to specific USB transfers. The Device core occupies a set of 33 registers that can be accessed by a CPU on the local bus.

5.3.2.7 DPR FIFO 2KB

OTG controller uses 8 dual port RAMs. Size of each two port RAM is 256 bytes, address is 8-bits width and the data bus is 8-bits width. Though the RAM is designed in common for device and host, the usage of RAM space differs when it acts as host or device.

- OTG Controller as Device: Address width - 11 bit, Data width - 8 bit
- OTG Controller as Host: Address width - 8 bit, Data width - 32 bit

5.3.2.8 DMA

The DMA read/write operation is based on the internal FIFO status. Whenever the OUT endpoint FIFO data is available (Max packet size or short packet), this block will initiate the dma_req to the external interface to perform the DMA Write to the external memory. For IN endpoint whenever the endpoint FIFO is empty the dma_req signal will be initiated to perform DMA read from the external memory

5.3.2.9 PCI Master/Target

The PCI Target/Master includes the DMA controller, configuration registers, PCI Power Management logic, and interrupt controller.

5.3.2.10 Custom Master/Target

Optional module for various bus interface to support various types of processors. It includes configuration registers, interrupt controllers and so on.

5.3.2.11 AMBA Master/Slave

This AHB/APB master is responsible for transferring data between the ARM Processor and USB 1.1 Device IP core. The AHB/APB slave Block consists of the Operational registers.

5.3.2.12 PCI Bus Interface

PCI Target/Master can be implemented that conforms to PCI specifications 2.2. The PCI Target/Master includes the DMA controller, configuration registers, PCI Power Management logic, and interrupt controller.

5.3.2.13 Custom Bus Interface

Optional module includes general purpose buses such as the 8-bit parallel bus, 16-bit parallel bus, I2C bus, and SPI buses. The wide variety of custom buses supported enable the selection of different types of processors such as the x86, SH3, 8051, ARC and other processors. Custom buses with special requirements can also be implemented.

5.3.2.14 AHB/APB Bus Interface

Optional AHB or APB bus can be used to provide a high-speed interface for the USB 1.1 Device IP Core. This AHB/APB master is responsible for transferring data between the ARM Processor and

USB 1.1 Device IP core. The AHB/APB slave Block consists of the Operational registers. Reading and writing of these registers are handled by the USB Device IP Core-AHB/APB bridge or ARM Processor.

5.4 Signal Interfaces

Table 7: Bus Interface Signals

Pin	Direction	Description
clk30	Input	All the outputs get asserted synchronous to the rising edge of this clock only
utm_rxvalid	Input	Signal indicating that the lower byte of data bus from the transceiver contains valid data
utm_rxvalidh	Input	Signal indicating that the higher byte of data bus from the transceiver contains valid data
utm_rxactive	Input	Signal indicating that SYNC has been detected and the packet bytes will follow. It gets deasserted after an EOP is detected
utm_rxerror	Input	Signal indicating that there is packet error in the received packet
utm_txready	Input	Signal indicating that the transceiver has clocked the data from the bus and is ready for the next transfer on the bus. Used as handshake signal for utm_txvalid from device
utm_linestate[1:0]	Input	These signals reflect the current state of the USB bus
utm_txvalidh	Output	When asserted, it means that the data[15:8] are valid
utm_dataout[15:0]	Output	The data out bus between the transceiver and the device core
utm_datain[15:0]	Input	The data in bus between the transceiver and the device core
utm_reset	Output	Signal used to reset all state machines of the transceiver
utm_xcvrselect[1:0]	Output	This signal selects between the FS and HS transceivers. This signal along with the signal utm_termselect is used to indicate the speed of device operation
utm_termselect	Output	This signal selects between the FS and HS terminations. This signal along with the signal utm_xcvrselect is used to indicate the speed of device operation
utm_opmode[1:0]	Output	These signals are used by the transceiver to select between different operating modes as follows 00 - Normal Operation 01 - Non - driving 10 - Disable Bit - Stuffing and NRZI encoding 11 - Reserved
utm_tx_bs_enable_h	Output	Indicates if the data on the dataout[15:8] needs to be bit stuffed or not.
		0: Bitstuffing is disabled

		1: Bitstuffing is enabled
utm_tx_bs_enable	Output	Indicates if the data on the dataout[7:0] needs to be bit stuffed or not. 0: Bitstuffing is disabled 1: Bitstuffing is enabled
utm_suspendm	Output	Signal used to place the transceiver in a suspend mode
utm_txvalid	Output	Signal to indicate a packet transmission is to be initiated and the lower byte of data bus has valid data
utm_a_valid	Input	Indicates if the session for an A-peripheral is valid (0.8V < Vth < 2V) 0b: Vbus < 0.8V 1b: Vbus > 2V
utm_b_valid	Input	Indicates if the session for an B-peripheral is valid (0.8V < Vth < 4V). 0b: Vbus < 0.8V 1b: Vbus > 4V
utm_sess_end	Input	Indicates if the voltage on Vbus (0.2V < Vth <0.8V). 1b : Vbus < 0.2V 0b : Vbus > 0.8V
utm_vbus_valid	Input	Indicates if the voltage on Vbus is at a valid level for operation (4.4V < Vth < 4.75V). 0b: Vbus < 4.4V 1b: Vbus > 4.75V
utm_id_dig	Input	Indicates whether the connected plug is a mini-A or mini-B. This is only valid when IdPullup is set to 1b. It must be valid within 50ms after IdPullup is set to 1b 0b: connected plug is a mini-A 1b: connected plug is a mini-B
utm_id_pullup	Output	Signal that enables the sampling of the analog Id line. 0b: Sampling of Id pin is disabled. IdDig is not valid 1b: Sampling of Id pin is enabled
utm_dppulldown	Output	This signal enables the 15k Ohm pull-down resistor on the DP line. 0b: Pull-down resistor not connected to DP 1b: Pull-down resistor connected to DP
utm_dmpulldown	Output	This signal enables the 15k Ohm pull-down resistor on the DM line. 0b: Pull-down resistor not connected to DM 1b: Pull-down resistor connected to DM
utm_chrg_vbus	Output	The signal enables charging Vbus. 1b: charge Vbus through a resistor (this has to be active for at least 30ms) 0b: do not charge Vbus through a resistor
utm_dischrg_vbus	Output	The signal enables discharging Vbus. 1b: discharge Vbus through a resistor (this has to be active for at least 50 ms)

		0b: do not discharge Vbus through a resistor
utm_drv_vbus	Output	This signal enables to drive 5V on Vbus 0b: do not drive Vbus 1b: drive 5V on Vbus
xcvr_disc_det_1	Input	This signal is used for all types of peripherals connected to it. It is only valid when DpPulldown and DmPulldown are 1b. If DpPulldown and DmPulldown are not 1b then the behaviour of HostDisconnect is undefined. As long as there is no peripheral connected, this signal will be 1b. If a peripheral is connected, then the value of this signal will be 0b.I
host	Output	This signal is given as a select signal to the otg model to read the host specific test pattern files. This signal is used only in simulation
device	Output	This signal is given as a select signal to the OTG model to read the device specific test pattern files. This signal is used only in simulation
clk_4x	Input	External clock Input of frequency 48Mhz.This clock is input to the size
rx_dp	Input	Single-ended receive data, positive terminal. The data is only valid if FsLs-SerialMode is set to 1b
rx_dm	Input	Single-ended receive data, negative terminal. The data is only valid if FsLs-SerialMode is set to 1b
rx_rcv	Input	Receive data. The data is only valid if FsLs- SerialMode is set to 1b.
fslsserialmode	Output	0b: FS and LS packets are sent using the parallel interface 1b: FS and LS packets are sent using the serial interface
tx_dat	Output	Differential data at D+/D- output
tx_se0	Output	Force Single-Ended Zero
tx_enable_n	Output	Active Low output enable signal. Signal Name Pin Type Description

Table 8: AHB Bus Interface Signals

Pin	Direction	Description
tctrl_hclk	Input	All outputs are synchronous to the rising edge of this clock. Inputs are sampled at the positive going edge of the clock
tahb_mas_hresetn	Input	This is the system reset to be given to the core. It is an active low signal.
tahb_dec_hsel1	Input	This signal indicates that the current transfer is intended for the selected slave.
tahb_adctrl_haddr[31:0]	Input	The 32-bit system address bus.
tahb_adctrl_hwrite	Input	When HIGH this signal indicates a write transfer and when LOW a read transfer.

tahb_adctrl_htrans [1:0]	Input	2-bit encoded signal that indicates to the slave about the type of transfer initiated by the master. The encoding is similar to the one specified in the AMBA-AHB specification. It is as follows: 00b IDLE 01b BUSY 10b NONSEQ 11b SEQ
ahb_adctrl_hsize [2:0]	Input	This input indicates the size of the transfer. In other words, this indicates the no. of valid bytes in the data bus.
tahb_adctrl_hburst [2:0]	Input	This 3-bit encoded input provides the slave, the information on whether the transfer initiated is a burst and the kind of burst transfer.
tahb_wmux_hwdata [31:0]	Input	This 32-bit input is the wdata bus using which the master writes data into the target register/memory.
tahb_rmux_hready_t	Input	Active high signal indicating the readiness of the selected slave to complete the current transfer.
tahb_arb_hmast[3:0]	Input	It is the Master number which is currently executing the transactions.
ahb_slv_hready	Output	When high indicates that the slave is ready to complete the current transfer. The slave shall de-assert this signal, in order to insert wait states on the bus.
ahb_slv_hresp[1:0]	Output	This 2-bit output bus is used by the slave to provide the status of the transfer. The encoding is as follows 00b OKAY 01b ERROR 10b RETRY 11b SPILT
ahb_slv_hrdata[31:0]	Output	32-bit output data bus used by the slave to return data on master reads.
ahb_mas_hburst[2:0]	Output	This 3-bit encoded input provides the slave, the information on whether the transfer initiated is a burst and the kind of burst transfer.
intr_req_n	Output	This bit indicates the interrupt from the backend.

5.5 SoC Level Integration

5.5.1 Verification Environment

USB 2.0 OTG IP Core is proven on:

- Arasan's FPGA based platform
- In production in multiple customer designs

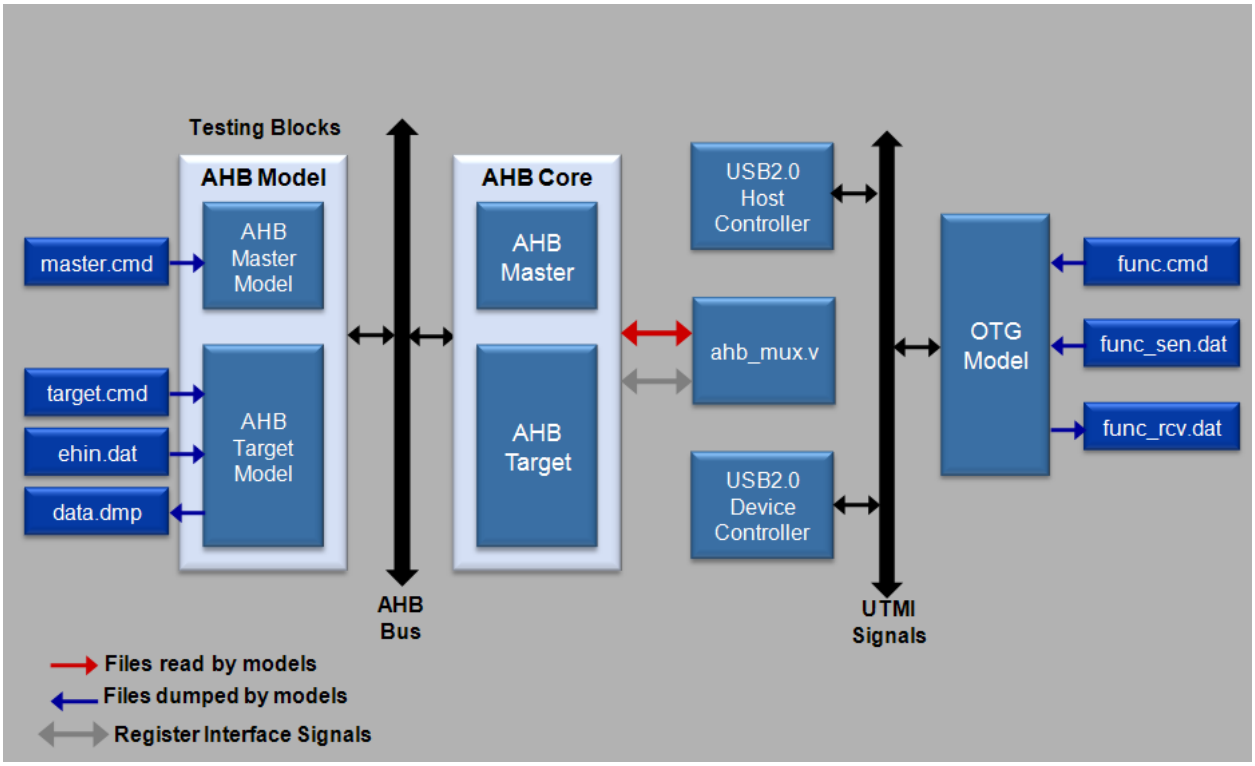

Figure 11: OTG Testing Environment

5.5.2 IP Deliverables

The IP package consists of the following:

- RMM-compliant synthesizable RTL design in Verilog
- Easy-to-use test environment
- Synthesis scripts
- Technical documents
- Simulation scripts

6 USB 2.0 PHY IP

6.1 Overview

The USB 2.0 PHY performs low-level protocol and signaling functions. While transmitting, the PHY serializes data, generates Synchronize (SYNC) and End-of-Packet (EOP) packet fields, and performs bit stuffing and Non-Return-to-Zero Inverted (NRZI) encoding. While receiving data, the PHY recovers incoming data and clock, de-serializes data, strips SYNC and EOP fields, and performs bit un-stuffing and NRZI decoding.

The USB 2.0 PHY is a full-featured on-chip physical transceiver. It has Electro Static Discharge (ESD) protection and fully supports all OTG and host functionality. On-board clock generation and PLL blocks provide for accurate, high-speed data transmission from and to the transceiver. When this USB PHY is used, a minimal number of external components are required.

The Arasan USB 2.0 PHY IP is a transceiver, compliant with the USB 2.0 Transceiver Macrocell Interface Plus (UTMI+) level 3 specifications, for use with host, embedded host, On-the-Go (OTG) and function controllers. Its high speed, mixed-signal circuitry supports 480 Mb/s USB 2.0 High Speed (HS) traffic, while remaining backward compatible with USB 1.1 legacy protocol for 12Mb/s Full Speed (FS) traffic and 1.5Mb/s Low Speed (LS) traffic.

6.2 Features

- Complies with USB specifications, rev. 2.0 and 1.1
- Complies with UTMI+ specification, level 3, rev. 1.0
- Supports 480Mb/s (HS), 12Mb/s (FS) and 1.5MB/s (LS) serial data transmission rates
- Supports 8-bit unidirectional Parallel Interface Engine (PIE) bus for HS, FS and LS modes, and Serial Interface Engine (SIE) for FS and LS modes
- Supports USB 2.0-defined test modes
- Supports OTG USB 2.0
- Supports single-ended data interface
- Digital I/O for Tx and Rx USB OTG cable status
- Rail-to-rail common mode differential receiver
- NRZI encoding and decoding
- SYNC and EOP generation and detection
- Bit stuffing and un-stuffing with error detection
- Supports full junction temperature range of -40°C to 125°C
- Dual 3.3V/1.2V supply
- Integrated HS and FS/LS terminations
- Area <1mm2 at 130nm
- On-chip decoupling capacitors to minimize digital switching noise

6.3 Architecture

6.3.1 System on Chip Description

The system block diagram in figure 1 depicts the USB 2.0 PHY integrated into a System-on-Chip (SoC) ASIC. The PHY acts like a bridge between USB-compliant external device(s) and the on-chip USB core. Communications with external devices, such as host, hub or peripherals, are conveyed via the serial USB 2.0 bus, in half duplex mode. On-chip, communications between the PHY and USB 2.0 core are handled via a parallel UTMI+ bus within the PHY, selectable to 8-bit or 16-bit width.

The USB 2.0 core receives data from, and transmits data to, the PHY. Per USB protocol, the core distributes data to, or receives data from, the appropriate End Point. End Points are buffers that store multiple bytes of data. The buffers are uniquely addressable. Upon power-up or detection of device attachment/removal, the USB 2.0 core, in conjunction with the host, establishes which particular device address will be allowed communication with the external USB device.

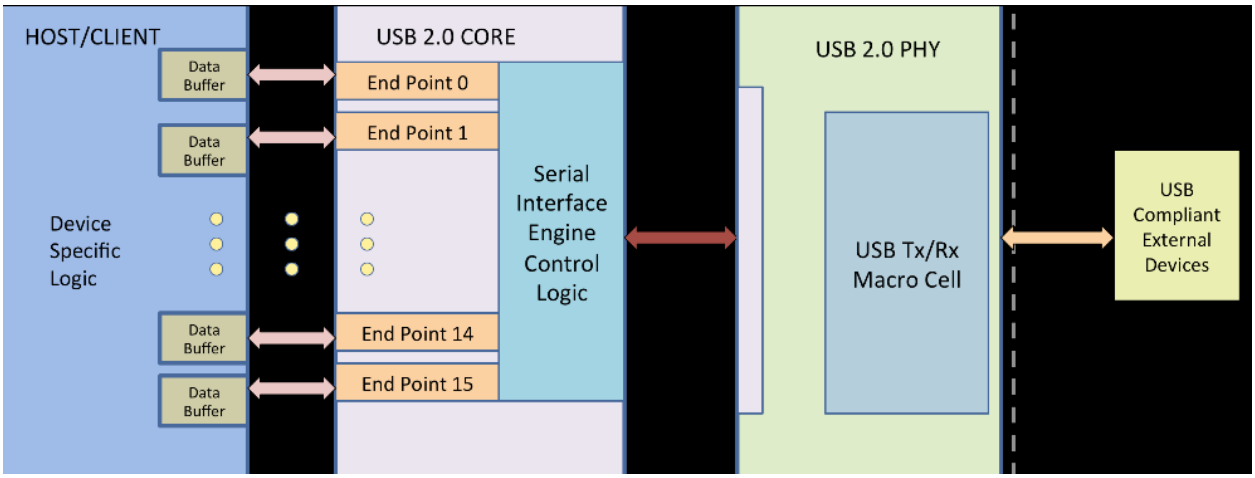

Figure 12: System on Chip Block Diagram

6.4 Functional Block Diagram

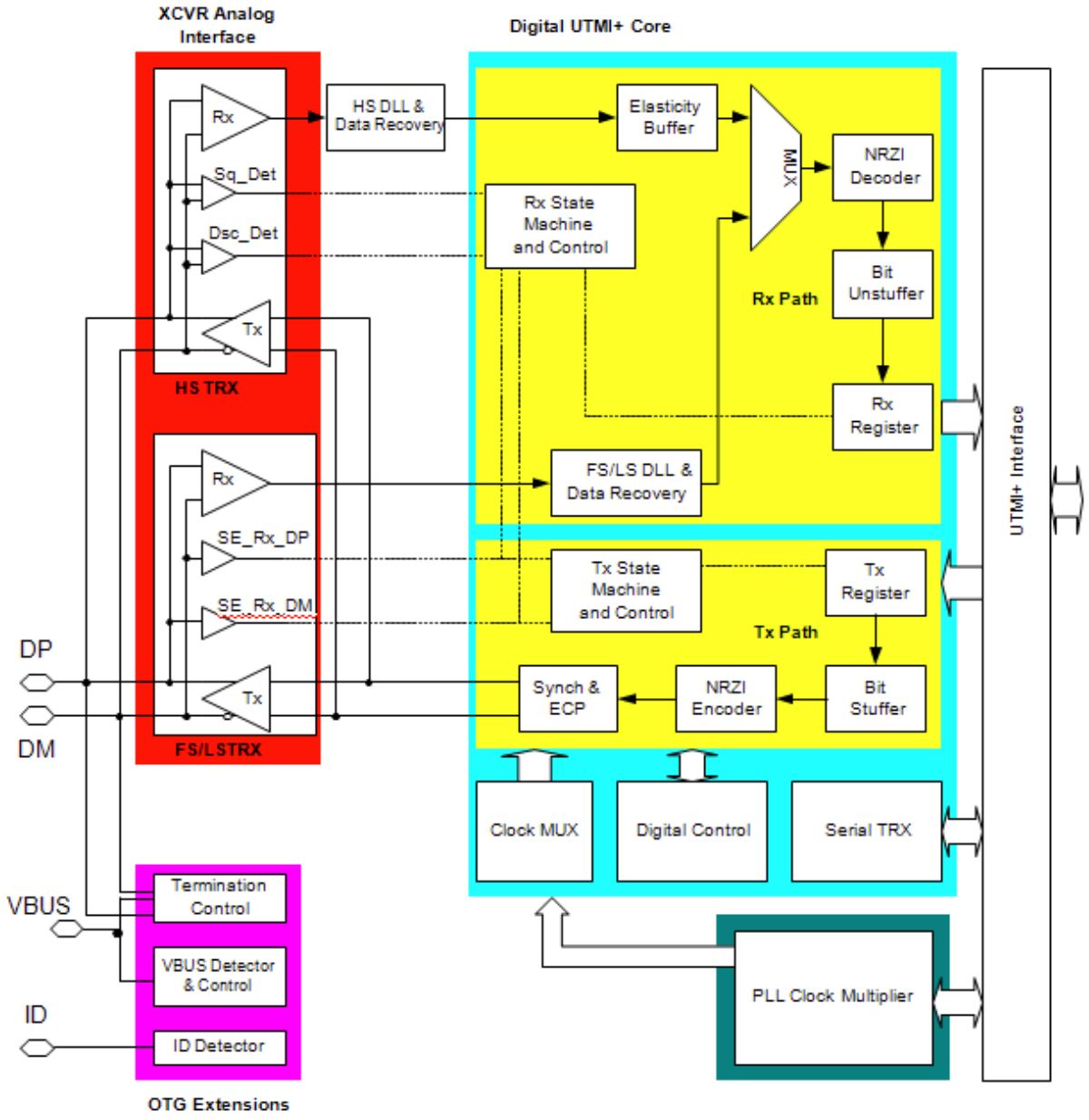

Figure 13: Functional Block Diagram

6.4.1 Functional Block Description

The USB 2.0 PHY primarily consists of five major blocks: a transceiver analog Rx/Tx interface, digital USB UTMI+ core, PLL and OTG circuitry.

6.4.1.1 XCVR Analog Interface

The transceiver analog interface consists of two sub-blocks, HS TRX and FS/LS TRX. HS TRX handles high speed traffic and FS/LS TRX handles full and low speed traffic. Both sub-blocks contain receiver and driver functions, though only receive or transmit mode will be enabled at any given time.

6.4.1.2 HS TRX

The high-speed transceiver (HS-TRX) is active only in HS mode. It contains the low level analog circuitry required to physically interface USB 2.0 signaling to the USB Dataline Plus (DP) and Dataline Minus (DM) signal lines. It contains the HS differential data transmitter and receiver, performs HS transmission envelope detection and host disconnection detection.

Tx

Tx is a HS transmit driver. When enabled, data from the transmit data path will be driven to the DP/DM signal lines. The HS transmit driver is active only when “transmit” is asserted, the XcvrSelect input is in HS transceiver enabled mode, and transmit state machine has data to send.

Rx

Rx is the HS differential receiver. When enabled, received HS data are multiplexed through the “receive” data path to the “receive” shift and hold registers. The receiver is active only in HS mode.

Sq_Det

Sq_Det is the transmission envelope (Squelch) detector. The transmission envelope detector indicates data are invalid when the amplitude of the differential signals at a receiver’s inputs falls below the squelch threshold. It must indicate a squelch state when the signal drops below 100mV differential amplitude, and it must indicate the line is not in the squelch state when the signal exceeds 150mV differential amplitude.

DSC_Det

DSC_Det is the disconnect envelope detector. The disconnect envelope detector is active only in host mode. It detects the high speed disconnect state on the line. Disconnection must be indicated when the amplitude of the differential signal at the downstream-facing driver’s connector is >625mV, and must not be indicated when the signal amplitude is <525mV.

6.4.1.3 FS/LS TRX

The full and low speed transceiver is active only in FS or LS modes. It contains the circuitry required to send and receive FS or LS data on the USB bus. It also supports reset, suspend and resume detection through dataline single-ended receivers.

Tx

Tx is a FS/LS transmitter. When enabled, data from the transmit data path is driven to the DP/DM signal lines. The FS/LS transmitter is active only when transmit is asserted, the XcvrSelect input is in FS or LS transceiver enabled mode, and transmit state machine has data to send.

Rx

Rx is a FS/LS differential receiver. When enabled, received FS or LS data are multiplexed through the receive data path to the receive shift and hold registers. The receiver is active only in FS or LS mode. SE_Rx_DP and SE_Rx_DM “SE_Rx_DP” and “SE_Rx_DM” are the Single-ended receivers. The single ended receivers are used for detection of full speed and low speed signaling. One receiver handles data line plus signaling, and the other handles data line minus.

6.4.1.4 HS DLL and Data Recovery

The High Speed (HS) delay line PLL is extracts clock and data from the data received over the USB 2.0 interface for reception by the receive deserializer. The HS DLL has two outputs: the extracted clean data, and the recovered data clock synchronized with the extracted data.

6.4.1.5 Digital UTMI+ Core

The digital USB UTMI+ core consists of Rx and Tx Paths, a clock MUX, digital control logic and serial TRX block.

6.4.1.6 Tx Path

The digital core Tx path includes blocks that perform SYNC and EOP generation, data encoding, bit stuffing and serialization. In addition, a Tx state machine manages communication with the controller.

Tx Register

The Tx shift/hold register reads parallel data from the parallel application bus interface and serializes it for transmission over the serial USB bus. The module consists of an 8-bit primary shift register for parallel/serial conversion and 8-bit hold register for buffering data awaiting serialization.

Bit Stuffer

In order to ensure adequate signal transitions, the transmitter performs bit stuffing when sending a packet over the USB bus. A “0” is inserted after each occurrence of six consecutive and “1”s in the data stream prior to NRZI encoding, in order to force a transition in the NRZI data stream.

NRZI encoder

The NRZI encoder encodes the HS, FS, or LS serial transmitted data. A “0” is encoded as a state transition and a “1” is encoded as no state transition.

Transmit State Machine and Control

The transmit state machine satisfies the state diagram shown in figure 3, controls communication between the controller and the PHY Tx path, synchronizes the data with the SYNC and EOP frames, and supports LF, FS and HS modes.

6.4.1.7 Rx Path

The digital core Rx path includes blocks that perform SYNC and EOP detection and stripping, data encoding, bit un-stuffing and de-serialization. In addition, an Rx stat machine manages communication with the digital controller. FS/LS data and clock are recovered in this block.

FS/LS DLL and Data Recovery

This delay line PLL extracts data from the FS/LS serial data stream. Output data are synchronized with the local clock.

Elasticity Buffer

The elasticity buffer compensates for differences between transmitting and receiving clocks. It is used to synchronize the HS extracted data with the PLL internal clock.

MUX

The Mux block allows data from the HS or FS/LS receivers to be routed to the shared “receive” logic. The state of the Mux is determined by the XcvrSelect input.

NRZI decoder

The NRZI decoder decodes incoming HS or FS NRZI-encoded data. A change in level is decoded as “0,” and no change in level is decoded as “1.”

Bit un-stuffer

The bit un-stuffer recognizes the stuffed bits in a data stream and discards them. It also detects bit stuffing error, which it interprets as HS EOP.

Rx Register

The Rx shift/hold register de-serializes received data recovered by the DLL and transmits 8-bit parallel data to the application bus interface. It consists of an 8-bit primary shift register for serial to parallel conversion and an 8-bit hold register for buffering the final de-serialized data byte.

Rx State Machine and Control

The receiver state machine satisfies the state diagram shown in figure 4, controls communication between the controller and the PHY Rx path, strips the SYNC and EOP frames from the data and supports LS, FS and HS modes.

6.4.1.8 Clock MUX

The clock multiplexer supplies both the Tx and Rx paths with the adequate bit clock, depending on the XcvrSelect signal, and ensures smooth clock switching during the XcvrSelect=11b mode. It also includes clock gating and power-down features.

6.4.1.9 Digital Control

The digital control logic block controls, enables and disables the different blocks within the system.

6.4.1.10 Serial TRX

The serial TRX interface makes it possible to reuse existing FS/LS host controller IP without changing the interface. Using this interface, packets are conveyed serially.

6.4.1.11 PLL Clock Multiplier

This module generates the appropriate internal clocks for the UTM and CLK output signals. All data transfer signals are synchronized with the CLK signal. It is composed of an off-chip crystal and on-chip clock multiplier.

6.4.1.12 External crystal

The external crystal consists of a precise resonance frequency crystal and a crystal oscillator. At the designer's option, the crystal oscillator may be integrated on-chip or reside off-chip. The crystal/crystal oscillator should provide a precise clock of 12MHz with accuracy of ±100ppm.

6.4.1.13 Clock Multiplier

The UTM interface is a uni-directional 8-bit parallel interface, and the frequency of the CLK signal is 60MHz. All data transfer signals are synchronized with the CLK signal. A "CLK usable" signal is internally implemented, blocking any CLK transitions until usable. In addition to the CLK signal, the clock multiplier provides three additional clocks, at 480MHz, 12Mhz and 1.5MHz.

6.4.1.14 OTG Extensions

The OTG circuitry provides the capability to dynamically switch between host and peripheral, enabling dual-role device behavior and point-to-point communication. OTG functions include VBUS generation and detection, ID detection, and terminations control.

6.4.1.15 ID detector

This block produces the ID signal that is used to indicate the state of the ID pin on the USB mini receptacle, in order to determine whether 1) the plug is connected, and 2) whether the device default state is "A" or "B."

6.4.1.16 VBUS Detector and Termination Control

The VBus detector employs comparators to monitor and sense the voltage on the USB bus power line. VBus signaling and discharging are also performed by VBus pull-up and pull-down resistors.

6.5 Pin Diagram

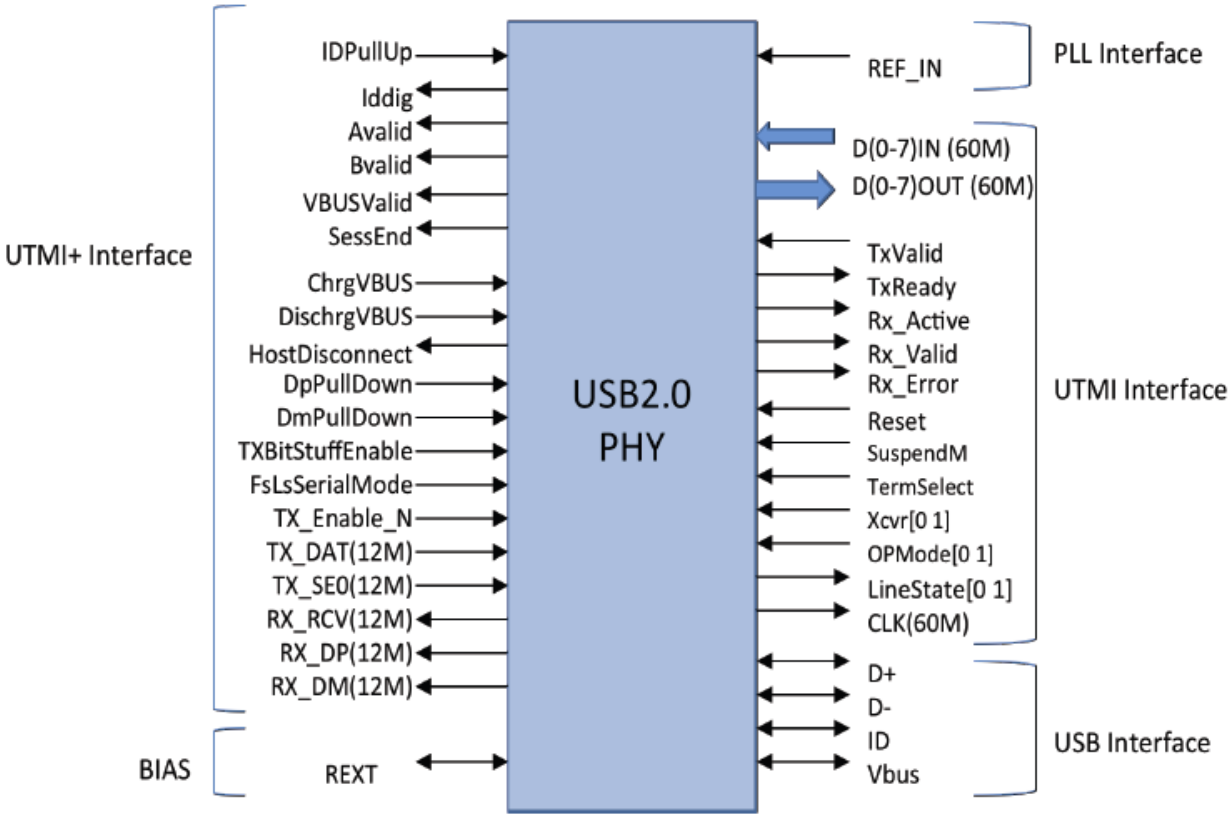

Figure 14: Pin Diagram

6.5.1 Pin Description

Table 9: Pin Description

Pin	Direction	Description
USB Interface		
D+, D–	A-IO	Data+, Data–. Differential data bus conforming to the USB standard
ID	A-IO	USB plug indicator
Vbus	Power	USB power line
PLL		
REF_IN	D-In	12MHz crystal input
BIAS		
REXT	A-IO	12.4 KΩ reference external resistance
UTMI		

D(0…7)IN (60M)	D-In	8 bits parallel input data bus
D(0…7)OUT(60M)	D-Out	8 bit parallel output data bus
TxValid	D-In	Transmit data valid
TxReady	D-Out	Transmit data ready
Rx_Active	D-Out	Receive data active
Rx_Valid	D-Out	Receive data valid
Rx_Error	D-Out	Error in received data
Reset	D-In	Reset signal to UTM state machines
SuspendM	D-In	Suspend Mode enable
TermSelect	D-In	Termination select between FS/LS and HS Terminations
Xcvr[0 1]	D-In	Transceiver Select between FS/LS and HS Transceivers
OPMode[0 1]	D-In	Operational mode selector between various modes
LineState[0 1]	D-Out	USB Bus Line State Monitor CLK(60M) CLK 60 MHz output clock
UTMI+		
IDPullUp	D-In	Signal that enable analog ID line sampling
Iddig	D-Out	USB Plug Indicator Output
Avalid	D-Out	Avalid Comparator output
Bvalid	D-Out	Bvalid Comparator output
VBUSValid	D-Out	VBUSvalid Comparator output
SessEnd	D-Out	Sessionvalid Comparator output
ChrgVBUS	D-In	VBUS charging enable
DischrgVBUS	D-In	VBUS discharging enable
HostDisconnect	D-Out	Indicates the disconnection of a peripheral
DpPullDown	D-In	Enable DPLUS Pull Down resistor
DmPullDown	D- In	Enable DMINUS Pull Down resistor
TXBitStuffEnable	D-In	Enables Bit stuffing if OPMODE is 11
FsLsSerialMode	D-In	Enables Serial Mode communication with a USB1.1 controller
TX_Enable_N	D-In	Active low output enable signal (Serial Mode)
TX_DAT(12M)	D-In	Differential data at D+/D- output (Serial Mode)
TX_SE0(12M)	D-In	Force Single-Ended zero (Serial Mode)
RX_RCV(12M)	D-Out	Differential Receive data (Serial Mode)
RX_DP(12M)	D-Out	Single Ended receive data, positive terminal (Serial Mode)
RX_DM(12M)	D-Out	Single Ended receive data, negative terminal (Serial Mode)
Supply		
AGND12	Power	0V analog Ground
AVDD12HF	Power	1.2 analog positive supply for CDR
AGND12HF	Power	0V analog Ground for CDR
AVDD33	Power	3.3V analog positive supply
AGND33	Power	0V analog Ground
DVDD33	Power	3.3V digital pad positive supply
DGND33	Power	0V digital pad Ground
DVDD12	Power	1.2V digital core positive supply

DGND12	Power	0V digital core Ground
VDD_ESD	Power	1.2V digital core positive supply
VSS_ESD	Power	0V digital core Ground
Power Cut	BRK	Break Power Supplies + Filler Pads
ESD_Power	Power	Dummy power pads for ESD rules

6.6 SoC Level Integration

6.6.1 Verification Environment

The USB 2.0 UTMI+ PHY has been integrated into reference platforms for interoperability testing and validation of USB compliance.

6.6.2 IP Deliverables

The IP package consists of the following:

- GDSII layout and layer map
- Place-and-route views (.lib, .lef)
- LVS and DRC verification reports
- Verilog SIM Model (NC-Verilog)
- Gate-level Netlist and SDF timing
- Test Guidelines, Layout Guidelines and Application Notes

7 USB 2.0 HSIC PHY IP

7.1 Overview

Universal Serial Bus (USB) is the ubiquitous peripherals interconnect of choice for large number of computing and consumer applications. Many systems provide a comprehensive set of drivers to support all commonly available USB peripherals. This enables consumer to purchase and use USB peripherals without having to install a new driver, thus strengthening the popularity of USB.

It is becoming increasingly attractive to use USB as a chip-to-chip interconnect within a product (without use of external cables and connectors). However, because USB was designed to enable hot-plugging of peripherals over cables up to five meters in length, there are certain power and implementation issues that are not attractive for many chip-to-chip interconnect solutions. To better meet the needs of a USB chip-to-chip interconnect, the High Speed Inter Chip (HSIC) PHY accomplishes this by removing the analog transceivers, thus reducing complexity, cost and manufacturing risk.

7.2 Features

- High-Speed 480Mbps data rate only
- Source-synchronous serial interface
- No power consumed unless a transfer is in progress.
- Maximum trace length of 10cm
- No hot Plug-n-Play support, no hot removal/attach
- Signals driven at 1.2V standard LVCMOS levels
- Designed for low-power applications
- No high-speed chirp protocol, the HSIC interface is always operated at high-speed
- HSIC host or peripheral can be powered in any order

7.3 Architecture

7.3.1 Functional Description

HSIC is a 2-signal (strobe, data) source synchronous serial interface which uses 240 MHz DDR signaling to provide High-Speed 480Mps USB transfers which are 100% host driver compatible with traditional USB cable-connected topologies. Full-Speed (FS) and Low-Speed (LS) USB transfers are not directly supported by the HSIC interface (a HSIC enabled hub can provide FS and LS support, as well as IC_USB support).

7.3.2 Functional Block Diagram

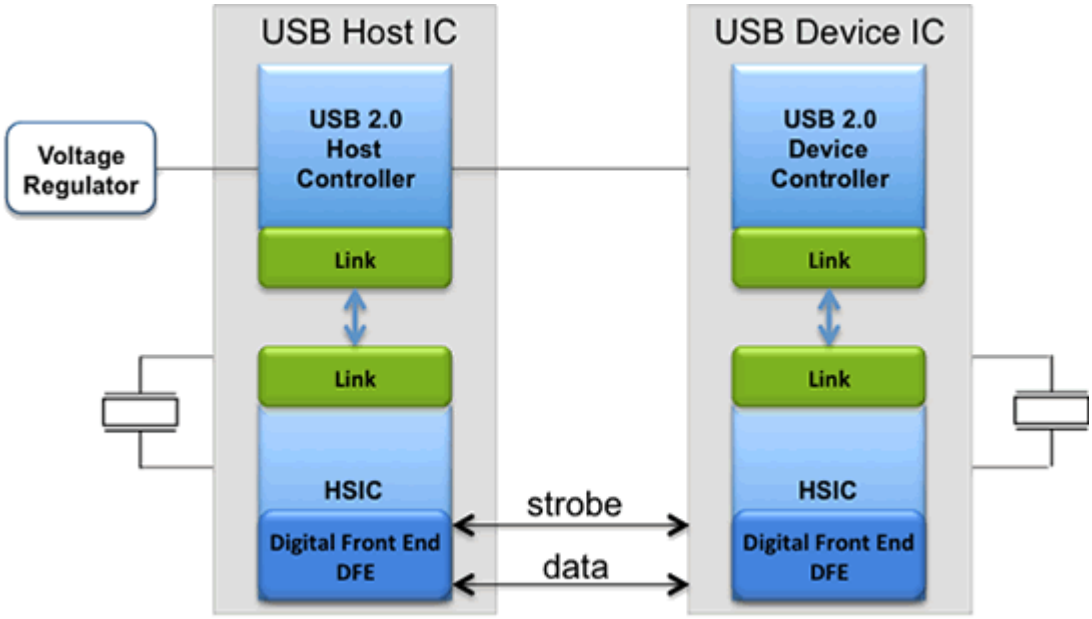

Figure 15: USB 2.0 HSIC PHY Functional Block Diagram

7.3.3 Functional Block Diagram Description

7.3.3.1 Link (UTMI+)

This block interfaces with the device/host controller and to the HSIC internal module. It generates UTMI+ specific signals adhering to the UTMI+ specification at the (host/device) controller end and passes these signals to the HSIC sub block.

7.3.3.2 High Speed Inter Chip (HSIC)

This block handles the NRZI encoding, bit stuff insertion and serialization of the data during transmission. It handles the NRZI decoding, bit stuff detection and parallelization of the data during reception.

During data transmission:

- The parallel input data from the link layer is serialized for transmission over the HSIC interface.
- It takes care of the bit-stuffing logic that detects six continuous ones in the bit stream and inserts a zero bit.
- NRZI encoding is done finally on the data before transmission.

During data reception:

- The serial data is parallelized and sent to the link interface.

- It takes care of the bit-stuffing logic that detects six continuous ones in the bit stream and drops the zero bit during reception.
- NRZI decoding is done when the data is received during reception.

7.3.3.3 Digital Front End (DFE)

This block handles:

- The Dual Data Rate signaling of the data and strobe that is being transmitted or received to/from the HSIC interface
- Driving the bi-directional strobe and data lines appropriately based on data transmission/reception
- PLL clock source for
  - 240Mhz DDR Clock Generation
  - 60Mhz UTMI+ Clock
- Transmit buffers for Strobe, Data and Transmit Enable control signals
- Receive Buffers
- I/O pads for Data, Strobe and PLL clock reference

7.4 Pin Diagram

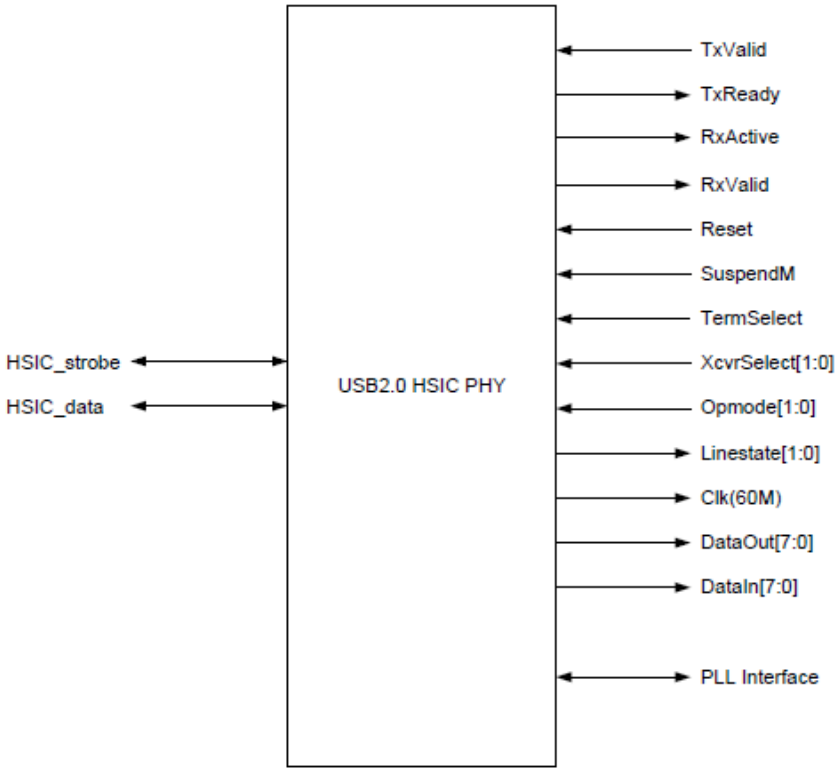

Figure 16: USB 2.0 HSIC PHY PIN Diagram

7.5 SoC Integration

7.5.1 Verification Environment

- Comprehensive suite of simulation tests for ease of System on Chip (SoC) integration
- Verification components and test files provided
- Verification environment and test suite well documented

7.5.2 IP Deliverables

- GDSII layout and layer map
- Place-and-route views (.lib, .lef)
- Layout Versus Schematic (LVS) and Design Rule Check (DRC) verification reports
- Verilog SIM Model (NC-Verilog)
- Gate-level Netlist and Standard Delay Format (SDF) timing
- Test guidelines, layout guidelines and application notes

8 Services & Support

8.1 Global Support

Arasan Chip Systems provide global support to its IP customers. The technical support is not geographically bound to any specific site or location, and therefore our customers can easily get support for design teams that are distributed in several locations at no extra cost.

8.2 Arasan Support Team

Our technical support is provided by the engineers who have designed the IP. That is a huge benefit for our customers, who can communicate directly with the engineers who have the deepest knowledge and domain expertise of the IP, and the standard to which it complies.

8.3 Professional Services & Customization

At Arasan Chip Systems we understand that no two Application Processors are the same. We realize that often the standard itself needs some tweaks and optimizations to fit your design better. Sometimes, the interface between the IP blocks and your design need some customization. Therefore, we provide professional services and customization to our IP customers. We do not sell our IP blocks as “black box” that cannot be touched. Please contact us for more details on our customization services.

8.4 The Arasan Porting Engine

Analog IP blocks, such as eMMC 5.1 HS400 PHY, are designed for a specific Fab and process technology. Arasan’s analog design team, utilizing its deep domain expertise and vast experience, is capable of porting the PHYs into any specific process technology required by the customer. That is “The Arasan Porting Engine”.

8.5 Pricing & Licensing

Arasan charges a one-time licensing fee, with no additional royalties. The licensing fee gives the right to use our IP for 1 project. Licensing fee for additional projects, using the same IP, is discounted. We also offer unlimited-use license. For any additional information regarding pricing and licensing – please contact our sales at: sales@arasan.com.