SmartFusion2 SoC FPGA Code Shadowing from eMMC Device to DDR Memory

Demo Guide

Superseded

April 2014

Microsemi

Revision History

Date	Revision	Change
16 April 2014	1	First Release

Confidentiality Status

This is a non-confidential document.

Superseded

Table of Contents

Preface . . . 4
About this Document . . . 4
Intended Audience . . . 4
References . . . 4
Microsemi Publications . . . 4

SmartFusion2 SoC FPGA - Code Shadowing from eMMC Device to DDR Memory . . . 5
Introduction . . . 5
Demo Design . . . 6
Introduction . . . 6
Demo Design Features . . . 6
Demo Design Description . . . 7
Setting Up the Demo Design . . . 11
Running the Demo Design for Multi-Stage Boot Process Method . . . 13
Running the Demo Design for Hardware Boot Engine Method . . . 21
Conclusion . . . 21
Appendix 1: DDR Configurations . . . 22
Appendix 2: Generating Executable Bin File . . . 24

Product Support . . . 25
Customer Service . . . 25
Customer Technical Support Center . . . 25
Technical Support . . . 25
Website . . . 25
Contacting the Customer Technical Support Center . . . 25
Email . . . 25
My Cases . . . 26
Outside the U.S. . . . 26
ITAR Technical Support . . . 26

Superseded

Preface

About this Document

This demo is for SmartFusion®2 system-on-chip (SoC) field programmable gate array (FPGA) devices. It provides instructions on how to use the corresponding reference design.

Intended Audience

SmartFusion2 devices are used by:

- FPGA designers
- Embedded designers
- System-level designers

References

Microsemi Publications

- SmartFusion2 Microcontroller Subsystem User Guide
- SmartFusion2 SoC FPGA Remapping eNVM, eSRAM, and DDR/SDR SDRAM Memories Application Note
- Configuring Serial Terminal Emulation Programs Tutorial

Refer to the following web page for a complete and up-to-date listing of SmartFusion2 device documentation:

www.microsemi.com/index.php?option=com_content&id=1664&lang=en&view=article#documents

Superseded

SmartFusion2 SoC FPGA - Code Shadowing from eMMC Device to DDR Memory

Introduction

This demo design shows SmartFusion2 SoC FPGA device capabilities for code shadowing from embedded multi-media card (eMMC) device to double data rate (DDR) synchronous dynamic random access memory (SDRAM) and executing the code from DDR SDRAM. Figure 1 shows the top-level block diagram for code shadowing from eMMC device to DDR memory.

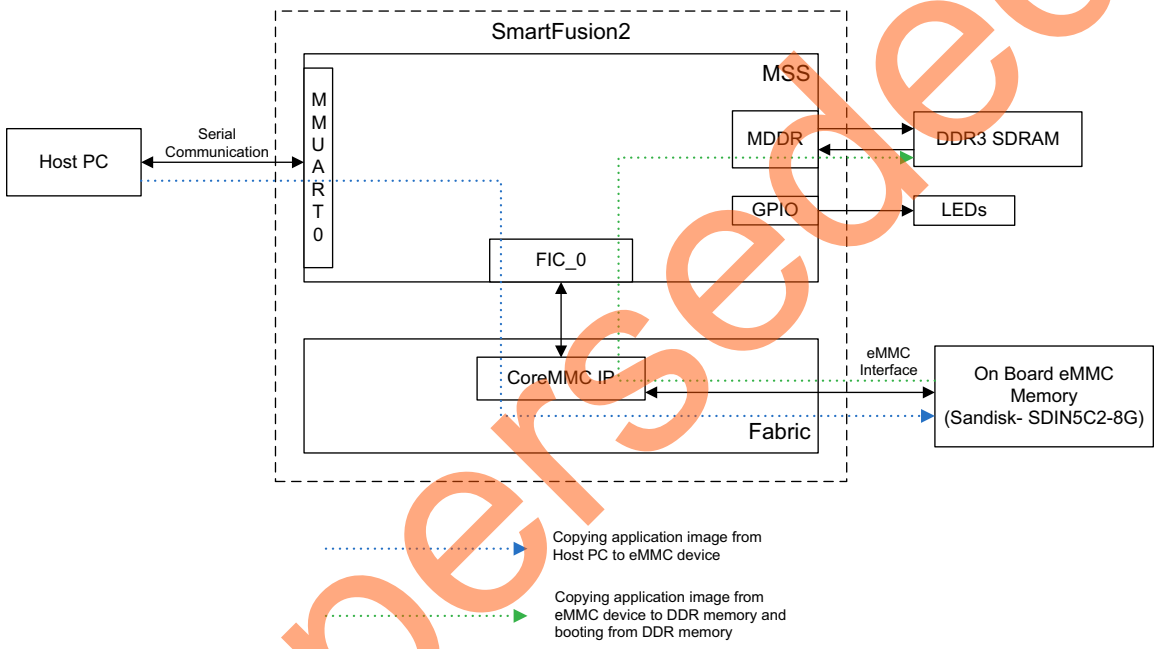

Figure 1 • **Top-Level Block Diagram of the Demo**

Code shadowing is a booting method that is used to run an image from external, faster, and volatile memories (DRAM). It is the process of copying the code from the non-volatile memory to the volatile memory for execution.

Code shadowing is required when the non-volatile memory associated with a processor does not support random access to the code for execute-in-place, or there is insufficient non-volatile random access memory. In performance-critical applications, the execution speed can be improved by code shadowing, where code is copied to higher throughput RAM for faster execution.

Single data rate (SDR)/DDR SDRAM memories are used in applications that have a large application executable image and require higher performance. Typically, the large executable images are stored in non-volatile memory, such as NAND flash or SPI flash, and copied to volatile memory, such as SDR/DDR SDRAM memory, at power up for execution.

SmartFusion2 SoC FPGA devices integrate fourth generation flash-based FPGA fabric, an ARM® Cortex™-M3 processor, and high performance communication interfaces on a single chip. The high speed memory controllers in the SmartFusion2 SoC FPGA devices are used to interface with the external DDR2/DDR3/LPDDR memories. The DDR2/DDR3 memories can be operated at maximum speed of 333 MHz. The Cortex-M3 processor can directly run the instructions from external DDR

Superseded

memory through the microcontroller subsystem (MSS) DDR (MDDR). The FPGA Cache Controller and MDDR bridge handles the data flow for a better performance.

Table 1 shows the reference design requirements and details for this demo.

Table 1 • **Reference Design Requirements and Details**

Reference Design Requirements and Details	Description
Hardware Requirements	
SmartFusion2 Development Kit • 12 V adapter • FlashPro4 programmer • USB A to Mini-B USB cable	Rev C or later
Host PC or Laptop	• Windows XP SP2 Operating System - 32-bit/64-bit • Windows 7 Operating System - 32-bit/64-bit
Software Requirements	
Libero® System-on-Chip (SoC) for viewing the design files	11.3
FlashPro Programming Software	11.3
SoftConsole	3.4
Host PC Drivers	USB to UART drivers
One of the following serial terminal emulation programs: • HyperTerminal • TeraTerm • PuTTY	-

Superseded

Demo Design

Introduction

The demo design files are available for download from the following path in the Microsemi® website:
http://soc.microsemi.com/download/rsc/?f=SF2_CODE_SHADOWING_DDR_EMMC_DF

The demo design files include:

- Libero SoC project
- STAPL programming files
- Sample application SoftConsole project
- eMMC host PC loader (`m2s_emmc_loader.exe`)
- Linker scripts
- DDR configuration files
- readme.txt file

Refer to the `readme.txt` file provided in the design files for the complete directory structure.

Demo Design Features

The demo design implements the following features:

- Loading the target application into eMMC device from host PC using UART serial communication
- Reading the eMMC device contents and loading into DDR3 memory
- Running the LEDs blinking application from DDR3 memory using bootloader

Demo Design Description

This demo design implements code shadowing technique to boot the application image from DDR memory. The design also provides host interface over SmartFusion2 SoC FPGA multi-mode universal asynchronous/synchronous receiver/transmitter (MMUART) to load the target application executable image (*.bin) to the on-board eMMC. The cache controller is enabled in MSS configurator of the demo design. This cache controller caches the instructions when the Cortex-M3 executes code from either eNVM or DDR memory.

The code shadowing is implemented in the following two methods:

- Multi-stage boot process method using Cortex-M3 processor
- Hardware boot engine method using FPGA fabric

Multi-Stage Boot Process Method

The application image is run from external DDR memories in the following two boot stages:

- Cortex-M3 processor boots the soft bootloader from embedded nonvolatile memory (eNVM), which copies the target application executable image from eMMC device to the DDR memory after configuring the DDR controller.
- The program execution control is transferred to external DDR memory by configuring the DDR_CR system register.
- Cortex-M3 processor boots the target application image from DDR memory.

The design implements a bootloader program to load the target application executable image from eMMC device to DDR memory for execution. The bootloader program running from eNVM jumps to the target application stored in DDR memory after the target application image is copied to DDR memory. Figure 2 shows the detailed block diagram of the demo design.

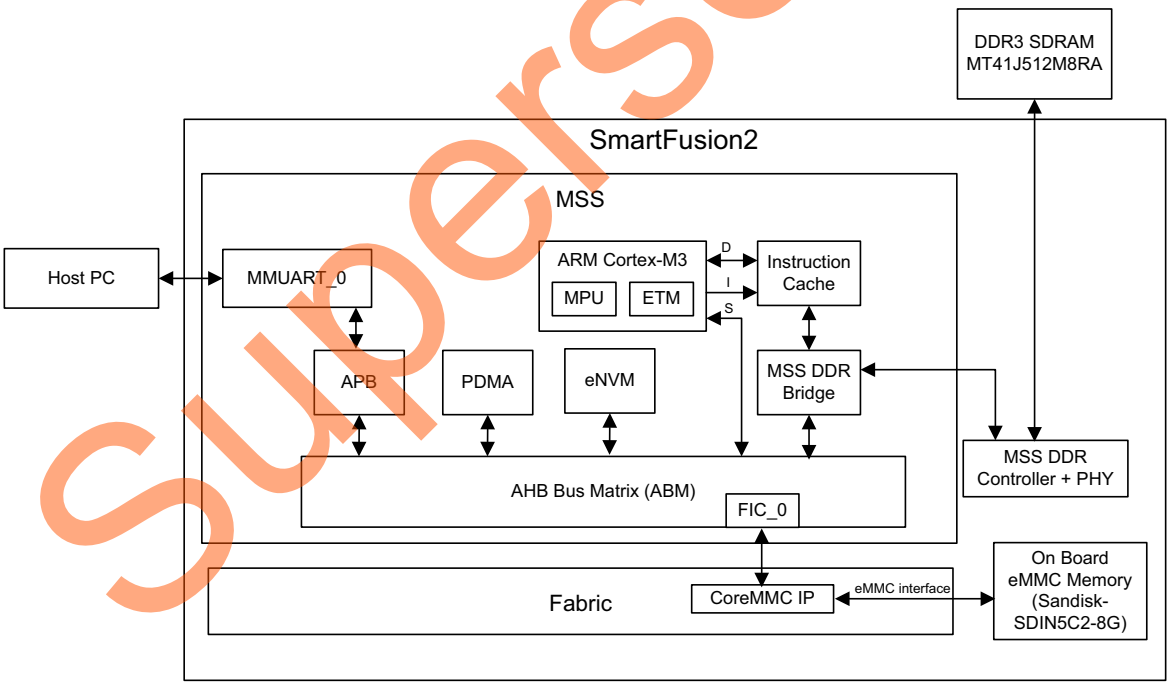

Figure 2 • **Code Shadowing - Multi Stage Boot Process Demo Block Diagram**

The MDDR is configured for DDR3 to operate at 333 MHz. "Appendix 1: DDR Configurations" on page 22 shows the register values that need to be configured for accessing DDR3. The DDR is configured before executing the main application code.

Superseded

Bootloader

The bootloader performs the following operations:

1. Copying the target application image from eMMC to DDR memory.
2. Remapping the DDR memory starting address from 0xA0000000 to 0x00000000 by configuring the DDR_CR system register.
3. Initializing the Cortex-M3 processor stack pointer as per the target application. The first location of the target application vector table contains the stack pointer value. The vector table of the target application is available starting from the address 0x00000000.Loading the program counter (PC) to reset handler of the target application for running the target application image from the DDR memory. Reset handler of the target application is available in the vector table at the address 0x00000004.

The demo design flow is described in Figure 3.

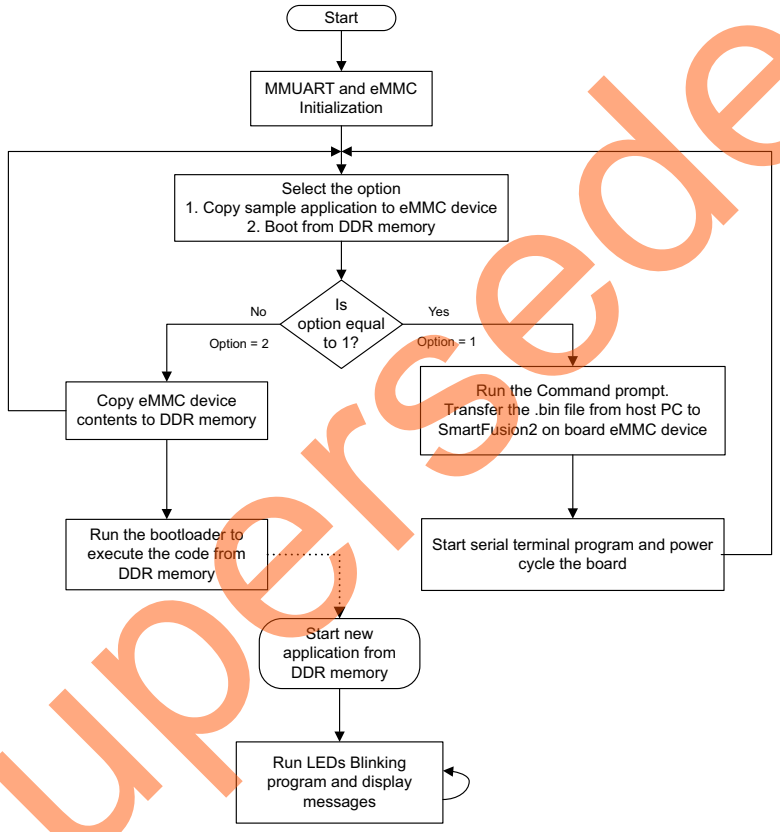

Figure 3 • **Design Flow for Multi-Stage Boot Process Method**

Hardware Boot Engine Method

In this method, the Cortex-M3 directly boots the target application image from external DDR memories. The hardware boot engine copies the application image from eMMC device to DDR memory, before releasing the Cortex-M3 processor reset. After releasing the reset, the Cortex-M3 processor boots directly from DDR memory.

This demo design implements boot engine logic in FPGA fabric to copy the target application executable image from eMMC flash to the DDR memory for execution. The design also implements writing into eMMC device, which can be executed by Cortex-M3 processor to load the target application executable image into eMMC device using serial communication through UART interface with host PC. The DIP switch1 on the SmartFusion2 Development Kit can be used to select whether to program the eMMC device or to execute the code from DDR memory.

Superseded

If the executable target application is available in eMMC device, the code shadowing from eMMC device to DDR memory is started on device power-up. The boot engine initializes the MDDR, copies the Image from eMMC device to DDR memory, and remaps the DDR memory space to 0x00000000 by keeping the Cortex-M3 processor in reset. After the boot engine releases the Cortex-M3 reset, the Cortex-M3 executes the target application from DDR memory.

Figure 4 shows the detailed block diagram of the demo design. The FIC_0 is configured in **Slave** mode to remap the DDR memory to 0x00000000 from FPGA fabric AHB master. The MDDR AXI interface (DDR_FIC) is enabled to access the DDR memory from FPGA fabric AXI master.

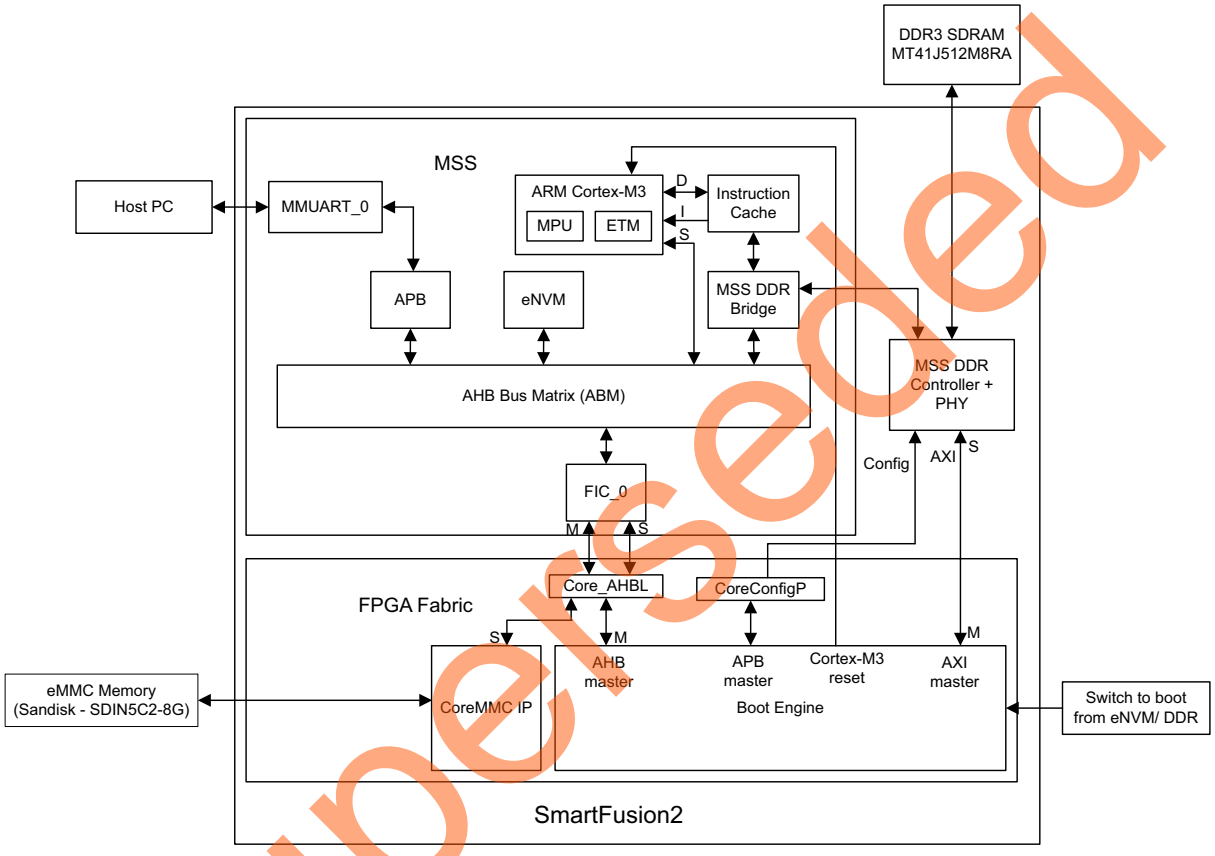

Figure 4 • **Code Shadowing - Hardware Boot Engine Demo Block Diagram**

Boot Engine

This is the major part of the code shadowing demo that copies the application image from eMMC device to the DDR memory. The boot engine performs the following operations:

1. Initializing MDDR for accessing DDR3 at 320 MHz by keeping the Cortex-M3 processor in reset.
2. Copying the target application image from eMMC device to DDR memory using the AXI master in the FPGA fabric through MDDR AXI interface.
3. Remapping the DDR memory starting address from 0xA0000000 to 0x00000000 by writing to the DDR_CR system register.
4. Releasing reset to Cortex-M3 processor to boot from DDR memory.

Superseded

The demo design flow is described in Figure 5.

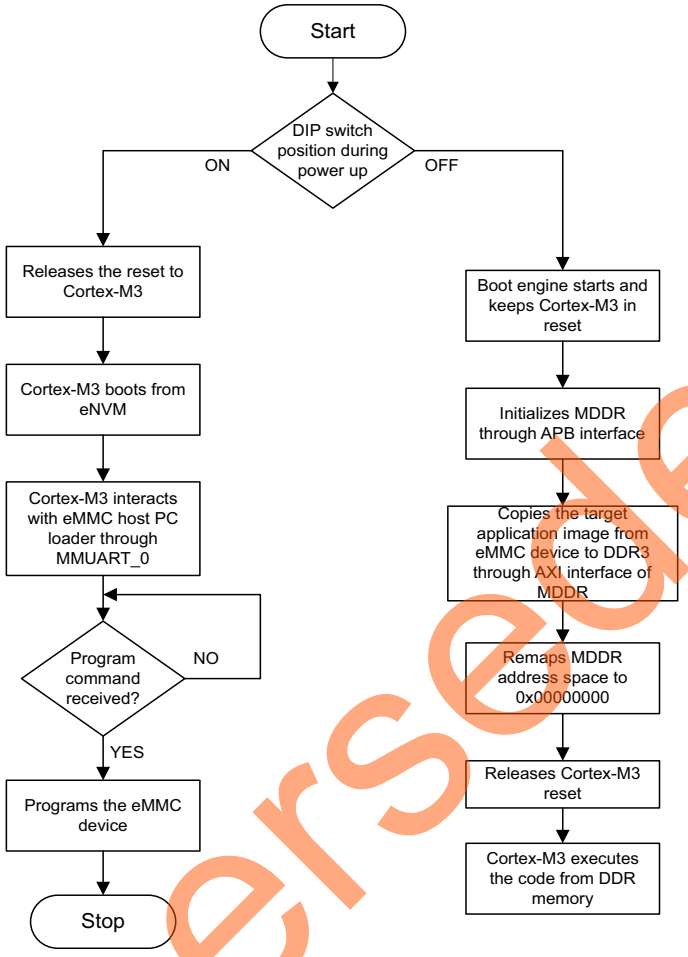

Figure 5 • **Design Flow for Hardware Boot Engine Method**

Creating Target Application Image for DDR Memory

A target application image is executed from the DDR memory in this demo. The linker description file `production-execute-in-place-externalDDRorSDR.ld` (included in the design files) is used to build the application image. This file defines the DDR memory starting address as 0x00000000, since the bootloader/boot engine performs DDR memory remapping from 0xA0000000 to 0x00000000. This linker script creates an application image with stack and data sections in memory whose starting address is 0xA0080000. A simple light-emitting diode (LED) blinking application image file with interrupt service routines to display the messages on to the serial terminal emulation program is provided for this demo.

The sample application SoftConsole project is provided with design files. It is located at: *<download_folder>\sf2_code_shadowing_to_ddr_from_emmc_df\sample_application*.

The sample application does not re-configure the DDR registers by disabling the macro **MSS_SYS_MDDR_CONFIG_BY_CORTEX** (#define MSS_SYS_MDDR_CONFIG_BY_CORTEX 0) in sys_config.h file located at: *<download_folder>\sf2_code_shadowing_to_ddr_from_emmc_df\sample_application\code_shadow_MSS_CM3_hw_platform\drivers_config\sys_config.h*

Refer to "Appendix 2: Generating Executable Bin File" on page 24 for generating executable *.bin file.

Superseded

eMMC Host PC Loader

The eMMC host PC loader is implemented to transfer the executable target application image (*.bin) from the host PC to SmartFusion2 on-board eMMC device using MMUART_0 interface. The `m2s_emmc_loader.exe` file is executed from the host PC command prompt. It is located at: *<download_folder>\sf2_code_shadowing_to_ddr_from_emmc_df\emmc_loader*.

The syntax is:

```
m2s_emmc_loader.exe <*.bin> <COM Port number>
```

where *.bin is the executable target application image.

For more information, refer to "Running the Demo Design for Multi-Stage Boot Process Method" section on page 13.

Setting Up the Demo Design

1. Connect the FlashPro4 programmer to the J59 connector of SmartFusion2 SoC FPGA Development Kit.
2. Connect one end of the USB mini-B cable to the J24 connector provided on the SmartFusion2 SoC FPGA Development Kit.
3. Connect the other end of the USB cable to the host PC.

   Make sure that the USB to UART bridge drivers are automatically detected (can be verified in the Device Manager), as shown in Figure 6.
4. From the detected four COM ports, select the one with its location on **Properties** window as **on USB Serial Converter D**. Make a note of the COM port number for serial port configuration and ensure that the COM port location is specified as **on USB serial Converter D**, as shown in Figure 6.

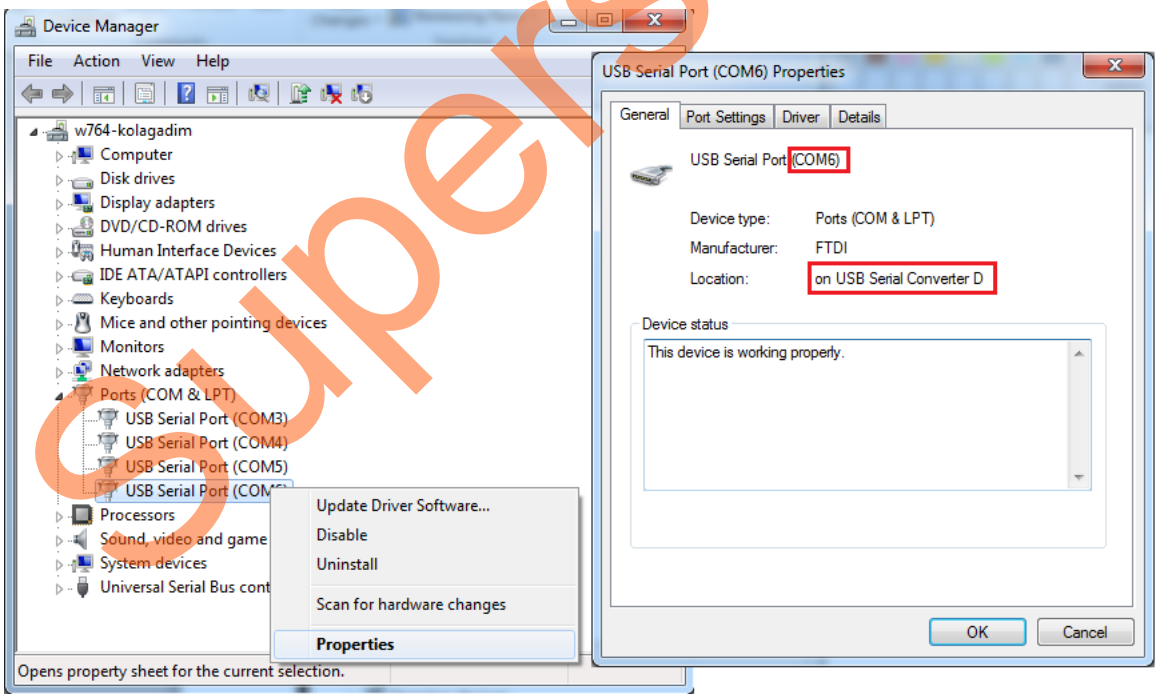

Figure 6 • **USB to UART Bridge Drivers**

5. If USB to UART bridge drivers are not installed, download and install the drivers from www.microsemi.com/soc/documents/CDM_2.08.24_WHQL_Certified.zip.

Superseded

6. Connect the jumpers on the SmartFusion2 SoC FPGA Development Kit, as shown in Table 2. While making the jumper connections, the power supply switch **SW7** on the board should be in **OFF** position.

Table 2 • **SmartFusion2 SoC FPGA Development Kit Jumper Settings**

Jumper Number	Settings	Notes
J70, J93, J94, J117, J123, J142, J157, J160, J167, J225, J226, J227	1-2 closed	These are the default jumper settings of the Development Kit board. Make sure these jumpers are set accordingly.
J2	1-3 closed	
J23	2-3 closed	
J129, J133	2-3 closed	These are the MMUART_0 jumper settings for serial communication.

7. Connect the power supply to the J18 connector.

Figure 7 shows the board setup for running the code shadowing from eMMC to DDR3 demo on the SmartFusion2 SoC FPGA Development Kit.

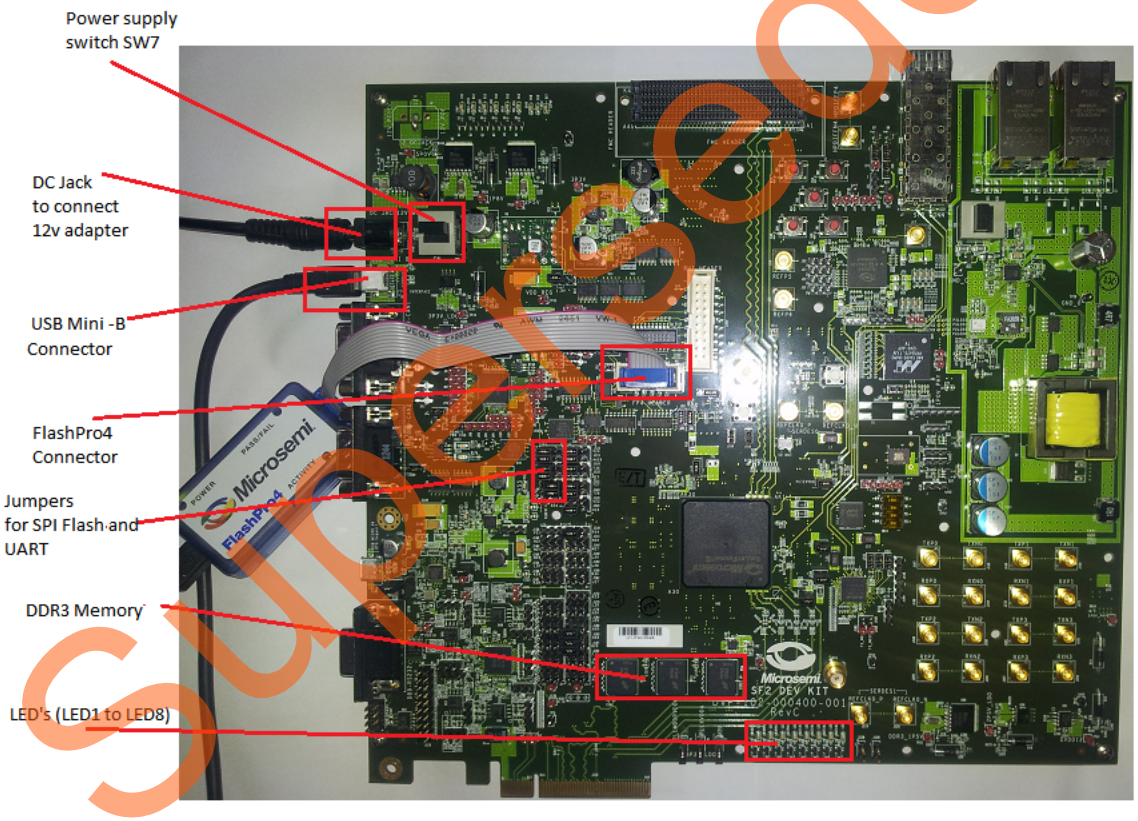

Figure 7 • **SmartFusion2 SoC FPGA Development Kit Setup**

Superseded

Running the Demo Design for Multi-Stage Boot Process Method

1. Download the demo design from: http://soc.microsemi.com/download/rsc/?f=SF2_CODE_SHADOWING_DDR_EMMC_DF
2. Start any serial terminal emulation program such as:
   - HyperTerminal
   - PuTTY
   - Tera Term

   The configuration for the program is:
   - Baud Rate: 57600
   - Eight data bits
   - One stop bit
   - No Parity
   - No flow control

   For more information on configuring the serial terminal emulation programs, refer to the *Configuring Serial Terminal Emulation Programs Tutorial*.
3. Switch **ON** the **SW7** power supply switch.
4. Launch the FlashPro software.
5. Click **New Project**.

Superseded

6. In the **New Project** window, type the project name.

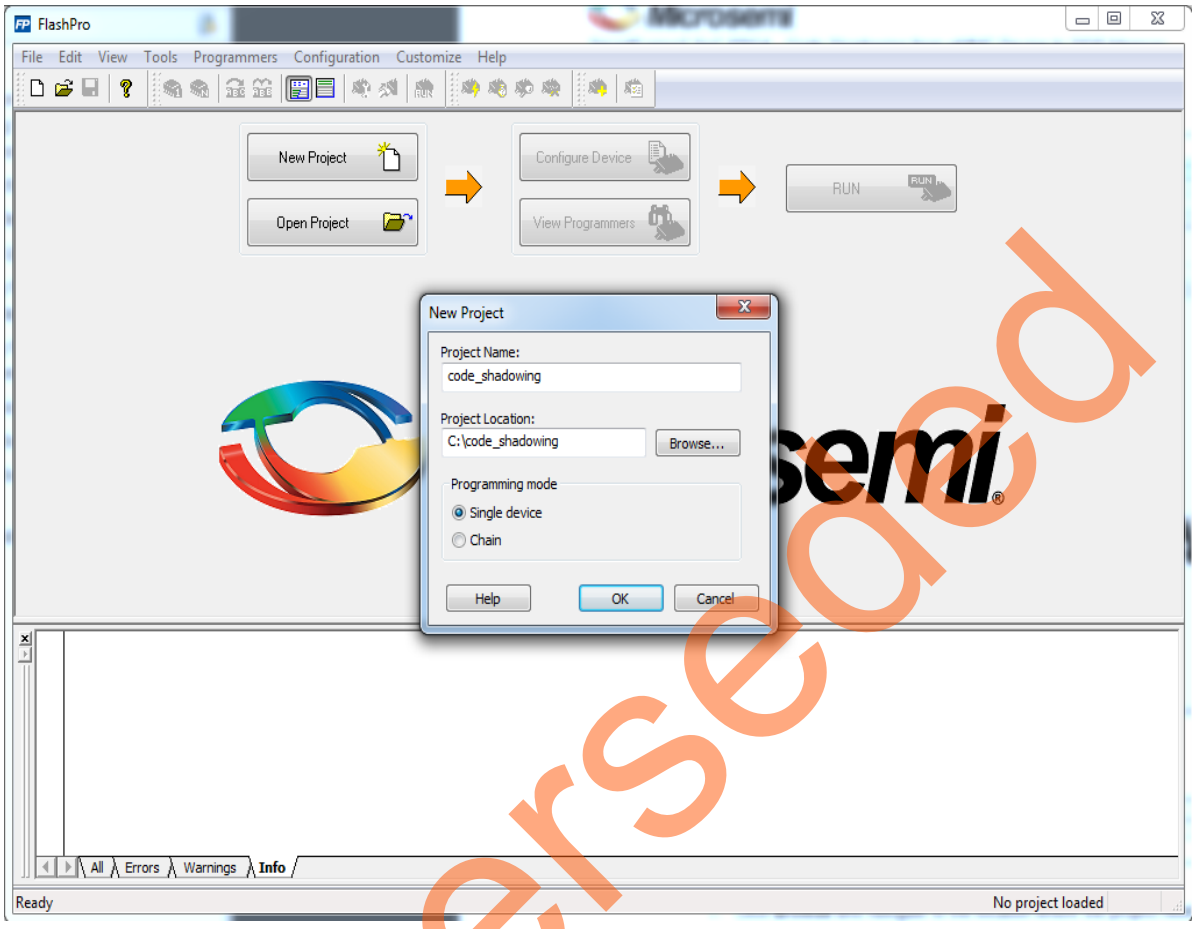

Figure 8 • **FlashPro New Project**

7. Click **Browse** and navigate to the location where the project needs to be saved.
8. Select **Single device** as the **Programming mode**.
9. Click **OK** to save the project.
10. Click **Configure Device**.

Superseded

11. Click **Browse** and navigate to the location where the `code_shadow_top.stp` file is located and select the file. The default location is: *<download_folder>\sf2_code_shadowing_to_ddr_from_emmc_df\stapl_programming_file\MultiStageBoot_method*. The required programming file is selected and is ready to be programmed in the device.

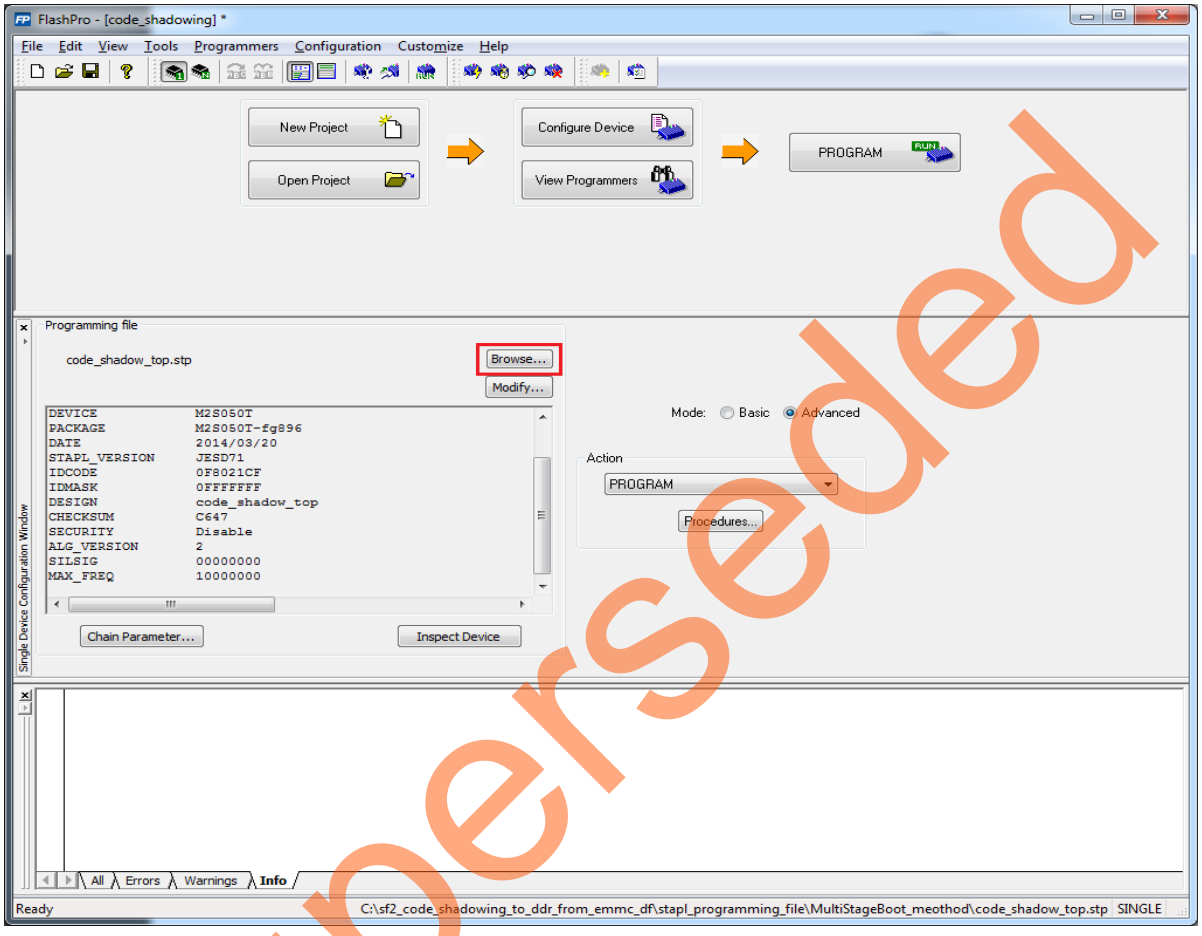

Figure 9 • **FlashPro Project Configured**

Superseded

12. Click **PROGRAM** to start programming the device.

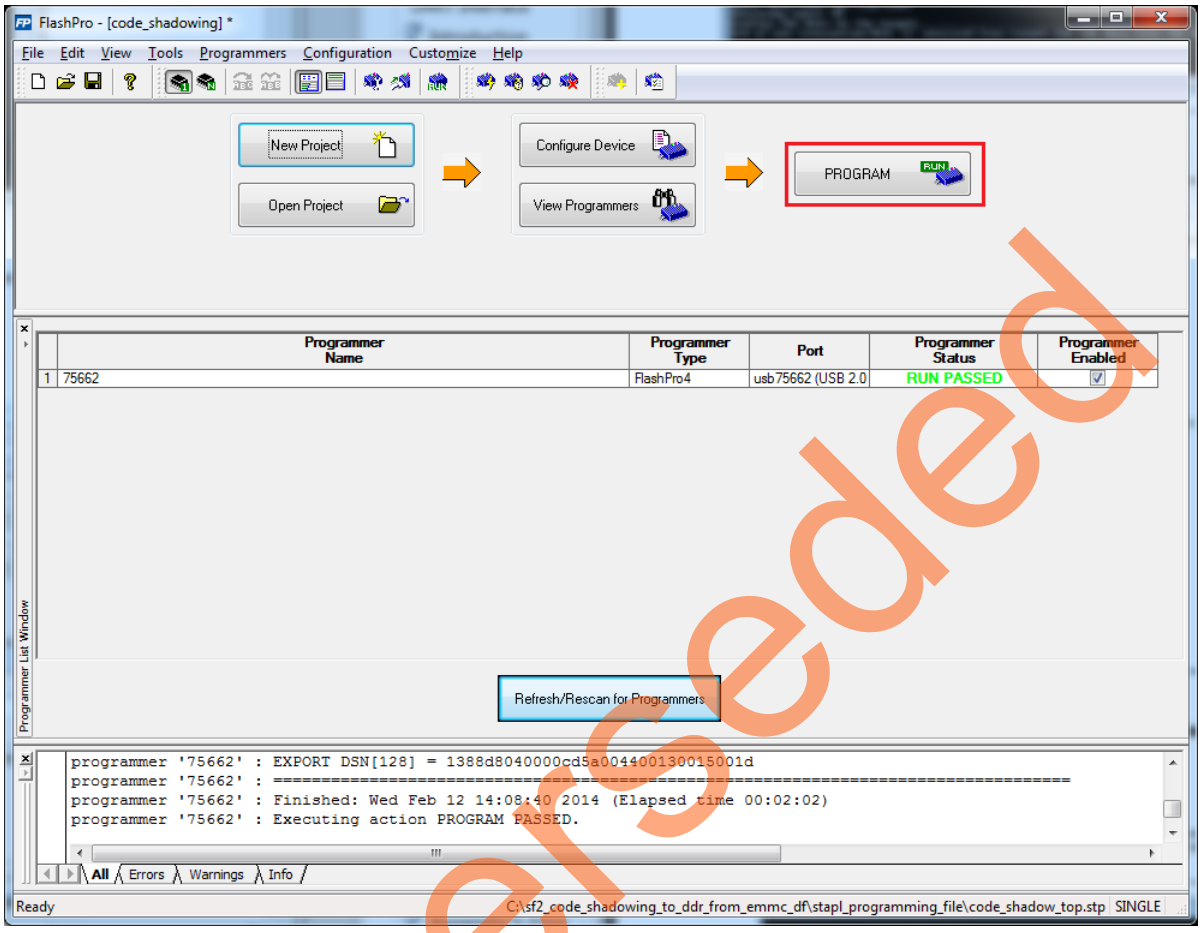

Figure 10 • **Programming the Device**

Superseded

13. Wait until a message is displayed, indicating that the program has passed. This demo requires the SmartFusion2 device to be programmed with the application, which runs from eNVM, to perform the code shadowing from eMMC to DDR3 memory. So, the SmartFusion2 device is programmed with the `code_shadow_top.stp` programming file, which has the application image embedded in the eNVM client.

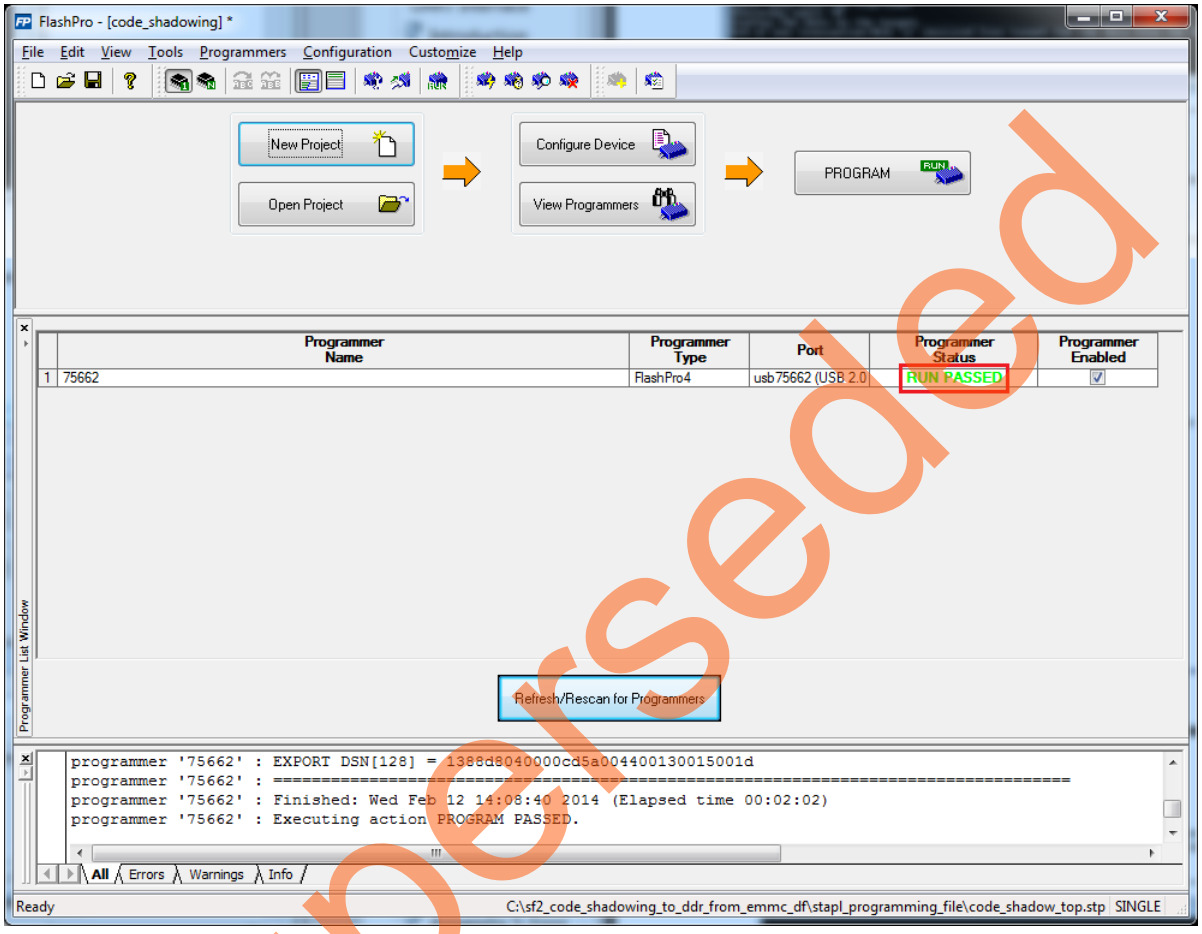

Figure 11 • **FlashPro Program Passed**

14. Power Cycle the SmartFusion2 Development Kit board.

Superseded

The serial terminal emulation program displays the options as shown in Figure 12.

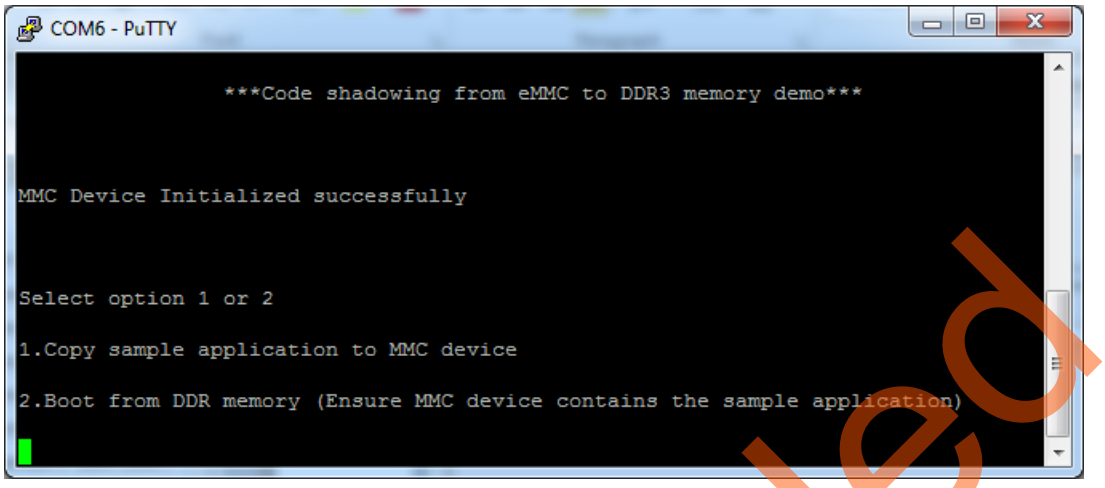

Figure 12 • **Demo Options**

15. Type **1** to copy the sample application image (*.bin) file to the eMMC device from host PC using UART interface.
16. Close the serial terminal emulation program window to copy the sample application image to SmartFusion2 eMMC device using COM port.

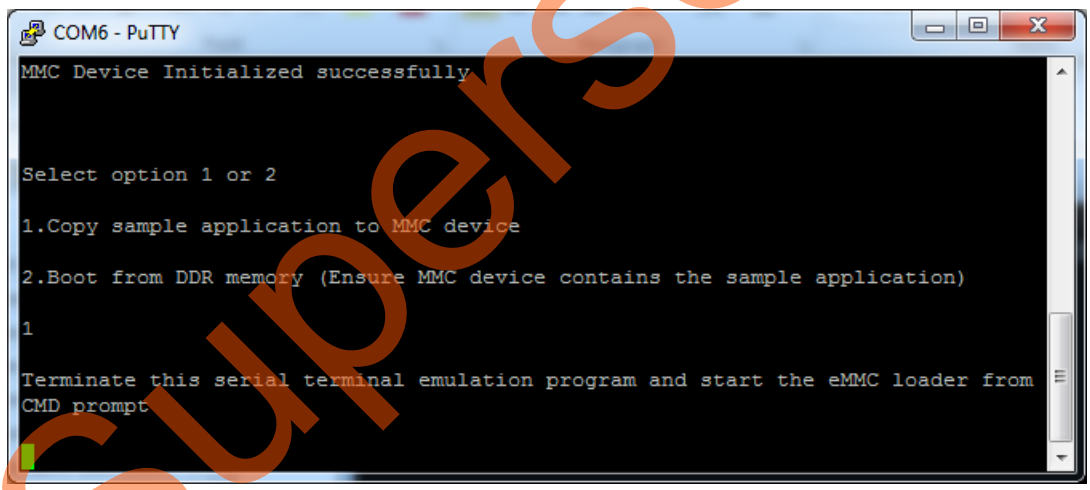

Figure 13 • **Selecting Option 1**

17. Open the command prompt in the host PC.
18. Navigate to the directory, where the eMMC host PC loader (`m2s_emmc_loader.exe`) is located. The default location is: *<download_folder>\sf2_code_shadowing_to_ddr_from_emmc_df\emmc_loader*.

Superseded

19. Run the `m2s_emmc_loader.exe` file and launch the eMMC host PC loader to transfer the sample application image file from host PC to SmartFusion2 on board eMMC device using UART interface.

    Command: `m2s_emmc_loader.exe sample_app.bin 6`
    where, `sample_app.bin` is the target executable application image and 6 is the COM port number.

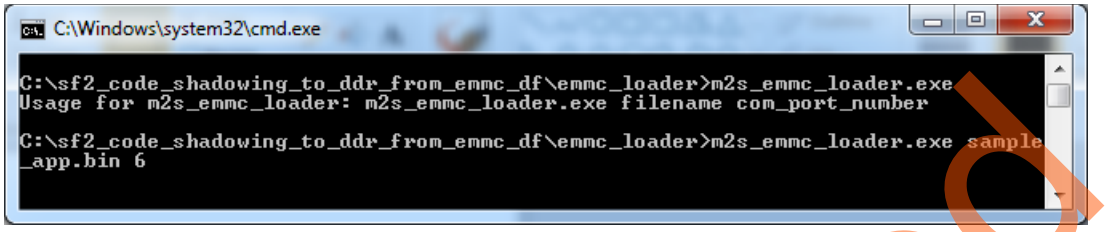

Figure 14 • **Command Prompt Window**

On successful file transfer using UART interface, the command prompt window displays message as shown in Figure 14.

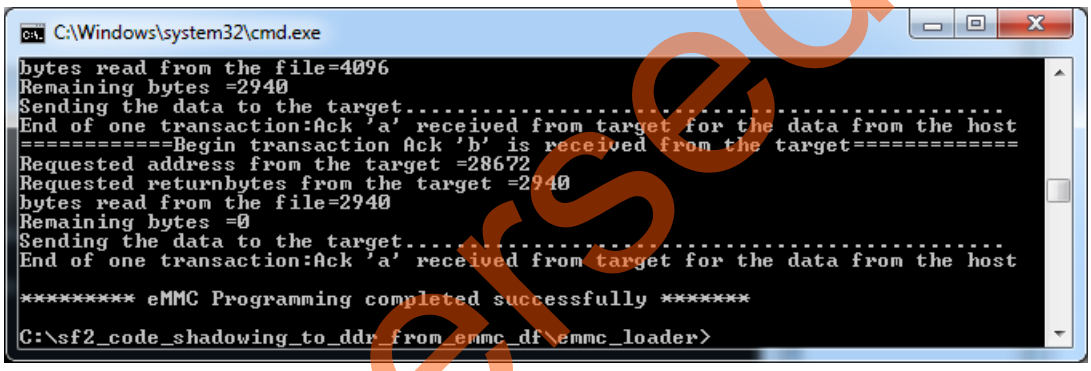

Figure 15 • **eMMC Program Successful**

The SmartFusion2 eMMC device is loaded with the executable application image.

Note: If the command prompt window shows any handshake error message, stop the command prompt by pressing **Ctrl+C**. Press **SW9** to reset the SmartFusion2 Development Kit board and go to Step 12.

20. Start the serial terminal emulation program as mentioned in Step 2. and power cycle the SmartFusion2 Development Kit board.

Superseded

21. Type **2** to boot from DDR memory. The application running from SmartFusion2 eNVM copies the eMMC device contents to DDR memory and runs the bootloader function to jump the execution from current eNVM to DDR memory, as shown in Figure 16.

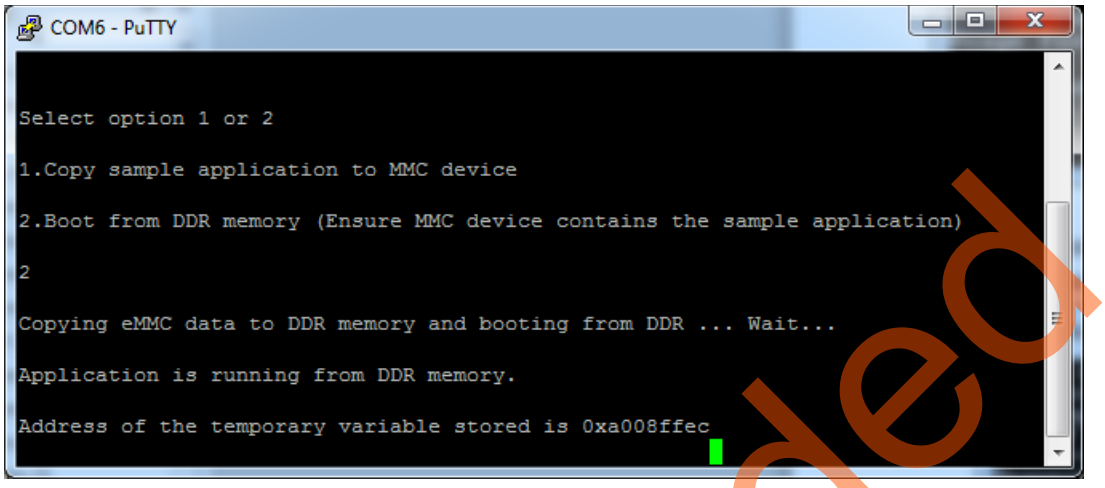

Figure 16 • **Code Running from DDR Memory**

Note: The serial terminal emulation program displays the address of temporary variable (the address may vary at run time), which indicates that the code is running from the DDR3 memory. DDR3 memory starts from the address 0XA0000000.

LEDs blinking (counter increment and decrement logic) can be observed in the SmartFusion2 Development Kit board.

22. Press **SW2** or **SW5** to trigger the interrupt in the application. On pressing SW2 or SW5, the interrupt service routine runs and displays the message on the serial terminal emulation program, as shown in Figure 17.

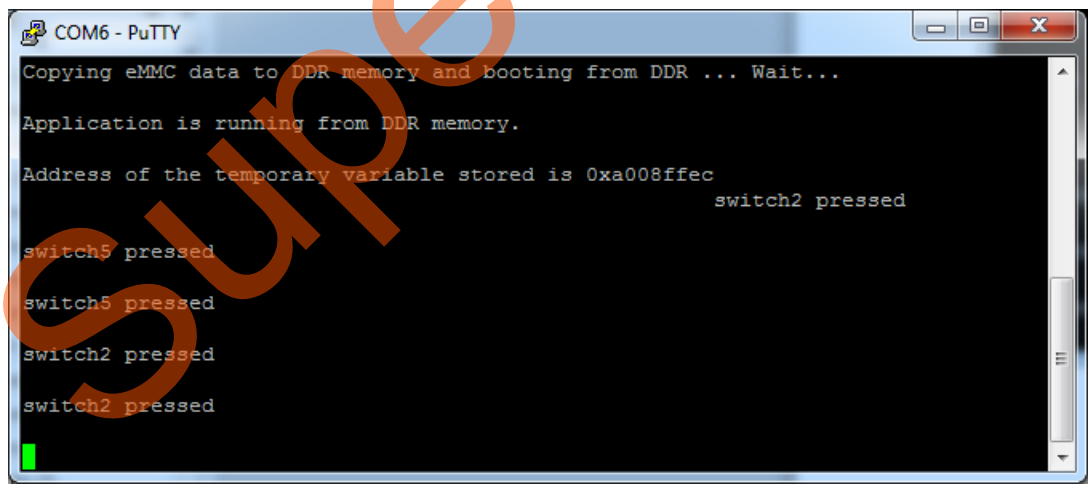

Figure 17 • **Message on Triggering the Interrupts**

Note: The application might display the message two times due to switch de-bounce.

Superseded

Running the Demo Design for Hardware Boot Engine Method

1. Program the SmarFusion2 SoC FPGA device with the programming file provided in the design files,
   <download_folder>\\sf2_code_shadowing_to_ddr_from_emmc_df\stapl_programming_file\HWBootEngine_method \CodeShadowing_Fabric.stp, using the FlashPro design software as described in Step 2. to
   Step 11. in "Running the Demo Design for Multi-Stage Boot Process Method" section on page 13.
2. To program the eMMC device make DIP switch1 to **ON** position. This selection makes the Cortex-M3 to boot from eNVM.
3. Power cycle the SmartFusion2 Development Kit board.
4. Follow Step 15. to Step 19. in "Running the Demo Design for Multi-Stage Boot Process Method" section on page 13 to load the target application image to eMMC memory.
5. After programming the eMMC device successfully as shown Figure 15 on page 19, change DIP switch1 to **OFF** position. This selection makes the boot engine to run from fabric after SmartFusion2 device reset.
6. Power cycle the SmartFusion2 Development Kit board. The boot engine running from fabric copies the target application image from eMMC device to the DDR memory and releases reset to Cortex-M3, which boots the target application image from DDR memory.

   The serial terminal emulation program displays the address of temporary variable (the address may vary in run time) which indicates that the code is running from the DDR3 memory, as shown in Figure 16 on page 20. DDR3 memory starts from the address 0XA0000000.

   LEDs blinking (counter increment and decrement logic) can be observed in the SmartFusion2 Development Kit board.Press SW2 or SW5 to trigger the interrupt in the application. On pressing SW2 or SW5, the Interrupt service routine runs and displays the message on the serial terminal emulation program, as shown in Figure 17 on page 20.

Conclusion

This demonstration shows the capability of SmartFusion2 SoC FPGA device to interface with DDR memory and to run the executable image from the DDR memory by shadowing code from eMMC device.

Superseded

Appendix 1: DDR Configurations

Figure 18, Figure 19, and Figure 20 on page 23 show the DDR3 configuration settings. For more information, refer to *SmartFusion2 SoC FPGA High Speed DDR Interfaces User Guide*

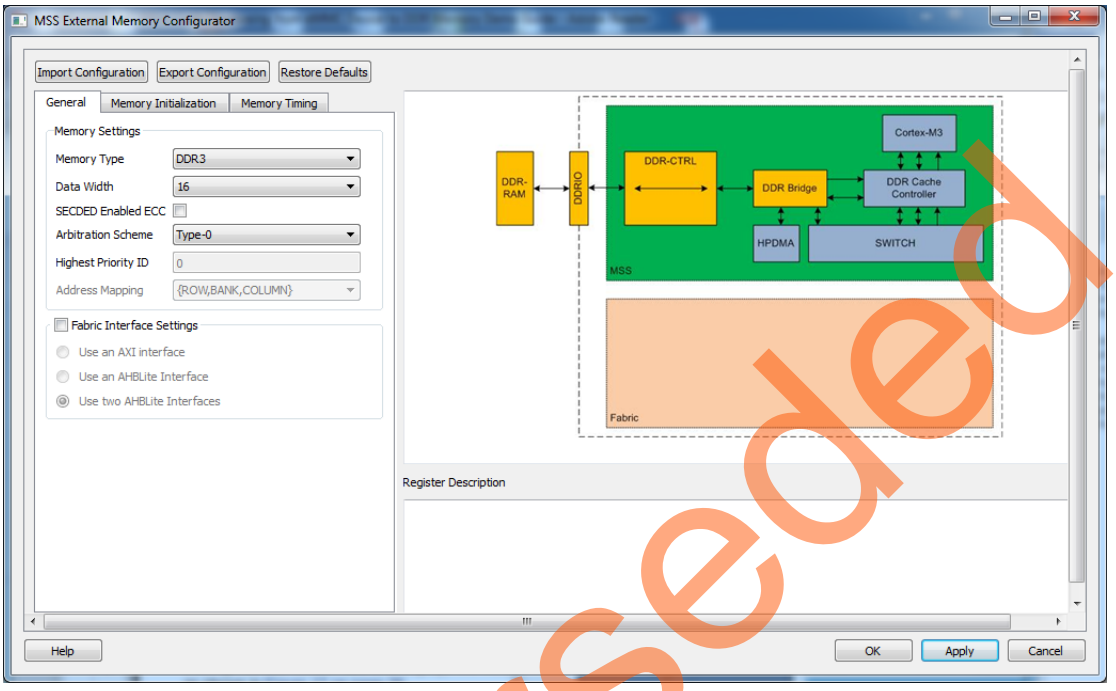

Figure 18 • **General DDR Configuration Settings**

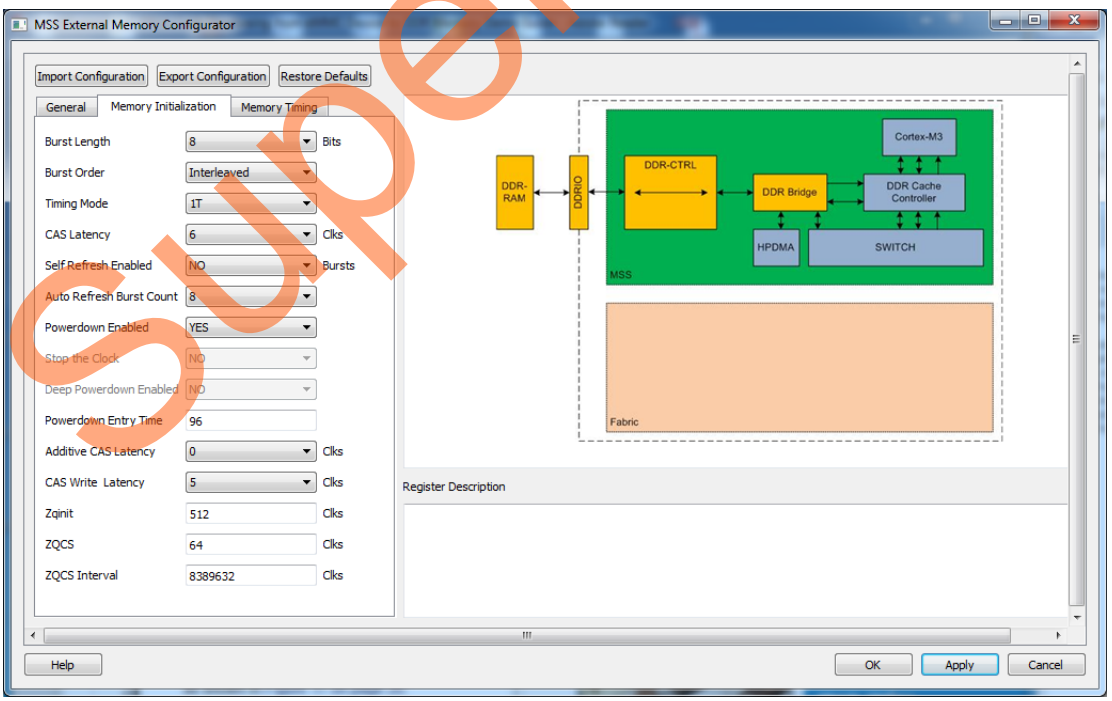

Figure 19 • **DDR Memory Initialization Settings**

Superseded

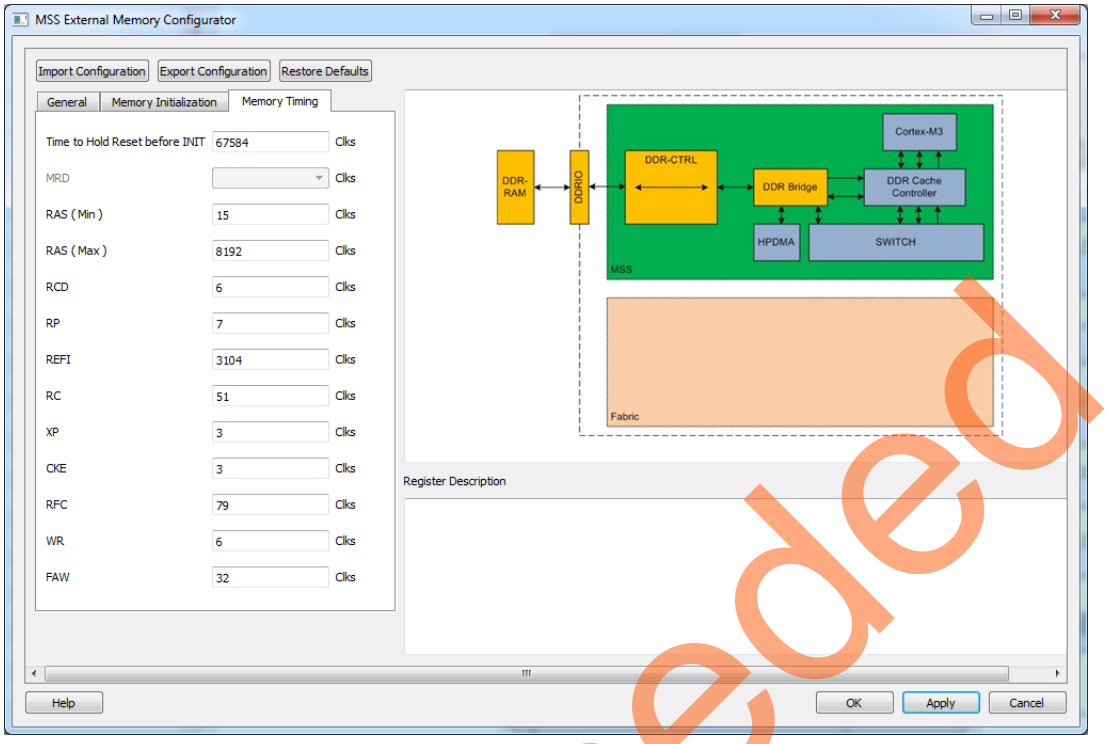

Figure 20 • **DDR Memory Timing Settings**

Superseded

Appendix 2: Generating Executable Bin File

The executable bin file is required to program the eMMC for running the code shadowing demo. To generate the executable bin file from SoftConsole, perform the following steps:

1. Open the SoftConsole project located at: *<download_folder>sf2_code_shadowing_to_ddr_from_emmc_df\sample_application* and build the project in release mode.

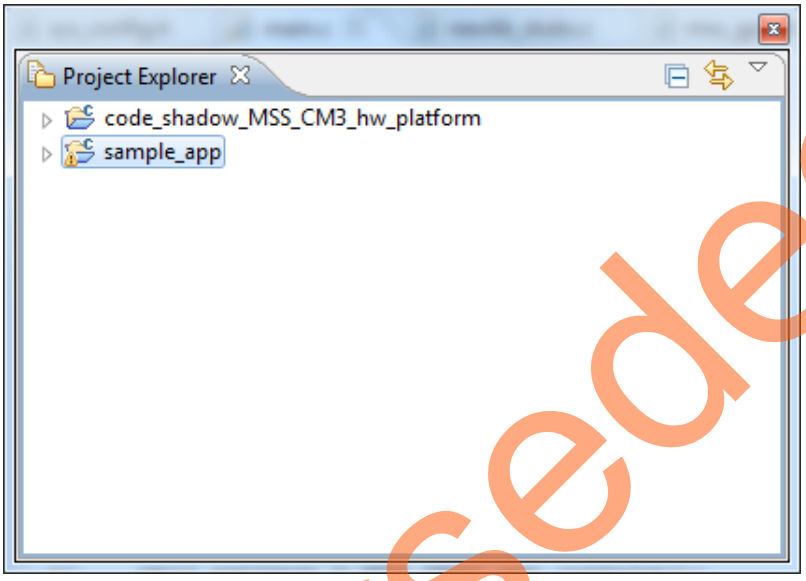

Figure 21 • **Sample Application Project Workspace**

2. Double-click the batch file `Bin-File-Generator.bat` located at: *<download_folder>sf2_code_shadowing_to_ddr_from_emmc_df\sample_application\sample_app* folder, as shown in Figure 22.

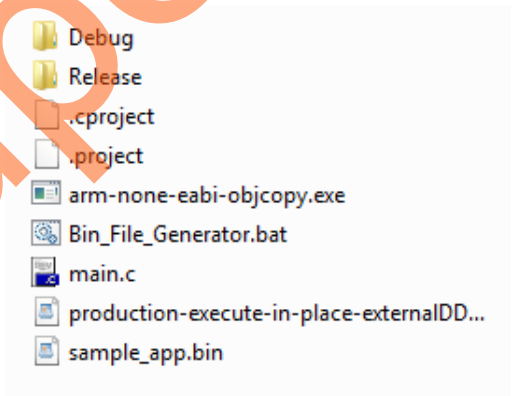

Figure 22 • **Bin File Generator**

The `Bin-File-Generator.bat` file creates `sample_app.bin` file and copies the same to *<download_folder>\sf2_code_shadowing_to_ddr_from_emmc_df\emmc_loader*.

Superseded

Product Support

Microsemi SoC Products Group backs its products with various support services, including Customer Service, Customer Technical Support Center, a website, electronic mail, and worldwide sales offices. This appendix contains information about contacting Microsemi SoC Products Group and using these support services.

Customer Service

Contact Customer Service for non-technical product support, such as product pricing, product upgrades, update information, order status, and authorization.

From North America, call 800.262.1060
From the rest of the world, call 650.318.4460
Fax, from anywhere in the world, 408.643.6913

Customer Technical Support Center

Microsemi SoC Products Group staffs its Customer Technical Support Center with highly skilled engineers who can help answer your hardware, software, and design questions about Microsemi SoC Products. The Customer Technical Support Center spends a great deal of time creating application notes, answers to common design cycle questions, documentation of known issues, and various FAQs. So, before you contact us, please visit our online resources. It is very likely we have already answered your questions.

Technical Support

Visit the Customer Support website (www.microsemi.com/soc/support/search/default.aspx) for more information and support. Many answers available on the searchable web resource include diagrams, illustrations, and links to other resources on the website.

Website

You can browse a variety of technical and non-technical information on the SoC home page, at www.microsemi.com/soc.

Contacting the Customer Technical Support Center

Highly skilled engineers staff the Technical Support Center. The Technical Support Center can be contacted by email or through the Microsemi SoC Products Group website.

Email

You can communicate your technical questions to our email address and receive answers back by email, fax, or phone. Also, if you have design problems, you can email your design files to receive assistance. We constantly monitor the email account throughout the day. When sending your request to us, please be sure to include your full name, company name, and your contact information for efficient processing of your request.

The technical support email address is soc_tech@microsemi.com.

Superseded

My Cases

Microsemi SoC Products Group customers may submit and track technical cases online by going to My Cases.

Outside the U.S.

Customers needing assistance outside the US time zones can either contact technical support via email (soc_tech@microsemi.com) or contact a local sales office. Sales office listings can be found at www.microsemi.com/soc/company/contact/default.aspx.

ITAR Technical Support

For technical support on RH and RT FPGAs that are regulated by International Traffic in Arms Regulations (ITAR), contact us via soc_tech_itar@microsemi.com. Alternatively, within My Cases, select **Yes** in the ITAR drop-down list. For a complete list of ITAR-regulated Microsemi FPGAs, visit the ITAR web page.

Superseded

Superseded

Superseded



Microsemi Corporate Headquarters
One Enterprise, Aliso Viejo CA 92656 USA
Within the USA: +1 (949) 380-6100
Sales: +1 (949) 380-6136
Fax: +1 (949) 215-4996
E-mail: sales.support@microsemi.com

Microsemi Corporation (Nasdaq: MSCC) offers a comprehensive portfolio of semiconductor and system solutions for communications, defense and security, aerospace, and industrial markets. Products include high-performance and radiation-hardened analog mixed-signal integrated circuits, FPGAs, SoCs, and ASICs; power management products; timing and synchronization devices and precise time solutions, setting the world's standard for time; voice processing devices; RF solutions; discrete components; security technologies and scalable anti-tamper products; Power-over-Ethernet ICs and midspans; as well as custom design capabilities and services. Microsemi is headquartered in Aliso Viejo, Calif. and has approximately 3,400 employees globally. Learn more at **www.microsemi.com**.

© 2014 Microsemi Corporation. All rights reserved. Microsemi and the Microsemi logo are trademarks of Microsemi Corporation. All other trademarks and service marks are the property of their respective owners.