Freescale Semiconductor
Application Note

Document Number: AN3424
Rev. 0, 05/2007

MSC8144 Device Reset Configuration For the MSC8144ADS Board

by *Andrew Temple*
NCSG DSP Applications
Freescale Semiconductor, Inc.
Austin, TX

This document is a quick configuration reference to help you take advantage of the flexible configuration of the StarCore™-based MSC8144 DSP.

The MSC8144 reset configuration words contain 64 bits to configure 64 clock modes, additional clock sources, peripheral enablement, peripheral and GPIO pin multiplexing, peripheral modes, and boot settings. There are a several ways to configure the device to get the same functionality. For example, there are four ways to boot the device from the I^2C ROM on the MSC8144ADS board—all of which are described in this application note. Examples illustrate how to configure device reset as well as the SmartDSP OS on the MSC8144 application development system (MSC8144ADS). Related documentation for the MSC8144 device and ADS configuration is referenced in Appendix B.

Contents

freescale™ semiconductor

1 MSC8144ADS Reset Basics

MSC8144ADS reset terms used throughout this document are as follows:

- *Reset configuration word (RCW)*. The 64 bits used by the MSC8144 to configure clocks, peripherals, booting, and pinning out of reset. For details on the RCW, consult the reset chapter of the MSC8144 reference manual.
- *Reset configuration word registers high and low (RCWHR and RCWLR)*. Two 32-bit registers in the MSC8144 device that contain the reset configuration word.
- *Reset configuration word source select pins (RCW_SRC)*. Three pins used by an internal multiplexer to select the reset configuration source. The device reads from these pins to fill the contents of the RCW registers. These pins are located on SW4 of the MSC8144ADS board.
- *Reset configuration (RC)*. Pins sampled on reset, which are one of the three configuration sources that can be chosen by RCW_SRC. These pins are located on SW1, SW3, and SW4 of the MSC8144ADS board.
- *I^2C ROM*. EEPROM on the MSC8144 I^2C bus for loading the device configuration mode and/or booting executable code.
- *Boot port*. A mode that selects the device boot source if the device is set to run free of the debugger (that is, not in Debug mode) based on bits set in the RCW registers. For quick reference, the table from the MSC8144 reference manual showing the different boot ports is included in Appendix A of this application note.
- *Pin multiplexing*. A mode that determines whether certain GPIO pins are assigned to a peripheral. This mode is required if Ethernet, PCI, or UTOPIA is used. This mode is also selected in the RCW. For quick reference, the table from the MSC8144 reference manual showing the pin multiplexing modes is included in Appendix A.
- *Debug mode*. Selected by a pin that, on reset, determines whether the device stops and waits for a debugger to connect or begins executing code at the location to which the boot port is pointing. (*Not part of the reset configuration word*)

2 MSC8144 Configuration Sources

Using the RCW_SRC multiplexing pins (DIP switch SW4 on the MSC8144ADS), there are three main ways a user can load the contents of the RCW registers (see Figure 1):

- I^2C ROM
- RC pins
- Predefined mode

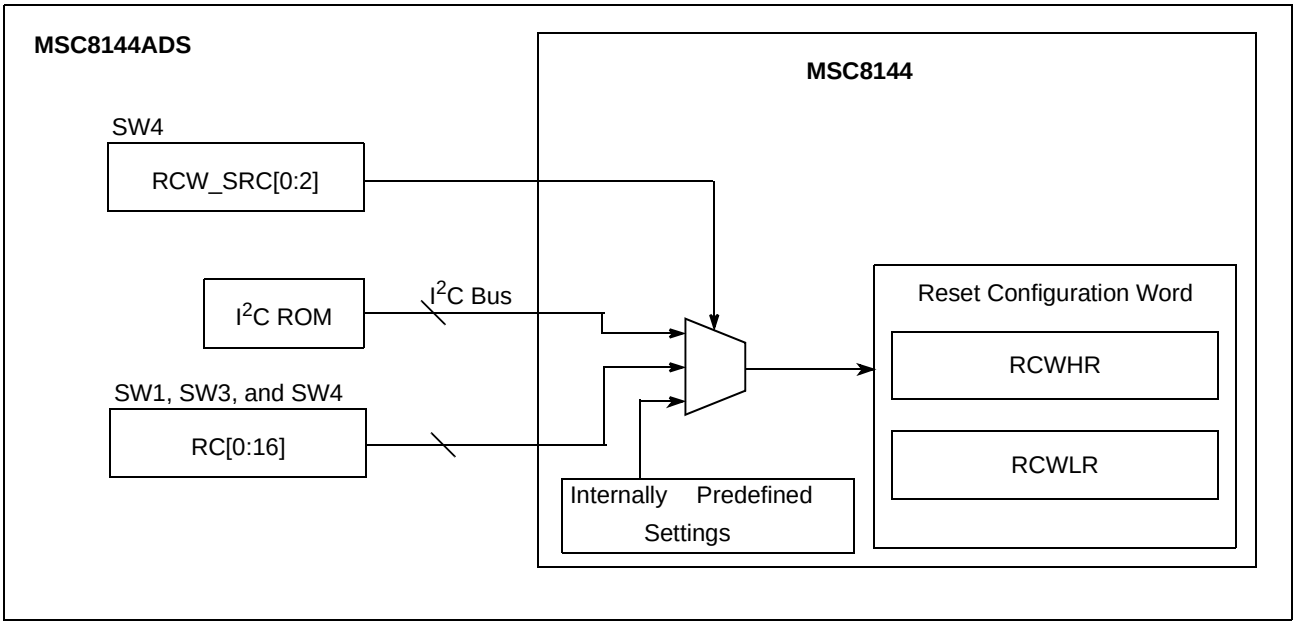

Figure 1. Configuration Source Multiplexing

2.1 I²C ROM: (RCW_SRC[0:2] = 001 or 010)

The only way to gain full access to all bits in RCWLR and RCWHR is to configure the RCW from I²C ROM. This configuration gives you access to all 64 clock modes, additional clock options (such as running M3 at the MSC8144 full class speed), and all booting and GPIO multiplexing settings.

2.1.1 Programming the I²C ROM

There are four ways you can boot the MSC8144 device to run from I²C ROM. Remember, booting from I²C ROM is different from loading the configuration from I²C ROM. The MSC8144 can be configured to load the reset configuration and/or boot from the I²C ROM. Programming the I²C ROM is currently not supported as a simple GUI feature in CodeWarrior™ v3.0 Beta, so you must refer to one of the C projects provided in the `example projects` folder of the CodeWarrior install directory at:

```
<CodeWarrior Install>\(CodeWarrior_Examples)\StarCore_Examples\I2CBoot
```

For details on burning, refer to the `HOWTO_Burn_I2C_Image.txt`.

2.1.2 Altering the Configuration Word Loaded into I²C ROM

When the I²C ROM is programmed, both the boot code and configuration word are burned into I²C ROM at the same time, so both are read from `data.txt`. You can change the configuration word by altering `data.txt` and rebuilding the `burner.mcp` project before debugging and executing `burner.mcp`. The default RCW settings for `data.txt` are as follows:

```
/* Master Reset Configuration Word Low          */
0xFF, 0xFF, 0xFF, 0x00, 0x1F, 0x18, 0x00,
/* Master Reset Configuration Word High         */
0xFF, 0xFF, 0xFF, 0x04, 0x6C, 0x18, 0x01,
```

There are three bytes of 0xFF padding before the actual contents of the RCW, which is organized from high-order bits to low-order bits. If you refer to the definition of the RCWHR bits in the MSC8144 reference manual, you can see in the preceding lines from `data.txt` that if RCW_SRC loads the configuration from I^2C, the RCW points the boot port back to the I^2C bus and boots the MSC8144 device from the I^2C ROM. If you want to boot from the default PCI port, you must alter `data.txt` as follows:

```
/* Master Reset Configuration Word Low          */
 0xFF, 0xFF, 0xFF, 0x00, 0x1F, 0x18, 0x00,
/* Master Reset Configuration Word High         */
 0xFF, 0xFF, 0xFF, 0x01, 0x6C, 0x18, 0x01,
```

2.2 RC Pins: (RCW_SRC[0:2] = 011)

For quick switch capability, a subset of the RCW bits can be configured through the RC pins. The settings you can change in this mode are listed in Table 1 and Table 2. In this mode, the following options are hard coded:

- System PLL input to PCI block (PCI block runs at 200 MHz for available clock modes)
- System PLL input to DDR block (DDR block runs at 400 MHz for available clock modes)
- System PLL not input to M3 memory block (M3 block is clocked by PLL2)
- RapidIO®/SGMII reference clock is 100 MHz, serializer/deserializer (SerDes) is 1.25 GHz
- RapidIO V_{DD} is set to 1.0 V
- PCI and RapidIO host access enabled
- RapidIO prescale timer enable: OCeaN clock is 200 MHz
- SerDes digital filter bandwidth is 200 ppm
- No loopback mode on SerDes
- Watchdog timer is disabled

Table 1. Settings Configurable from DIP Switches (RCW_SRC[0:2]=011): RCWLR

RCWLR Bits	RC Pins (DIP Switches)	Usage
19	RC[16] || RC[3]	==0: RapidIO disabled on SerDes ==1: RapidIO enabled on SerDes
18	RC[16]	==0: RapidIO 4x protocol. ==1: RapidIO 1x protocol
17	RC[16]	==0: Disable SGMII1 on SerDes ==1: Enable SGMII1 on SerDes
16	RC[16] && RC[3]	==0: Disable SGMII2 on SerDes ==1: Enable SGMII2 on SerDes
2–0	RC[2:0]	MODCK[2:0], **Note:** MODCK[5:3] = 000

Note: Bits are displayed using little-endian format although the device is big-endian. In this mode, the PLL source for PCI and DDR are the system PLL, but M3 remains sourced from PLL2

Table 2. Settings Configurable from DIP Switches SRC[0:2]=011): RCWHR

RCWHR Bits	RC Pins (DIP Switches)	Usage
27–26	RC[15:14]	Boot port select [4:3]
24–23	RC[13:12]	Boot port select [1:0] **Note:** Boot Port Select[5] and [2] = 0 **Note:** For mode definitions, refer to Table 10
11–10	RC[11:10]	Pin Mux[1:0] **Note:** See Table 11 in Appendix A
9–4	RC[9-4]	Device ID

Note: Bits are displayed using little-endian format although the device is big-endian.

2.3 Predefined Mode: (RCW_SRC[0:2] = 100, 101, 110, 111)

In the predefined modes, RCW is loaded with a fully predefined setting based on only the RCW_SRC bits. Three predefined modes affect the following:

- Boot port (see Table 10)
- Pin multiplexing (see Table 11)
- Clock mode

Using these pins is the easiest way to connect a debugger quickly to the device without learning about the full configuration. In the predefined modes, the following options are hardcoded:

- System PLL input to PCI block (PCI block runs at 200 MHz for the available clock modes)
- System PLL input to DDR block (DDR block runs at 400 MHz for the available clock modes)
- System PLL not input to M3 memory block (M3 block is clocked by PLL2)
- RapidIO prescale timer enable: OCeaN clock is 200 MHz
- RapidIO/SGMII reference clock is 100 MHz, SerDes is 1.25 GHz
- RapidIO interface is enabled
- RapidIO uses the 1x protocol
- RapidIO host access enabled
- RapidIO V_{DD} is set to 1.0 V
- SGMII1 is enabled on SerDes
- SGMII2 is enabled on SerDes
- PCI host access enabled
- SerDes digital filter BW is 200 ppm
- No loopback mode on SerDes
- Watchdog timer is disabled

For quick reference, the optional settings are provided in Table 8.

Table 3. Predefined Settings

RCW_SRC	MODCK[5:0] (Clock Mode)	PIN MUX[13:10]	BOOT PORT[28:23]
100	001011 (1GHz Core, 333 MHz M3)	0010 (PCI Enabled)	000010 (Boot from PCI)
101	001011 (1GHz Core, 333 MHz M3)	0001 (Ethernet and UTOPIA Enabled)	001000 (Boot from I2C)
110	110101 REQUIRES 33.3 MHz CLKIN (800MHz Core, 333 MHz M3)	0010 (PCI Enabled)	000010 (Boot from PCI)
111	110101 REQUIRES 33.3 MHz CLKIN (800 MHz Core, 333 MHz M3)	0001 (Ethernet and UTOPIA Enabled)	001000 (Boot from I2C)

Note: Bits here are displayed in little-endian format although the device is big-endian.

3 Reading the Reset Configuration Word

Verify that the MSC8144 device is properly configured, as follows:

1. connect to the MSC8144ADS using CodeWarrior
2. Debug a project.
3. Select the View → Registers menu.
   The RCW registers (high and low) are located under **RESET**.

4 Power-On Reset and Boot Flow

The flow chart in Figure 2 shows how the reset configuration and device boot fit into the main device power-up flow. This flow chart is simplified to show only information necessary to the software programmer testing application code and peripherals on the MSC8144ADS.

5 Example Configurations

Because some peripherals—such as the PCI, Ethernet, and UTOPIA—share pins with GPIO, specific configurations are needed to connect to them. This section presents example configurations that are useful to a programmer running applications out of ROM on the MSC8144ADS or working with the SmartDSP OS demonstrations.

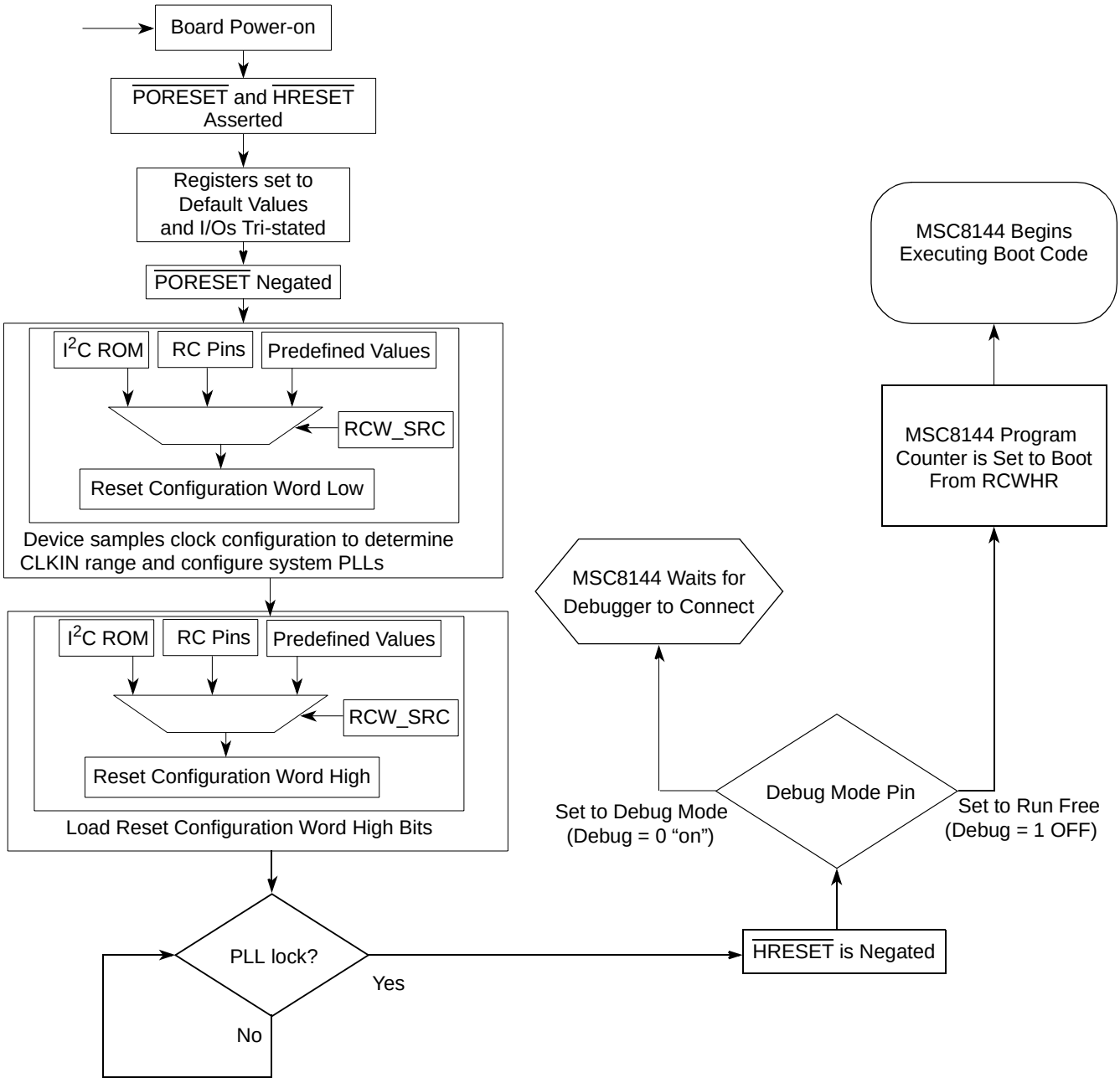

Figure 2. MSC8144 Power-on Reset and Boot

5.1 Configure MSC8144ADS to Boot using the LEDs Example

If DIP switches malfunction while you are showing a demonstration on a board, the MSC8144ADS has many ways around this kind of configuration problem that do not require board reprogramming. Consider the case of bootloading from the I^2C ROM on the MSC8144ADS. With the standard 66.667 MHz crystal oscillator on the MSC8144ADS board, there are four different ways to boot out of I^2C ROM. The differences lie in how the MSC8144 DSP reads the reset configuration word into the RCWHR and RCWLR.

5.1.1 Load the RCW from I²C ROM

Because the bootloader and LED example must be loaded into the I²C ROM, we start with loading the configuration from I²C ROM as well:

1. As noted in Section 2.1.1, "Programming the I²C ROM, refer to the CodeWarrior examples for the steps in programming the I²C ROM, but stop before Step 5.
2. Before loading `burner.mcp`, you must alter the RCW being programmed to I²C ROM, so open `data.txt`.
3. Alter the `data.txt` RCWHR so that the boot port points to I²C ROM. Refer to Table 10 to see which boot port mode is required. The configuration should look like this:

```
/* Master Reset Configuration Word Low          */
 0xFF, 0xFF, 0xFF, 0x00, 0x1F, 0x18, 0x00,
/* Master Reset Configuration Word High         */
 0xFF, 0xFF, 0xFF, 0x04, 0x6C, 0x18, 0x01,
```

4. Save `data.txt`.
5. Re-build `burner.mcp` in CodeWarrior

   `data.txt` is an include file to `burner.mcp`.
6. Verify that the MSC8144ADS DIP switches are in a known configuration. Table 4 shows an example default configuration:

Table 4. Default DIP Switch Settings Configured from I²C ROM

	1	2	3	4	5	6	7	8
SW1	x	x	x	x	x	x	x	x
SW2	0	1	1	0	1	1	1	1
SW3	x	x	x	x	0	1	1	1
SW4	0	1	0	x	x	x	x	x

Note: Logic 1 = OFF on the MSC8144ADS

7. Power on the MSC8144ADS board and run `burner.mcp` until it completes.
8. Power off the board.
9. Now that the I²C ROM is set to load RCWHR with boot port = I2C ROM as shown in the flowchart in Section 4, "Power-On Reset and Boot Flow, you must use the DIP switches to ensure the following if the device is to boot from I²C ROM:
   - RCW_SRC loads the configuration from I²C ROM (see Section 2.1, "I²C ROM: (RCW_SRC[0:2] = 001 or 010))
   - The MSC8144 runs the boot code (not in debug mode), as highlighted in bold on SW2 in Table 5.

Table 5. DIP Switch Setting: Configure and Boot from I²C ROM

	1	2	3	4	5	6	7	8
SW1	x	x	x	x	x	x	x	x
SW2	0	1	1	**1**	1	1	1	1
SW3	x	x	x	x	0	1	1	1
SW4	**0**	**1**	**0**	x	x	x	x	x

Note: Logic 1 = OFF on the MSC8144ADS.

10. Power on the board.
11. The LEDs should begin flashing.

5.1.2 Load the RCW from I²C ROM (2)

There are two RCW_SRC settings to load the RCW from I²C ROM, RCW_SRC[0:2] = 001 or 010. The difference between them is the minimum and maximum CLKIN frequency. If the crystal oscillator on the ADS (CLKIN) is between 25 and 100 MHz, RCW_SRC = 001 mode is expected to work. RCW_SRC is expected to work with a CLKIN value between 44 and 133 MHz. Because the MSC8144ADS uses a 66 MHz crystal, you can use both load settings. Also, because you have already programmed the I²C ROM, the only remaining step is to change the RCW_SRC DIP switches to test this load option, as shown in Table 6. Re-power the board and verify that the LED program still boots and flashes LEDs.

Table 6. DIP Switch Setting: Configure and Boot from I²C ROM (2)

	1	2	3	4	5	6	7	8
SW1	x	x	x	x	x	x	x	x
SW2	0	1	1	**1**	1	1	1	1
SW3	x	x	x	x	0	1	1	1
SW4	**0**	**0**	**1**	x	x	x	x	x

Note: Logical 1 = OFF on the MSC8144ADS

5.1.3 Set the RCW using the RC Pins

Because the I²C ROM already has a bootloader and example program in it, you need only verify the following to ensure that the MSC8144 can load and run from I²C ROM:

- RCW_SRC[0:2] = 011 points the MSC8144 to load the RCW from the RC pins.
- RC[14] = 1, RC[15,13,12] = 0 so that the boot port points to I²C ROM
- Debug mode is OFF (logic 1)

Referring to Table 1 and Table 2, configure the DIP switches as shown in Table 7.

Table 7. DIP Switch Settings Configured from the RC Pins, With Boot from the I²C ROM

	1	2	3	4	5	6	7	8
SW1	0	0	0	0	0	1	1	**0**
SW2	0	1	1	**1**	1	1	1	1
SW3	**0**	**1**	**0**	1	0	1	1	1
SW4	**0**	**1**	**1**	0	0	0	1	0

Note: Logic 1 = OFF on the MSC8144ADS

After the DIP switch settings match those in Table 7, cycle power on the MSC8144ADS board and boot the LED demonstration again through this new avenue.

5.1.4 Set the RCW using Predefined Settings

As Table 3 shows, using predefined mode RCW_SRC = 101 sets the RCW to point the boot port to I²C ROM. Therefore, with the I²C ROM already loaded with code, you need only complete the following steps:

1. Verify that RCW_SRC is configured to a value of 0b101.
2. Verify that the device is set to run free (Debug mode = OFF, which is a logic 1).

Table 8. DIP Switch Settings Configured from Predefined Settings, with Boot from I²C ROM

	1	2	3	4	5	6	7	8
SW1	x	x	x	x	x	x	x	x
SW2	0	1	1	**1**	1	1	1	1
SW3	x	x	x	x	0	1	1	1
SW4	**1**	**0**	**1**	x	x	x	x	x

Note: Logic 1 = OFF on the MSC8144ADS

There is a fifth setting for RCW_SRC to make the MSC8144 boot from I²C ROM, but it requires a different board oscillator, so it is not discussed here.

5.2 Configure the MSC8144ADS for SmartDSP OS Demonstrations

The SmartDSP OS and example MSC8144ADS demonstrations are optionally installed along with CodeWarrior. To select this option, click **YES** to `Install SmartDSP OS` during CodeWarrior installation. The demonstrations of Ethernet (UEC, NET, and NET_PATTERN) require additional hardware configuration for pin multiplexing. That is, in the SmartDSP OS documentation, all of these demonstrations request the user to alter the RCW to pin multiplexing mode 6.

As described in Section 5.1.1, “Load the RCW from I²C ROM, the I²C ROM must be re-programmed because pin multiplexing mode 6 is not available to the other configurations. Repeat the procedure in Section 5.1.1, but this time ensure that the configuration portion of `data.txt` matches the following:

```
/* Master Reset Configuration Word Low          */
 0xFF, 0xFF, 0xFF, 0x00, 0x1F, 0x18, 0x00,
```

```
/* Master Reset Configuration Word High          */
 0xFF, 0xFF, 0xFF, 0x04, 0x6C, 0x18, 0x01,
```

After the I^{2}C ROM is programmed, the RCW_SRC DIP switches should be set to load the RCW from the I^{2}C ROM, and the board should be left in Debug mode. Therefore, the DIP switches are configured as shown in Table 9.

Table 9. DIP Switch Settings: SmartDSP OS Demonstrations for Ethernet

	1	2	3	4	5	6	7	8
SW1	x	x	x	x	x	x	x	x
SW2	0	1	1	**0**	1	1	1	1
SW3	x	x	x	x	0	1	1	1
SW4	**0**	**1**	**0**	x	x	x	x	x

Note: Logic 1 = OFF on the MSC8144ADS.

At this point, the MSC8144 is configured to run with the SmartDSP OS demonstrations.

6 Common MSC8144 Misconceptions

Aspects of the MSC8144 that may cause confusion when the MSC8144ADS is configured are as follows:

- The MSC8144 DSP and most Freescale devices are big-endian devices, meaning that the bits are organized from low-order to high. For example, a 32-bit register is organized from bit 0 to 31 rather than from bit 31 to 0. For purposes of clarity, this document states the bit order each time a variable is listed. For example: “RCW_SRC[0:2] lists bits from low order to high.”
- On the MSC8144ADS, a DIP switch turned to OFF = logic 1.
- Booting and configuring from I^{2}C ROM should not be confused with each other. They are two different procedures. *Booting* from I^{2}C ROM can be done without *configuring* from I^{2}C ROM and *vice versa*, as illustrated in the examples presented in this document.

Appendix A Boot Port Modes and Pin Multiplexing

In this appendix, Table 10 presents the boot port modes and Table 11 presents the pin multiplexing configurations.

Table 10. Boot Port Modes

Boot Port	Description	BPRT[5:0]
PCI	No DDR	000000
	Single 32MB DDR	000001
	256 Mbyte DDR (Default)	000010
	Single 64 Mbyte DDR	000011
	Single 128 Mbyte DDR	000100
	512 Mbyte DDR	000101

Table 10. Boot Port Modes

Boot Port	Description	BPRT[5:0]
I^2C	Boot over I^2C bus	001000
SRIO	No I^2C	001001
	with I^2C	001010
Ethernet1 without I^2C	SMII	001111
	RMII	010000
	RGMII	010001
	MII	010010
	SGMII	010011
Ethernet1 and I^2C	SMII	010100
	RMII	010101
	RGMII	010110
	MII	010111
	SGMII	011000

Table 11. GPIO Pin Multiplexing Configurations

GPIO	I/O Mode (PIN_MUX field lowest four bits)							
	Mode 0 (0000)	Mode 1 (0001)	Mode 2 (0010)	Mode 3 (0011)	Mode 4 (0100)	Mode 5 (0101)	Mode 6 (0110)	Mode 7 (0111)
0	GPIO							
1	GPIO							
2	GPIO							
3	GPIO							
4	GPIO			PCI		GPIO		
5	GPIO			PCI		GPIO		
6	GPIO			PCI		GPIO		
7	GPIO			PCI		GPIO		
8	GPIO			PCI		GPIO		
9	GPIO			PCI		GPIO		
10	GPIO			PCI		GPIO		
11	GPIO			PCI		GPIO		
12	GPIO			PCI		GPIO		
13	GPIO							
14	GPIO							

Table 11. GPIO Pin Multiplexing Configurations (continued) (continued)

GPIO	I/O Mode (PIN_MUX field lowest four bits)							
	Mode 0 (0000)	**Mode 1 (0001)**	**Mode 2 (0010)**	**Mode 3 (0011)**	**Mode 4 (0100)**	**Mode 5 (0101)**	**Mode 6 (0110)**	**Mode 7 (0111)**
15	GPIO							
16	GPIO							
17	UTOPIA		GPIO	UTOPIA	PCI	UTOPIA		
18	GPIO				PCI	GPIO		UTOPIA
19	GPIO				PCI	GPIO		UTOPIA
20	GPIO				PCI	GPIO		UTOPIA
21	GPIO							
22	GPIO							
23	GPIO							
24	GPIO							
25	GPIO	Ethernet	PCI			GPIO	Ethernet	GPIO
26	GPIO							
27	GPIO							
28	GPIO			PCI		GPIO		
29	GPIO		PCI			GPIO		
30	GPIO		PCI			GPIO		
31	GPIO		PCI			GPIO		

Note: GPIO includes any signal configured by the GPIO configuration registers, including GPI, GPO, $\overline{\text{IRQ}}$, and other dedicated functions.

Appendix B Further Reading

Related documentation for the MSC8144 device and the MSC8144ADS currently available on the Freescale web site and through the Literature Distribution Center is as follows:

- *MSC8144ADS Processor Board Reference Manual* (MSC8144ADSRM)
- *MSC8144ADS Hardware Getting Started Guide*
- *MSC8144 Reference Manual: Quad Core Media Signal Processor* (MSC8144RM)
- *Using an I^2C EEPROM During MSC814x Initialization* (AN3421)

THIS PAGE INTENTIONALLY LEFT BLANK

THIS PAGE INTENTIONALLY LEFT BLANK

How to Reach Us:

Home Page:
www.freescale.com

Web Support:
http://www.freescale.com/support

USA/Europe or Locations Not Listed:
Freescale Semiconductor, Inc.
Technical Information Center, EL516
2100 East Elliot Road
Tempe, Arizona 85284
+1-800-521-6274 or
+1-480-768-2130
www.freescale.com/support

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

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
support.asia@freescale.com

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

Document Number: AN3424
Rev. 0
05/2007

freescale™ semiconductor