OpenCores.Org

USBHostSlave IP Core Specification

Author: Steve Fielding
sfielding@base2designs.com

Rev. 0.7
June 8, 2005

This page has been intentionally left blank.

Revision History

Rev.	Date	Author	Description
0.1	10/01/04	Sfielding	Created
0.2	12/05/04	Sfielding	Fixed frame_num register description
0.3	01/11/05	Sfielding	Added host slave mode register
0.4	01/25/05	Sfielding	Added Version number register, and changed host and slave frameNum registers
0.5	02/10/05	Sfielding	Added instructions for clearing interrupt registers
0.6	02/25/05	Sfielding	Added isochronous mode control. Altered descriptions of endpoint control and status registers
0.7	06/04/05	Sfielding	Added SOF timer register Split version number into major, and minor Added HOST_RX_FIFO_DATA reg to register memory map

Contents

1 Introduction

USBHostSlave is a USB 1.1 Host and Device IP core.

- Supports full speed (12Mbps) and low speed (1.5Mbps) operation.
- USB Device has four endpoints, each with their own independent FIFO.
- Supports the four types of USB data transfer; control, bulk, interrupt, and isochronous transfers.
- Host can automatically generate SOF packets.
- 8-bit Wishbone slave bus interface.
- FIFO depth configurable via parameters.

2 Architecture

The USBhostSlave IP core consists of five major functional blocks (see Figure (1)).

USBSerialInterfaceEngine – Supports the lowest level of the USB 1.1 protocol layer. On the transmit path, USBSerialInterfaceEngine implements, sync insertion, CRC calculation and insertion, parallel to serial conversion, bit stuffing, and NRZI encoding. On the receive path, USBSerialInterfaceEngine, implements connection state detection, sync detection and stripping, clock recovery, NRZI decoding, bit de-stuffing, CRC calculation and checking, and serial to parallel conversion.

HostSlaveMux – Allows host and slave controllers to share access to the USBSerialInterfaceEngine.

USBSlaveControl – Supports the USB 1.1 Device specific portion of the USB 1.1 protocol layer. Supports all USB 1.1 transaction types; bulk, setup, interrupt, and isochronous.

USBHostControl – Supports the USB 1.1 Host specific portion of the USB 1.1 protocol layer. Supports all USB 1.1 transaction types; bulk, setup, interrupt, and isochronous. Supports automatic preamble insertion, and automatic SOF generation and transmission

WishBoneBI – Provides Wishbone compatible interface to host/slave controllers and the transmit/receive FIFOs.

WISHBONE COMPATIBLE

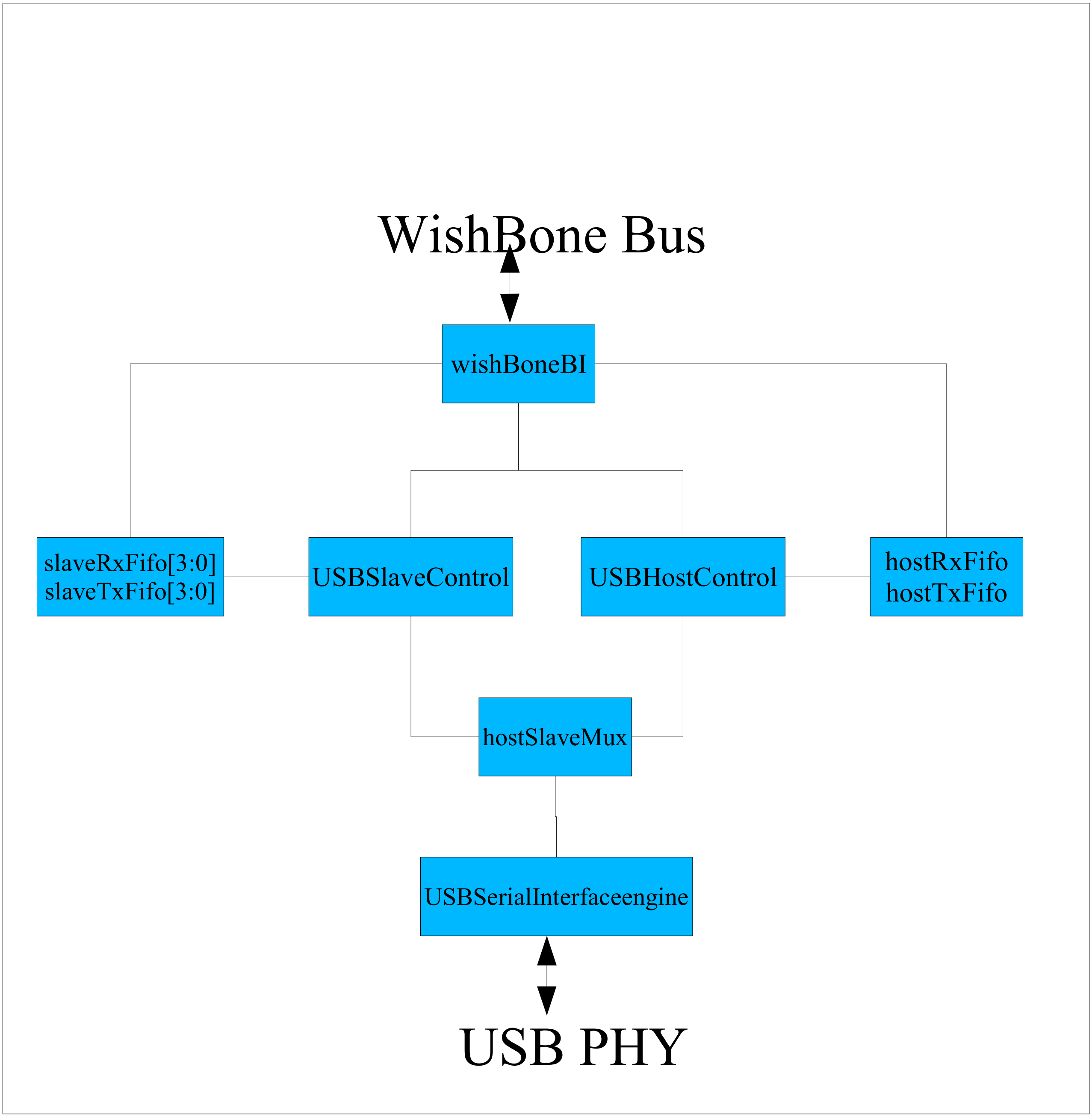

Figure 1USBHostSlave block diagram

3 Operation

Consider a system consisting of a USBHostSlave IP core, a USB PHY (eg Philips ISP1102), and a microprocessor (see figure (2)).

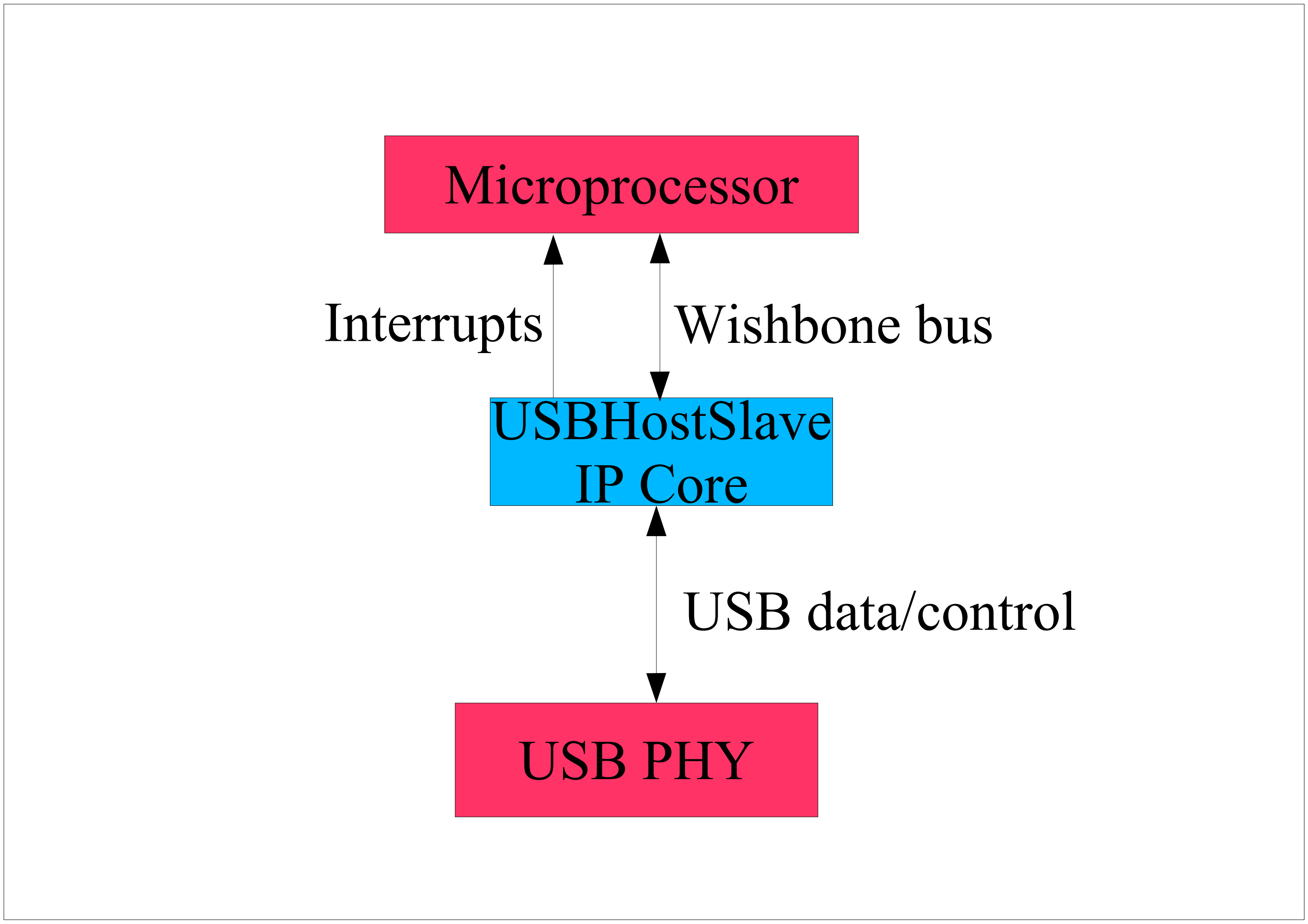

Figure 2USBHostSlave in a typical system

There are two scenarios to consider. Firstly where a USBHostSlave is configured as a USB Host, and the second where it is configured as USB Device.

Let's consider the Host configuration first. On power up, the microprocessor will configure USBHostSlave as a USB Host. If the Host is not connected to a USB Device, then USBHostSlave will report that the USB line is in a disconnect state. When the Host is connected to a Device, USBHostSlave will report the connection event and the connection speed. Now the following USBHostSlave parameters can be configured;

- USB speed.
- USB polarity.
- The USB address of the attached Device (Address zero is the default Device address).
- The USB endpoint (endpoint zero is used for set up) within the attached Device.

Now that the Host has been configured, the Host can attempt to complete a transfer with the attached USB Device. Let's use the example of the Host sending data to the USB Device. First the USBHostSlave transmit FIFO will be loaded with data, then the transaction type will be set, and finally the transaction request flag will be set. The microprocessor detects that the transaction is complete by checking for the transaction done flag to be set, or waiting for the transaction done interrupt.

Now let's take a look at the case where USBHostSlave is configured as a USB Device. On power up, the microprocessor will configure USBHostSlave as a USB Device. If the Device is not connected to a USB Host, then USBHostSlave will report that the USB line is in a reset state. When the Device is connected to a Host, USBHostSlave will report the connection event and the connection speed. Now the following USBHostSlave parameters can be configured;

- USB speed.
- USB polarity.
- The USB address (Address zero is the default Device address).
- Set the global endpoint enable flag

If the Device is expecting a transaction then it must be prepared to accept the transaction. If an incoming data packet is expected then simply set the enable flag for the endpoint that is expecting a transaction, and wait for the transaction done flag (or transaction done interrupt) to be set. If the expected transaction involves outgoing data, then the endpoint FIFO will have to be loaded with data before the enable flag is set.

4

Registers

USB Host Slave Common Registers

Register Address	Name
0xe0	HOST_SLAVE_CONTROL_REG
0xe1	HOST_SLAVE_VERSION_REG

HOST_SLAVE_CONTROL_REG

Bit Position	Name	Description
0	HOST_MODE	Set to 1 to enable host mode. Set to 0 to enable slave mode.

HOST_SLAVE_VERSION_REG

Bit Position	Name	Description
[7:4]	VERSION_NUM_MAJOR	Major revision number
[3:0]	VERSION_NUM_MINOR	Minor revision number

USB Host

Register Address	Name
0x0	HOST_TX_CONTROL_REG
0x1	HOST_TX_TRANS_TYPE_REG
0x2	HOST_TX_LINE_CONTROL_REG
0x3	HOST_TX_SOF_ENABLE_REG
0x4	HOST_TX_ADDR_REG
0x5	HOST_TX_ENDP_REG
0x6	HOST_FRAME_NUM_MSP_REG
0x7	HOST_FRAME_NUM_LSP_REG
0x8	HOST_INTERRUPT_STATUS_REG
0x9	HOST_INTERRUPT_MASK_REG
0xa	HOST_RX_STATUS_REG

Register Address	Name
0xb	HOST_RX_PID_REG
0xc	HOST_RX_ADDR_REG
0xd	HOST_RX_ENDP_REG
0xe	HOST_RX_CONNECT_STATE_REG
0xf	HOST_SOF_TIMER_MSB_REG
0x20	HOST_RX_FIFO_DATA
0x21	HOST_RX_FIFO_STATUS
0x22	HOST_RX_FIFO_DATA_COUNT_MSB
0x23	HOST_RX_FIFO_DATA_COUNT_LSB
0x24	HOST_RX_FIFO_CONTROL_REG
0x30	HOST_TX_FIFO_DATA
0x31	HOST_TX_FIFO_STATUS
0x32	HOST_TX_FIFO_DATA_COUNT_MSB
0x33	HOST_TX_FIFO_DATA_COUNT_LSB
0x34	HOST_TX_FIFO_CONTROL_REG

HOST_TX_CONTROL_REG

Bit Position	Name	Description
0	TRANS_REQ_BIT	Set to 0 to disable transaction. Set to 1 to enable transaction, automatically cleared when transaction complete.
1	SOF_SYNC_BIT	Set to 1 to syncrhonize transaction with the end of SOF transmission.Transaction will be scheduled for transmission immediately after SOF transmission.
2	PREAMBLE_ENABLE_BIT	Set to 1 to enable preamble. Should only be enabled when the host is connected to a low speed device via a hub. The preamble is a token which is prefixed to all packet transmissions, and is transmitted at full speed irrespective of the state of the FULL_SPEED_LINE_RATE_BIT.

Bit Position	Name	Description
3	ISO_ENABLE_BIT	Set to 1 to enable isochronous mode. In isochronous mode, no acknowledgements are sent or received. Note, the TRANS_TYPE_REG must be set to either IN_TRANS or OUTDATA0_TRANS. Isochronous mode is not allowed with any other transaction types.

HOST_RX_CONNECT_STATE_REG

Bit Position	Name	Description

[1:0]	RX_LINE_STATE	The contents of RX_CONNECT_STATE_REG reflect the current connection state, where;
		DISCONNECT = 0
		LOW_SPEED_CONNECT = 1
		FULL_SPEED_CONNECT = 2

HOST_INTERRUPT_STATUS_REG

Bit Position	Name	Description
0	TRANS_DONE_BIT	Automatically set to 1 when a transaction is completed. Must be cleared by writing 1.
1	RESUME_INT_BIT	Automatically set to 1 when resume state is detected. Must be cleared by writing 1.
2	CONNECTION_EVENT_BIT	Automatically set to 1 when a connect or disconnect occurs. Must be cleared by writing 1.
3	SOF_SENT_BIT	Automatically set to 1 when a SOF transmission occurs. Must be cleared by writing 1.

HOST_INTERRUPT_MASK_REG

Bit Position	Name	Description
0	TRANS_DONE_BIT	Set to 1 to enable interrupt on transaction completion.
1	RESUME_INT_BIT	Set to 1 to enable interrupt on resume detected.

Bit Position	Name	Description
2	CONNECTION_EVENT_BIT	Set to 1 to enable interrupt on connect or disconnect event.
3	SOF_SENT_BIT	Set to 1 to enable interrupt on SOF transmission.

HOST_RX_STATUS_REG

Bit Position	Name	Description
0	CRC_ERROR_BIT	When set to 1, indicates CRC error detected on the last transaction.
1	BIT_STUFF_ERROR_BIT	When set to 1, indicates bit stuff error detected on the last transaction.
2	RX_OVERFLOW_BIT	When set to 1, indicates insufficient free space in RX fifo to accept entire data packet.
3	RX_TIME_OUT_BIT	When set to 1, indicates no response from USB device.
4	NAK_RXED_BIT	When set to 1, indicates NAK received from USB device.
5	STALL_RXED_BIT	When set to 1, indicates STALL received from USB device.
6	ACK_RXED_BIT	When set to 1, indicates ACK received from USB device.
7	DATA_SEQUENCE_BIT	If the last transaction was of type IN_TRANS, then this bit indicates the sequence number of the last receive packet. DATA0 = 0, DATA1 = 1.

HOST_TX_TRANS_TYPE_REG

Bit Position	Name	Description

[1:0]	TRANSACTION_TYPE	SETUP_TRANS = 0
		IN_TRANS = 1
		OUTDATA0_TRANS = 2
		OUTDATA1_TRANS = 3
		These are the four basic types of transaction. The transaction types detailed in USB 1.1 (Setup, bulk, and isochronous) are composed of a series of one or more of these basic atomic transactions.

HOST_TX_LINE_CONTROL_REG

Bit Position	Name	Description
[1:0]	TX_LINE_STATE	When DIRECT_CONTROL_BIT=1,TX_LINE_STATE directly controls the state of the USB physical wires, where; TX_LINE_STATE [1] = D+ TX_LINE_STATE [0] = D-
2	DIRECT_CONTROL_BIT	Set to 1 to allow direct control the state of the USB physical wires. Clear to 0 for normal operation.
3	FULL_SPEED_LINE_POLARITY_BIT	Set to 1 to enable full speed line polarity. That is J= differential 1, K= differential 0. Clear to zero to enable low speed line poarity. That is J= differential 0, K= differential 1. If the host is communicating with a full speed device, then full speed line polarity should be enabled. If the host is communicating with a low speed device **directly** then full speed line polarity should be disabled. If the host is communicating with a low speed device **via a hub**, then full speed line polarity should be enabled.
4	FULL_SPEED_LINE_RATE_BIT	Set to 1 to enable full speed line rate of 12Mbps. Clear to 0 to enable low speed line rate of 1.5Mbps. If the host is communicating with a full speed device, then full speed line rate should be enabled. If the host is communicating with a low speed device full speed line rate should be disabled.

HOST_TX_SOF_ENABLE_REG

Bit Position	Name	Description
0	SOF_EN_BIT	If FULL_SPEED_LINE_POLARITY_BIT is set, then setting this bit to 1 to enables automatic transmission of SOF tokens every 1mS. Note that SOF tokens will be transmitted at full speed line rate irrespective of the state of FULL_SPEED_LINE_RATE_BIT. If FULL_SPEED_LINE_POLARITY_BIT is clear, then setting this bit to 1 to enables automatic transmission of low speed EOP keep alive every 1mS. Transition from 0 to 1 causes transmission of resume state prior to EOP transmission. Note that this mode is only used when the host is connected directly to a low speed device. Clear to 0 to disable automatic SOF/EOP transmission, and allow any devices attached to the host to enter the suspend state.

HOST_TX_ADDR_REG

Bit Position	Name	Description
[6:0]	DEVICE_ADDRESS	USB Device address.

HOST_TX_ENDP_REG

Bit Position	Name	Description
[3:0]	ENDP_ADDRESS	Endpoint address.

HOST_FRAME_NUM_MSP_REG

Bit Position	Name	Description
[2:0]	FRAME_NUM_MSP	Most significant part of the frame number used for SOF transmission. That is FRAME_NUM_MSP = FRAME_NUM[10:8].

HOST_FRAME_NUM_LSP_REG

Bit Position	Name	Description
[7:0]	FRAME_NUM_LSP	Least significant part of the frame number used for SOF transmission. That is FRAME_NUM_LSP = FRAME_NUM[7:0].

HOST_SOF_TIMER_MSB_REG

Bit Position	Name	Description
[7:0]	HOST_SOF_TIMER_MSB	Most significant byte of the SOF timer used for SOF transmission. Timer is incremented at 48MHz, thus there are 48000 ticks in a 1mS frame. This register can be used to calculate the number of ticks remaining in a frame. Trem = 0xbb - HOST_SOF_TIMER_MSB

HOST_RX_PID_REG

Bit Position	Name	Description
[3:0]	RECEIVE_PID	Packet identifier for the last packet received.

HOST_RX_ADDR_REG

Bit Position	Name	Description
[6:0]	RECEIVE_ADDRESS	Address from which the last receive packet was sent.

HOST_RX_ENDP_REG

Bit Position	Name	Description
[3:0]	RECEIVE_ENDP	End point from which the last receive packet was sent.

HOST_RX_FIFO_DATA

Bit Position	Name	Description
[7:0]	RX_FIFO_DATA	If the last transaction was an IN_TRANS, then the receive payload can be retrieved by reading RX_FIFO_DATA

HOST_TX_FIFO_DATA

Bit Position	Name	Description
[7:0]	TX_FIFO_DATA	Prior to requesting an OUTDATA0_TRANS or an OUTDATA1_TRANS , load transmit fifo with data by writing to TX_FIFO_DATA.

HOST_[T,R]X_FIFO_STATUS

Bit Position	Name	Description
[0]	FIFO_FULL	1 = Full. 0 = Not full.

HOST_[T,R]X_FIFO_DATA_COUNT_MSB

Bit Position	Name	Description
[7:0]	FIFO_DATA_COUNT_MSB	MSByte of FIFO_DATA_COUNT. Indicates the number of data entries within the fifo.

HOST_[T,R]X_FIFO_DATA_COUNT_LSB

Bit Position	Name	Description
[7:0]	FIFO_DATA_COUNT_LSB	LSByte of FIFO_DATA_COUNT. Indicates the number of data entries within the fifo.

HOST_[T,R]X_FIFO_CONTROL

Bit Position	Name	Description
[0]	FIFO_FORCE_EMPTY	Write only. Set to 1 to delete all the data samples within the fifo.

USB Slave (Device)

Register Address	Name
0x40	ENDPOINT0_CONTROL_REG
0x41	ENDPOINT0_STATUS_REG
0x42	ENDPOINT0_TRANSTYPE_STATUS_REG
0x43	ENDPOINT0_NAK_TRANSTYPE_STATUS_REG
0x44	ENDPOINT1_CONTROL_REG
0x45	ENDPOINT1_STATUS_REG
0x46	ENDPOINT1_TRANSTYPE_STATUS_REG
0x47	ENDPOINT1_NAK_TRANSTYPE_STATUS_REG

Register Address	Name
0x48	ENDPOINT2_CONTROL_REG
0x49	ENDPOINT2_STATUS_REG
0x4a	ENDPOINT2_TRANSTYPE_STATUS_REG
0x4b	ENDPOINT2_NAK_TRANSTYPE_STATUS_REG
0x4c	ENDPOINT3_CONTROL_REG
0x4d	ENDPOINT3_STATUS_REG
0x4e	ENDPOINT3_TRANSTYPE_STATUS_REG
0x4f	ENDPOINT3_NAK_TRANSTYPE_STATUS_REG
0x50	SC_CONTROL_REG
0x51	SC_LINE_STATUS_REG
0x52	SC_INTERRUPT_STATUS_REG
0x53	SC_INTERRUPT_MASK_REG
0x54	SC_ADDRESS
0x55	SC_FRAME_NUM_MSP
0x56	SC_FRAME_NUM_LSP
0x60	EP0_RX_FIFO_DATA
0x61	EP0_RX_FIFO_STATUS
0x62	EP0_RX_FIFO_DATA_COUNT_MSB
0x63	EP0_RX_FIFO_DATA_COUNT_LSB
0x64	EP0_RX_FIFO_CONTROL_REG
0x70	EP0_TX_FIFO_DATA
0x71	EP0_TX_FIFO_STATUS
0x72	EP0_TX_FIFO_DATA_COUNT_MSB
0x73	EP0_TX_FIFO_DATA_COUNT_LSB
0x74	EP0_TX_FIFO_CONTROL_REG
0x80	EP1_RX_FIFO_DATA
0x81	EP1_RX_FIFO_STATUS
0x82	EP1_RX_FIFO_DATA_COUNT_MSB
0x83	EP1_RX_FIFO_DATA_COUNT_LSB
0x84	EP1_RX_FIFO_CONTROL_REG
0x90	EP1_TX_FIFO_DATA
0x91	EP1_TX_FIFO_STATUS
0x92	EP1_TX_FIFO_DATA_COUNT_MSB
0x93	EP1_TX_FIFO_DATA_COUNT_LSB
0x94	EP1_TX_FIFO_CONTROL_REG
0xa0	EP2_RX_FIFO_DATA
0xa1	EP2_RX_FIFO_STATUS
0xa2	EP2_RX_FIFO_DATA_COUNT_MSB

Register Address	Name
0xa3	EP2_RX_FIFO_DATA_COUNT_LSB
0xa4	EP2_RX_FIFO_CONTROL_REG
0xb0	EP2_TX_FIFO_DATA
0xb1	EP2_TX_FIFO_STATUS
0xb2	EP2_TX_FIFO_DATA_COUNT_MSB
0xb3	EP2_TX_FIFO_DATA_COUNT_LSB
0xb4	EP2_TX_FIFO_CONTROL_REG
0xc0	EP3_RX_FIFO_DATA
0xc1	EP3_RX_FIFO_STATUS
0xc2	EP3_RX_FIFO_DATA_COUNT_MSB
0xc3	EP3_RX_FIFO_DATA_COUNT_LSB
0xc4	EP3_RX_FIFO_CONTROL_REG
0xd0	EP3_TX_FIFO_DATA
0xd1	EP3_TX_FIFO_STATUS
0xd2	EP3_TX_FIFO_DATA_COUNT_MSB
0xd3	EP3_TX_FIFO_DATA_COUNT_LSB
0xd4	EP3_TX_FIFO_CONTROL_REG

ENDPOINT[3..0]_CONTROL_REG

Bit Position	Name	Description
[0]	ENDPOINT_ENABLE_BIT	Set to 1 to enable the endpoint. If endpoint is not enabled then it will not respond to any transactions. If endpoint is enabled, not ready, and not in isochronous mode, then all transactions will be NAK'd.
[1]	ENDPOINT_READY_BIT	Set to 1 make the endpoint ready. If endpoint is enabled and ready then it can respond to a host inititiated transaction. Automatically cleared to 0 when transaction is complete.
[2]	ENDPOINT_OUTDATA_SEQUENCE_BIT	If set to 1 then the endpoint will respond to a host IN request with a DATA1 packet, otherwise it will respond with a DATA0 packet.

Bit Position	Name	Description
[3]	ENDPOINT_SEND_STALL_BIT	If set to 1 and endpoint is enabled, ready, and not in isochronous mode, then endpoint will send STALL in response to a host inititiated transaction.
[4]	ENDPOINT_ISO_ENABLE_BIT	Set to 1 to enable isochronous transfers. In isochronous mode the endpoint does not send acknowledgements, nor does it expect to receive acknowledgements.

ENDPOINT[3..0]_STATUS_REG

Bit Position	Name	Description
0	SC_CRC_ERROR_BIT	When set to 1, indicates CRC error detected on the last transaction.
1	SC_BIT_STUFF_ERROR_BIT	When set to 1, indicates bit stuff error detected on the last transaction.
2	SC_RX_OVERFLOW_BIT	When set to 1, indicates insufficient free space in RX fifo to accept entire data packet.
3	SC_RX_TIME_OUT_BIT	When set to 1, indicates no response from USB host.
4	SC_NAK_SENT_BIT	When set to 1, indicates NAK sent to USB host.
5	SC_STALL_SENT_BIT	When set to 1, indicates STALL sent to USB host.
6	SC_ACK_RXED_BIT	When set to 1, indicates ACK received from USB host.
7	SC_DATA_SEQUENCE_BIT	If the last transaction was of type OUT_TRANS, then this bit indicates the sequence number of the last receive packet. DATA0 = 0, DATA1 = 1.

Note that SC_NAK_SENT_BIT refers to the last host inititiated transaction that occurred whilst the endpoint was enabled, but not ready. The other bits in this register are not effected by host transactions that are NAK'd, and they refer to the last transaction which resulted in ENDPOINT_READY_BIT being changed from 1 to 0. SC_NAK_SENT_BIT is cleared to zero immediately after an enabled/ready transaction is completed. Note that the usb slave endpoint cannot NAK a host transmission when the endpoint is enabled and ready. So the NAK_SENT_BIT never refers to a transaction which resulted in ENDPOINT_READY_BIT being changed from 1 to 0

ENDPOINT[3..0]_TRANSTYPE_STATUS_REG

Bit Position	Name	Description

[1:0]	TRANSACTION_TYPE	SC_SETUP_TRANS = 0
		SC_IN_TRANS = 1
		SC_OUTDATA_TRANS = 2
		This is the transaction type of the last transaction which resulted in ENDPOINT_READY_BIT being changed from 1 to 0

ENDPOINT[3..0]_NAK_TRANSTYPE_STATUS_REG

Bit Position	Name	Description

[1:0]	TRANSACTION_TYPE	SC_SETUP_TRANS = 0
		SC_IN_TRANS = 1
		SC_OUTDATA_TRANS = 2
		This is the transaction type of the last transaction which resulted in a NAK being sent to the host.

SC_CONTROL_REG

Bit Position	Name	Description
[0]	SC_GLOBAL_ENABLE_BIT	When cleared to 0, all endpoints are disabled, and the slave will not respond to any host inititiated transactions.
[2:1]	SC_TX_LINE_STATE	When SC_DIRECT_CONTROL_BIT=1 ,SC_TX_LINE_STATE directly controls the state of the USB physical wires, where; SC_TX_LINE_STATE [2] = D+ SC_TX_LINE_STATE [1] = D-
[3]	SC_DIRECT_CONTROL_BIT	Set to 1 to allow direct control the state of the USB physical wires. Clear to 0 for normal operation.

Bit Position	Name	Description
[4]	SC_FULL_SPEED_LINE_POLARITY_BIT	Set to 1 to enable full speed line polarity. That is J= differential 1, K= differential 0. Clear to zero to enable low speed line poarity. That is J= differential 0, K= differential 1.
[5]	SC_FULL_SPEED_LINE_RATE_BIT	Set to 1 to enable full speed line rate of 12Mbps. Clear to 0 to enable low speed line rate of 1.5Mbps.

SC_LINE_STATUS_REG

Bit Position	Name	Description

[1:0]	RX_LINE_STATE	The contents of RX_CONNECT_STATE_REG reflect the current connection state, where;
		
		RESET = 0
		
		LOW_SPEED_CONNECT = 1
		
		FULL_SPEED_CONNECT = 2

SC_INTERRUPT_STATUS_REG

Bit Position	Name	Description
[0]	SC_TRANS_DONE_BIT	Set to 1 when a transaction is completed. Must be cleared by writing 1.
[1]	SC_RESUME_INT_BIT	Set to 1 when resume state is detected. Must be cleared by writing 1.
[2]	SC_RESET_EVENT_BIT	Set to 1 when reset state is detected. Must be cleared by writing 1.
[3]	SC_SOF_RECEIVED_BIT	Set to 1 when a SOF packet is received. Must be cleared by writing 1.
[4]	SC_NAK_SENT_INT_BIT	Set to 1 when a NAK sent. Must be cleared by writing 1.

SC_INTERRUPT_MASK_REG

Bit Position	Name	Description
[0]	SC_TRANS_DONE_BIT	Set to 1 to enable interrupt on transaction complete.
[1]	SC_RESUME_INT_BIT	Set to 1 to enable interrupt on resume detected.

Bit Position	Name	Description
[2]	SC_RESET_EVENT_BIT	Set to 1 to enable interrupt on reset detected.
[3]	SC_SOF_RECEIVED_BIT	Set to 1 to enable interrupt on SOF received.
[4]	SC_NAK_SENT_INT_BIT	Set to 1 to enable interrupt on NAK'd transaction.

SC_ADDRESS

Bit Position	Name	Description
[6:0]	DEVICE_ADDRESS	USB Device address.

SC_FRAME_NUM_MSP

Bit Position	Name	Description
[2:0]	SC_FRAME_NUM_MSP	Most significant part of the frame number received in the last SOF transmission. That is SC_FRAME_NUM_MSP = FRAME_NUM[10:8].

SC_FRAME_NUM_LSP

Bit Position	Name	Description
[7:0]	SC_FRAME_NUM_LSP	Least significant part of the frame number received in the last SOF transmission. That is SC_FRAME_NUM_LSP = FRAME_NUM[7:0].

EP[3..0]_TX_FIFO_DATA

Bit Position	Name	Description
[7:0]	TX_FIFO_DATA	Prior to receiving a IN_TRANS, load transmit fifo with data by writing to TX_FIFO_DATA.

EP[3..0]_RX_FIFO_DATA

Bit Position	Name	Description
[7:0]	RX_FIFO_DATA	After receiving an OUTDATA_TRANS, or a SETUP_TRANS, get receive data from fifo by reading from RX_FIFO_DATA.

EP[3..0]_[T,R]X_FIFO_STATUS

Bit Position	Name	Description
[0]	FIFO_FULL	1 = Full, 0 = Not full

EP[3..0]_[T,R]X_FIFO_DATA_COUNT_MSB

Bit Position	Name	Description
[7:0]	FIFO_DATA_COUNT_MSB	MSByte of FIFO_DATA_COUNT. Indicates the number of data samples within the fifo.

EP[3..0]_[T,R]X_FIFO_DATA_COUNT_LSB

Bit Position	Name	Description
[7:0]	FIFO_DATA_COUNT_LSB	LSByte of FIFO_DATA_COUNT. Indicates the number of data samples within the fifo.

EP[3..0]_[T,R]X_FIFO_CONTROL

Bit Position	Name	Description
[0]	FIFO_FORCE_EMPTY	Write only. Set to 1 to delete all the data samples within the fifo.

5 Clocks

Name	Source	Rates (MHz)			Remarks	Description
		Max	Min	Resolution		
clk	Input Pad	48	48	-	Duty cycle 50/50.	System clock.

Table 1: List of clocks

6 IO Ports

Port	Width	Direction	Description
clk	1	input	48MHz System clock. Future revisions of the core should have a seperate wishbone clock.
rst	1	input	Synchronous system reset.
address_i	8	input	WISHBONE address input
data_i	8	input	WISHBONE data input
data_o	8	output	WISHBONE data output
writeEn	1	input	WISHBONE write enable
strobe_i	1	input	WISHBONE strobe input
ack_o	1	output	WISHBONE acknowledge output
hostSOFSentIntOut	1	output	Host SOF sent interrupt
hostConnEventIntOut	1	output	Host connection event interrupt
hostResumeIntOut	1	output	Host resume interrupt
hostTransDoneIntOut	1	output	Host transaction done interrupt
slaveSOFRxedIntOut	1	output	SlaveSOF received interrupt
slaveResetEventIntOut	1	output	Slave reset event interrupt
slaveResumeIntOut	1	output	Slave resume interrupt
slaveTransDoneIntOut	1	output	Slave transaction done interrupt
slaveNAKSentIntOut	1	output	Slave NAK sent interrupt
USBWireDataIn	2	input	USB wire differential data input USBWireDataIn[1] = D+ USBWireDataIn[0] = D-
USBWireDataOut	2	output	USB wire differential data output USBWireDataOut[1] = D+ USBWireDataOut[0] = D-
USBWireDataOutTick	1	output	Debug output. 6MHz in full speed mode, 750KHz in low speed mode.
USBWireDataInTick	1	output	Debug output. 6MHz in full speed mode, 750KHz in low speed mode.
USBWireCtrlOut	1	output	USB tansmit output enable. Connect to OE pin on USB PHY (eg invert and connect to OE_n on ISP1102)

Table 2: List of IO ports

7 Wishbone Datasheet

WISHBONE DATASHEET for USBHostSlave IP Core		
Description	Specification	
General Description:	8-bit slave input and output port	
Supported cycles:	SLAVE READ/WRITE	
Data port Size:	8-bit	
Data port granularity:	8-bit	
Data port, max operand size:	8-bit	
Data transfer ordering:	N/A	
Data transfer sequencing:	Undefined	
Supported signal list and cross reference to equivalet WISHBONE signals:	Signal Name	WISHBONE Equiv.
	address_i	ADR_I
	data_i[7:0]	DAT_I()
	data_o[7:0]	DAT_O()
	WriteEn	WE_I
	strobe_i	STB_I
	ack_o	ACK_O

Table 3: WISHBONE data sheet

8 Resource Utilization

Design Entity	Logic Cells	Memory bits
USBHostControl	585	0
USBSlaveControl	527	0
USBSerialInterfaceEngine	1148	0
WishBoneBI	127	0
10 FIFOs each 64x8-bit	530	5120
USBHostSlave (top level)	**2924**	**5120**

Table 4 Resource utilization for Altera CycloneEP1C20