. .eescale Semiconductor
Application Note

AN2945
Rev. 0, 3/2005

Booting an MSC711x Device from an MPC8272 Host Using the HDI16 Interface

By Manoj Bapat

In DSP applications such as wireless infrastructure and packet telephony, the host processor writes the data and source programs to slave devices during boot loading. Therefore, the option to transmit source programs through the host interface is an important feature of the Freescale MSC711x DSPs. The boot loader program stored in the ROM provides the code necessary to perform this procedure after power-on reset. This application note describes a general programming model for boot loading an MSC711x device through the HDI16 host interface. For purposes of illustration, it focuses on the MSC7115 DSP.

The boot loader program stored in MSC7115 ROM loads and executes the source programs that are distributed by the host processor connected to the HDI16 port. While the boot loading procedure can be performed using any compatible host processor, a host MPC8272 processor is used in the application discussed here.

This document assumes a working knowledge of the MSC7115 and MPC8272 architectures, along with the ability to program the MPC8272. For information on the MSC7115 HDI16 pins, registers, and programming model, refer to the *MSC711x Reference Manual*. For details on the MSC711xADS, refer to the *MSC711xADS Reference Manual*. For information on the MPC8272 processor architecture, registers, memory controller, and programming model, refer to the *MPC8272 Reference Manual*. All of these manuals are available at the Freescale Semiconductor web site listed on the back cover of this application note.

CONTENTS

1 Booting Basics

When the MSC7115 device first exits reset, there is not yet user code available for it to execute. The boot program, which resides in the MSC7115 internal ROM, initializes the MSC7115 after it completes a reset sequence. The purpose of booting is to load data, usually executable code, from an external source, such as an external device or external memory, into the memories in the MSC7115 memory map. After booting, control transfers from the MSC711x boot program to the newly loaded user application code. The MSC7115 can boot from an external host through the HDI16 port or download a user boot program through the I^{2}C port. The boot operating mode is set by configuring the BM[1–0] pins, which are sampled on the rising edge of $\overline{\text{PORESET}}$. **Table 1** shows the mode options for BM[1–0]. **Figure 1** shows an overview of the boot process.

Table 1. Boot Sources

External Connection		Boot Source
BM1	**BM0**	
0	0	External host via the HDI16 with the PLL disabled
0	1	I^{2}C port
1	0	External host via the HDI16 with the PLL enabled
1	1	Reserved

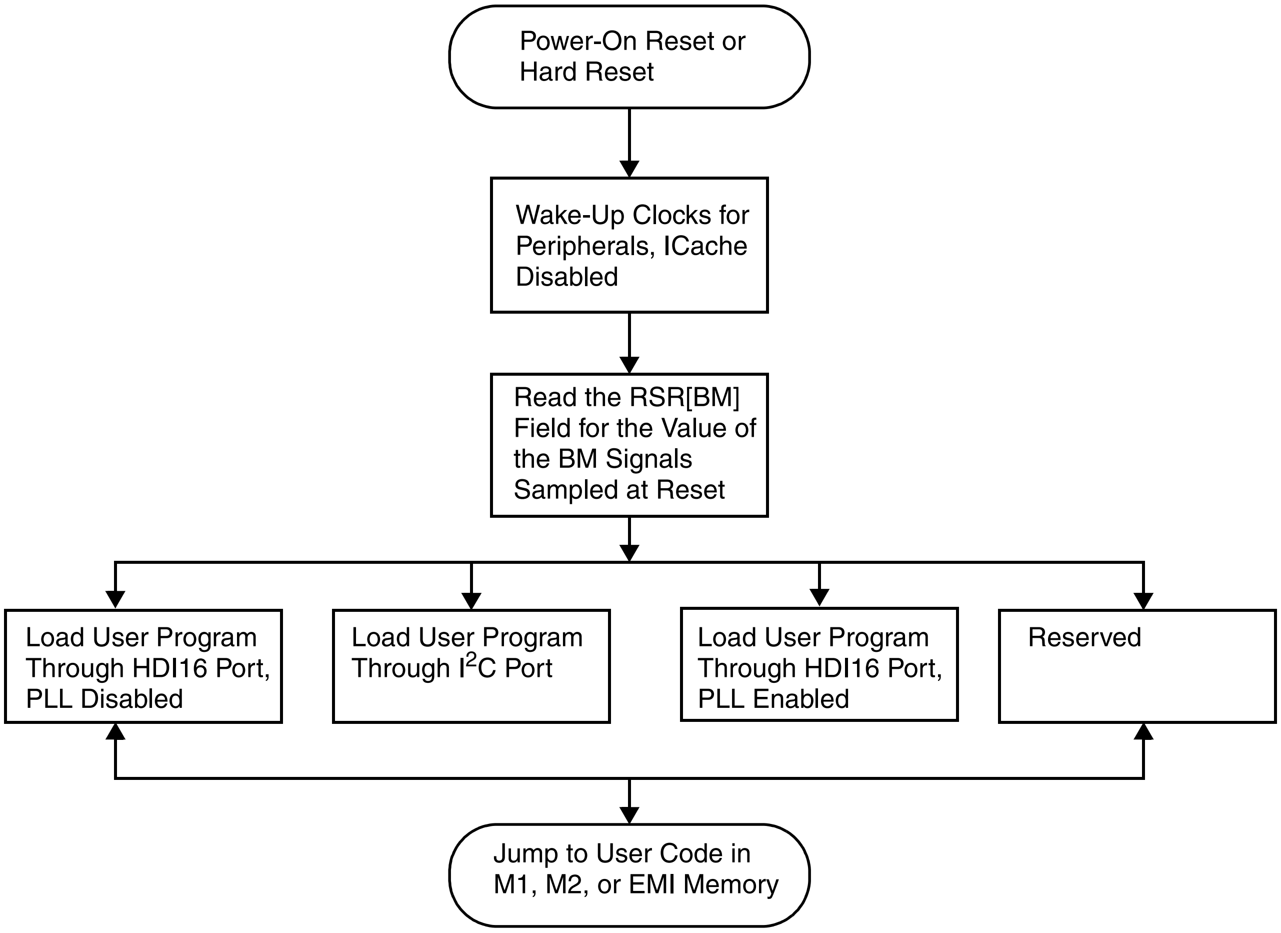

Power-On Reset or Hard Reset
Wake-Up Clocks for Peripherals, ICache Disabled
Read the RSR[BM] Field for the Value of the BM Signals Sampled at Reset
Load User Program Through HDI16 Port, PLL Disabled
Load User Program Through I²C Port
Load User Program Through HDI16 Port, PLL Enabled
Reserved
Jump to User Code in M1, M2, or EMI Memory

Figure 1. Boot Process

A common MSC711x booting technique is to load a small user boot program through one of the valid boot sources shown in **Table 1** into M1 memory and then to execute code from this newly loaded user boot program to load a much larger user program into the other memories of the MSC711x memory map. Loading code into M1 memory provides more flexibility for booting large programs into an MSC711x DSP because a user boot program can be better configured for booting from a user system. The overall steps in this procedure for the MSC7115 device are as follows:

1. Reset the MSC7115 device.
2. Exit from reset into the MSC711x boot program.
3. Use this boot program to load your boot program.
4. Jump to this user boot program when it finishes loading.
5. Use the user boot program to load the application code.
6. Jump to this application code when it is loaded.

An alternative to loading your user boot program is loading the application code directly from one of the MSC711x boot sources. The MSC7115 device is configured as follows out of power-on reset when the boot program is executing:

- ICache is disabled.
- The PLL is bypassed or enabled based on the sampled values of BM[1–0] at power-on reset.
- Software watchdog timer is disabled.
- Execution begins with the power-on reset vector located in the boot ROM.
- Maskable interrupts are disabled.
- Non-maskable interrupts are enabled.

The MSC7115 device is configured as follows out of hard reset when the boot program is executing:

- ICache is disabled.
- The device is clocked the same way it was clocked before the hard reset.
- Software watchdog timer is disabled.
- Execution begins with the hard reset vector in the boot ROM.
- Maskable interrupts are disabled.
- Non-maskable interrupts are enabled.

The software watchdog timer is disabled at power-on reset or is reset by any technique that generates a hard reset. As a result, the software watchdog timer is not active during boot loading. However, the watchdog timer is not disabled by a soft reset, so appropriate care should be taken that the watchdog does not expire.

2 HDI16 Boot Programming

This section considers boot programming from the MSC711x side and from the host HDI16 side.

2.1 Slave MSC7115 Side

The host data interface provides a simple mechanism for connecting a host processor to an MSC711x device. For a boot through the HDI16 interface, the port is configured as follows:

- Non-DMA mode.
- Operate in Polled mode on the MSC711x side.
- Operate in Polled mode on the external host side.
- The external host must write four 16-bit values at a time from the boot data record, as follows:
  - TX0 contains the first word.
  - TX1 contains the second word.
  - TX2 contains the third word.
  - TX3 contains the fourth word.

  TX0 is the most significant (MS) word, and TX3 is the least significant (LS) word.

A boot from power-on reset is additionally configurable as follows:

- 8- or 16-bit mode as specified by the device H8BIT pin.
- Data strobe as specified by the device HDDS and HDSP pins.

2.1.1 Record Structure for Boot Data

The source program can be organized into blocks and placed into an HDI16 boot data record. Each block is either a data block or an instruction block, and it can be loaded to a different specified destination. The checksum method ensures correct data loading. Each block specifies its size and destination address and ends with a checksum, as shown in **Figure 2**.

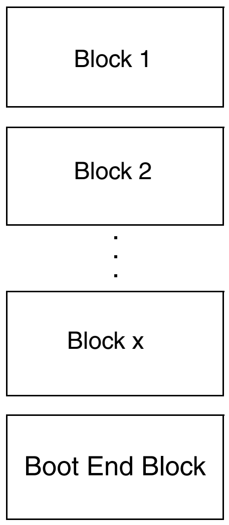


Figure 2. Boot Code Stream Structure

2.1.2 Host Flags

The HDI16 host flags are used to facilitate communication between the MSC711x boot program and external host processor during booting. Several different HDI16 host flags are used.

- ICR[HF3] is set by the host to indicate that the checksums should be compared. This function is optional and is user-programmable.
- HCR[HF5] is set by the SC1400 core to indicate that the HDI16 port is initialized and ready to receive data.
- HCR[HF4] is set by the SC1400 core to indicate that the code has been loaded.
- HCR[HF7] is set by the SC1400 core to indicate that an error occurred during the loading since the checksums do not match. This bit is sticky. That is, if the bit is set for one block, it remains set until the end of the code transfer. The host can ascertain whether errors occurred during code transfer by reading the ISR[HF7] bit.

2.2 Host MPC8272 Side

The MPC8272 device is a versatile communications processor that integrates a high-performance PowerPC™ RISC microprocessor, a very flexible system integration unit, encryption hardware, and many communications peripheral controllers that can be used in a variety of applications, particularly in communications and networking systems. The memory controller controls a maximum of eight memory banks sharing a high performance SDRAM machine, a general-purpose chip-select machine (GPCM), and three user-programmable machines (UPMs). It supports a glue less interface to synchronous DRAM (SDRAM), SRAM, EEPROM, Flash EEPROM, burstable RAM, regular DRAM devices, extended data output DRAM devices, and other peripherals. This flexible memory controller allows the implementation of memory systems with very specific timing requirements.

- The SDRAM machine provides an interface to synchronous DRAMs, using SDRAM pipelining, bank interleaving, and back-to-back page mode to achieve the highest performance.
- The GPCM provides interfacing for simpler, lower-performance memory resources and memory-mapped devices. The GPCM has inherently lower performance because it does not support bursting.
- The UPM supports address multiplexing of the external bus, refresh timers, and generation of programmable control signals for row address and column address strobes to allow for a glueless interface to DRAMs, burstable SRAMs, and almost any other kind of peripheral. The refresh timers allow refresh cycles to be initiated. The UPM can generate different timing patterns for the control signals those govern a memory device. These patterns define how the external control signals behave during a read, write, burst-read, or burst- write access request. Refresh timers are also available to generate user-defined refresh cycles periodically.

For the bootload program, the host-side UPM memory controller is programmed to provide the necessary delay.

2.2.1 Programming the UPMs

The three user-programmable machines (UPMs) are flexible interfaces that connect to a wide range of memory devices. At the heart of each UPM is an internal-memory RAM array that specifies the logical value driven on the external memory controller pins for a given clock cycle. Each word in the RAM array provides bits that allow a memory access to be controlled with a resolution of up to one quarter of the external bus clock period on the byte-select and chip-select lines. The UPM is a micro-sequencer that requires microinstructions or RAM words to generate signal timings for different memory cycles.

The UPMs are programmed as follows:

1. Set up BRx and ORx.
2. Write patterns into the RAM array.
3. Program MPTPR and L/PURT if refresh is required.
4. Program the machine mode register (MxMR).

To write patterns to the RAM array, set MxMR[OP] = 01 and access the UPM with a single byte transaction. For details on UPM programming, refer to the *MPC8272 Reference Manual*.

2.2.2 Tasks Performed by External Host

When an MSC711x device is booted from an external host, the external host waits for the MSC711x boot program to finish its default initialization and then initializes the device by loading code and data to the internal memory. The external host polls a valid bit accessible through the HDI16 host flags. A second valid bit is set when the MSC711x boot code finishes the default initialization and the external host can access the internal resources, including internal memory. When the external host finishes its initialization sequence, it should notify the MSC7115 device by asserting a host port bit to be read by the SC1400 core. In summary, the user boot program running on the external host performs the following overall steps:

1. Waits for the assertion of the valid bit in the HDI16.
2. Loads code and data to internal RAM.
3. Signals the SC1400 core to initiate a jump to the target address provided by the final record.

2.2.3 External Host Side Boot Load Flow

The external host performs the following steps when an MSC711x device is booted through the host interface:

1. Writes the boot code in blocks conforming to the specified format.
2. Specifies pointers for ISR, ICR, and Tx registers. If checksum calculation and verification is required, the host sets the ICR[HF3] bit to indicate the requirement to the MSC711x boot loader.
3. Resets the MSC711x device.
4. If checksum calculation and verification is required, the host sets the ICR[HF3] bit to indicate the requirement to the MSC711x boot loader.
5. Polls the ISR[HF5] bit to determine whether the HDI16 is ready to receive data. The data transfers to the Tx register 64 bits at a time. After each transfer, the host polls the ISR[TXDE] bit, and when ready, receives and then transfers the next 64 bits.
6. The MSC7115 device indicates that all bits are transferred by setting the ISR[HF4] bit.
7. When the checksum function is used, the host checks the ISR[HF7] bit to detect checksum errors at the end of the code transfer.

Figure 3 summarizes these steps.

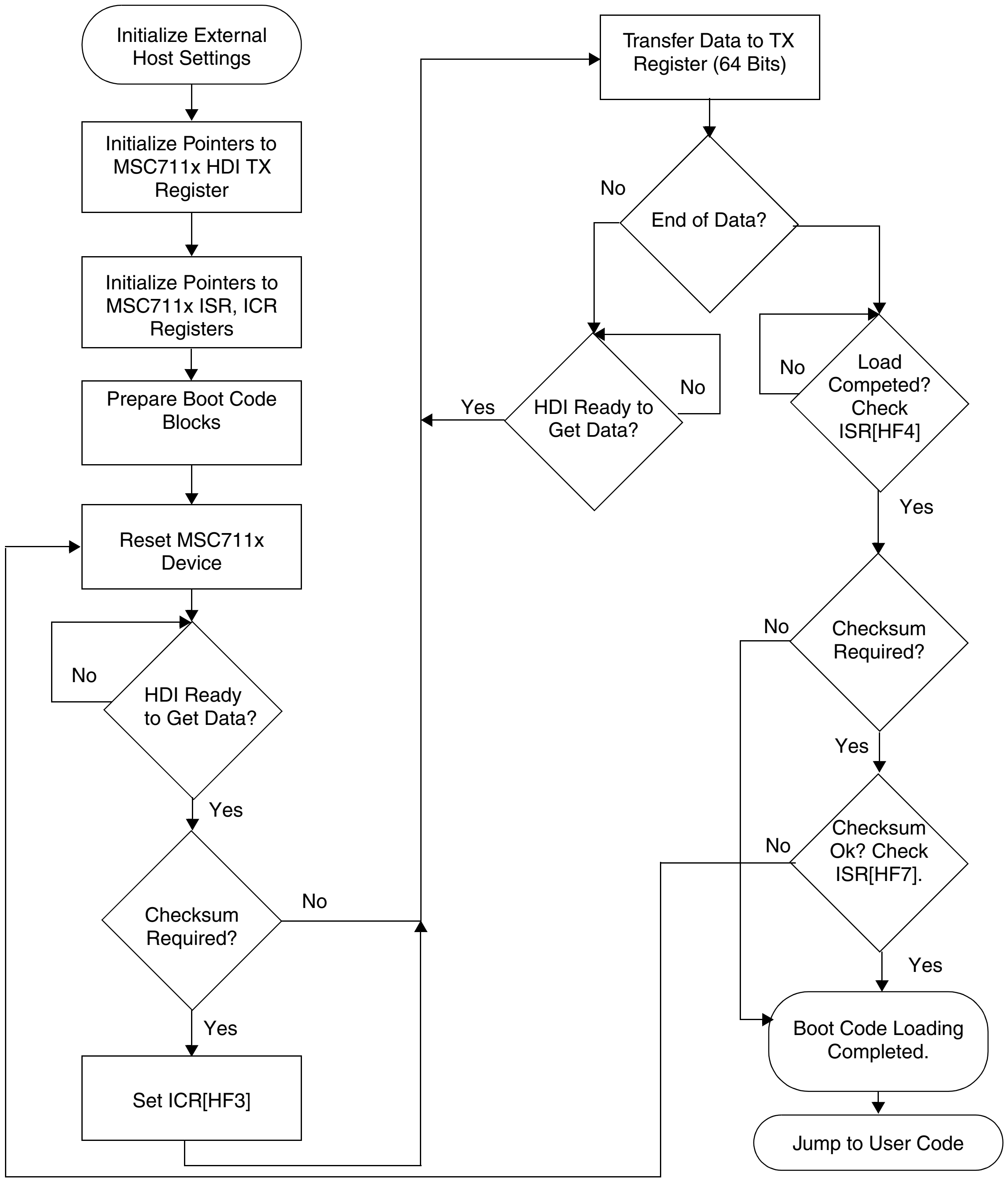

Initialize External Host Settings
Initialize Pointers to MSC711x HDI TX Register
Initialize Pointers to MSC711x ISR, ICR Registers
Prepare Boot Code Blocks
Reset MSC711x Device
No
HDI Ready to Get Data?
Yes
Checksum Required?
No
Yes
Set ICR[HF3]
Transfer Data to TX Register (64 Bits)
No
End of Data?
Yes
HDI Ready to Get Data?
No
No
Load Competed? Check ISR[HF4]
Yes
No
Checksum Required?
Yes
No
Checksum Ok? Check ISR[HF7].
Yes
Boot Code Loading Completed.
Jump to User Code

Figure 3. External Host Procedure During HDI16 Boot

2.2.4 Host Interface Boot Procedure

The HDI16 boot code loads the boot data records, stores the boot data at a given address, and performs error checking via checksums. The boot loader is enabled and initialized as follows:

1. The boot loader routine enables the HDI16 port by setting the HPCR[HEN] bit and checks the value of HPCR[H8BIT] to determine which port size the host is using (see **Figure 4**).
2. The external host initializes the HDI16 when it is ready to start the boot source code transfer by setting the ICR[INIT] bit.

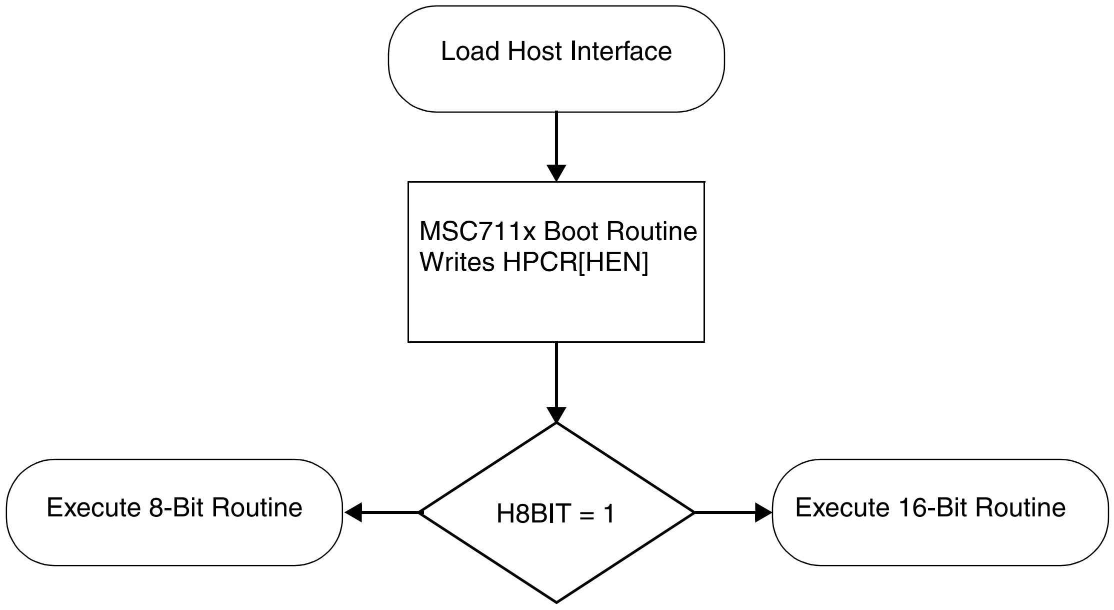


Figure 4. Host Port Size Configuration Check

2.2.5 HDI16 Boot Data Records

The data stream that is booted through the HDI16 is organized as a set of records. Each HDI16 boot data record contains the fields listed in **Table 2**. You must organize the boot data into records beforehand so that the boot program receives the data correctly through the HDI16 port.

Table 2. HDI Boot Record Fields in the Order Received by the HDI16

Field Name	Size	Description
Block size	32 Bits	Contains the number of 16-bit boot data entries, N, within this record. For example, a record with two 16-bit boot data entries specifies a value of 0x00000002 for the size of record. The overall size of each block, starting from the first bit of the block size to the last bit of the checksum, must be a multiple of eight bytes. The block size N must satisfy the following equation (where M is an integer): $N = 4 \times M+2$. This rule applies because the boot program always receives data from the HDI in 64-bit quantities. The last two words are reserved for the final checksum values. **Note:** The minimum number of data entries in a record is 2 (for the case where $M = 0$).
Load address	32 Bits	Specifies where the boot data is to be loaded in its destination memory. This value must be aligned on a 16-byte boundary.

Table 2. HDI Boot Record Fields in the Order Received by the HDI16 (Continued)

Field Name	Size	Description
Boot data entries	N x 16-bits	Contains the actual data values that are loaded into the destination memory. The number of values loaded, N, is the block size. These values usually contain the desired user program that is booted into the MSC711x device. This data must be organized in Big-Endian format. That is, the most significant portion is loaded at the lower order address.
$\overline{\text{Checksum}}$	16 Bits	Contains the expected value for the one’s complement of the checksum.
Checksum	16 Bits	Contains the expected checksum for the boot record to be compared against the checksum.
Note: Checksum comparison enable: Enabled by the ICR[HF3] host flag.		

The record structure does not indicate the total number of records. Instead, the end of the data records is indicated by a special final record with a 32-bit block size of 0x00000000. **Table 3** shows the structure of this final record.

Table 3. Structure of the Final Record

Word	Description
1	0x0000
2	0x0000
3	Target Address — jump to this location when boot completes (most significant 16-bits)
4	Target Address — jump to this location when boot completes (least significant 16-bits)
5	$\overline{\text{Checksum}}$—$\overline{\text{XOR}}$ including address
6	Checksum—XOR including address
7	0x0000
8	0x0000

Table 4 shows three records containing boot data followed by a final record to indicate the end of the HDI16 boot data records. An application can contain any number of records as long as there is at least one record. The example in **Table 4** shows a generalized size for record 1, a block size of 2 for record 2, and a block size of 6 for record 3. The final record’s target address indicates the address at which the program execution continues when the boot process completes. This address must be aligned on a 16-byte boundary.

Table 4. Example Record Structure of Boot Data Received Through HDI16 Port

Record	Word[1]	Description
1	1	Size of record 1 (most significant 16-bits)
	2	Size of record 1 (least significant 16-bits)
	3	Load address where data from record 1 is to be loaded (most significant 16-bits)
	4	Load address where data from record 1 is to be loaded (least significant 16-bits)
	5	Boot data: First 16-bit word
	...	**...**
	n	Boot data: Last 16-bit word
	n+1	$\overline{\text{Checksum}}$—$\overline{\text{XOR}}$ for record 1
	n+2	Checksum—XOR for record 1
2	1	Size of record 2 (most significant 16-bits)
	2	Size of record 2 (least significant 16-bits)
	3	Load address where data from record 2 is to be loaded (most significant 16-bits)
	4	Load address where data from record 2 is to be loaded (least significant 16-bits)
	5	Boot data: first 16-bit word
	6	Boot data: second 16-bit word
	7	$\overline{\text{Checksum}}$—$\overline{\text{XOR}}$ for record 2
	8	Checksum—XOR for record 2
3	1	Size of record 3 (most significant 16-bits)
	2	Size of record 3 (least significant 16-bits)
	3	Load address where data from record 3 is to be loaded (most significant 16-bits)
	4	Load address where data from record 3 is to be loaded (least significant 16-bits)
	5	Boot data: First 16-bit word
	6	Boot data: Second 16-bit word
	7	Boot data: Third 16-bit word
	8	Boot data: Fourth 16-bit word
	9	Boot data: Fifth 16-bit word
	10	Boot data: Sixth 16-bit word
	11	$\overline{\text{Checksum}}$—$\overline{\text{XOR}}$ for record 3
	12	Checksum—XOR for record 3

Table 4. Example Record Structure of Boot Data Received Through HDI16 Port (Continued)

Record	Word[1]	Description
Final Record	1	0x0000
	2	0x0000
	3	Target address — jump to this location when boot completes (most significant 16-bits)
	4	Target Address — jump to this location when boot completes (least significant 16-bits)
	5	$\overline{\text{Checksum}}$—$\overline{\text{XOR}}$ for last record
	6	Checksum—XOR for last record
	7	0x0000
	8	0x0000

Notes: **1.** Each word represents 16 bits.

When more than one code block is included in the source program data stream, word n + 5 contains the address of the second block, as shown in **Table 4**. The sequence repeats for subsequent blocks until the final block in the data stream arrives.

2.2.6 Broadcast Boot Facility

The MSC711x broadcast boot facility is useful when the host needs to boot several devices in the same way. For this method, the broadcast chip-select pin is ORed with the chip-select pin. To broadcast the commands to all devices, the host asserts the broadcast chip-select pin simultaneously for all MSC711x devices. To access a single MSC711x, the relevant chip-select pin is asserted.

3 MSC711xADS Settings and Connections for Boot Load Test

On the MSC711x device, the HPCR[HLEND] bit must always have a value of 0, which configures the HDI16 module for big-endian operation. This is the default value out of reset. The HDI16 port on the MSC711x differs from the MSC8101 HDI16 port in several ways. The main differences are as follows:

- Bits on the HD bus are numbered with bit 0 as the LSB.
- Bits on the HA bus are numbered with bit 0 as the LSB.
- HDI16 register bits are numbered with bit 0 as the LSB.
- The Host Port (HPE) pin does not exist as a pin on MSC711x devices. Instead, this signal is internally tied as asserted. That is, the host port is enabled. To disable this port, use the HPCR[HEN] bit.
- The HDSP pin value is sampled only at reset.
- The H8BIT pin value is sampled only at reset.
- The host address bus uses only three pins HA[2–0]. The HA3 is not in use and is internally tied to 0.
- The reset configuration registers are not accessible to an external host and are not supported.

The remainder of this section covers the HDI16 bus signals, the HDI16 bus connection to the host processor header, and the MSC711x connection to the MPC8272. **Table 5** lists the HDI16 bus signals.

Table 5. HDI16 Bus Signals

Signal	Description
HD[15–0]	Host data bus in 16-bit mode.
HD[7–0]	Host data bus in 8-bit mode.
HA[2–0]	Host address line.
HRW/HRD	Read/write select or read strobe.
$\overline{\text{HCS1}}$	Host chip-select 1.
$\overline{\text{HCS2}}$	Host chip-select 2.
HREQ/HTRQ	Host request or host transmit request.
HACK/HRRQ	Host acknowledge or host receive request.
HDDS	Dual data strobe control input.
HDSP	Data strobe polarity control input.
H8BIT	8-bit mode control input.

Figure 5 shows the signals that connect the MSC7115 device to the HDI header.

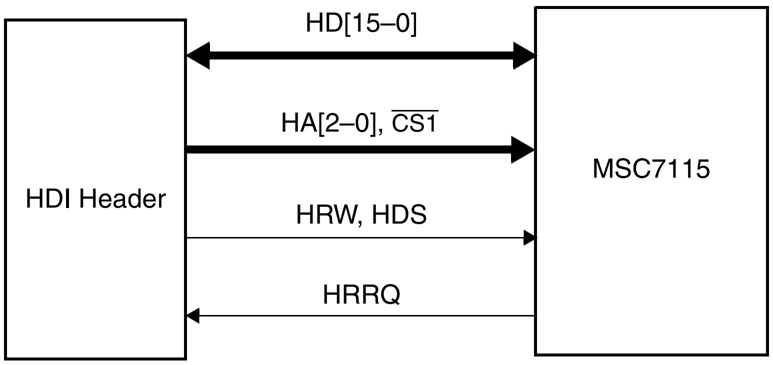


Figure 5. Signals for HDI16 Bus to Host Processor Header

Table 6. HDI Bus Connection to the Host Processor Header

Pin Number	Signal Name	Attribute	Description
1	GND	P	Digital GND. Main GND plane.
2			
3	HD0	I/O	Host interface bidirectional tri-stated data bus port: HD[15–0]
4	HD1		
5	HD2		
6	HD3		
7	HD4		
8	HD5		

Table 6. HDI Bus Connection to the Host Processor Header

Pin Number	Signal Name	Attribute	Description
9	HD6	I/O	Host interface bidirectional tri-stated data bus port: HD[15–0]
10	HD7		
11	HD8		
12	HD9		
13	HD10		
14	HD11		
15	HD12		
16	HD13		
17	HD14		
18	HD15		
19	GND	P	Digital GND. Main GND plane.
20			
21	HA0	I	Host Interface Address Line, HA[3–0]. Bit 0 corresponds to the LSB of the bus.
22	HA1		
23	HA2		
24	HA3		
25	$\overline{\text{HCS1}}$	I	Host Chip Select 1. The HDI $\overline{\text{CS}}$ is determined by the logical OR between $\overline{\text{HCS1}}$ and $\overline{\text{HCS2}}$.
26	$\overline{\text{HCS2}}$	I	Host Chip Select 2. The HDI CS is determined by the logical OR between $\overline{\text{HCS1}}$ and $\overline{\text{HCS2}}$.
27	HACK	I/O	Host DMA Acknowledge or Host Receive Request output. When the HDI16 is programmed to interface to a single host request, this pin is the host acknowledge Schmitt trigger input in host DMA mode (HACK). The polarity of the host DMA acknowledge is programmable.
28	HREQ	Q, T, S	Host Request or Transmit Host Request output. When the HDI16 is programmed to interface to a single host request, this pin is the host request output (HREQ). This pin can be used for host DMA requests in host DMA mode.
29	HRW	I	Host Read/Write or Host Read input.
30	HDS	I	Host Read/Write or Host Read input.
31	$\overline{\text{HRESET}}$	I/O, P, U	MSC7115 hard reset.
32	$\overline{\text{PORESET}}$	I/O, P, U	Power-on reset.
33	3V3	P	+3.3 V power out. These lines are connected to the main 3.3 V plane of the MSC711xEVM.
34	N.C.	—	Not connected.
35	GND	P	Digital GND. Main GND plane.
36			

As **Figure 6** shows, most of the hardware options on the MPC711xADS are controlled or monitored by the Board Control and Status Register (BCSR), which is a 32-bit wide read/write register file. The BCSR is accessed via the MPC8272 memory controller and in fact includes 8 registers: BCSR[0–7]. Since the minimum block size for a $\overline{\text{CS}}$ region is 32 KB and only the A[27–29] lines are decoded by the BCSR for register selection, BCSR[0–7] are duplicated inside that region.

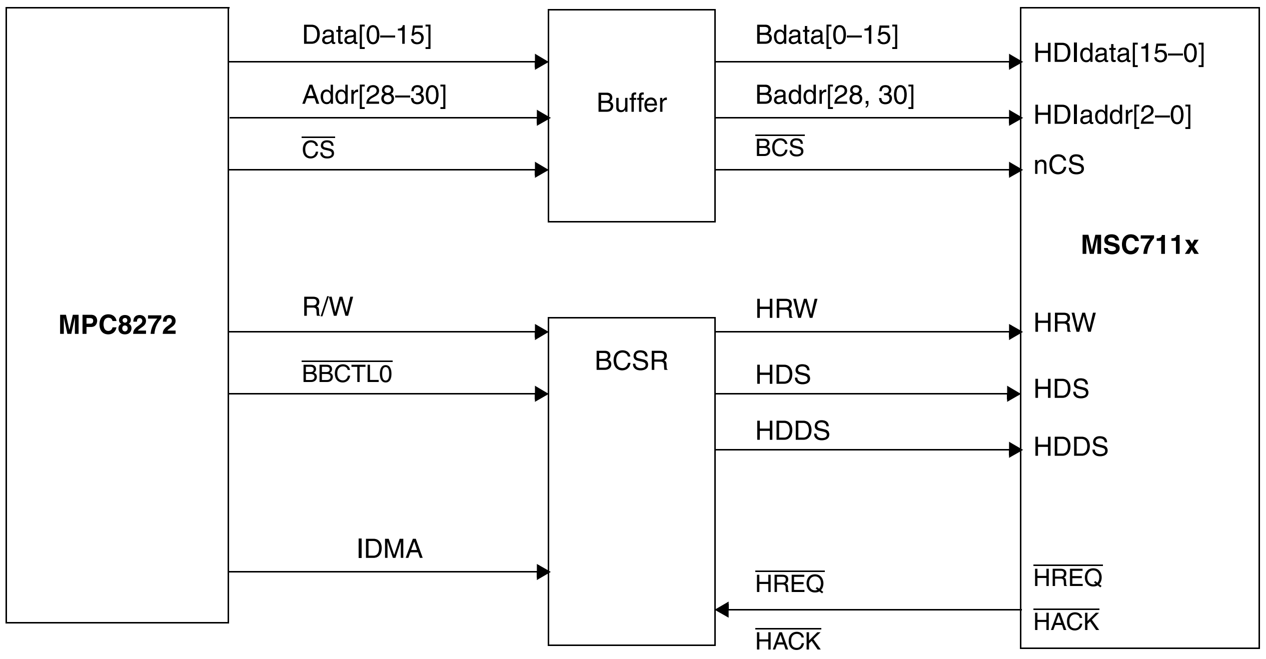


Figure 6. MSC711x Connection to the MPC8272

Altera CPLD buffer programming affects the values and functionality of the signals as described in **Table 7**.

Table 7. Effect of Altera CPLD Buffer on Signal Functionality

Signals	Description
Data Strobes and Chip Selects	
HRW/HRD	Single data strobe. The Altera CPLD asserts this signal according to the states of HDSP and HDDS.
HDS/HWR	
HDSP	Determined by dip switch SW5–7. The current setting on the board is 0 so that the polarity of the signals is active low.
HDDS	The value of this signal is determined by the host via a BCSR bit in the Altera. The default value is 1, meaning double strobe.
HCS1	Although going through the Altera, these signals are transparent to $\overline{\text{CS6}}$ and $\overline{\text{CS7}}$ from the host, respectively.
HCS2	
Port Signals	
HREQ/HTRQ	HREQ goes through the Altera and connects to either DREQ2 or DREQ3 in the host. The default connection is DREQ2.
HACK/HRRQ	HACK goes through the Altera and connects either to DACK2 or DACK3 in the host. The default connection is DACK2.
Miscellaneous	
HPE	This is not an external pin. It is internally asserted to a value of 1 and cannot be changed.
H8BIT	The value of this signal is determined by dip switch SW5–8. It is cleared to 0 to designate a 16-bit HDI bus. The MSC711xADS does not support an 8-bit HDI bus.
GPCM Connection	

Table 7. Effect of Altera CPLD Buffer on Signal Functionality (Continued)

Signals	Description
BCTL0 →HRD	Double strobe.
BE0 →HRW	
BCTL0 →HRW	Single strobe
HDICSx →HDS	
UPM Connection for Both Single and Double Strobe	
GPL0 →HRW/HRD	
GPL1 →HDS/HRW	
CPLD	Handles the polarity of signals according to HDPOL, which is set by dip switch.

4 MSC711xADS Software Settings

This section describes the MSC711xADS switch and jumper settings for the MPC8272 connection to the MSC7115 DSP. The switch settings are as follows:

- *SW1*. MSC711x Abort ($\overline{\text{ABORT}}$). Normally used to abort program execution by issuing a level 0 interrupt to the MSC711x. The ABORT switch signal is debounced.
- *SW2*. Slave Hard Reset ($\overline{\text{HRESET}}$). When SW2 is pressed, a hard reset is generated to the MSC711x but does not affect the host. The hard reset signal is debounced.
- *SW3*. Power-on Reset ($\overline{\text{PORESET}}$). Performs a power-on reset to the MPC8272 and to the MSC7115. All configuration and all data residing in volatile memories are lost.
- *SW4*. MSC711x Power-Up Configuration. **Figure 7** shows the switch settings for SW4.

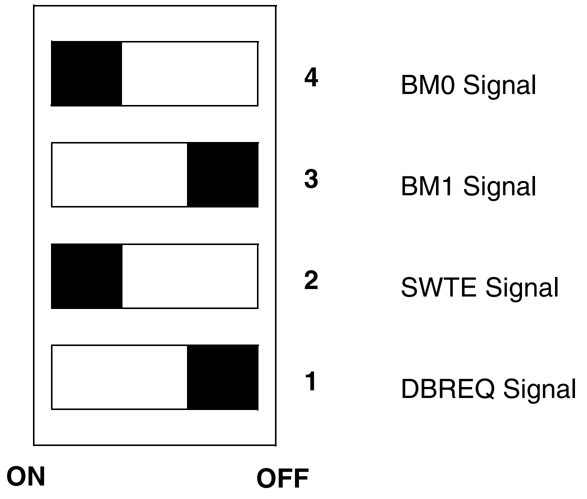


Figure 7. SW4 Settings for the MPC8272 Connection to the MSC7115 Device

- *SW5*. MSC711x Event Pins Configuration. **Figure 8** shows the SW5 switch settings.

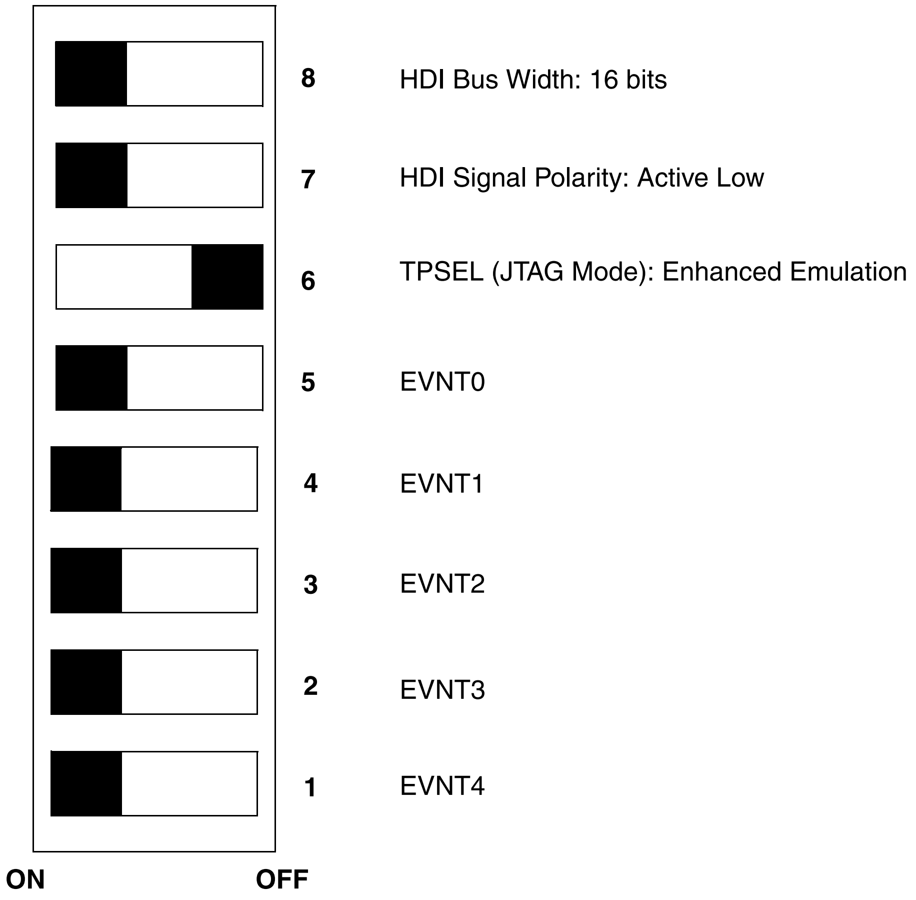

8
HDI Bus Width: 16 bits
7
HDI Signal Polarity: Active Low
6
TPSEL (JTAG Mode): Enhanced Emulation
5
EVNT0
4
EVNT1
3
EVNT2
2
EVNT3
1
EVNT4
ON
OFF

Figure 8. SW5 Settings for the MPC8272 Connection to the MSC7115 Device

- *SW6*. JTAG Chain Options. **Figure 9** shows the SW6 switch settings.

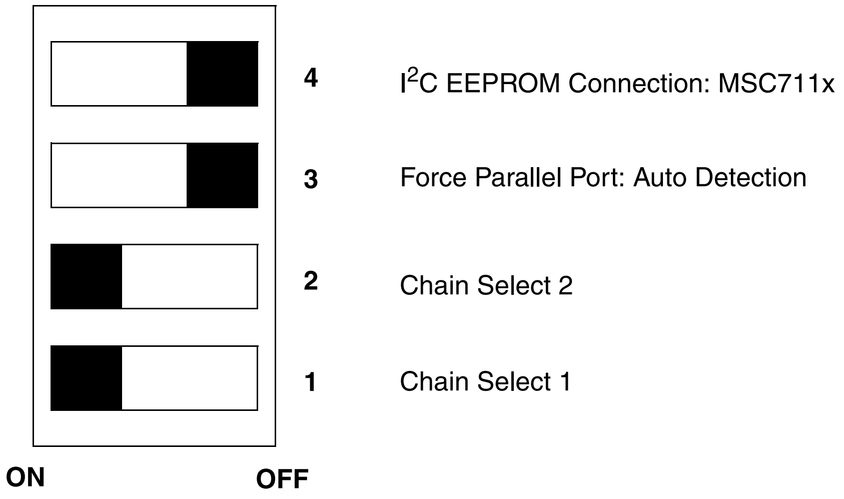

4
I2C EEPROM Connection: MSC711x
3
Force Parallel Port: Auto Detection
2
Chain Select 2
1
Chain Select 1
ON
OFF

Figure 9. SW6 Settings for the MPC8272 Connection to the MSC7115 Device

- *SW7*. I^{2}C EEPROM Address and Protection Mode. **Figure 10** shows the SW7 switch settings.

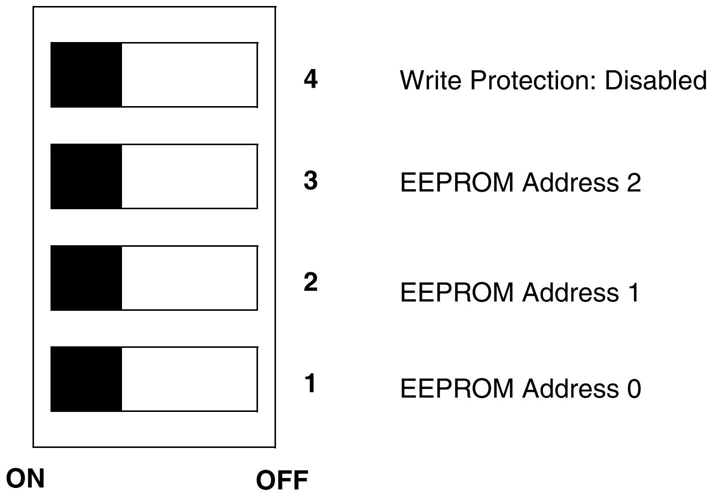
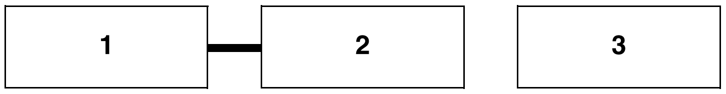


Figure 10. SW7 Settings for the MPC8272 Connection to the MSC7115 Device

- *SW8*. MPC8272 Clock Mode Settings.
- *SW9*. Toggle switch for power.
- *SW10*. Host Soft Reset. When SW10 is pressed, it generates a soft reset to the MPC8272. The soft reset switch signal is debounced.
- *SW11*. Host Hard Reset. When SW11 is pressed, it generates a hard reset to the MPC8272, which in turn generates a hard reset to the MSC711x, according to the JP6 configuration. The hard reset switch signal is debounced.
- *SW12*. Host Abort (NMI). Used to abort program execution by issuing a level 0 interrupt to the MPC8272. The ABORT switch signal is debounced.

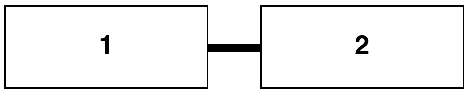
Jumper settings for the MPC8272 connection to the MSC7115 devices are listed as follows:

- *JP1*. MSC711x Ethernet PHY MII/RMII mode.



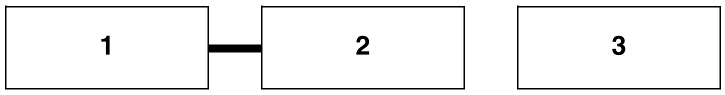
MII Mode (Factory Default)

- *JP2*. H110 Back Plane Reset.



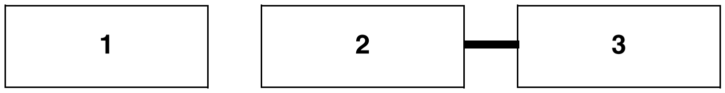
Hard Reset Connected to Back Plane (Factory Default)

- *JP3*. TDM Master Selection.



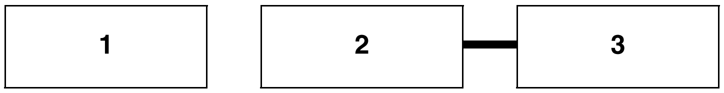
TSI is the TDM Master (Factory Default).

- *JP4*. MSC711x Clock In Source.



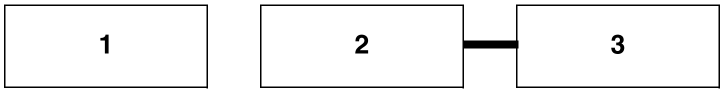
Clock In Source is the On-Board Oscillator.

- *JP5*. MPC8272 Hard Reset Configuration Word Source.



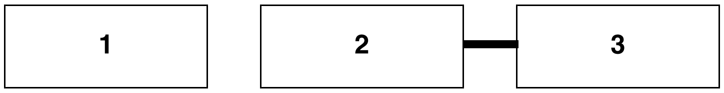
The HRCW is Sourced from the BCSR (Factory Default).

- *JP6*. MPC8272 Hard Reset Connection to MSC711x Hard Reset.



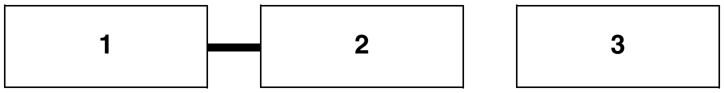
The MPC8272 Hard Reset is Disconnected from the MSC711x Hard Reset (Factory Default).

- *JP7*. PCI Expansion Enable/Disable.



The MPC8272 60x Bus (and HDI) is Visible on the J1 and J2 Connectors (Factory Default).

- *JP8*. Host (MPC8272) Enable/Disable.



The MPC8272 is Enabled and Connected to the MSC711x DSP (Factory Default).

5 Power-On Reset Configuration

At the end of power-on reset, the MPC8272 samples MODCK[1–3] to configure its various clock modes (core, CPM, bus, PCI). The MODCK[1–3] combination options are selected by means of dedicated dip switches. After power-on reset, the hard reset sequence starts, and many other options are configured, such as additional options for the clock generator. Although the MODCK[1–3] bits are sampled whenever there is a hard reset sequence, they are influential only once, after power-on reset. If a hard reset sequence is entered later, MODCK[1–3], although sampled, are don't care.The PCI_MODCK signal, which is sampled concurrently with the MODCK[1–3] pins, determines the PCI bus clock frequency. When it set high, it divides the PCI bus frequency by two. When it is reset low, the PCI bus frequency is as determined by the MODCK[1–3] and MODCKH[0–3] signals.

At the rising edge of the power-on reset signal, the MSC711x samples four signal pins to determine their configuration: BM0, BM1, SWTE, and HDPOL. These signal pins are sampled only once after power-on reset.

6 Boot Load Program

The host-side programming procedure is as follows:

1. Initialize the host-side controllers and registers to configure the port and the UPM memory controller that directly connects to the MSC7115 slave device. Set the UPM timings to fast or slow mode (to accommodate the PLL bypass or PLL enabled mode of the MSC7115 DSP out of reset).
2. Initialize the pointers to the MSC7115 HDI16 Transfer (TX) registers to enable these pointers for data transfers to the TX registers of the HDI16 port.
3. Initialize the pointers to the ICR and ISR of the slave device to enable these pointers for ICR initialization and reading the status of the MSC7115 Host Receive (HORX) data register.
4. Poll the Transmit Data Empty (TXDE) bit in the ISR to ensure that the TX registers are empty and ready for writes from the host processor.
5. Initialize the ICR[INIT] bit so the boot loader program can proceed to transfer the source program. In addition to the INIT bit, the ICR[HF3] bit can be set for an optional checksum. If HF3 is set, the boot loader performs a checksum to ensure that data is transferred properly. If HF3 is clear, the boot loader program ignores the checksum routine.
6. Transmit data to the MSC7115 TX registers. The host MPC8272 transmits the source program to the slave MSC7115 device. The MSC7115 HCR is configured by default to have ICR/HCR priority for DMA/Last Address Mode (HICR) bit cleared, which signifies that the last address mode is defined in the HCR. In addition, the Host DMA/Last Address Mode Control (HDM[0–2]) bits are cleared by default in the HCR to indicate that the trigger address in the TX registers is 0x7. The boot loader program assumes that the 64 bits are written to the TX registers before the address is triggered for transfer. That is, 16 bits are written to TX3 (address 0x7), TX2 (address 0x6), TX1 (address 0x5), and TX0 (address 0x4), which then triggers the transfer to the HORX register.
7. Complete the data transfer by checking HCR[HF4] and HCR[HF7] of the slave MSC7115 device to ensure that data loaded and transmitted without errors. HF4 is set if the load completed properly. The checksum is an optional feature in the bootload procedure. If a checksum is required, then HF7 determines whether it is necessary to re-transmit the data.
8. After verifying that the boot code has downloaded correctly, change the settings of the UPM memory controller to fast mode (if switching from PLL disabled mode to PLL enabled mode).
9. Transmit data from the host side to MSC7115 by writing to the TX[3–0] registers.
10. If the bootload occurred correctly, the application downloaded during the MSC711x HDI16 bootload process increments the data transmitted by the host in step 9 and sends it back to the host.
11. Read the RX[3–0] registers on the host MPC8272 side and compare the result with the reference vectors. If the results match, it indicates that the boot load has occurred correctly and the slow and fast UPM modes have been successfully verified.

Refer to the attached files for the bootload code. These C files have embedded comments to help in understanding 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.: AN2945
Rev. 0
3/2005