Document No: PD-3065-9111	Issue: 03	Effective date: 20/03/2017

Document Title:

USB Module Specification

Summary / Scope:

This document specifies the VHDL module which will interface between a USB interface and the Dual Port RAM (DPR) contained in the FlexComms Interface (FCI).

Reason for Issue / Nature of Change :

New USB commands added.

Distribution:

Project folder: \\renishaw\global\GB\PLC\CMMPD\Data\CMM\Projects\3065 - Head Interface Card\Controlled Documents - BLUE\Project Docs PD-\FlexComms Interface\

20/03/2017

X Richard Jones

Author: Richard Jones
Senior Electronics Design Engineer

20/03/2017

X Ellie Burnett

Reviewer: Ellie Burnett
Development Test Engineer
Signed by: Ellie Burnett

29/03/2017

X E C Barrow

Reviewer: Ed Barrow
Design and Development Engineer
Signed by: Ed Barrow

1 Introduction

This document specifies the FCI[1] USB module. This is a VHDL design that is implemented in the internal FPGA [2] and interfaces between three separate modules using a standard memory interface[3] and an FTDI FT2232H [4] device (see Figure 1).

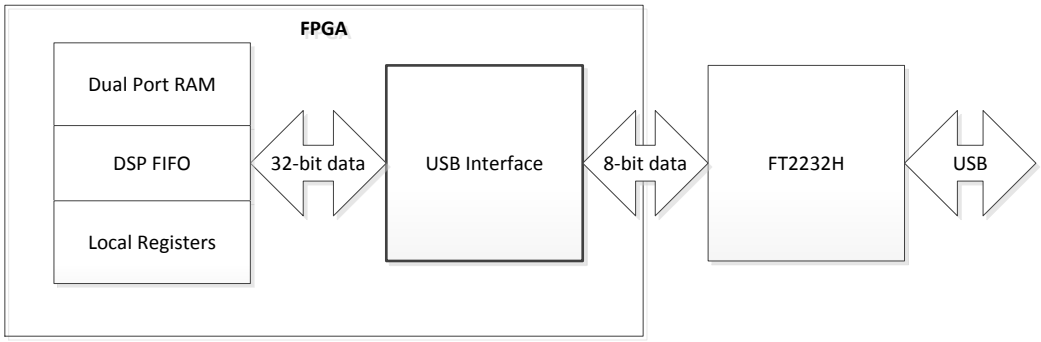

Figure 1: Diagram showing the position of the USB module in the FPGA

The USB module is responsible for interpreting commands from the USB host and sending back the appropriate response. This document defines the commands that are implemented in the module and their functionality.

2 References

[1] R. Jones, "PD-3065-9107-02 FlexComms Interface Specification," December 2015.

[2] N. Schollar, "PD-3065-9112-01 USB FlexComms Interface FPGA Datasheet," September 2016.

[3] Altera Corporation, "Cyclone IV Handbook: 3. Memory Blocks in Cyclone IV Devices," 2011. [Online]. Available: https://www.altera.com/en_US/pdfs/literature/hb/cyclone-iv/cyiv-51003.pdf. [Accessed 1st December 2015].

[4] Future Technology Devices International Ltd., "FT2232H Dual High Speed USB to Multipurpose UART/FIFO IC Datasheet," 2012. [Online]. Available: http://www.ftdichip.com/Support/Documents/DataSheets/ICs/DS_FT2232H.pdf. [Accessed 1st December 2015].

[5] FTDI, "D2XX Drivers," October 2016. [Online]. Available: http://www.ftdichip.com/Drivers/D2XX.htm. [Accessed March 2017].

[6] Future Technology Devices International Ltd., "AN232B-04 Data Throughput, Latency and Handshaking," 10 October 2006. [Online]. Available: http://www.ftdichip.com/Documents/AppNotes/AN232B-04_DataLatencyFlow.pdf. [Accessed 5 February 2016].

[7] J. Wilkes, "PD-5208-9011 SPA2 - Software Design Description," July 2013. [Online]. Available: \\renishaw\global\GB\PLC\CMMPD\Data\CMMnwprodcon\Project Files\5208 (SPA2)\Project Team Members File - (RED)\DSP Design\Documents\SPA2 SDD 7.pdf.

3 Revision History

Issue	Date	Notes	Author
01	4/1/2016	First Issue	Richard Jones
02	25/7/2016	Fixed mistake regarding register addresses incrementing by 4 (Section 7) Changed order of register headings to match those in the table (Section 7) Changed description for Number of addresses to stream register so that a value of 0 means one address will be streamed (Section 7) Fixed typo in references (Section 2) Changed data interface to avoid using tristates except in the top level Added FPGA-side clock input CLK to port list and removed USB clock (section 4) Increased number of registers that can be streamed from 16 to 32 (Section 6) Added streaming mode synchronised to head data (Section 7) Added DSP streaming mode and configuration registers (Section 7) Plus other changes.	Nick Schollar
03	20/03/2017	Updated Figure 1. Removed port 'DSP_SC_RX_FIFO_EMPTY_IN' (Section 4). Addition of service channel read and service channel write USB commands (Section 6) Addition of DSP service channel read and write commands (Section 6).	Richard Jones

4 Port List

The following table defines the module ports. Direction is relative to the module itself:

Name	Dir	Data Type	Description
General			
CLK	in	std logic	FPGA clock (50MHz)
RESET	in	std logic	Global asynchronous reset
FPGA Side			
DATA_IN	in	32 bit std logic	Data bus from the DPR
READ_EN_OUT	out	std logic	Read enable to the DPR
WRITE_EN_OUT	out	std logic	Write enable to the DPR
ADDRESS_OUT	out	16 bit std logic	DPR address bus
DATA_OUT	out	32 bit std logic	Data bus to the DPR
LED_OUT	out	std_logic	Indicates USB activity
HAVE_REC_FRAME_IN	in	std_logic	Indicates new frame received from head
USB_READ_COMPLETE_OUT	out	std_logic	High for one clock cycle after read completed – required for FIFOs
DSP_SC_RX_FIFO_EMPTY_IN	in	std_logic	High when DSP RX FIFO is empty
USB Side			
RXF_L_IN	in	std logic	Indicates the receive buffer is empty
TXE_L_IN	in	std logic	Indicates the transmit buffer is empty
RDL_OUT	out	std logic	Active-low read data strobe
WRL_OUT	out	std logic	Active-low write data strobe
SIWU_OUT	out	std logic	Send immediate /wake up
USB_DATA_IN	in	8-bit std_logic	Data from FT2232H
USB_DATA_OUT	out	8-bit std_logic	Data to FT2232H (via tristate)
USB_DATA_EN	out	std_logic	USB data tristate enable

5 USB Interface

The module interfaces with the FT2232H device using the FT245 Asynchronous FIFO mode. The FT2232H interfaces with the computer using USB 2.0 High Speed using the FTDI D2XX driver [5].

USB bus activity can be scheduled according to a 1ms frame rate where the USB scheduler services requests to transfer data using packets of at least 64 bytes (62 user bytes) every frame. The FT2232H presents a byte interface to the FPGA, and will hold amounts of data below the packet size until a definable timeout occurs or enough data has been written to fill a packet.[6]

If a small number of bytes are written by the computer, then a small number are read, and finally some more bytes are written; the interface will be very slow and laggy. The

implementation of block read, block write and streaming modes enables much higher data transfer speeds to be obtained.

6 List of USB Commands

Each command starts with a one-byte identifier which differentiates the commands. Address and data is described as 16 and 32 bit words respectively to facilitate compatibility with the existing PCI bus infrastructure. A summary of the different commands is shown below:

Number	Name
1	Read
2	Write
3	Block read
4	Block write
5	FlexComms service read
6	FlexComms service write
7	Two byte DSP service read
8	Four byte DSP service read
9	Two byte DSP service write
10	Four byte DSP service write

When streaming is enabled, the module continually transmists data packets in the format shown below. At the end of each data packet, the USB interface resumes the task it was doing when it was interrupted. Each packet starts with an eight-byte identifier and the length will vary depending on how many addresses have been set to stream (see section 6.5). All commands will continue to work with streaming mode enabled, although it is advisable to switch off streaming before executing block read and write commands.

Byte	Contents
1-8	Preamble - ASCII characters: "WAHSINER"
9	Data from mapped address 1 MSB
10	Data from mapped address 1
11	Data from mapped address 1
12	Data from mapped address 1 LSB
13	Data from mapped address 2 MSB
…	…
136	Maximum number of bytes possible

6.1 Read

Reads a data word from the FCI at the specified address.

Command		Response	
Byte no.	**Content**	**Byte no.**	**Content**
1	0x01	1	Data MSB
2	Address MSB	2	Data
3	Address LSB	3	Data
		4	Data LSB

If streaming is in progress when a read command is sent, the module will wait until the streaming packet has been transmitted before sending the read response.

6.2 Write

Writes a data word to the FCI at the specified address. There is no response from a write data command.

Byte no.	Content
1	0x02
2	Address MSB
3	Address LSB
4	Data MSB
5	Data
6	Data
7	Data LSB

6.3 Block Read

Reads 512 contiguous bytes (128 words) from the FCI memory and transits them in sequence. This size is chosen so that it is compatible with the block data received from the head. Note that addresses increase in multiples of four.

It is recommended to switch off streaming before executing a block read command.

Command		**Response**	
Byte no.	**Content**	**Byte no.**	**Content**
1	0x03	1-8	ASCII characters: "WAHSINER"
2	Start Address LSB	9	Data Word 1 MSB
3	Start Address MSB	10	Data Word 1
		11	Data Word 1
		12	Data Word 1 LSB
		13	Data Word 2 MSB
		…	…
		520	Data word 128 LSB

6.4 Block Write

Writes 512 bytes (128 words) to the FCI at the address specified. Addresses increase in multiples of four.

Byte no.	**Content**
1	0x04
2	Start Address MSB
3	Start Address LSB
4	Data Word 1 MSB
5	Data Word 1
6	Data Word 1
7	Data Word 1 LSB
8	Data Word 2 MSB
…	…
515	Data Word 128 LSB

6.5 FlexComms Service Channel Read

Reads data from a FlexComms node over the service channel.

Command		Response	
Byte no.	**Content**	**Byte no.**	**Content**
1	0x05	1	FlexComms Addrees MSB
2	Node Address MSB	2	FlexComms Address LSB
3	Node Address LSB	3	Data MSB
4	FlexComms Address MSB	4	Data LSB
5	FlexComms Address LSB		

Note that if there is no FlexComms node with the specified address available, the command will timeout and no response will be transmitted.

6.6 Service Channel Write

Writes data to a FlexComms node over the service channel.

Command		Response	
Byte no.	**Content**	**Byte no.**	**Content**
1	0x06	1	FlexComms Address MSB
2	Node Address MSB	2	FlexComms Address LSB
3	Node Address LSB	3	Data MSB
4	FlexComms Address MSB	4	Data LSB
5	FlexComms Address LSB		
6	Data MSB		
7	Data LSB		

Note that if there is no FlexComms node with the specified address available, the command will timeout and no response will be transmitted.

6.7 DSP Two Byte Service Read

Performs a two-byte fixed length read through the DSP's service channel. For more information refer to [7] section 2.

Command		Response	
Byte no.	**Content**	**Byte no.**	**Content**
1	0x07	1	Data high byte
2	IDN high byte	2	Data low byte
3	IDN low byte	3	Ack byte

6.8 DSP Four Byte Service Read

Performs a four-byte fixed length read through the DSP's service channel. For more information refer to [7] section 2.

Command		Response	
Byte no.	**Content**	**Byte no.**	**Content**
1	0x08	1	Data high byte
2	IDN high byte	2	Data byte 3
3	IDN low byte	3	Data byte 2
		4	Data low byte
		5	Ack byte

6.9 DSP Two Byte Service Write

Performs a two-byte fixed length write through the DSP's service channel. For more information refer to [7] section 2.

Command		Response	
Byte no.	**Content**	**Byte no.**	**Content**
1	0x09	1	Ack byte
2	IDN high byte		
3	IDN low byte		
4	Data high byte		
5	Data low byte		

6.10 DSP Four Byte Service Write

Performs a four-byte fixed length write through the DSP's service channel. For more information refer to [7] section 2.

Command		**Response**	
Byte no.	**Content**	**Byte no.**	**Content**
1	0x0A	1	Ack byte
2	IDN high byte		
3	IDN low byte		
4	Data high byte		
5	Data byte 3		
6	Data byte 2		
7	Data low byte		

7 Streaming Configuration Registers

The majority of addresses are mapped directly into the Dual Port RAM to facilitate integration with the existing VHDL code, however a small number are implemented by the USB interface to control data streaming. These are summarised in the following table and detailed in the sections below:

Address	Name	R/W
0xFF00	Streaming control	RW
0xFF08	Number of addresses to stream - 1	RW
0xFF0C	Streaming interval	RW
0xFF10–0xFF8C	Streaming data memory map	RW
0xFF90	Stream config 1	RW
0xFF94	Stream config 2	RW

0xFF00 USB Control

Bits 31-2	Bit 2	Bit 1	Bit 0
Reserved	Stream Mode	Reserved	Streaming enable
			Default 0x00000000

Bit 2 Stream Mode

0: USB stream interval is in units of milliseconds, i.e. a stream interval of 10 will result in a stream being sent every 10 milliseconds. Use this mode when streamed data is required at constant intervals.

1: USB stream interval is in units of number of FlexComms frames received from the head, i.e. a stream interval of 10 will result in a stream being sent every 10 times a rising edge is detected on the HAVE_RECEIVED_FRAME signal. Use this mode when streamed data must be synchronized with data received from the head.

Bit 0 Streaming enable

Enables and disables the streaming function.

0xFF08 Number of addresses to stream - 1

Bits 31- 5	Bits 4-0
Reserved	Number of addresses to stream - 1
	Default 0x00000000

Defines how many four-byte data words are included in the streaming transmission. The default value of 0 means that one address will be streamed when streaming is enabled, and a value of 31 means 32 addresses will be streamed.

0xFF0C Streaming interval

Bits 31- 16	Bits 15-0
Reserved	Streaming interval
	Default 0x000003E8

Defines the interval, measured in ms, between a new data stream being transmitted. Due to the variable latency of USB, the interval between data being received by the PC application will vary. The default streaming interval is 1,000 ms.

If Bit 2 Stream Mode of the USB Control register is set, then this register is in units of frames received from the head instead of milliseconds.

0xFF10 – 0xFF8C Streaming Data Memory Map (R/W)

This set of 32 locations holds the addresses for the data that will be included in the streaming data packet.

Bits 31-16	Bits 15-0
Reserved (0x0000)	Address
	Default: 0x00000000

0xFF90 Stream Config 1

Bit 31	...	Bit 0
32nd word DPR/DSP selection		1st word DPR/DSP selection
		Default: 0x00000000

This register configures whether each word is read from the dual-port RAM or the DSP. If the nth bit is 0 (default), data is read from the DPR at the address specified in the Streaming Data Memory Map for the nth word and written to the FT2232H.

If the nth bit is a 1, a read request is made to the DSP by writing to the FIFO and using the address specified in the Streaming Data Memory Map, then the data received from the DSP and read from the FIFO is written to the FT2232H.

The DSP comms scheduler must be enabled and the DSP comms synchronised before DSP streaming is used. The address for DSP reads is specified in the format described in [7].

0xFF94 Stream Config 2

Bit 31	...	Bit 0
32nd word DSP 2/4 byte selection		1st word DSP 2/4 byte selection
		Default: 0x00000000

This register configures whether each word read from the DSP is a two byte read or a four byte read. If a bit is a 0, it is a two byte read. If a bit is a 1, it is a four byte read. This register has no effect for words that are read from the dual-port RAM.

Four bytes are always written to the FT2232H, even for a two byte read. This is so that the streamed data length is always the same for a given number of addresses read, regardless of source.