AN2244
Rev. 2, 7/2005

Interconnecting Two MSC8101ADS Boards Across a 60x-Compatible Bus to the Host Interface

By Manoj Bapat

Within the telecommunications infrastructure, communication between devices is an essential requirement. For example, banks of DSP resource are often used for applications such as voice transcoders within base stations, and these devices must receive or transmit data to/from the outside world. Typically, a central controller terminates protocol layers and distributes the payload to one or more DSP banks (or arrays) for further processing (see **Figure 1**). This application note focuses on a subset of the system architecture, describing the interface between an MSC8101 device acting as an integrated communications processor (host MSC8101) and a single MSC8101 device (called the HDI16 MSC8101) acting as a standard DSP via its 16-bit host parallel interface (HDI16). The host MSC8101 accesses the HDI16 MSC8101 host interface registers via a memory mapping on its own 60x-compatible bus.

CONTENTS

1 MSC8101 Device Overview

The 16-bit Freescale MSC8101 processor is based on the StarCore™ SC140 core. On a single device, it integrates the high-performance SC140 four-ALU (arithmetic logic unit) DSP core with 512 KB of internal memory, a communications processor module (CPM), a 64-bit 60x-compatible bus, a very flexible system integration unit (SIU), and a 16-channel DMA controller. The MSC8101 DSP can execute up to four multiply-accumulate (MAC) operations in a single clock cycle. The MSC8101 CPM is a 32-bit RISC-based communications protocol engine that can network to time-division multiplexed (TDM) highways, Ethernet, and asynchronous transfer mode (ATM) backbones.

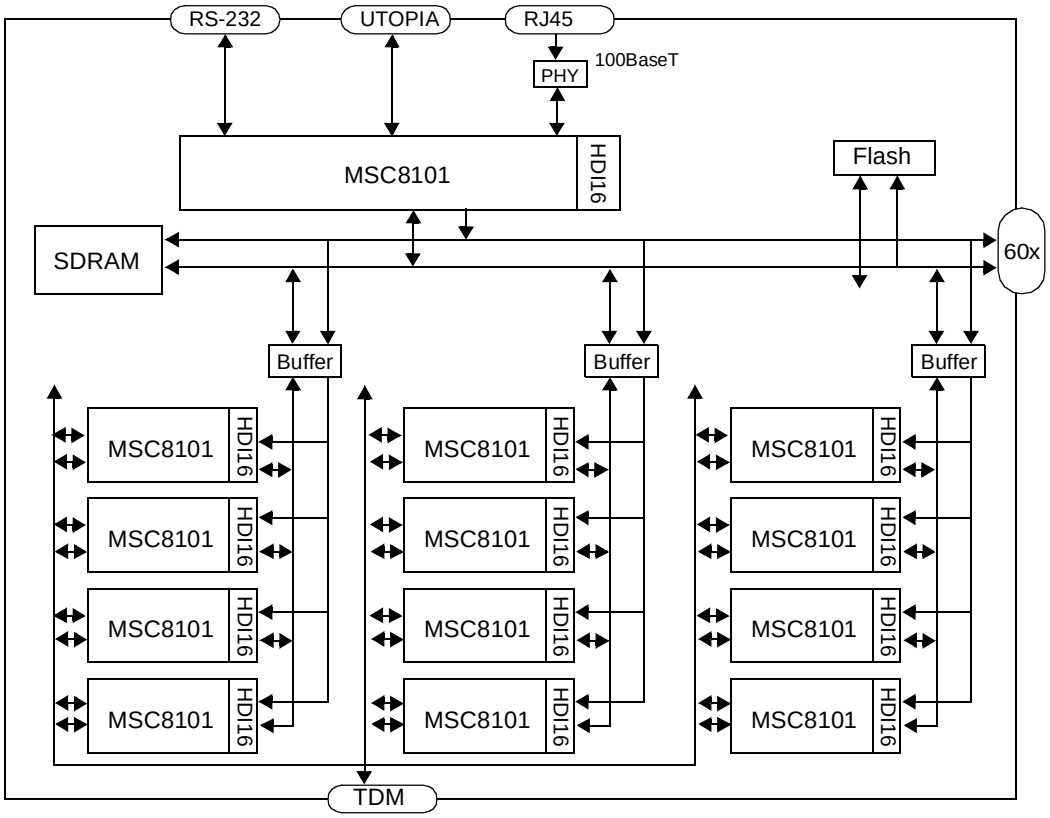


Figure 1. Controller to Multiple-HDI16 DSP Architecture

The large internal memory, 512 KB, reduces the need for external program and data memories. The MSC8101 offers 1200 DSP MIPS or 3000 RISC MIPS performance using an internal 300 MHz clock with a 1.6 V core and independent 3.3 V input/output (I/O). MSC8101 power dissipation is estimated at 0.5 W. The following MSC8101 modules are crucial to the application discussed in this document:

- *Host interface (HDI16).* A 16-bit wide, full-duplex, double-buffered parallel port that can connect directly to the data bus of a host processor. The HDI16 supports a variety of buses and gluelessly connects to a number of industry-standard microcomputers, microprocessors, and DSPs. The HDI16 also supports the 8-bit host data bus, which makes it fully compatible with the DSP56300 HDI08 (as viewed by the host side, not from the DSP side). The host bus can operate asynchronously to the SC140 core clock, and the HDI16 registers are divided into two banks. The host register bank is accessible to the external host, and the core register bank is accessible to the SC140 core.
- *Memory controller*. Supports a glueless interface to the external memory and peripheral devices on the external system bus. It also enables interfacing with the internal DSP memory and DSP peripherals residing on the internal local bus. Located on the external system bus, the memory controller controls a maximum of eight memory banks shared by a high-performance SDRAM machine, a general-purpose chip-select machine (GPCM), and two user-programmable machines (UPMs). It supports a glueless interface to synchronous DRAM (SDRAM), SRAM, EPROM, flash EPROM, burstable RAM, regular DRAM devices, extended data output DRAM devices, and other peripherals. Two additional memory banks control access to resources using the local bus.

The host MSC8101 device must configure its memory controller to map the HDI16 host-side registers into memory locations within its 60x system bus memory space. The HDI16 MSC8101 device configures the HDI16 directly, treating it as an internal peripheral.

This application note describes how to configure and use the HDI16 host interface and the memory controller to perform data transfer in non-DMA mode.[1] Also provided are the following:

- The register settings for these modules as used by this application (**Section 7**, *Source Code, Software Flow, and Register Settings*).
- The hardware connections between the host MSC8101 device and the HDI16 MSC8101 device.
- The physical connections for interfacing the host and HDI16 MSC8101 devices are described for the Freescale MSC8101 Application Development System (MSC8101ADS) (**Section 6**). The MSC8101ADS board is a general development platform with the MSC8101 device installed in a socket, on-board Flash and SDRAM memory, plus communication transceivers for Fast Ethernet, ATM 155-Uni, E1/T1, RS-232, and a Crystal stereo audio codec. With this system, developers can start developing software or use the schematics as a reference for their own system development. [2]
- Example source code in two parts, one for the host MSC8101 device and one for the HDI16 MSC8101 device. These transfers are not optimized. The source code is presented for illustrative purposes, and not for use as an HDI16 driver.

2 System Bus–HDI16 Host Interface

To understand the function of the parallel control interface between the host MSC8101 system bus and the HDI16 MSC8101 host interface, it is important to know the device requirements on each side.

2.1 Host Memory Controller

The memory controller SDRAM machine enables back-to-back memory read or write operations using page mode, pipelined operation, and bank interleaving for high-performance systems. GPCM-based chip selects interface predominantly to simple asynchronous devices such as ROM, Flash memory, SRAM, and so on. A GPCM-derived chip select ensures a glueless interface to such devices over a range of port sizes (8, 16, and 32-bit) and speed grades. However, the UPM offers much more flexibility in timing to target a broader range of system devices, and it offers access to more peripherals, making it more suited to our application. Through the UPM-controlled memory interface, software can define the chip selects and control strobes on each bus clock to a 1/4 clock and 1/2 clock granularity, respectively. Developers commonly use this flexibility for user-defined interfaces to application-specific integrated circuits (ASICs) or, as in our implementation example, to DSPs. The UPM-defined interface can be used with any of the host MSC8101 eight chip selects to give a programmable port size and strobe generation matching that required by the HDI16 MSC8101.

2.2 HDI16 Host Interface

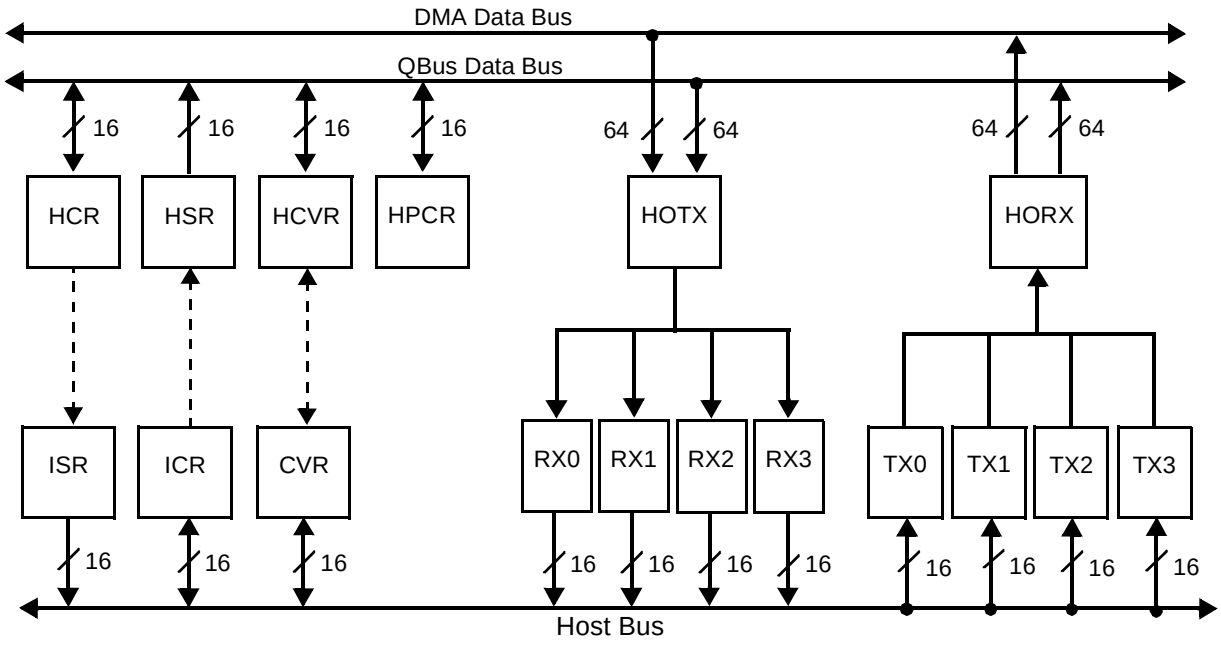
The HDI16 host interface has two sets of 16-bit wide registers, one set visible only within the MSC8101 and the other set visible only to the external host processor. **Figure 2** illustrates the relationship between the two sides. All HDI16 registers are mapped directly onto the MSC8101 QBus, and the transmit and receive FIFOs are mapped onto the DMA data bus so that the DMA controller can access them directly without intervention by the SC140 core. The QBus, which is part of the SC140 extended core interface, is a high-speed pipeline bus with separate address and data phases. It is a single-master bus with the same frequency as the SC140 core. The MSC8101 peripherals, in particular the HDI16 interface, are slaves to the QBus.

1. For details on these MSC8101 modules, consult the HDI16 and memory controller chapters of the *MSC8101 Reference Manual*. For details on programming the HDI16 port, consult the *MSC8101 User's Guide*.
2. For details on the MSC1810ADS board, consult the *MSC8101ADS User's Manual*.

The most important aspect of the HDI16 host interface for our purposes is that it is an asynchronous interface, reducing concerns over clock skew between the HDI16 host interface and the host device buses. Furthermore, since the host MSC8101 accesses all the HDI16 registers in the HDI16 MSC8101 with a single chip select and four address lines, the DSP host interface is in effect an asynchronous memory-mapped region. For the HDI16 host interface in single-strobe mode, the host device asserts a chip select, a single data strobe, and a read/write line to select HDI16 read or write bus operations. The read and write strobes are also used as the data latch control to complete the bus transactions, avoiding the need for any handshake termination signal from the DSP. Through appropriate mode selection, the host MSC8101 UPM-controlled signals fully support the HDI16 feature set. The UPM-defined interface can be used with any of the host MSC8101 eight chip selects to give the 16-bit port size and strobe generation matching that of the HDI16 MSC8101 device host interface.

Internally-Visible Registers

Accessed by the SC140 core		
HCR	Host Control Register	0x0000
HSR	Host Status Register	0x0040
HCVR	Host Command Vector Register	0x0060
HPCR	Host Port Control Register	0x0020
HOTX	Host Transmit Data Register	0x0080
HORX	Host Receive Data Register	0x00A0
Notes: Both HOTX and HORX are FIFOs with a capacity of four 64-bit words.		



Registers Visible to Host

Accessed by the Host Processor					
ISR	Interface Status Register	0x2			
ICR	Interface Control Register	0x0			
CVR	Command Vector Register	0x1			
RX[0–3]	Receive Data Registers	RX0 (0x7)	RX1 (0x6)	RX2 (0x5)	RX3 (0x4)
TX[0–3]	Transmit Data Registers	TX0 (0x7)	TX1 (0x6)	TX2 (0x5)	TX3 (0x4)

Figure 2. MSC8101 HDI16 Port Registers

2.3 Physical Interconnections

Table 1 shows the physical interconnections between the host MSC8101ADS UPM-controlled system bus and the host interface port of the HDI16 MSC8101ADS. The connectors specified in this table refer to the corresponding connector names, as defined in the *MSC8101 Application Development System User's Manual*.

Table 1. Physical MSC8101ADS Interconnects

Host MSC8101 Side				HDI16 MSC8101 Side		
P1 Pin No.	P2 Pin No.	System + CPM Edge Connector		P4 = H0 Pin No.	HDI16 Connector	
		Signal Name	Signal Description		Signal Name	Signal Description
C14		EXPD0	60x-Compatible Data Bus	3	HD0	Host Bidirectional Data Port
C15		EXPD1		4	HD1	
C16		EXPD2		5	HD2	
C17		EXPD3		6	HD3	
C18		EXPD4		7	HD4	
C19		EXPD5		8	HD5	
C20		EXPD6		9	HD6	
C21		EXPD7		10	HD7	
C22		EXPD8		11	HD8	
C23		EXPD9		12	HD9	
C24		EXPD10		13	HD10	
C25		EXPD11		14	HD11	
C26		EXPD12		15	HD12	
C27		EXPD13		16	HD13	
C28		EXPD14		17	HD14	
C29		EXPD15		18	HD15	
A10		EXPA25	Address Bus	21	HA0	Host Interface Address Lines
A11		EXPA26		22	HA1	
A14		EXPA29		23	HA2	
A15		EXPA30		24	HA3	
C4		BTOLCS1[1]	Chip Select 6	25	HCS1	Host Chip Select 1
				26	HCS2	Tie this to 3.3V (P4, Pin 33)
	D10	DREQ1	DMA Request 1 (PC22)	27	HRRQ/HACK	Receive Host Request OP
	D8	DREQ2	DMA Request 2 (PC24)	28	HTRQ/HREQ	Transmit Host Request OP
D10		EXPGPL3	UPM GPL Line 3	29	HRW	Host Read/Write
D12		EXPGPL5	UPM GPL Line 5	30	HDS	Host Data Strobe
B[1–3] C1, C3, C6, C13, D[1–3], D[6–13] D[16–D32]	C31, C32	0V	Ground	1, 2, 19, 20, 35, 36	0V	Ground

Notes: **1.** This signal is designated as BTOLSC1 in the *MSC8101ADS User's Manual* but as $\overline{\text{CS6}}$ in the *MSC8101 Reference Manual*.

Notable features of the connections depicted in **Table 1** are as follows:

- One chip select, CS6 (or BTOLCS1 as it is called here), is used to map memory accesses from the host MSC8101 to the HDI16 MSC8101 HDI16 port and is connected to the HDI16 chip-select line (HCS).
- Two separate General-Purpose Strobe Lines (GPLx) are used in this interface:
  - PGPL3 is programmed to generate the HDI16 read/write line (HRW), which is typically high for a read access and low for a write access.
  - PGPL5 is programmed to generate the HDI16 Data strobe (HDS), which must be asserted for every 16-bit read or write transaction.
- *Data Lines*. The host MSC8101 device 60x-compatible bus data lines (EXPDx) directly connect to the HDI16 data lines (HDx).
- *DMA Request/Service Request Lines*. Two separate DMA Request lines (DREQx) connect to the Service Request signals (HRRQ and HTRQ) on the HDI16.
- *Address Lines*. The address lines between the host MSC8101 and the HDI16 MSC8101 connect to enable burst transfers across the HDI16. The connections are defined as follows:
  - 60x EXPA25 → HA0
  - 60x EXPA26 → HA1
  - 60x EXPA29 → HA2
  - 60x EXPA30 → HA3
- *Ground Lines*. To provide the best ground plane while connecting the two MSC8101ADS boards, it is highly recommended that all grounds be common and connected together.

Although these interconnections include the signals for DMA transfers across the HDI16 interface, this application note and the accompanying software cover only the non-DMA data transfers across the HDI16 interface.

2.4 Host Data Transfers

To facilitate burst transfers, the host 60x bus address lines are connected to dispose of certain 60x signals:

- The host 60x A31 signal is not required because the HDI16 registers are 16-bit word addressed.
- The 60x A27 and A28 signals are disposed so that the host-side transmit and receive registers wrap around the same four 16-bit word addresses.

3 HDI16 Configuration, Synchronization, and Set-Up

This section describes how to configure the HDI16 slave device to enable non-DMA data transfers across the HDI16 interface. The next section describes how to configure the host device. The HDI16 MSC8101 is configured to enable the HDI16 host interface for communications with an external host MSC8101 as follows:

1. Configure the memory controller to map the HDI16 registers into memory.
2. Synchronize the HDI16 MSC8101 and host MSC8101 devices.

To synchronize the host and HDI16 software communications, we use host flags to monitor the status of the communications. We can access eight HDI16 host flags (HF) by polling from the host and HDI16 devices. Either the SC140 core or the host can set or clear these general-purpose flags for HDI16-to-host communications. If any

of HF[0–7] is set, depending on how the host flags are used, this may indicate an application-specific state within the HDI16 or host requiring intervention by the host processor or the HDI16 processor. The values of HF[0–3] and HF[4–7] are reflected as follows:

- For the HDI16 SC140 core side, in the Host Control Register (HCR) and Host Status Register (HSR).
- For the host side, in the Interface Control Register (ICR) and in the Interface Status Register (ISR).

For example, if the HDI16 MSC8101 software modifies these HF values, the host MSC8101 can read the modified values by reading the ISR. HF[0–7] can be used individually or as encoded pairs in a simple HDI16-to-host communication protocol, implemented in both the HDI16 MSC8101 software and host MSC8101 software. Consequently, the HDI16 side acts as a master during the synchronization process. When the HDI16-side software is ready to process commands sent from the host, it sets HF4 to signal to the host that it is awaiting a synchronization acknowledgment. The host acknowledges synchronization by sending back HF0 and HF1. When this sequence completes, the interface is synchronized.

The HDI16 MSC8101treats the HDI16 interface is an internal peripheral, making it easy to transfer data to and from the HDI16 interface and internal memory. Data transfer occurs between the data buffer in memory and the HDI16 host transmit FIFO.

4 Host Device Configuration

The basic operation of the external host is illustrated in the state transition diagram shown in **Figure 3**.

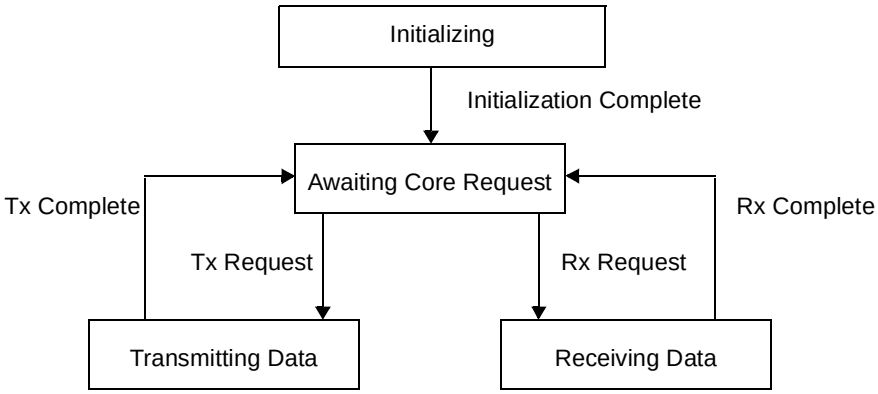


Figure 3. Overview of Host States

To achieve these states, perform the following steps:

1. Configure the host memory controller to enable mapping of the HDI16 registers in memory.
2. Configure the host HDI16 registers, synchronize the host and HDI16 sides, and enable the HDI16 service request signals from the host side.

4.1 Host Memory Controller

The MSC8101 memory controller is used to map each HDI16 host register to a specific memory location. Bank 6 of the memory controller handles these accesses since this bank is free within the current MSC8101ADS memory controller set-up. The Base and Option Registers are programmed to map the HDI16 registers to a base memory address of 0x30000000, select a port size of 16 bits, and enable bursts to the HDI16.

UPM A is selected to issue the required HDI16 signals, so the RAM array for this UPM must be loaded. Where possible, UPM A uses burst transfers across the HDI16, but first a burst DMA transfer size must be selected. Also, the data must be 32-byte aligned in memory. Within the example code, the host-side HDI16 registers are accessed via a type definition (HPORT) that simply overlays the memory-mapped host register locations.

4.2 Host HDI16 Registers

When the memory controller is initialized, the HDI16 registers are accessible and configured to meet the needs of this application. Before initializing any of the HDI16-specific flags, the host must initialize the HDI16 MSC8101 by transmitting its Reset Configuration Word, writing a byte value to each successive reset configuration register. The host then waits for the HDI16 MSC8101 to set a host flag, which indicates that the HDI16 MSC8101 is up and running. To complete this handshake, the host sets a couple of host flags to notify the HDI16 MSC8101 that it is running, too. Using the application source code, you can test the physical HDI16 interconnections via a simple polling mechanism that operates as a continuous loop composed of the following steps, all performed by the host MSC8101:

1. Wait for the HDI16 transmit buffer empty flag.
2. Write sixteen 16-bit words to the HDI16 Transmit Word Registers (these are processed as 16 separate single-beat transactions, not a burst transfer).
3. Wait for the Receive Data Full flag, which indicates that the HDI16 MSC8101 has data ready.
4. Read sixteen 16-bit words to the HDI16 Receive Word Registers.
5. Compare the data transmitted with the data received and store the number of discrepancies.

5 Hardware Timings

The host MSC8101 UPM-controlled bus and the HDI16 MSC8101 host interface are both programmable. Careful programming of the host MSC8101 chip-select registers and UPM can accommodate the HDI16 MSC8101 host port timings. The timings in **Table 1** are based on an implementation with a 40 MHz host MSC8101 60x-compatible bus-to-200 MHz HDI16 MSC8101. On any bus access the critical timing for both reads and writes is typically the data latch point. For the UPM-based read access, the host MSC8101 has the flexibility to latch data on a rising or falling CLKOUT edge. The falling CLKOUT edge is used here to latch the HDI16 data into the host MSC8101 as soon as possible. After the data is latched, an appropriate HDI16 host interface data hold time is ensured before the data strobe (DS) and Chip Select (CS1) are negated. For the UPM-based write access, the critical action is enveloping the DS assertion with CS asserted to ensure a proper write data hold time after latching by the HDI16 host port. Special attention is given to both the host read and write access strobe (DS) negation times (HDS assertion).

The HDI16 MSC8101 specifies some restrictions for consecutive register access, which results in a hold-off negation time for the read and write access strobes. Rather than restrict the firmware to avoid consecutive bus accesses to host port registers, the negation hold-off times are accommodated in the UPM hardware interface settings. Additional clocks are built into the end of UPM-based cycle, giving appropriate time before the next bus cycle starts.

For an application with a 40 MHz host MSC8101 60x-compatible bus and a 200 MHz HDI16 MSC8101, the host port read and write accesses are both five 40 MHz clocks. The timings are based on a buffered connection between the host MSC8101ADS and the HDI16 MSC8101ADS host port. The timings can be readily adapted to allow external decode logic to be added to support chip selects for a larger number of DSP HDI16 host ports.

Note: Within the UPM RAM array definition in the `upminit.c` source file are many `#if 0-#else-#endif` statements that can be changed to `#if 1` to select faster HDI16 transfer UPM patterns. These faster patterns have been successfully tested.

6 Physical MSC8101ADS Settings

Table 2 defines the default switch configuration for the HDI16 MSC8101 platform.[1]

Table 2. HDI16 MSC8101ADS Switch Settings

Switch	Settings used	Comments
SW1 – HOST	On-on-on–on	
SW2 – PPC_CTRL	Off-on-on-on-on-on-on-on	
SW5	32-bit	Mandatory to use the HDI16
SW6	32-bit	Mandatory to use the HDI16
SW9	On-on-on-off-on-off-off-on	SW9/8 is ON to enable the HDI16
SW10	Off-on-off-on	
SW11 – S/W_OPT	On-on-on-on	

Table 3 defines the default switch configuration for the host MSC8101 platform.

Table 3. Host MSC8101ADS Switch Settings

Switch	Settings used	Comments
SW1 – HOST	On-on-on–on	
SW2 – PPC_CTRL	On-on-on-on-on-on-on-on	
SW5	32 bit	
SW6	32 bit	
SW9	On-off-on-off-on-on-off-off	Load the reset configuration word from the Altera Device, disabled HDI16.
SW10	On-on-on-on	
SW11 – S/W_OPT	On-on-on-on	

Our application uses a 16.384 MHz CLKIN crystal on both ADS platforms. With the SW9 settings defined previously, MODCLK 40 is selected, yielding a host 60x bus clock speed of ~40 MHz on the host MSC8101 device and a DSP core speed of ~200 MHz on the HDI16 MSC8101 device.

7 Source Code, Software Flow, and Register Settings

This application was developed using the CodeWarrior™ for StarCore, production release 2.6. The CodeWarrior IDE provides a set of tools for developing software using a GUI. The project file references an `8101_Initialization.cfg` file with which it can configure SDRAM and so on via the debugger. The location of this file can be modified to suit your own implementation. **Table 4** defines both the host-side and DSP-side source code project files associated with this application note.

1. For details on how to power up and set up the MSC8101ADS board, consult the *MSC8101ADS User's Manual*. One chapter of this manual presents installation instructions for the MSC8101ADS board.

Table 4. Source Code Project Files

Module	Filename	Description
HDI16 MSC8101	8101.mcp	CodeWarrior Project File
	main.c	Main program
	hi16.c	HDI16 procedures
	MmapQ001.h	8101 Register Memory Mapping
	reg8101.h	8101 Registers
	hi16.h	HDI16 procedure and variable declarations
	type8101.h	Specific variables types used in this application
Host MSC8101	HDI16Host.mcp	Metrowerks project file
	hdi16.c	Initializes HDI16, initializes HDI16 MSC8101, sets up transmit test pattern and received test pattern destination address.
	upminit.c	The functions within are used to memory map the HDI16 (Host Interface) for the host MSC8101 by initializing UPM A.
Host MSC8101 Cont.		
	msc8101.h	MSC8101 Register Memory Mapping
	types.h	Specific variables types used in this application
	hdi16.h	Includes common header files, defines HDI16 memory map and card_initialize (UPM set-up) function prototype.
	hdi16masks.h	Defines bit masks for HDI16 host registers and a time-out count value for “wait” loops.

Table 5 and **Table 6** list the register settings for the HDI16 MSC8101 device and the host MSC8101 device. Unless indicated otherwise, all registers are described in detail in the *MSC8101 Reference Manual*, so most of the chapter/section references in the first column of the table are to this manual.

Table 5. HDI16 MSC8101 Register Settings

Register	Bits	Setting	Description
MSC8101ADS BCSR (0x40000000) *MSC8101 Application Development System User’s Manual*, section on “Board Control and Status Registers” in the Support Information chapter.	BCSR0/1	1	Select Host Request mode.
HDI16 Host Port Control Register (HPCR) (0x0081) “Core-Side Programming Model” in the Host Interface (HDI16) chapter.	8 15	1 1	Enable HDI16 peripheral. Host address 0x4 is used as host DMA transfer acknowledge input.
HDI16 Host Control Register (HCR) (0x8000 | 0x0000) “Core-Side Programming Model” in the Host Interface (HDI16) chapter.	15	1 | 0	Host flag 4 ON or OFF.

Table 6. Host MSC8101 Register Settings

Register	Bits	Setting	Description
Memory Controller Option Register 6 (OR6) (0XFFF00000) “Memory Controller Programming Model” in the Memory Controller chapter.	0–16 23	11111111 11110000 0 0	The corresponding address bits are used in the comparison with address pins. The banks support bursts.
Memory Controller Base Register 6 (BR6) (0x30001081) “Memory Controller Programming Model” in the Memory Controller chapter.	0–16 19–20 24–26 31	00110000 00000000 0 10 100 1	Bank address. 16-bit port size. UPM A machine select. Valid bank.
Machine A Mode Register (MAMR) (0 | 0x10000000) “Memory Controller Programming Model” in the Memory Controller chapter.	2–3	00 | 01	Normal operation | Write to array.

8 Testing the HDI16 Data Transfer Application

Following are the steps for setting up and testing the application provided in the software accompanying this application note. The comments embedded in the source code provide more details:

1. After the two MSC8101ADS boards are interconnected as described earlier, load the `HDI16Host.mcp` project on to the host MSC8101ADS using JTAG.
2. Ensure that in the CodeWarrior settings panel on the host side, under SC100 Debugger Target, the **RESET ON CONNECT** option is selected in the panel.
3. Ensure that the correct `8101_initialization.cfg` file is used.
4. Execute the host-side project code.

   The host-side MSC8101 code provides the hard reset configuration word to the slave HDI16 MSC8101.
5. After the reset, which is visually indicated when LD9 and LD10 turn OFF on the slave MSC8101ADS, load the slave-side project (`8101.mcp`) on the slave HDI16 MSC8101ADS using JTAG.
6. Ensure that in the CodeWarrior settings panel for the slave-side HDI16 MSC8101, under SC100 Debugger Target, the **RESET ON CONNECT** option is selected in the panel.
7. Ensure that the correct `8101_initialization.cfg` file is used.
8. Execute the slave side code.
9. Observe the code running on the host MSC8101. If the data transfer through the HDI16 occurs correctly, no error is counted by the error count variable in the code.

How to Reach Us:

Home Page:
www.freescale.com

E-mail:
support@freescale.com

USA/Europe or Locations not listed:
Freescale Semiconductor
Technical Information Center, CH370
1300 N. Alma School Road
Chandler, Arizona 85224
+1-800-521-6274 or +1-480-768-2130
support@freescale.com

Europe, Middle East, and Africa:
Freescale Halbleiter Deutschland GMBH
Technical Information Center
Schatzbogen 7
81829 München, Germany
+44 1296 380 456 (English)
+46 8 52200080 (English)
+49 89 92103 559 (German)
+33 1 69 35 48 48 (French)
support@freescale.com

Japan:
Freescale Semiconductor Japan Ltd.
Headquarters
ARCO Tower 15F
1-8-1, Shimo-Meguro, Meguro-ku,
Tokyo 153-0064, Japan
0120 191014 or +81 3 5437 9125
support.japan@freescale.com

Asia/Pacific:
Freescale Semiconductor Hong Kong Ltd.
Technical Information Center
2 Dai King Street
Tai Po Industrial Estate
Tai Po, N.T. Hong Kong
+800 2666 8080

For Literature Requests Only:
Freescale Semiconductor Literature Distribution Center
P.O. Box 5405
Denver, Colorado 80217
1-800-441-2447 or 303-675-2140
Fax: 303-675-2150
LDCForFreescaleSemiconductor@hibbertgroup.com

Document Order No.: AN2244
Rev. 2
7/2005