Freescale Semiconductor
Application Note

Document Number: AN4321
Rev. 0, 08/2011

MSC815x Serial RapidIO Boot

SmartDSP Boot of MSC8156AMC from an ADS Host

by *Networking and Multimedia Group*
Freescale Semiconductor, Inc.
Austin, TX

This application note is a reference for the MSC815x boot process through the use of serial RapidIO (SRIO) transactions and the SmartDSP OS. It includes a booting example that serves as a learning exercise in setting up the MSC815x ATMUs, configuring the MSC8156AMC card, and booting a multi-device serial RapidIO system.

Although typically the MSC815x DSP is booted from a non-DSP host, such as a device built on Power Architecture® technology (for example, the P4080), this application note uses an MCS8156ADS as a boot host and the multi-device MSC8156AMC as the boot target.

While this document demonstrates a host booting the master DSP over serial RapidIO, it also provides the full configuration to put all three AMC DSPs into serial RapidIO boot mode.

This document applies to the following devices:

- MSC8144
- MSC8144E
- MSC8151
- MSC8152
- MSC8154

Contents

1. Background information 2
2. Setup for the MSC8156ADS to MSC8156AMC boot demo example 3
3. Performing serial RapidIO transactions 8
4. Using the LED example application 13
5. Running the ADS to AMC SRIO boot application . . 13
6. Revision history 15

- MSC8154E
- MSC8156
- MSC8156E
- MSC8157
- MSC8157E
- MSC8158
- MSC8158E
- MSC8251
- MSC8252
- MSC8254
- MSC8256
- MSC8256E

1 Background information

1.1 Reference materials relevant to boot example

This table lists reference material useful for serial RapidIO boot and serial RapidIO usage relevant to this document. The documentation IDs for Freescale documents are included for ease of ordering.

Table 1. Relevant reference materials

ID	Title	Purpose
Applicable device ID	Applicable device reference manual	Discusses the serial RapidIO boot process in detail (see the corresponding “Boot Program” section in the applicable device reference manual), the required device setup (see the corresponding “Reset” section in the applicable device reference manual), and the serial RapidIO and High Speed Serial Interface (HSSI) blocks details. The manual also provides relevant register address information and programming descriptions.
MSC8156ADSRM	*MSC8156ADS Reference Manual*	Provides configuration, module, and connectivity information for the MSC8156 ADS board. This document was used to provide information on MSC8156 Reset Device Configuration and SerDes to AMC backplane connectivity.
MSC8156AMCUM	*MSC8156 Advanced Mezzanine Card User’s Manual*	Used for reference on DIP switch configuration, DSP device ID, and SerDes switch configuration and connectivity.
AN4256	*Understanding OCN ATMU with SmartDSP OS Examples*	Explains the MSC815x’s OCN ATMU and how it pertains to SmartDSP. Also provides programming examples.

Table 1. Relevant reference materials (continued)

ID	Title	Purpose
AN3661	*RapidIO® Technology in Wireless Base Stations: Programming DSPs over a RapidIO Interconnect*	Provides background and reference material for serial RapidIO in the AMC system, and configuration/boot over a PPC host device.
TARGETING_ STARCORE_DSPS	*CodeWarrior Development Studio for StarCore DSP Architectures Targeting Manual*	Provides information on manipulating and reading the executable files associated with user projects for Freescale StarCore-based DSP devices. This document was used to generate boot images to be sent over serial RapidIO.

1.2 Acronyms and Abbreviations

This table contains acronyms and abbreviations used in this document.

Table 2. Acronyms and Abbreviated Terms

Term	Meaning
ADS	Application Development System
AMC	Advanced Mezzanine Card
ATMU	Address Translation Mapping Unit
CLASS	Chip Level Arbitration and Switching System
DIP	Dual-Inline Package
DSP	Digital Signal Processor
EPROM	Erasable Programmable Read-Only Memory
EEPROM	Electrically Erasable Programmable Read-Only Memory
OCN DMA	OCeaN DMA. Also known as On-chip Network DMA.
PEFCAR	Performance Capabilities Register
RCWHR	Reset Configuration Word High Register in the DSP
RCWLR	Reset Configuration Word Low Register in the DSP
SDOS	SmartDSP Operating System
uTCA	Micro Telecommunications Computing Platform

2 Setup for the MSC8156ADS to MSC8156AMC boot demo example

In the following example, the MSC8156ADS is used as a boot host to boot the MSC8156AMC card with a program that flashes the AMC card's LEDs.

NOTE

The example application code in this section was tested using the SmartDSP OS v3.7.0 and CodeWarrior 10.1.5. Updates in the OS and tools since then are not expected to impact these specific projects.

2.1 Connectivity considerations

The MSC815x can be configured for up to 4x SRIO, which is used in this example because it is supported on both the MSC8156ADS and MSC8156AMC cards. Both ADS and AMC cards can be plugged into a Schroff uTCA chassis to enable connectivity.

2.2 MSC8156AMC connectivity and architecture

This figure shows the MSC8156AMC block diagram. Note the SerDes lane numbers available on the AMC card's connector; these are used to connect over the uTCA. There are three DSPs on the MSC8156AMC card that are connected via a serial RapidIO switch to the AMC connector.

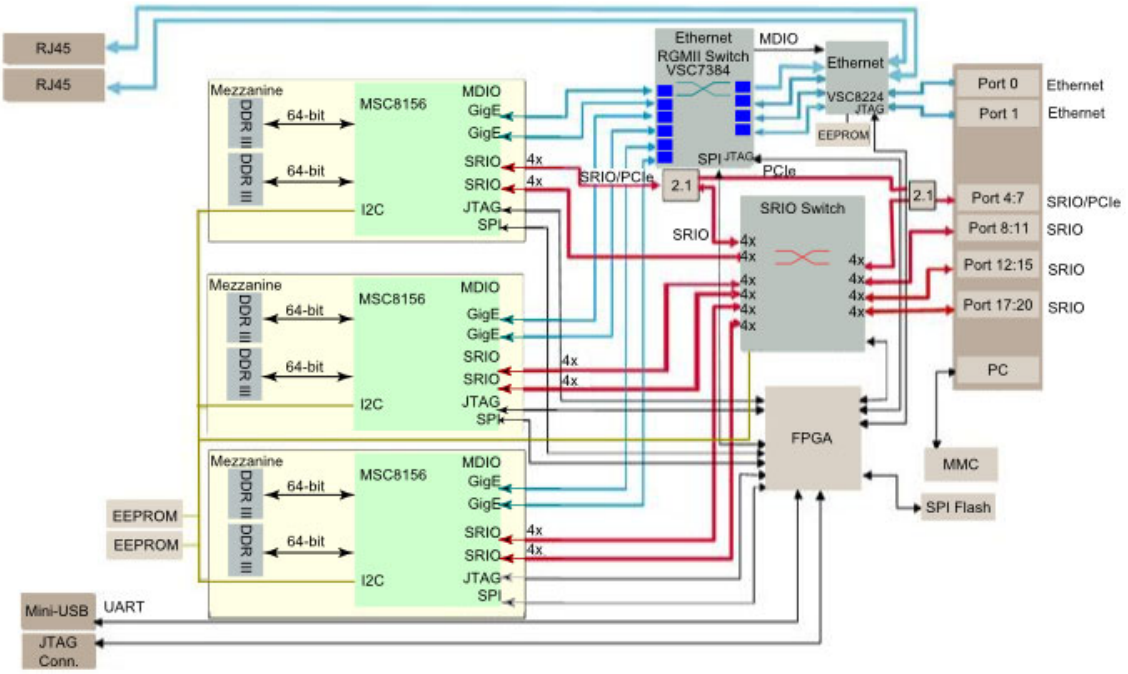

Figure 1. MSC8156AMC block diagram

2.3 MSC8156ADS connectivity and architecture

The MSC8156ADS provides the user with the option to connect either via AMC ports 4:7 or via AMC ports 8:11. The uTCA chassis slots #1 and #4 allow the user to connect over lanes 8:11 to provide space for fans and other devices; this also provides the user with space to change DIP switches, if needed, without removing cards. The figure below shows the MSC8156ADS serial RapidIO-only block diagram.

Ports 8:11 connect to the switch on the AMC and to SerDes1 on the ADS. Because ports 8:11 connect to a switch on the AMC, the user has the option of using either of the MSC8156's SerDes ports for AMC DSPs. AMC port 8:11 connects to the switch's port #4. The host is properly configured when ports 8:11 are connected to switch port #4 on the AMC, and ports 8:11 are connected to the SerDes1 on the ADS.

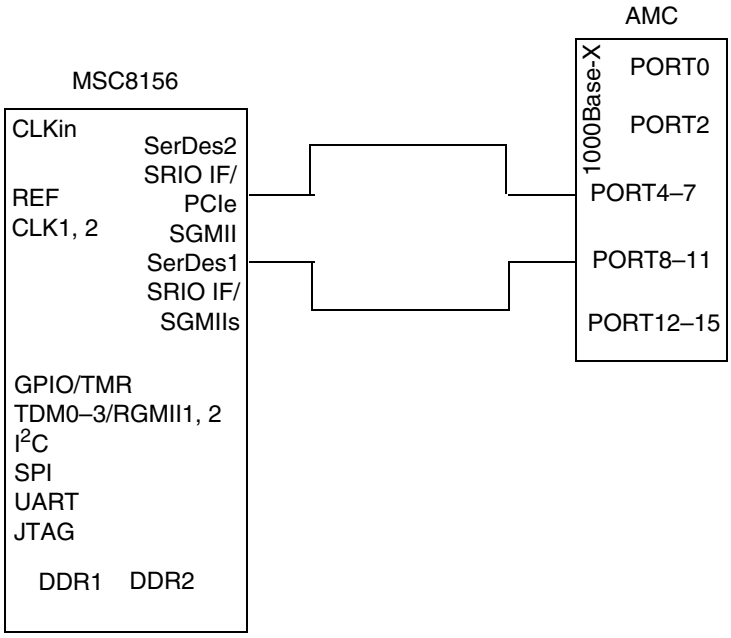

Figure 2. MSC8156ADS serial RapidIO-only block diagram

When the lanes (in this case, lanes 8:11) and uTCA slots have been determined, the hardware will be set up as shown in this figure.

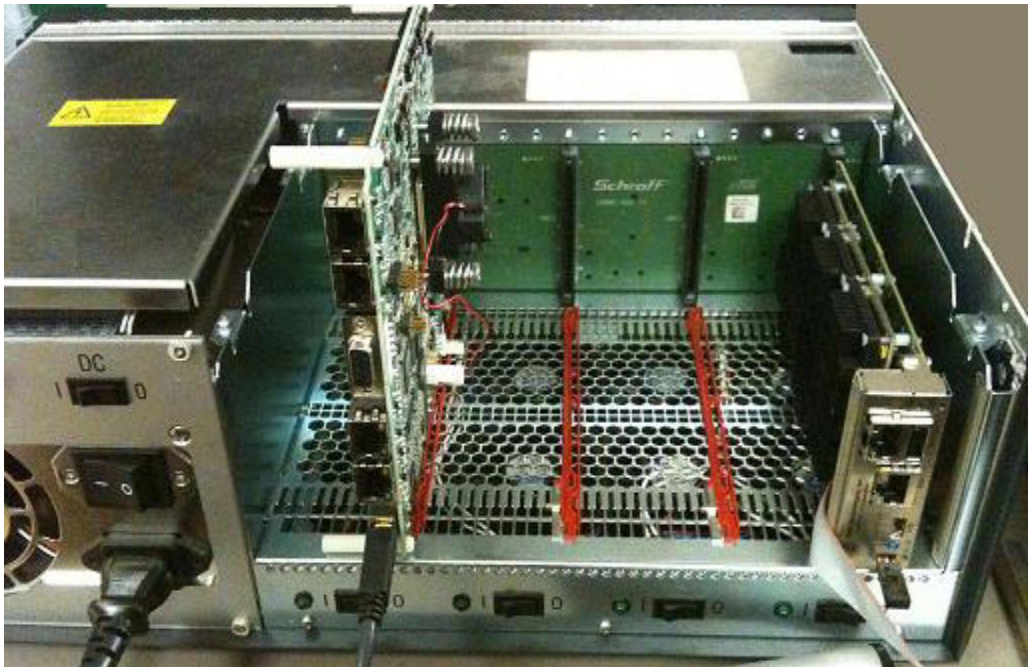

Figure 3. Proper setup of uTCA chassis with MSC8156ADS and MSC8156AMC connected over lanes 8:11

2.4 AMC-specific setup

When the hardware is set up as shown in Figure 3, configure the AMC card so that the DSPs are ready to boot over serial RapidIO.

2.4.1 Configuring the AMC DSPs out of reset

The AMC DSPs are configured out of reset via an I^2C EEPROM shared by all DSPs. The EEPROM contains the RCWHR and RCWLR configuration words to determine boot port, peripheral configuration, and device ID.

To load from I^2C EEPROM, the MSC8156AMC DIP switches are set up as follows (where 1 = OFF):

```
SW2 = 00010100
SW1 = 00000000
```

Additionally, the user must configure RCWLR[S1P/S2P] appropriately. Because an MSC8156 contains only one serial RapidIO ID, when the serial RapidIO switch is enumerating, there will be a problem if both MSC8156 serial RapidIO ports are enabled and connected to the switch; this is because two switch ports must not contain the same serial RapidIO device ID. If the user attempts to boot through more than one port, unpredictable results occur. If the two switch ports contain the same device ID, the user must make sure only one is enabled. The user must only configure S1P for 4x SRIO; the other SerDes port will be disabled.

The data contents, which the user must load into the EEPROM and which configure the RCWLR and RCWHR for 4x SRIO, along with the executable to configure DSP1 for serial RapidIO boot are provided with this application note, in the following zip file:

```
BOOT1A_Workaround_forDSP1_rest_normal-AMC_v3.txt
```

Booting out of EEPROM—using the above file—configures all DSPs for serial RapidIO boot and moves them into standard serial RapidIO boot operation.

NOTE

The AMC switch out of reset is not enumerated; therefore, this is left to the boot host to manage. For details, see Section 3.1, “SmartDSP OS enumeration of MSC8156AMC’s SRIO switch.”

2.4.2 Standard process for booting over serial RapidIO

This figure illustrates the standard process for MSC815x booting over serial RapidIO.

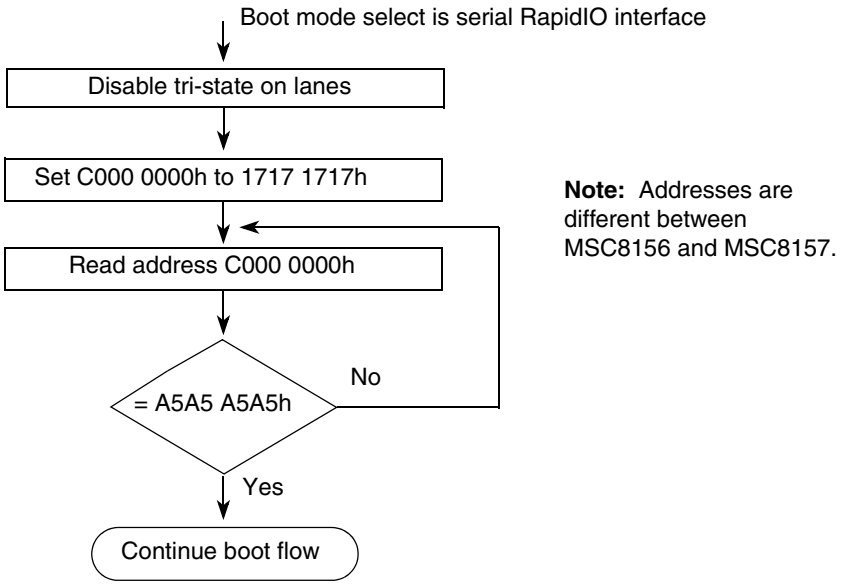

Figure 4. MSC8157 serial RapidIO boot flow

The MSC815x comes out of reset, opens and enables serial RapidIO and SerDes lanes, and then writes 1717 1717h to its own M3 memory (address C010 1C00h for MSC8156). At this point, the MSC815x polls that same memory location until the boot host has updated the value from 1717 1717h to A5A5 A5A5h.

The role of the boot host is to load the application into the MSC8156's memory and then update the "start address" (also known as the "entry point"), to which the PC jumps, so that the PC jumps to the new application loaded by the host; that is, the boot host sets the address and then the controlling program jumps to the new address to begin execution of the application loaded by the host starting at that address in memory.

The memory location where the entry point is indicated is C010 1C10h on the MSC8156.

After the full program and program start address have been loaded, the boot host's final function is to write A5A5 A5A5h to the address the MSC8156 is polling (address C010 1C00h for MSC8156). At this point, the MSC8156 executes the loaded application.

2.5 ADS-specific setup (serial RapidIO boot host)

The MSC8156ADS must be set up to configure SerDes1 for 4x SRIO, which is done using the following DIP switch settings. Note that 1 = OFF:

```
SW1 = 00000000
SW2 = 00000110
SW3 = 00010000
SW4 = 11000000
```

```
SW5 = 10010000
SW6 = 00000000
SW7 = 00000011
```

Details of the boot host code are covered in the following sections:

- Section 3, “Performing serial RapidIO transactions”
- Section 4, “Using the LED example application”
- Section 5, “Running the ADS to AMC SRIO boot application”

3 Performing serial RapidIO transactions

The MSC8156 serial RapidIO interface provides the capability to perform standard serial RapidIO transactions (NWRITE, NWRITE_R, SWRITE, and NREAD). These transactions are used in this example to perform loading and checking of boot code, along with signaling of boot load completion.

3.1 SmartDSP OS enumeration of MSC8156AMC’s SRIO switch

The SmartDSP OS initializes the MSC815x serial RapidIO by setting up internal registers and by doing a standard port discovery process. When performing these tasks, the SmartDSP OS checks what is connected to the MSC8156. When the ADS is connected to the MSC8156AMC, the SmartDSP OS detects the switch connection and assigns serial RapidIO IDs to the switch lanes, as shown in the following table.

This table shows how the SmartDSP OS assigns serial RapidIO IDs to the switch lanes.

Table 3. SDOS enumeration of MSC8156AMC’s serial RapidIO switch

Switch Lane	Port Mode	Serial RapidIO Switch Port Number	AMC End Point	Serial RapidIO ID
3:0	4x	0	DSP2 SRIO0	ID = 1
7:4	4x	1	DSP2 SRIO1	NA
11:8	4x	2	DSP3 SRIO0	ID = 2
15:12	4x	3	DSP3 SRIO1	NA
19:16	4x	4	AMC SRIO1	Connected directly to the ADS
23:20	4x	5	AMC SRIO2	NA
27:24	4x	6	AMC SRIO0	NA
31:28	4x	7	SMC SRIO3	NA
35:32	4x	8	DSP1 SRIO0	ID = 3
39:36	4x	9	DSP1 SRIO1	NA

Note:
1. Serial RapidIO IDs for the disabled serial RapidIO ports are listed as NA.

After the SmartDSP OS assigns IDs to the switch lanes, the user may direct transactions towards these IDs. Note that, in Section 2.4.1, "Configuring the AMC DSPs out of reset," only one port of serial RapidIO is enabled for each DSP.

3.2 Setting up the ATMU windows

To perform serial RapidIO transactions, the user must configure the ATMUs. The MSC8156 has ATMUs for the following:

- For the two OCN DMA controllers
- For each serial RapidIO unit

These ATMUs are used basically as a multiplexer to direct data traffic to different destinations.

3.2.1 Understanding the OCN DMA controller ATMU window

The OCN DMA controller ATMU allows up to ten access windows. The programmer must set up at least a subset of these windows (see msc815x_config.c in the SmartDSP demos) to allow the DMA controller to access the memory space.

A DMA ATMU window includes the following:

- Start address
- Window size
- Interface

The window start address and size act effectively as the multiplex controller. If an address is within the range between an ATMU window's start address to the start address + window size then that ATMU window is selected and the transaction is forwarded to the output port determined by the ATMU window's interface (output port can be to the PCI Express block, SRIO0, SRIO1, or to the MSC815x's internal subsystem CLASS via one of the OCN/CLASS ports).

3.2.2 Configuring the OCN DMA controller ATMU windows

In the case of the serial RapidIO boot demo for this application note, the host's ATMU windows in the SmartDSP's msc815x_config.c is defined as follows:

```
ocn_dma_init_params_t ocn_dma_init_params1 ={
     OCN_DMA_ID1,
     OS_HWI_PRIORITY0,
     {   /* On which core to enable the channel. */
         ANY_CORE_ID, ANY_CORE_ID, ANY_CORE_ID, ANY_CORE_ID,
     },
     {
         {
             RIO0_ATMU_BASE,
             OCN_DMA_WIN_2M,//DMA_ATMU_WIN_SIZE,
             OCN_DMA_INTERFACE_RIO0
         },
```

```
        {
            RIO0_ATMU_BASE*2,
            OCN_DMA_WIN_2M,//DMA_ATMU_WIN_SIZE,
            OCN_DMA_INTERFACE_RIO0
        },
        {
            RIO0_ATMU_BASE*3,
            OCN_DMA_WIN_2M,//DMA_ATMU_WIN_SIZE,
            OCN_DMA_INTERFACE_RIO0
        },
        {
            RIO0_ATMU_BASE*4,
            OCN_DMA_WIN_2M,//DMA_ATMU_WIN_SIZE,
            OCN_DMA_INTERFACE_RIO0
        },
        {
            M3_BASE_ADDR,
            DMA_ATMU_WIN_SIZE,
            OCN_DMA_INTERFACE_LOCAL1
        },
        {
        0x40000000ULL,//DDR0 base address
        OCN_DMA_WIN_1G,//DMA_ATMU_WIN_SIZE,
            OCN_DMA_INTERFACE_LOCAL1
        }
    }};
```

In the example above, six OCN DMA controller ATMU windows are created. The first four windows provide accesses to data over SRIO0; the last two windows provide data accesses to memory connected to M3 and DDR on this chip.

There are four separate windows with four separate start addresses, all of which are pointed to SRIO0. This is because when these transfers reach SRIO0, they are then directed by a second ATMU, the serial RapidIO's ATMU, towards specific serial RapidIO IDs for serial RapidIO transactions.

3.2.3 Understanding the serial RapidIO ATMU windows

A serial RapidIO ATMU window contains the following:

- ATMU window number
- Serial RapidIO ID
- OCN base address
- Translation address
- Window size
- Priority level
- Access type

NOTE

The OCN base address is the address used as an input to the “ATMU MUX.” The translation address is the output address from the “ATMU MUX.”

3.2.4 Configuring the serial RapidIO ATMU windows

In the case of the serial RapidIO boot demo for this application note, the serial RapidIO ATMU windows in msc815x_config.c have been configured as follows:

```
{//AMC Master DSP M3 memory window
        1,                  /* Window number */
        3,                  /* SRIO ID towards which to direct the flow */
        RIO0_ATMU_BASE,     /* 36 bit Ocean address. - Base address*/
       M3_BASE_ADDR,      /* 34 bit RapidIO translation address - destined for M3 */
        RIO_ATMU_WIN_SIZE,  /* Window size */
        MEDIUM_FLOW_LEVEL,  /* Priority level */
        NWRITE              /* Access type on RapidIO */
    },
    {//AMC DSP2 M3 memory window
        2,                  /* Window number */
        1,                  /* SRIO ID towards which to direct the flow */
        RIO0_ATMU_BASE * 2,     /* 36 bit Ocean address. - Base address*/
       M3_BASE_ADDR,      /* 34 bit RapidIO translation address - destined for M3 */
        RIO_ATMU_WIN_SIZE,  /* Window size */
        MEDIUM_FLOW_LEVEL,  /* Priority level */
        NWRITE              /* Access type on RapidIO */
    },
    {//AMC DSP3 M3 memory window
        3,                  /* Window number */
        2,                  /* SRIO ID towards which to direct the flow */
        RIO0_ATMU_BASE*3,     /* 36 bit Ocean address. - Base address*/
       M3_BASE_ADDR,      /* 34 bit RapidIO translation address - destined for M3 */
        RIO_ATMU_WIN_SIZE,  /* Window size */
        MEDIUM_FLOW_LEVEL,  /* Priority level */
        NWRITE              /* Access type on RapidIO */
    },
    {//AMC Master DSP M2 memory window
        4,                  /* Window number */
        3,                  /* SRIO ID of AMC Master device */
        RIO0_ATMU_BASE*4,     /* 36 bit Ocean address. - Base address*/
       0x30000000ULL,     /* 34 bit RapidIO translation address - destined for M3 */
```

```
            SRIO_ATMU_OUT_WIN_512K,//RIO_ATMU_WIN_SIZE,  /* Window size */
            MEDIUM_FLOW_LEVEL,  /* Priority level */
            NWRITE              /* Access type on RapidIO */
        }, ....
```

All four SRIO ATMU windows are capable of being programmed to be decoded into four separate serial RapidIO endpoints, or four different address ranges of a single SRIO endpoint. The sample code above illustrates how four windows are directed to three different SRIO endpoints. The first and fourth windows both go to SRIO ID3, which points to DSP1 of the AMC card. For more information on how the SmartDSP OS assigns serial RapidIO IDs to the switch lanes, see Table 3. The difference between the first and fourth window is the translation address.

3.3 Creating an OCN DMA controller transaction

When the OCN DMA controller ATMU and the SRIO ATMU are both set up, an example transaction, such as performing a single write of a block of data from the ADS's DDR to M3 of the AMC's DSP1, can be performed. To perform this task, the user needs to create an OCN DMA transaction that:

- Reads data from the DDR (with an appropriate address offset),
- And sends data to the SRIO aimed for SRIO ATMU window four (with an appropriate offset).

To create an OCN DMA that meets the above requirements, the user must create a DMA transaction. Because it is only a single transaction, the user should create a direct transaction. The full direct transaction example is provided in the host code provided in the zip file for this application note.

3.3.1 Setting up the direct DMA transaction

The direct DMA transaction for the example in Section 3.3, "Creating an OCN DMA controller transaction," is set up as follows:

```
ocn_dma_transfer_config.source.addr.high_addr = 0;
        ocn_dma_transfer_config.source.addr.low_addr = (uint32_t)boot_seg0;//DDR
        ocn_dma_transfer_config.size = sizeof(boot_seg0);
        ocn_dma_transfer_config.destination.addr.low_addr= (uint32_t)0x00000000;
//0x30000000 intended address -> base offset provided by ATMU
        ocn_dma_transfer_config.destination.addr.high_addr = 4;
        /* Performing transfer directly. */
        status = ocnDmaTransfer(ocn_dma[TRANSACTION_DMA], &ocn_dma_transfer_config, 0);
```

3.3.2 Understanding the sample code

In the sample code provided in Section 3.3.1, "Setting up the direct DMA transaction," the high address of the source is set to 0, and the lower 32 bits are assigned to a variable stored in DDR memory. The DMA controller decodes this using its last (sixth) ATMU window (as set up in Section 3.2.2, "Configuring the OCN DMA controller ATMU windows"), and considers this as an access through the CLASS bus to DDR memory.

The destination address highest 32 bits are set to 4, and the OCN DMA controller decodes this using its fourth ATMU window to forward the data on to SRIO0.

SRIO0 reads the high 32-bit address as 4, which matches the 36-bit OCN address for SRIO ATMU window #4. SRIO0 forwards the transaction on using the description of ATMU window 4: NWRITE to SRIO endpoint with ID #3, Address = 3000 0000h + the lower 32-bit address offset, which was 0 in the example code.

If the source and destination addresses of the transfer were reversed, the transaction would have acted as an NREAD instead of an NWRITE, pulling data from SRIO ATMU window 4 (SRIO ID 3, which is the AMC DSP1) and writing to DDR memory.

4 Using the LED example application

The LED example application, which flashes AMC LED D1 (the green LED, not the yellow LED), is provided in the zip file contained with this application note, under the “Files for AMC” folder.

To obtain the segment values required to load over serial RapidIO, the ELD is extracted from the code example in a way similar to the standard I^2C EEPROM boot demo; using the output from the i2cboot.exe application, each segment is extracted from the EEPROM text file. The first 32 bytes are the “start address” of the segment. For detailed information on reading ELD (ELF format files) for the purpose of creating a boot image, see the CodeWarrior Development Studio for StarCore DSP Architectures Targeting Manual, provided with the CodeWarrior development tools in the **<CodeWarrior Install>\SC\Help\PDF** folder. The I2CBOOT utility tool is used for the small LED Flashing demo. Users with more complex applications are advised to refer to the sections on the ELF File Dump Utility and the ELF2XX utility in the Targeting Manual.

Instead of creating an intelligent ELF reader, the required program, data, and segments for the LED flash ELD are loaded into a header file. As noted above, when moving to a larger, more complicated application, it is advised that the programmer use an ELF reader utility to read directly from an ELD file at boot time and boot over SRIO.

5 Running the ADS to AMC SRIO boot application

The AMC card requires the setup described in the *MSC8156 Advanced Mezzanine Card User’s Manual*.

To burn the EEPROM for the AMC board:

1. Create a SmartDSP demo folder based on the standard msc815x demo folder. The user must ensure that the folder contains the AMC support related folder package required for all AMC projects (that is, the amc_support package). The folder created should reside in a directory parallel to the msc815x directory. Its location should resemble the following: demos\starcore\msc815x_amc\
2. Extract the AMC_i2c_burn_DSP_EEPROM.zip file to this folder.
3. Ensure the I^2C code includes the BOOT1A_Workaround_forDSP1_rest_normal-AMC_v3.txt.
4. Build and run this application on the AMC card.
5. Repower or PORESET the AMC card using the blue push button on the front of the card.

To run the serial RapidIO boot demo on the ADS:

1. After the MSC8156AMC is programmed with the new EEPROM image and repowered, extract the rio_dma-amc_boot_host.zip file into the standard SmartDSP demo's folder for ADS code: demos\starcore\msc815x\
2. Build and run this demo.
3. The demo will stop on a debug statement break in appRioDmaInit after using direct DMA transactions to send the boot application to the AMC's DSP1.
4. The demo provides the following output:

```
Discovered a switch with 10 ports. Connected to port 4 (hop count=0)
Mapped Dev ID 0x0 to port 4 on switch, hop count=0
Port 0 on switch is active (hop count=0)
Mapped Dev ID 0xffff to port 0 on switch, hop count=0
Discovered endpoint. Enumerated with ID 1 (hop count=1).
Update routing table with found endpoint
Mapped Dev ID 0x1 to port 0 on switch, hop count=0
Port 2 on switch is active (hop count=0)
Mapped Dev ID 0xffff to port 2 on switch, hop count=0
Discovered endpoint. Enumerated with ID 2 (hop count=1).
Update routing table with found endpoint
Mapped Dev ID 0x2 to port 2 on switch, hop count=0
Port 8 on switch is active (hop count=0)
Mapped Dev ID 0xffff to port 8 on switch, hop count=0
Discovered endpoint. Enumerated with ID 3 (hop count=1).
Update routing table with found endpoint
Mapped Dev ID 0x3 to port 8 on switch, hop count=0
```

Note that re-enumerating the switch when it is already enumerated does not work. At this point, the MSC8156AMC card's green LED should start blinking, indicating that boot has successfully loaded to the AMC DSP1, and executed.

6 Revision history

This table provides a revision history for this document.

Table 4. Document revision history

Rev. Number	Date	Substantive Change(s)
0	08/2011	Initial public release

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 China Ltd.
Exchange Building 23F
No. 118 Jianguo Road
Chaoyang District
Beijing 100022
China
+86 10 5879 8000
support.asia@freescale.com

For Literature Requests Only:
Freescale Semiconductor
Literature Distribution Center
1-800 441-2447 or
+1-303-675-2140
Fax: +1-303-675-2150
LDCForFreescaleSemiconductor
@hibbertgroup.com

Document Number: AN4321
Rev. 0
08/2011