1.1 Overview

- **USB 2.0 Compliant**
- **3 IN Endpoints (including EP0, bulkin, interrupt)**
- **2 OUT Endpoints (including EP0, bulkout)**
- **Max Packet Size: Bulk-IN/-OUT 512Bytes, Control/Interrupt 64Bytes**
- **Support for DMA access to FIFOs**
- **Support Soft Connect/Disconnect**

1.2 Register Description

The UDC register map is split into the following sections:

Common USB registers (00h–0Fh) – These registers provide control and status for the complete core.

Indexed Endpoint Control/Status registers (10h–1Fh) – These registers provide control and status for the endpoints. The registers mapped into this section depend on the value of the Index register.

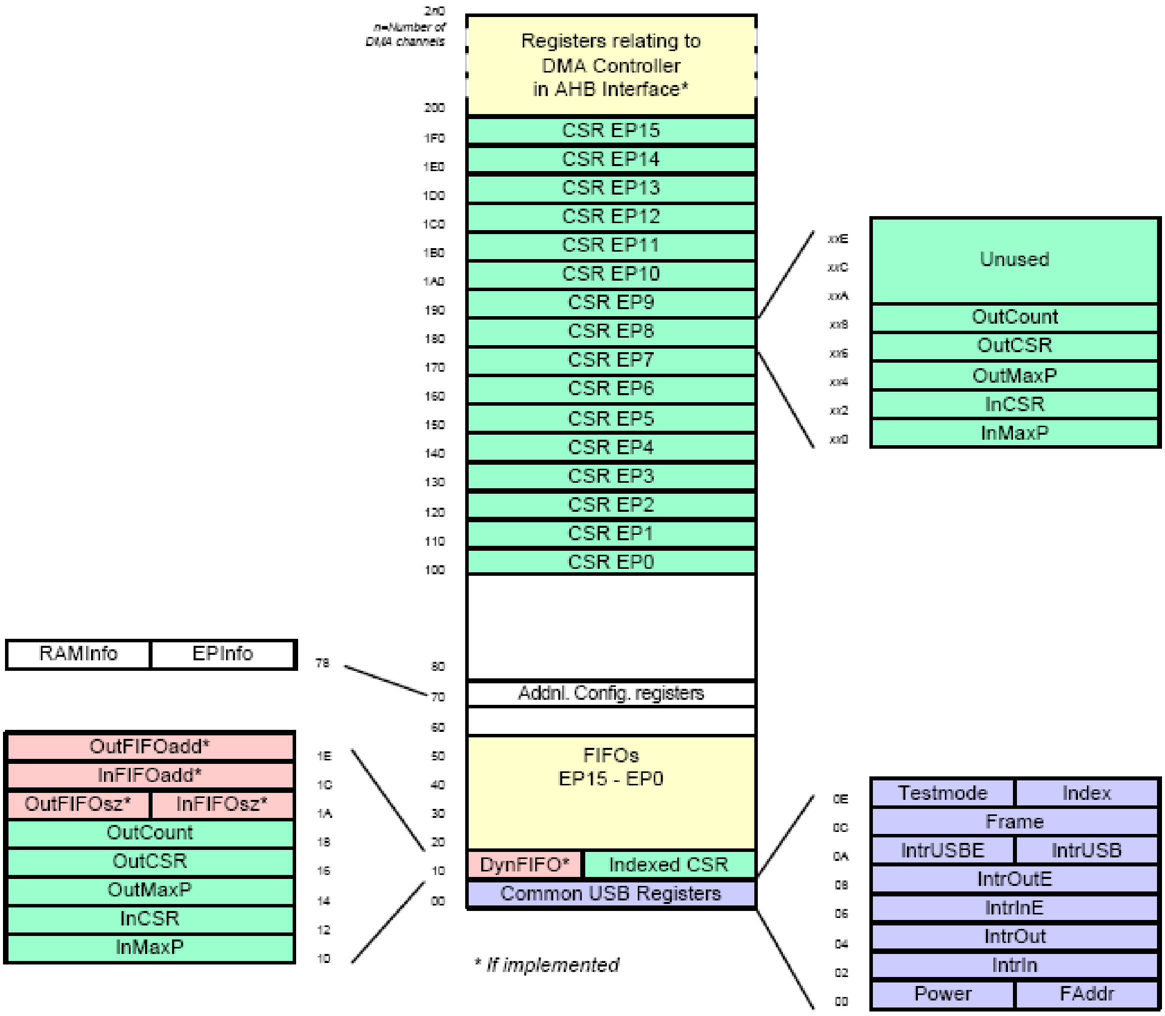
FIFOs (20h–5Fh) – This address range provides access to the endpoint FIFOs.

Additional Configuration registers (70h–7Fh) – Registers in this area of the memory map provide additional device statusinformation.

Non-Indexed Endpoint Control/Status registers (100h and above) – The registers available at 10h–1Fh, accessible independently of the setting of the Index register. 100h–10Fh EP0 registers; 110h–11Fh EP1 registers; 120h–12Fh EP2; and so on.

DMA Control Registers (200h and above) – These registers only appear in versions of the design that offer the AHB interface. The resulting Memory Map is detailed on the following pages and illustrated in the diagram below

The resulting Memory Map is detailed on the following pages and illustrated in the diagram below.



** If implemented*

MUSBHSFC REGISTER MAP – Common USB registers

ADDR	NAME	DESCRIPTION
00	FAddr	Function address register.
01	Power	Power management register.
02,03	IntrIn	Interrupt register for Endpoint 0 and IN Endpoints 1 to 15.
04,05	IntrOut	Interrupt register for OUT Endpoints 1 to 15.
06,07	IntrInE	Interrupt enable register for IntrIn.
08,09	IntrOutE	Interrupt enable register for IntrOut.
0A	IntrUSB	Interrupt register for common USB interrupts.
0B	IntrUSBE	Interrupt enable register for IntrUSB.
0C,0D	Frame	Frame number.
0E	Index	Index register for selecting the endpoint status and control registers.
0F	Testmode	Enables the USB 2.0 test modes.
MUSBHSFC REGISTER MAP – Indexed registers (Control Status registers for endpoint selected by the Index register)		
10,11	InMaxP	Maximum packet size for IN endpoint. (Index register set to select Endpoints 1 – 15 only)
12,13	CSR0	Control Status register for Endpoint 0. (Index register set to select Endpoint 0)
	InCSR	Control Status register for IN endpoint. (Index register set to select Endpoints 1 – 15)
14,15	OutMaxP	Maximum packet size for OUT endpoint. (Index register set to select Endpoints 1 – 15 only)
16,17	OutCSR	Control Status register for OUT endpoint. (Index register set to select Endpoints 1 – 15 only)
18,19	Count0	Number of received bytes in Endpoint 0 FIFO. (Index register set to select Endpoint 0)
	OutCount	Number of bytes in OUT endpoint FIFO. (Index register set to select Endpoints 1 – 15)
1A–1F	–	Only used if Dynamic FIFO sizing option is selected (see Section 18). Otherwise return 0.
MUSBHSFC REGISTER MAP – FIFOs		
20–XX	FIFOx	FIFOs for Endpoints 0 to 15. XX = 3F if CPU bus is 16-bit; XX = 5F if CPU bus is 32-bit.
Additional Configuration Registers		
78	EPInfo	Information about numbers of IN and OUT endpoints.
79	RAMInfo	Information about the width of the RAM and the number of DMA channels.

Note: In the following bit descriptions: ‘r’ means that the bit is read only ‘set’ means that the bit can only be written to set it ‘clear’ means that the bit can only be written to clear it ‘self-clearing’ means the bit will be cleared automatically ‘rw’ means that the bit can be both read and written ‘r/set’ means that the bit can be read or set but it can’t be cleared ‘r/clear’ means that the bit can be read or cleared but it can’t be set when the associated action has been executed.

1.2.1 Common Registers

1.2.1.1 FAD D R

FAddr is an 8-bit register that should be written with the function’s 7-bit address (received through a SET_ADDRESS descriptor). It is then used for decoding the function address in subsequent token packets.

Address: 00h*; Reset value:* 8’h00

	D7	D6	D5	D4	D3	D2	D1	D0
	Update	(MSB)			Function Address			(LSB)
From CPU	r	rw	rw	rw	rw	rw	rw	rw
From USB	r/clear	r	r	r	r	r	r	r

Bit	Name	Function
D7	**Update**	Set when FAddr is written. Cleared when the new address takes effect (at the end of the current transfer).
D6–D0	**Func Addr**	The function address.

1.2.1.2 POWER

Power is an 8-bit register that is used for controlling Suspend and Resume signaling, and High-speed operation.

Address: 01h; *Reset value:* 8’h20

	D7	D6	D5	D4	D3	D2	D1	D0
	ISO Update	Soft Conn	HS Enab	HS Mode	Reset	Resume	Suspend Mode	Enable SuspendM
From CPU	rw	rw	rw	r	r	rw	r	rw
From USB	r	r	r	rw	rw	r	set	r

Bit	Name	Function
D7	**ISO Update**	When set by the CPU, the MUSBHSFC will wait for an SOF token from the time InPktRdy is set before sending the packet. If an IN token is received before an SOF token, then a zero length data packet will be sent. Note: This bit only affects endpoints performing Isochronous transfers.
D6	**Soft Conn**	If Soft Connect/Disconnect feature is enabled, then the USB D+/D- lines are enabled when this bit is set by the CPU and tri-stated when this bit is cleared by the CPU. (See Section 6.1)
D5	**HS Enab**	When set by the CPU, the MUSBHSFC will negotiate for High-speed mode when the device is reset by the hub. If not set, the device will only operate in Full-speed mode.
D4	**HS Mode**	This read-only bit is set when the MUSBHSFC has successfully negotiated for High-speed mode.
D3	**Reset**	This read-only bit is set when Reset signaling has been detected on the bus (after 2.5µs of SE0). It is cleared when either HS negotiation has completed successfully or after 2.1ms of Reset signaling if HS negotiation fails.
D2	**Resume**	Set by the CPU to generate Resume signaling when the function is in Suspend mode. The CPU should clear this bit after 10 ms (a maximum of 15 ms) to end Resume signaling.

D1	**Suspend Mode**	This read-only bit is set when Suspend mode is entered. It is cleared when the CPU reads the interrupt register, or sets the Resume bit of this register.
D0	**Enable SuspendM**	Set by the CPU to enable the SUSPENDM signal.

1.2.1.3 INTRIN

IntrIn is a 16-bit read-only register that indicates which of the interrupts for IN Endpoints 1 – 15 are currently active. It also indicates whether the Endpoint 0 interrupt is currently active. *Note:* Bits relating to endpoints that have not been configured will always return 0. Note also that all active interrupts are cleared when this register is read.

Address: 02h*; Reset value:* 16'h0000

	D15	D14	D13	D12	D11	D10	D9	D8
	EP15 In	**EP14 In**	**EP13 In**	**EP12 In**	**EP11 In**	**EP10 In**	**EP9 In**	**EP8 In**
From CPU	r	r	r	r	r	r	r	r
From USB	set	set	set	set	set	set	set	set

	D7	D6	D5	D4	D3	D2	D1	D0
	EP7 In	**EP6 In**	**EP5 In**	**EP4 In**	**EP3 In**	**EP2 In**	**EP1 In**	**EP0**
From CPU	r	r	r	r	r	r	r	r
From USB	set	set	set	set	set	set	set	set

Bit	Name	Function
D15	**EP15 In**	IN Endpoint 15 interrupt.
D14	**EP14 In**	IN Endpoint 14 interrupt.
D13	**EP13 In**	IN Endpoint 13 interrupt.
D12	**EP12 In**	IN Endpoint 12 interrupt.
D11	**EP11 In**	IN Endpoint 11 interrupt.
D10	**EP10 In**	IN Endpoint 10 interrupt.
D9	**EP9 In**	IN Endpoint 9 interrupt.
D8	**EP8 In**	IN Endpoint 8 interrupt.
D7	**EP7 In**	IN Endpoint 7 interrupt.
D6	**EP6 In**	IN Endpoint 6 interrupt.
D5	**EP5 In**	IN Endpoint 5 interrupt.
D4	**EP4 In**	IN Endpoint 4 interrupt.
D3	**EP3 In**	IN Endpoint 3 interrupt.
D2	**EP2 In**	IN Endpoint 2 interrupt.
D1	**EP1 In**	IN Endpoint 1 interrupt.
D0	**EP0**	Endpoint 0 interrupt.

1.2.1.4 INTROUT

IntrOut is a 16-bit read-only register that indicates which of the interrupts for OUT Endpoints 1 – 15 are currently active. *Note:* Bits relating to endpoints that have not been configured will always return 0. Note also that all active interrupts are cleared when this register is read.

Address: 04h*; Reset value:* 16'h0000

	D15	D14	D13	D12	D11	D10	D9	D8
	EP15 Out	**EP14 Out**	**EP13 Out**	**EP12 Out**	**EP11 Out**	**EP10 Out**	**EP9 Out**	**EP8 Out**
From CPU	r	r	r	r	r	r	r	r
From USB	set	set	set	set	set	set	set	set

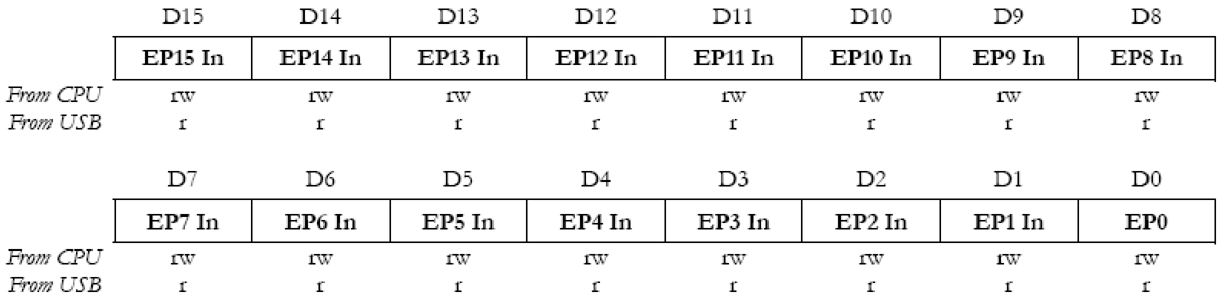
	D7	D6	D5	D4	D3	D2	D1	D0
	EP7 Out	**EP6 Out**	**EP5 Out**	**EP4 Out**	**EP3 Out**	**EP2 Out**	**EP1 Out**	**–**
From CPU	r	r	r	r	r	r	r	r
From USB	set	set	set	set	set	set	set	r

Bit	**Name**	**Function**
D15	**EP15 Out**	OUT Endpoint 15 interrupt.
D14	**EP14 Out**	OUT Endpoint 14 interrupt.
D13	**EP13 Out**	OUT Endpoint 13 interrupt.
D12	**EP12 Out**	OUT Endpoint 12 interrupt.
D11	**EP11 Out**	OUT Endpoint 11 interrupt.
D10	**EP10 Out**	OUT Endpoint 10 interrupt.
D9	**EP9 Out**	OUT Endpoint 9 interrupt.
D8	**EP8 Out**	OUT Endpoint 8 interrupt.
D7	**EP7 Out**	OUT Endpoint 7 interrupt.
D6	**EP6 Out**	OUT Endpoint 6 interrupt.
D5	**EP5 Out**	OUT Endpoint 5 interrupt.
D4	**EP4 Out**	OUT Endpoint 4 interrupt.
D3	**EP3 Out**	OUT Endpoint 3 interrupt.
D2	**EP2 Out**	OUT Endpoint 2 interrupt.
D1	**EP1 Out**	OUT Endpoint 1 interrupt.
D0	–	Unused, always returns 0

1.2.1.5 INTRINE

IntrInE is a 16-bit register that provides interrupt enable bits for each of the interrupts in IntrIn. Where a bit is set to 1, MC_NINT will be asserted on the corresponding interrupt in the IntrIn register becoming set. Where a bit is set to 0, the interrupt in IntrIn is still set but MC_NINT is not asserted. On reset, D0 – D*n* are set to 1 where *n* is the number of IN Endpoints (in addition to Endpoint 0) that are included in the design, while the remaining bits are set to 0. *Note:* Bits relating to endpoints that have not been configured will always return 0.

Address: 06h*; Reset value:* 16’hFFFF masked with the IN endpoints implemented



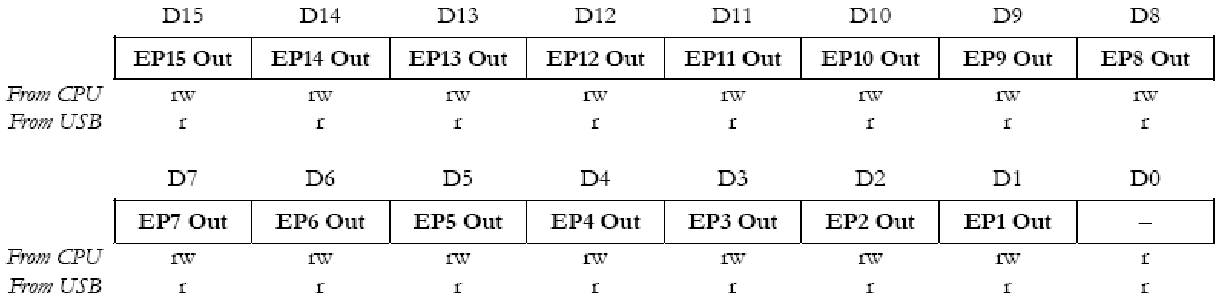
	D15	D14	D13	D12	D11	D10	D9	D8
	EP15 In	**EP14 In**	**EP13 In**	**EP12 In**	**EP11 In**	**EP10 In**	**EP9 In**	**EP8 In**
From CPU	rw	rw	rw	rw	rw	rw	rw	rw
From USB	r	r	r	r	r	r	r	r

	D7	D6	D5	D4	D3	D2	D1	D0
	EP7 In	**EP6 In**	**EP5 In**	**EP4 In**	**EP3 In**	**EP2 In**	**EP1 In**	**EP0**
From CPU	rw	rw	rw	rw	rw	rw	rw	rw
From USB	r	r	r	r	r	r	r	r

1.2.1.6 INTROUTE

IntrOutE is a 16-bit register that provides interrupt enable bits for each of the interrupts in IntrOut. Where a bit is set to 1, MC_NINT will be asserted on the corresponding interrupt in the IntrOut register becoming set. Where a bit is set to 0, the interrupt in IntrOut is still set but MC_NINT is not asserted. On reset, D1 – D*m* are set to 1 where *m* is the number of OUT Endpoints (in addition to Endpoint 0) that are included in the design, while the remaining bits are set to 0. *Note:* Bits relating to endpoints that have not been configured will always return 0.

Address: 08h*; Reset value:* 16’hFFFE masked with the OUT endpoints implemented



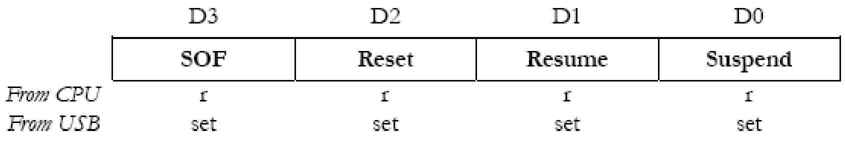
	D15	D14	D13	D12	D11	D10	D9	D8
	EP15 Out	**EP14 Out**	**EP13 Out**	**EP12 Out**	**EP11 Out**	**EP10 Out**	**EP9 Out**	**EP8 Out**
From CPU	rw	rw	rw	rw	rw	rw	rw	rw
From USB	r	r	r	r	r	r	r	r

	D7	D6	D5	D4	D3	D2	D1	D0
	EP7 Out	**EP6 Out**	**EP5 Out**	**EP4 Out**	**EP3 Out**	**EP2 Out**	**EP1 Out**	–
From CPU	rw	rw	rw	rw	rw	rw	rw	r
From USB	r	r	r	r	r	r	r	r

1.2.1.7 INTRUSB

IntrUSB is a 4-bit read-only register that indicates which USB interrupts are currently active. *Note:* All active interrupts are cleared when this register is read.

Address: 0Ah; *Reset value:* 4’b0000



	D3	D2	D1	D0
	SOF	**Reset**	**Resume**	**Suspend**
From CPU	r	r	r	r
From USB	set	set	set	set

Bit	Name	Function
D3	**SOF**	Set at the start of each frame – see Section 16.
D2	**Reset**	Set when reset signaling is detected on the bus.
D1	**Resume**	Set when resume signaling is detected on the bus while the MUSBHSFC is in Suspend mode.
D0	**Suspend**	Set when suspend signaling is detected on the bus.

1.2.1.8 INTRUSBE

IntrUSBE is a 4-bit register that provides interrupt enable bits for each of the interrupts in IntrUSB.

Address: 0Bh; *Reset value:* 4’b0110

	D3	D2	D1	D0
	SOF	**Reset**	**Resume**	**Suspend**
From CPU	rw	rw	rw	rw
From USB	r	r	r	r

1.2.1.9 FRAME

Frame is a 16-bit read-only register that holds the last received frame number.

Address: 0Ch; *Reset value:* 16’h0000

	D15	...	D11	D10	...	D0
	0	0 0	0 0	(MSB)	Frame Number	(LSB)
From CPU	r	...	r	r	...	r
From USB	w	...	w	w	...	w

1.2.1.10 INDEX

Index is a 4-bit register that determines which endpoint control/status registers are accessed at addresses 10h to 19h.

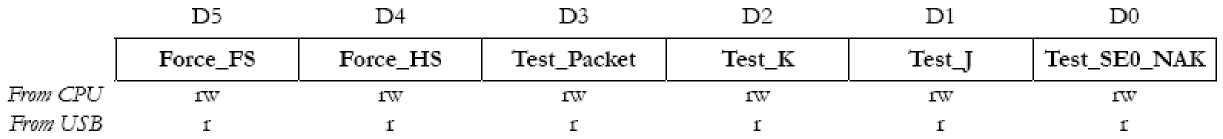
Address: 0Eh; *Reset value:* 4’b0000

	D3	D2	D1	D0
	(MSB)	Selected Endpoint		(LSB)
From CPU	rw	rw	rw	rw
From USB	r	r	r	r

1.2.1.11 TESTMODE

Testmode is a 6-bit register that is primarily used to put the MUSBHSFC into one of the four test modes for High-speed operation described in the USB 2.0 specification – in response to a SET FEATURE: TESTMODE command. It is not used in normal operation.

Address: 0Fh*; Reset value:* 6’b000000



	D5	D4	D3	D2	D1	D0
	Force_FS	**Force_HS**	**Test_Packet**	**Test_K**	**Test_J**	**Test_SE0_NAK**
From CPU	rw	rw	rw	rw	rw	rw
From USB	r	r	r	r	r	r

Bit	**Name**	**Description**
D5	**Force_FS**	The CPU sets this bit to force the MUSBHSFC into Full-speed mode when it receives a USB reset.
D4	**Force_HS**	The CPU sets this bit to force the MUSBHSFC into High-speed mode when it receives a USB reset.
D3	**Test_Packet**	(High-speed mode) The CPU sets this bit to enter the Test_Packet test mode. In this mode, the MUSBHSFC repetitively transmits on the bus a 53-byte test packet, the form of which is defined in the Universal Serial Bus Specification Revision 2.0, Section 7.1.20 (and in Section 16.4 of the MUSBHSFC Programmer’s Guide). Note: The test packet has a fixed format must be loaded into the Endpoint 0 FIFO before the test mode is entered.
D2	**Test_K**	(High-speed mode) The CPU sets this bit to enter the Test_K test mode. In this mode, the MUSBHSFC transmits a continuous K on the bus.
D1	**Test_J**	(High-speed mode) The CPU sets this bit to enter the Test_J test mode. In this mode, the MUSBHSFC transmits a continuous J on the bus.
D0	**Test_SE0_NAK**	(High-speed mode) The CPU sets this bit to enter the Test_SE0_NAK test mode. In this mode, the MUSBHSFC remains in High-speed mode but responds to any valid IN token with a NAK.

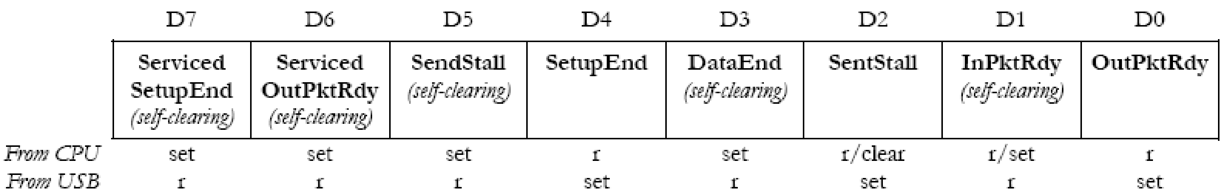
1.2.2 INDEXED REGISTERS

Note: The action of the following registers when the selected endpoint has not been configured is undefined.

1.2.2.1 CSR0

CSR0 is an 8-bit register that provides control and status bits for Endpoint 0. *Note:* Users should be aware that the value returned when the register is read reflects the status attained e.g. as a result of writing to the register.

Address: 12h (with the Index register set to 0)*; Reset value:* 8’h00



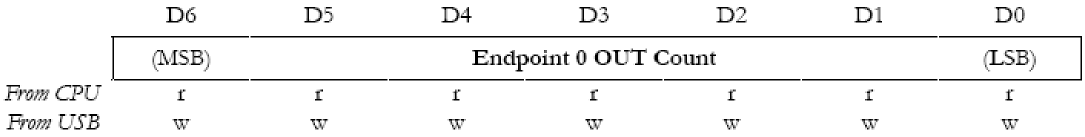
	D7	D6	D5	D4	D3	D2	D1	D0
	Serviced SetupEnd *(self-clearing)*	**Serviced OutPktRdy** *(self-clearing)*	**SendStall** *(self-clearing)*	**SetupEnd**	**DataEnd** *(self-clearing)*	**SentStall**	**InPktRdy** *(self-clearing)*	**OutPktRdy**
From CPU	set	set	set	r	set	r/clear	r/set	r
From USB	r	r	r	set	r	set	r	set

Bit	Name	Function
D7	**ServicedSetupEnd**	The CPU writes a 1 to this bit to clear the SetupEnd bit. It is cleared automatically.
D6	**ServicedOutPktRdy**	The CPU writes a 1 to this bit to clear the OutPktRdy bit. It is cleared automatically.
D5	**SendStall**	The CPU writes a 1 to this bit to terminate the current transaction. The STALL handshake will be transmitted and then this bit will be cleared automatically. Note: This behavior differs from that of the SendStall bits associated with additional IN/OUT endpoints, which need to be cleared by the CPU.
D4	**SetupEnd**	This bit will be set when a control transaction ends before the DataEnd bit has been set. An interrupt will be generated and the FIFO flushed at this time. The bit is cleared by the CPU writing a 1 to the ServicedSetupEnd bit.
D3	**DataEnd**	The CPU sets this bit: 1. When setting InPktRdy for the last data packet. 2. When clearing OutPktRdy after unloading the last data packet. 3. When setting InPktRdy for a zero length data packet. It is cleared automatically.
D2	**SentStall**	This bit is set when a STALL handshake is transmitted. The CPU should clear this bit.
D1	**InPktRdy**	The CPU sets this bit after loading a data packet into the FIFO. It is cleared automatically when the data packet has been transmitted. An interrupt is generated when this bit is cleared (if enabled).
D0	**OutPktRdy**	This bit is set when a data packet has been received. An interrupt is generate when this bit is set (if enabled). The CPU clears this bit by setting the ServicedOutPktRdy bit.

1.2.2.2 COUNT0

Count0 is a 7-bit read-only register that indicates the number of received data bytes in the Endpoint 0 FIFO. *Note:* The value returned changes as the contents of the FIFO change and is only valid while OutPktRdy (CSR0.D0) is set.

Address: 18h (with the Index register set to 0)*; Reset value:* 7'b0000000



	D6	D5	D4	D3	D2	D1	D0
	(MSB)			**Endpoint 0 OUT Count**			(LSB)
From CPU	r	r	r	r	r	r	r
From USB	w	w	w	w	w	w	w

1.2.2.3 INMAXP

The InMaxP register defines the maximum amount of data that can be transferred through the selected IN endpoint in a single operation. There is an InMaxP register for each IN endpoint (except Endpoint 0).

Address: 10h*; Reset value:* 11/13/16'h0000

	D12/15		D11	D10	...	D0
	$m-1$			(MSB)	Maximum Payload/transaction	(LSB)
From CPU	rw	...	rw	rw	...	rw
From USB	r	...	r	r	...	r

Bits 10:0 define (in bytes) the maximum payload transmitted in a single transaction. The value set can be up to 1024 bytes but is subject to the constraints placed by the USB Specification on packet sizes for Bulk, Interrupt and Isochronous transfers in Fullspeed and High-speed operations.

Where the option of High-bandwidth Isochronous endpoints or of packet splitting on Bulk endpoints has been taken when the core is configured, the register includes either 2 or 5 further bits that define a multiplier *m* which is equal to one more than the value recorded.

In the case of Bulk endpoints with the packet splitting option enabled, the multiplier *m* can be up to 32 and defines the maximum number of 'USB' packets (i.e. packets for transmission over the USB) of the specified payload into which a single data packet placed in the FIFO should be split, prior to transfer. (If the packet splitting option is not enabled, D15–D13 is not implemented and D12–D11(if included) is ignored.) **Note:** The data packet is required to be an exact multiple of the payload specified by bits 10:0, which is itself required to be either 8, 16, 32, 64 or (in the case of High Speed transfers) 512 bytes.

For Isochronous endpoints operating in High-Speed mode and with the High-bandwidth option enabled, *m* may only be either 2 or 3 (corresponding to bit 11 set or bit 12 set, respectively) and it specifies the maximum number of such transactions that can take place in a single microframe. If either bit 11 or bit 12 is non-zero, the MUSBHSFC will automatically split any data packet written to the FIFO into up to 2 or 3 'USB' packets, each containing the specified payload (or less). The maximum payload for each transaction is 1024 bytes, so this allows up to 3072 bytes to be transmitted in each microframe. (For Isochronous transfers in Fullspeed mode or if High-bandwidth is not enabled, bits 11 and 12 are ignored.)

The value written to bits 10:0 (multiplied by *m* in the case of high-bandwidth Isochronous transfers) must match the value given in the *wMaxPacketSize* field of the Standard Endpoint Descriptor for the associated endpoint (see *USB Specification* Revision 2.0, Chapter 9). A mismatch could cause unexpected results.

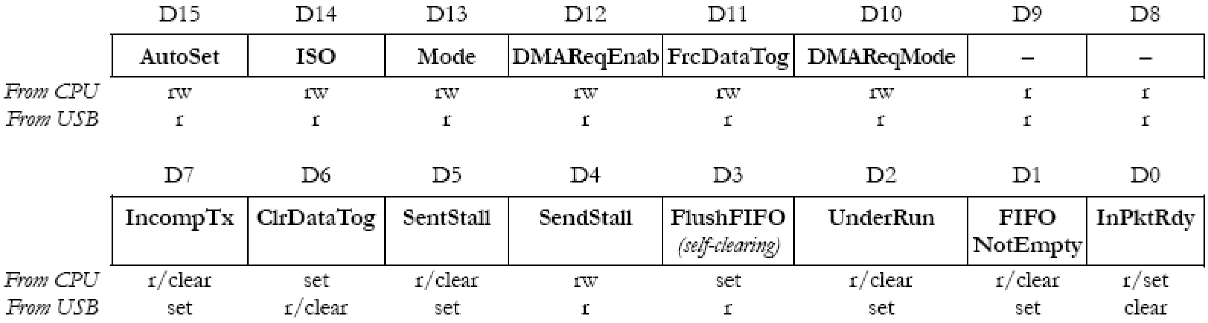
The total amount of data represented by the value written to this register (specified payload × *m*) must not exceed the FIFO size for the IN endpoint, and should not exceed half the FIFO size if double-buffering is required.

If this register is changed after packets have been sent from the endpoint, the IN endpoint FIFO should be completely flushed (using the FlushFIFO bit in InCSR) after writing the new value to this register.

1.2.2.4 INCSR

InCSR is a 16-bit register that provides control and status bits for IN transactions through the currently-selected endpoint. *Note:* Users should be aware that the value returned when the register is read reflects the status attained e.g. as a result of writing to the register.

Address: 12h*; Reset value:*16'h0000



	D15	D14	D13	D12	D11	D10	D9	D8
	AutoSet	**ISO**	**Mode**	**DMAReqEnab**	**FrcDataTog**	**DMAReqMode**	**–**	**–**
From CPU	rw	rw	rw	rw	rw	rw	r	r
From USB	r	r	r	r	r	r	r	r

	D7	D6	D5	D4	D3	D2	D1	D0
	IncompTx	**ClrDataTog**	**SentStall**	**SendStall**	**FlushFIFO** *(self-clearing)*	**UnderRun**	**FIFO NotEmpty**	**InPktRdy**
From CPU	r/clear	set	r/clear	rw	set	r/clear	r/clear	r/set
From USB	set	r/clear	set	r	r	set	set	clear

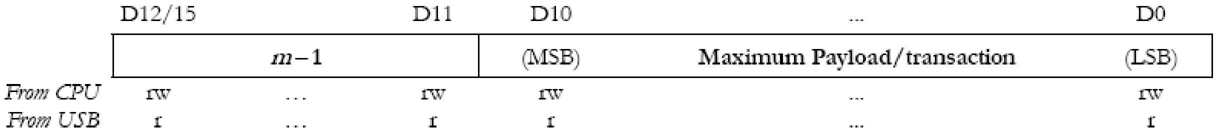
Bit	Name	Function
D15	**AutoSet**	If the CPU sets this bit, InPktRdy will be automatically set when data of the maximum packet size (as set in InMaxP) is loaded into the IN FIFO. If a packet of less than the maximum packet size is loaded, InPktRdy will have to be set manually. Note: Should not be set for high-bandwidth Isochronous endpoints.
D14	**ISO**	The CPU sets this bit to enable the IN endpoint for Isochronous transfers (ISO mode), and clears it to enable the IN endpoint for Bulk/Interrupt transfers.
D13	**Mode**	The CPU sets this bit to enable the endpoint direction as IN, and clears it to enable the endpoint direction as OUT. Note: Only valid where the endpoint FIFO is used for both IN and OUT transactions, otherwise ignored.
D12	**DMAReqEnab**	The CPU sets this bit to enable the DMA request for the IN endpoint.
D11	**FrcDataTog**	The CPU sets this bit to force the endpoint's IN data toggle to switch after each data packet is sent regardless of whether an ACK was received. This can be used by Interrupt IN endpoints that are used to communicate rate feedback for Isochronous endpoints.
D10	**DMAReqMode**	The CPU sets this bit to select DMA Request Mode 1 and clears it to select DMA Request Mode 0. Note: This bit must not be cleared either before or in the same cycle as the above DMAReqEnab bit is cleared.
D9–D8	**–**	Unused, always return 0.
D7	**IncompTx**	When the endpoint is being used for high-bandwidth Isochronous transfers, this bit is set to indicate where a large packet has been split into 2 or 3 packets for transmission but insufficient IN tokens have been received to send all the parts. The remainder of the current packet is then flushed from the FIFO (but any second packet in the FIFO will remain). Note: In anything other than a high-bandwidth Isochronous transfer, this bit will always return 0.
D6	**ClrDataTog**	The CPU writes a 1 to this bit to reset the endpoint IN data toggle to 0.
D5	**SentStall**	This bit is set when a STALL handshake is transmitted. The FIFO is flushed and the InPktRdy bit is cleared (see below). The CPU should clear this bit.
D4	**SendStall**	The CPU writes a 1 to this bit to issue a STALL handshake to an IN token. The CPU clears this bit to terminate the stall condition. Note: This bit has no effect where the endpoint is being used for Isochronous transfers.
D3	**FlushFIFO**	The CPU writes a 1 to this bit to flush the next packet to be transmitted from the endpoint IN FIFO. The FIFO pointer is reset and the InPktRdy bit (below) is cleared. May be set simultaneously with InPktRdy to abort the packet that has just been loaded into the FIFO. Note: (i) FlushFIFO should only be set when InPktRdy is set (at other times, it may cause data corruption). (ii) If the FIFO contains two packets, FlushFIFO may need to be set twice to completely clear the FIFO.

D2	**UnderRun**	In ISO mode, this bit is set when a zero length data packet is sent after receiving an IN token with the InPktRdy bit not set. In Bulk/Interrupt mode, this bit is set when a NAK is returned in response to an IN token. The CPU should clear this bit.
D1	**FIFONotEmpty**	This bit is set when there is at least 1 packet in the IN FIFO.
D0	**InPktRdy**	The CPU sets this bit after loading a data packet into the FIFO. It is cleared automatically when a data packet has been transmitted. If the FIFO is double-buffered, it is also automatically cleared when there is space for a second packet in the FIFO (see Section 9.2). An interrupt is generated (if enabled) when the bit is cleared (suppressed by the built-in DMA controller in DMA Mode 1).

1.2.2.5 OUTMAXP

The OutMaxP register defines the maximum amount of data that can be transferred through the selected OUT endpoint in a single operation. There is an OutMaxP register for each OUT endpoint (except Endpoint 0).

Address: 14h*; Reset value:* 11/13/16'h0000



	D12/15		D11	D10	...	D0
	m−1			(MSB)	**Maximum Payload/transaction**	(LSB)
From CPU	rw	…	rw	rw	...	rw
From USB	r	…	r	r	...	r

Bits 10:0 define (in bytes) the maximum payload transmitted in a single transaction. The value set can be up to 1024 bytes but is subject to the constraints placed by the USB Specification on packet sizes for Bulk, Interrupt and Isochronous transfers in Fullspeed and High-speed operations.

Where the option of High-bandwidth Isochronous endpoints or of combining Bulk packets has been taken when the core is configured, the register includes either 2 or 5 further bits that define a multiplier m which is equal to one more than the value recorded.

For Bulk endpoints with the packet combining option enabled, the multiplier *m* can be up to 32 and defines the number of USB packets of the specified payload which are to be combined into a single data packet within the FIFO. (If the packet splitting option is not enabled, D15–D13 is not implemented and D12–D11 (if included) is ignored.)

For Isochronous endpoints operating in High-Speed mode and with the High-bandwidth option enabled, *m* may only be either 2 or 3 (corresponding to bit 11 set or bit 12 set, respectively) and it specifies the maximum number of such transactions that can take place in a single microframe. If either bit 11 or bit 12 is non-zero, the MUSBHSFC will automatically combine the separate USB packets received in any microframe into a single packet within the OUT FIFO. The maximum payload for each transaction is 1024 bytes, so this allows up to 3072 bytes to be received in each microframe. (For Isochronous transfers in Full-speed mode or if High-bandwidth is not enabled, bits 11 and 12 are ignored.)

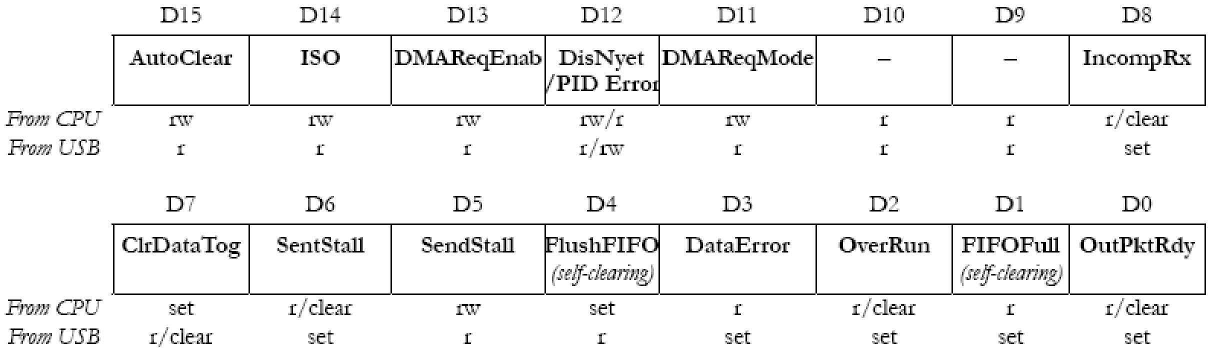
The value written to bits 10:0 (multiplied by *m* in the case of high-bandwidth Isochronous transfers) must match the value given in the *wMaxPacketSize* field of the Standard Endpoint Descriptor for the associated endpoint (see *USB Specification* Revision 2.0, Chapter 9). A mismatch could cause unexpected results.

The total amount of data represented by the value written to this register (specified payload × *m*) must not exceed the FIFO size for the OUT endpoint, and should not exceed half the FIFO size if double-buffering is required.

1.2.2.6 OUTCSR

OutCSR is a 16-bit register that provides control and status bits for OUT transactions through the currently-selected endpoint. It is reset to 0. *Note:* Users should be aware that the value returned when the register is read reflects the status attained e.g. as a result of writing to the register.

Address: 16h*; Reset value:*16’h0000



	D15	D14	D13	D12	D11	D10	D9	D8
	AutoClear	ISO	DMAReqEnab	DisNyet /PID Error	DMAReqMode	–	–	IncompRx
From CPU	rw	rw	rw	rw/r	rw	r	r	r/clear
From USB	r	r	r	r/rw	r	r	r	set

	D7	D6	D5	D4	D3	D2	D1	D0
	ClrDataTog	SentStall	SendStall	FlushFIFO *(self-clearing)*	DataError	OverRun	FIFOFull *(self-clearing)*	OutPktRdy
From CPU	set	r/clear	rw	set	r	r/clear	r	r/clear
From USB	r/clear	set	r	r	set	set	set	set

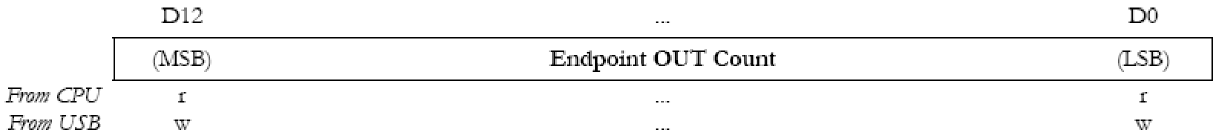
Bit	Name	Function
D15	**AutoClear**	If the CPU sets this bit then the OutPktRdy bit will be automatically cleared when a packet of OutMaxP bytes has been unloaded from the OUT FIFO. When packets of less than the maximum packet size are unloaded, OutPktRdy will have to be cleared manually. Note: Should not be set for high-bandwidth Isochronous endpoints.
D14	**ISO**	The CPU sets this bit to enable the OUT endpoint for Isochronous transfers, and clears it to enable the OUT endpoint for Bulk/Interrupt transfers.
D13	**DMAReqEnab**	The CPU sets this bit to enable the DMA request for the OUT endpoint.
D12	**DisNyet PID Error**	**Bulk/Interrupt Transactions: The CPU sets this bit to disable the sending of NYET handshakes. When set, all successfully received OUT packets are ACK'd including at the point at which the FIFO becomes full. Note: This bit only has any effect in High-speed mode, in which mode it should be set for all Interrupt endpoints. ISO Transactions: The core sets this bit to indicate a PID error in the received packet.**
D11	**DMAReqMode**	Two modes of DMA Request operation are supported: DMA Request Mode 0 in which a DMA request is generated for all received packets, together with an interrupt (if enabled); and DMA Request Mode 1 in which a DMA request (but no interrupt) is generated for OUT packets of size OutMaxP bytes and an interrupt (but no DMA request) is generated for OUT packets of any other size. The CPU sets this bit to select DMA Request Mode 1 and clears this bit to select DMA Request Mode 0.
D10–D9	–	Unused, always return 0.
D8	**IncompRx**	This bit is set in a high-bandwidth Isochronous transfer if the packet in the OUT FIFO is incomplete because parts of the data were not received. It is cleared when OutPktRdy is cleared. Note: In anything other than a high-bandwidth Isochronous transfer, this bit will always return 0.
D7	**ClrDataTog**	The CPU writes a 1 to this bit to reset the endpoint data toggle to 0.
D6	**SentStall**	This bit is set when a STALL handshake is transmitted. The CPU should clear this bit.

D5	**SendStall**	The CPU writes a 1 to this bit to issue a STALL handshake to a DATA packet. The CPU clears this bit to terminate the stall condition. Note: This bit has no effect where the endpoint is being used for Isochronous transfers.
D4	**FlushFIFO**	The CPU writes a 1 to this bit to flush the next packet to be read from the endpoint OUT FIFO. The FIFO pointer is reset and the OutPktRdy bit (below) is cleared. Note: FlushFIFO should only be used when OutPktRdy is set. At other times, it may cause data to be corrupted. Note also that, if the FIFO is double-buffered, FlushFIFO may need to be set twice to completely clear the FIFO.
D3	**DataError**	This bit is set at the same time that OutPktRdy is set if the data packet has a CRC error. It is cleared when OutPktRdy is cleared. Note: This bit is only valid when the endpoint is operating in ISO mode. In Bulk mode, it always returns zero.
D2	**OverRun**	This bit is set if an OUT packet arrives while FIFOFull is set i.e. the OUT packet cannot be loaded into the OUT FIFO. The CPU should clear this bit. Note: This bit is only valid when the endpoint is operating in ISO mode. In Bulk mode, it always returns zero.
D1	**FIFOFull**	This bit is set when no more packets can be loaded into the OUT FIFO.
D0	**OutPktRdy**	This bit is set when a data packet has been received. The CPU should clear this bit when the packet has been unloaded from the OUT FIFO. An interrupt is generated (if enabled) when the bit is set.

1.2.2.7 OUTCOUNT

OutCount is a 13-bit read-only register that holds the number of received data bytes in the packet in the OUT FIFO. *Note:* The value returned changes as the FIFO is unloaded and is only valid while OutPktRdy (OutCSR.D0) is set.

Address: 18h*; Reset value:* 13'b0000000000000



	D12	...	D0
	(MSB)	**Endpoint OUT Count**	(LSB)
From CPU	r	...	r
From USB	w	...	w

1.2.2.8 FIFOx (Addresses 20h – xxh)

This address range provides 16 addresses for CPU access to the FIFOs for each endpoint. Writing to these addresses loads data into the IN FIFO for the corresponding endpoint. Reading from these addresses unloads data from the OUT FIFO for the corresponding endpoint.

If the CPU bus is 16-bit, the address range is 20h – 3Fh and the FIFOs are located on 16-bit word boundaries (Endpoint 0 at 20h, Endpoint 1 at 22h ... Endpoint 15 at 3Eh). If the CPU bus is 32-bit, the address range is 20h – 5Fh and the FIFOs are located on 32-bit double-word boundaries (Endpoint 0 at 20h, Endpoint 1 at 24h ... Endpoint 15 at 5Ch).

Note: Transfers to and from FIFOs may be 8-bit, 16-bit, 24-bit or 32-bit as required, and any combination of access is allowed provided the data accessed is contiguous. However, all the transfers associated with one packet must be of the same width so that the data is consistently byte-, word- or double-word-aligned. The last transfer may however contain fewer bytes than the previous transfers in order to complete an odd-byte or odd-word transfer.

1.3 SOFT CONNECT/DISCONNECT

JZ4740 UDC supports soft connect/disconnect.

After a hardware reset (NRST = 0), Soft Conn is cleared to 0. The UDC will therefore appear disconnected until the software has set Soft Conn to 1. Once the Soft Conn bit has been set to 1, the software can also simulate a disconnect by clearing this bit to 0.

1.4 DMA CONTROLLER REGISTERS

The UDC DMA controller has two modes of operation, referred to as DMA Modes 0 and 1. The controller can further be programmed to conduct transfers using INCR4, INCR8 and INCR16 4/8/16-beat incrementing bursts rather than bursts of unspecified length.

When operating in DMA Mode 0, the DMA controller can be only programmed to load/unload one packet, so processor intervention is required for each packet transferred over the USB. This mode can be used with any endpoint, whether it uses Control, Bulk, Isochronous, or Interrupt transactions.

When operating in DMA Mode 1, the DMA controller can be programmed to load/unload a complete bulk transfer (which can be many packets). Once set up, the DMA controller will load/unload all packets of the transfer, interrupting the processor only when the transfer has completed. DMA Mode 1 can only be used with endpoints that use Bulk transactions.

Each channel can be independently programmed for the selected operating mode.

The DMA controller has one interrupt register which indicates which channels have a pending interrupt, and a set of three control registers for each configured channel. The following table shows the DMA controller register map for two DMA channels:

Address	**Register**	**Description**
200h	INTR	Indicates pending DMA interrupts, one bit per DMA channel implemented. D0 used for DMA Channel 1, D1 for DMA Channel 2 etc. Cleared when read.
204h	CNTL (1)	DMA Channel 1 Control: D0: Enable DMA D1: Direction. 0= DMA Write (OUT endpoint), 1=DMA Read (IN endpoint) D2: DMA Mode D3: Interrupt Enable D7–4: Endpoint number D8: Bus Error D10–9: Burst Mode 00 = Burst Mode 0 : Bursts of unspecified length 01 = Burst Mode 1 : INCR4 or unspecified length 10 = Burst Mode 2 : INCR8, INCR4 or unspecified length 11 = Burst Mode 3 : INCR16, INCR8, INCR4 or unspecified length

208	ADDR (1)	DMA Channel 1 AHB Memory Address (32 bits)
20C	COUNT (1)	DMA Channel 1 Byte Count (32 bits)*
210	–	Unused
214	CNTL (2)	DMA Channel 2 Control
218	ADDR (2)	DMA Channel 2 AHB Memory Address (32 bits)
21C	COUNT (2)	DMA Channel 2 Byte Count (32 bits)*
	etc.	

* Note: If DMA is enabled with a count of 0, the bus will not be requested and a DMA interrupt will be generated.

1.4.1 TRANSFERRING PACKETS

Use of the built-in DMA controller to access the MUSBHSFC FIFOs requires both the DMA controller and the MUSBHSFC endpoint to be appropriately programmed. Many variations are possible. The following sections detail the standard set-ups used for the basic actions of transferring individual packets and multiple packets.

1.4.1.1 INDIVIDUAL PACKET: OUT ENDPOINT

The transfer of individual packets will normally be carried out using DMA Mode 0.

For this, the MUSBHSFC OUT endpoint should be programmed as follows:

The relevant interrupt enable bit in the IntrOutE register set to 1.
The DMAReqEnab bit (D13) of the appropriate OutCSR register set to 0. (**Note:** There is no need to set the MUSBHSFC to generate DMA requests for this operation.)

When a packet has been received by the MUSBHSFC, it will generate the appropriate Endpoint interrupt. The processor should then program selected channel of the DMA controller as follows:

ADDR : Memory address to store packet
COUNT : Size of packet (determined by reading the MUSBHSFC OutCount register)
CNTL : DMA Enable (D0) =1; Direction (D1) =0; DMA Mode (D2) =0; Interrupt Enable (D3) =1; Required Burst Mode (D10–9)

The DMA controller will then request bus mastership and transfer the packet to memory. When it has completed the transfer, it will generate a DMA interrupt (DMA_NINT taken low). The processor should then clear the OutPktRdy bit in the OutCSR register.

1.4.1.2 INDIVIDUAL PACKET: IN ENDPOINT

To carry out this operation using DMA Mode 0, an MUSBHSFC IN endpoint should be programmed as follows:

The relevant interrupt enable bit in the IntrInE register set to 1.
The DMAReqEnab bit (D12) of the appropriate InCSR register set to 0. (**Note:** There is no need to set

the MUSBHSFC to generate DMA requests for this operation.)

When the FIFO in the UDC becomes available, the UDC will interrupt the processor with the appropriate IN Endpoint interrupt. The processor should then program the DMA controller as follows:

ADDR : Memory address of packet to send
COUNT : Size of packet to be sent
CNTL : DMA Enable (D0) =1; Direction (D1) =1; DMA Mode (D2) =0; Interrupt Enable (D3) =1; Required Burst Mode (D10–9).

The DMA controller will then request bus mastership and transfer the packet to the MUSBHSFC FIFO. When it has completed the transfer, it will generate a DMA interrupt. The processor should then set the InPktRdy bit in the MUSBHSFC InCSR register.

1.4.1.3 MULTIPLE PACKET: OUT ENDPOINT

The transfer of multiple packets will normally be carried out using DMA Mode 1.

Where multiple packets are to be received using DMA Mode 1, the DMA Controller should be programmed as follows:

ADDR : Memory address of the buffer in which to store transfer
COUNT : Maximum size of data buffer
CNTL : DMA Enable (D0) =1; Direction (D1) =0; DMA Mode (D2) =1; Interrupt Enable (D3) =1; Required Burst Mode (D10–9).

and the MUSBHSFC OUT endpoint should be programmed as follows:

The relevant interrupt enable bit in the IntrOutE register should be set to 1.
The AutoClear (D15), DMAReqEnab (D13) and DMAReqMode (D11) bits of the appropriate OutCSR register should be set to 1. In Host mode, the AutoReq (D14) bit should also be set to 1.

As each packet is received by the MUSBHSFC, the DMA controller will request bus mastership and transfer the packet to memory. With AutoClear set, the MUSBHSFC will automatically clear the OutPktRdy bit. This process will continue automatically until the MUSBHSFC receives a ‘short packet’ (one of less than the maximum packet size for the endpoint) signifying the end of the transfer. This ‘short packet’ will not be transferred by the DMA controller: instead the MUSBHSFC will interrupt the processor by generating the appropriate Endpoint interrupt. The processor can then read the MUSBHSFC OutCount register to see the size of the ‘short packet’ and either unload it manually or reprogram the DMA controller in Mode 0 to unload the packet.

The DMA controller ADDR register will have been incremented as the packets were unloaded so the processor can determine the size of the transfer by comparing the current value of ADDR against the start address of the memory buffer.

Note: If the size of the transfer exceeds the data buffer size, the DMA controller will stop unloading the FIFO and interrupt the processor via the DMA_NINT line.

1.4.1.4 MULTIPLE PACKET: IN ENDPOINT

To carry out this operation using DMA Mode 1, the DMA controller should be programmed as follows:

ADDR : Memory address of data block to send
COUNT : Size of data block
CNTL : DMA Enable (D0) =1; Direction (D1) =1; DMA Mode (D2) =1; Interrupt Enable (D3) =1; Required Burst Mode (D10–9).

and the MUSBHSFC IN endpoint should be programmed as follows:

The relevant interrupt enable bit in the IntrInE register should be set to 1 (simply so that errors can be detected).
The AutoSet (D15), DMAReqEnab (D12) and DMAReqMode (D10) bits of the appropriate InCSR register should be set to 1.

When the FIFO in the MUSBHSFC becomes available, the DMA controller will request bus mastership and transfer a packet to the FIFO. With AutoSet set, the MUSBHSFC will automatically set the InPktRdy bit. This process will continue until the entire data block has been transferred to the MUSBHSFC. The DMA controller will then interrupt the processor by taking DMA_NINT low. If the last packet to be loaded was less than the maximum packet size for the endpoint, the InPktRdy bit will not have been set for this packet: the processor should therefore respond to the DMA interrupt by setting the InPktRdy bit to allow the last 'short packet' to be sent. If the last packet to be loaded was of the maximum packet size, then the action to take depends on whether the transfer is under the control of an application such as the mass storage software on a Windows system that keeps count of the individual packets sent. If the transfer isn't under such control, the processor should still respond to the DMA interrupt by setting the InPktRdy bit. This has the effect of sending a null packet for the receiving software to interpret as indicating the end of the transfer.

1.5 USB RESET

When a reset condition is detected on the USB, the MUSBHSFC performs the following actions:

- Sets FAddr to 0.
- Sets Index to 0.
- Flushes all endpoint FIFOs.
- Clears all control/status registers.
- Enables all endpoint interrupts.
- Generates a Reset interrupt.

1.6 HIGH SPEED MODE

If the HSEnab bit of the Power register is set (it is set by default at power-up), then when the MUSBHSFC is reset by a USB reset signal from the hub, it will negotiate for High-speed mode. If the USB host, and all hubs between the MUSBHSFC and the host, support High-speed operation then the HSMode bit of the Power register will be set and the MUSBHSFC will operate in High-speed mode. If the High-speed negotiation fails, the HSMode bit will not be set and the MUSBHSFC will operate in Full-speed mode only.