SmartFusion2 SoC FPGA Code Shadowing from SPI Flash to DDR Memory - Libero SoC v11.7

DG0386 Demo Guide

Microsemi
Power Matters.™

Contents

Figures

Tables

1 Preface

1.1 Purpose

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.

1.2 Intended Audience

This demo guide is intended for:

- FPGA designers
- Embedded designers
- System-level designers

1.3 References

See the following web page for a complete and up-to-date listing of SmartFusion2 device documentation: *http://www.microsemi.com/products/fpga-soc/soc-fpga/smartfusion2#documentation*

The following documents are referred in this demo guide.

- *UG0331: SmartFusion2 Microcontroller Subsystem User Guide*
- *SmartFusion2 System Builder User Guide*

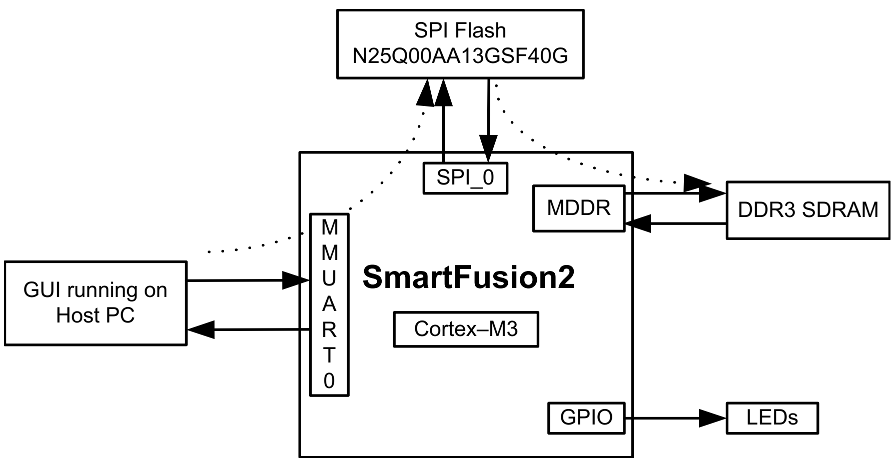
2 SmartFusion2 SoC FPGA - Code Shadowing from SPI Flash to DDR Memory

2.1 Introduction

This demo design shows SmartFusion2 SoC FPGA device capabilities for code shadowing from the serial peripheral interface (SPI) flash memory 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 SPI flash device to DDR memory.

Figure 1 • **Top-Level Block Diagram**



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 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 a maximum speed of 333 MHz. The Cortex-M3 processor can directly run the instructions from external DDR memory through the microcontroller subsystem (MSS) DDR (MDDR). The FPGA cache controller and MSS DDR bridge handle the data flow for a better performance.

2.2 Design Requirements

Table 1 shows the design requirements for this demo.

Table 1 • **Design Requirements**

Design Requirements	Description
Hardware Requirements	
SmartFusion2 Advanced Development Kit: • 12 V adapter • FlashPro5 • USB A to Mini - B USB cable	Rev A or later
Desktop 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)	v11.7
FlashPro Programming Software	v11.7
SoftConsole	v3.4 SP1*
PC Drivers	USB to UART drivers
Microsoft .NET Framework 4 client for launching demo GUI	–
Note: **For this tutorial, SoftConsole v3.4 SP1 is used. For using SoftConsole v4.0, see the TU0546: SoftConsole v4.0 and Libero SoC v11.7 Tutorial.*	

2.3 Demo Design

2.3.1 Introduction

The demo design files are available for download from the following path in the Microsemi website: *http://soc.microsemi.com/download/rsc/?f=m2s_dg0386_liberov11p7_df*

The demo design files include:

- Libero SoC project
- STAPL programming files
- GUI executable
- Sample application images
- Linker scripts
- DDR configuration files
- Readme.txt file

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

2.3.2 Description

This demo design implements code shadowing technique to boot the application image from DDR memory. This design also provides host interface over SmartFusion2 SoC FPGA multi-mode universal asynchronous/synchronous receiver/transmitter (MMUART) to load the target application executable image into SPI flash connected to the MSS SPI0 interface.

The code shadowing is implemented in the following two methods:

1. Multi-stage boot process method using the Cortex-M3 processor
2. Hardware boot engine method using the FPGA fabric

2.3.2.1 Multi-Stage Boot Process Method

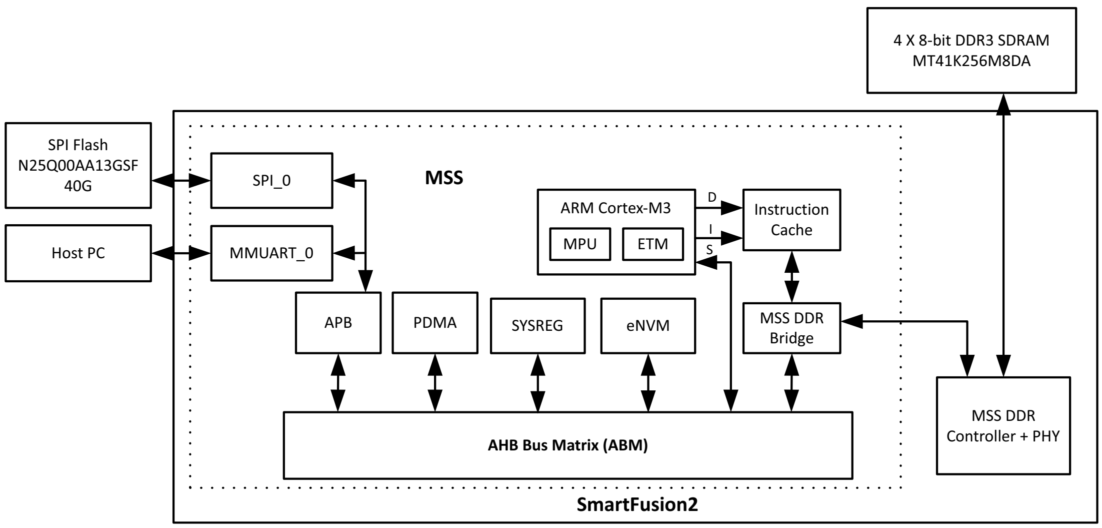
The application image is run from external DDR memories in the following two boot stages:

- The Cortex-M3 processor boots the soft boot loader from embedded non-volatile memory (eNVM), which performs the code image transfer from SPI flash device to DDR memory.
- The Cortex-M3 processor boots the application image from DDR memory.

This design implements a bootloader program to load the target application executable image from SPI flash 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 320 MHz. "Appendix: DDR3 Configurations" on page 22 shows the DDR3 configuration settings. DDR is configured before executing the main application code.

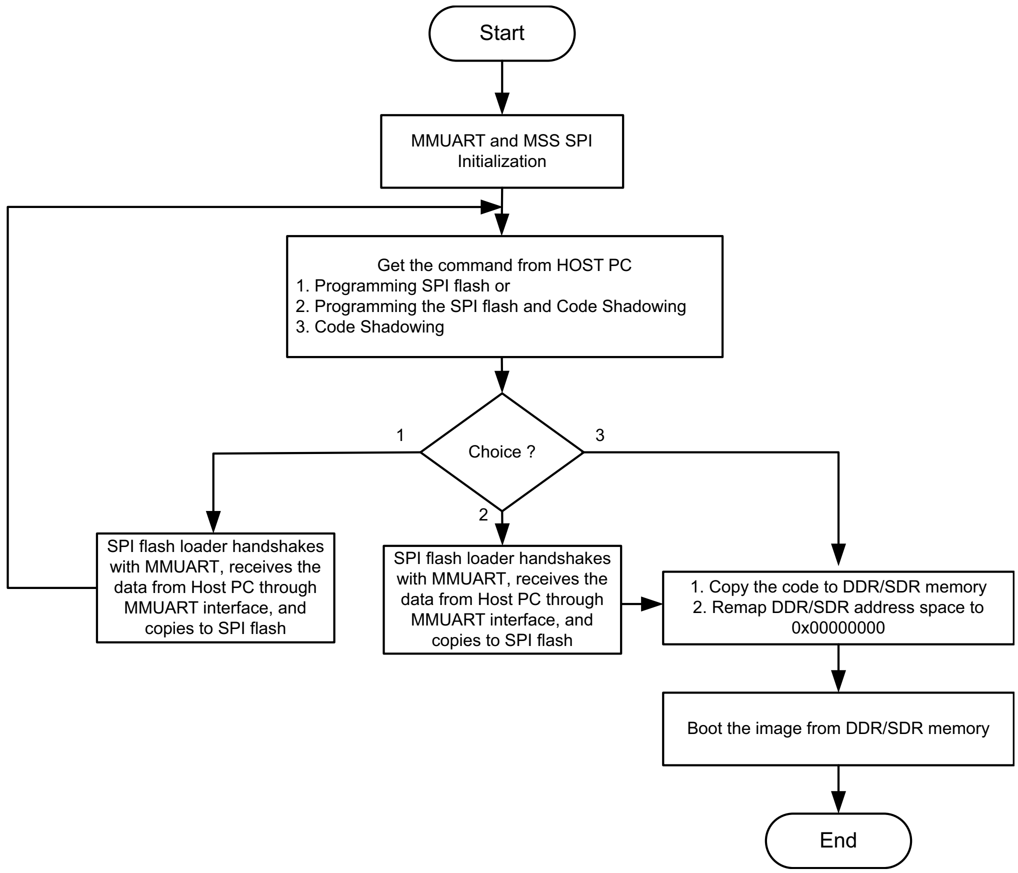
2.3.2.1.1 Bootloader

The bootloader performs the following operations:

1. Copying the target application image from SPI flash memory 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.
4. 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.

Figure 3 shows the demo design.

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



2.3.2.1.2 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 the SPI flash 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 method requires less boot-up time than multi-stage boot process as it avoids multiple boot stages and copies application image to DDR memory in less time.

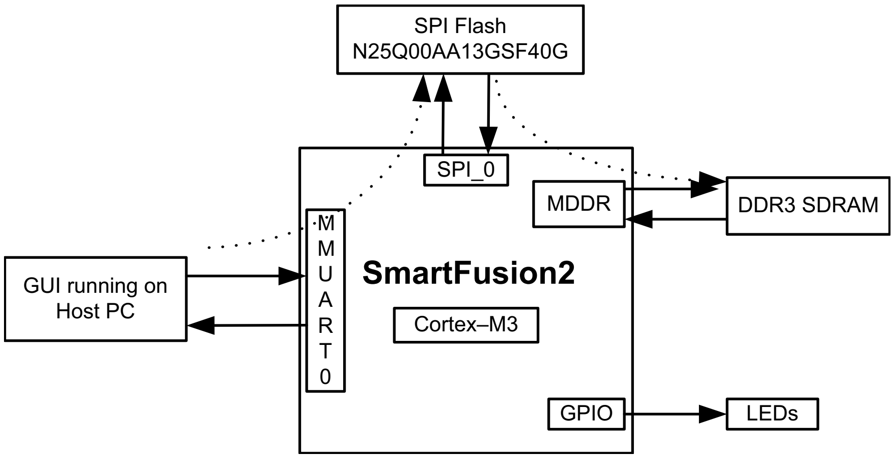
This demo design implements boot engine logic in FPGA fabric to copy the target application executable image from SPI flash to the DDR memory for execution. This design also implements SPI flash loader, which can be executed by Cortex-M3 processor to load the target application executable image into SPI flash device using the provided host interface over SmartFusion2 SoC FPGA MMUART_0. The DIP switch1 on the SmartFusion2 Advanced Development Kit can be used to select whether to program the SPI flash device or to execute the code from DDR memory.

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

The FIC_0 is configured in Slave mode to access the MSS SPI_0 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 shows the detailed block diagram of the demo design.

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



2.3.2.1.3 Boot Engine

This is the major part of the code shadowing demo that copies the application image from the SPI flash 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 SPI flash memory device to DDR memory using the AXI master in the FPGA fabric through the 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.

Figure 5 shows the demo design flow.

Figure 5 • **Top-Level Block Diagram**

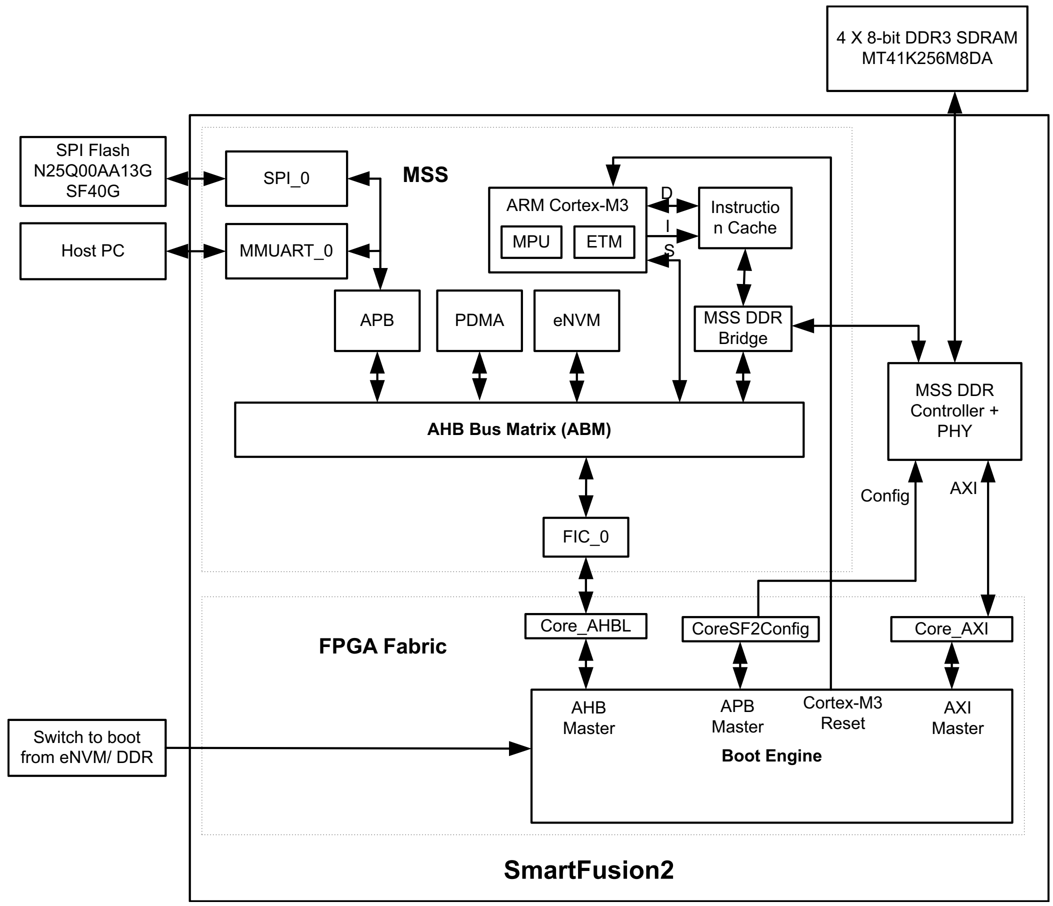
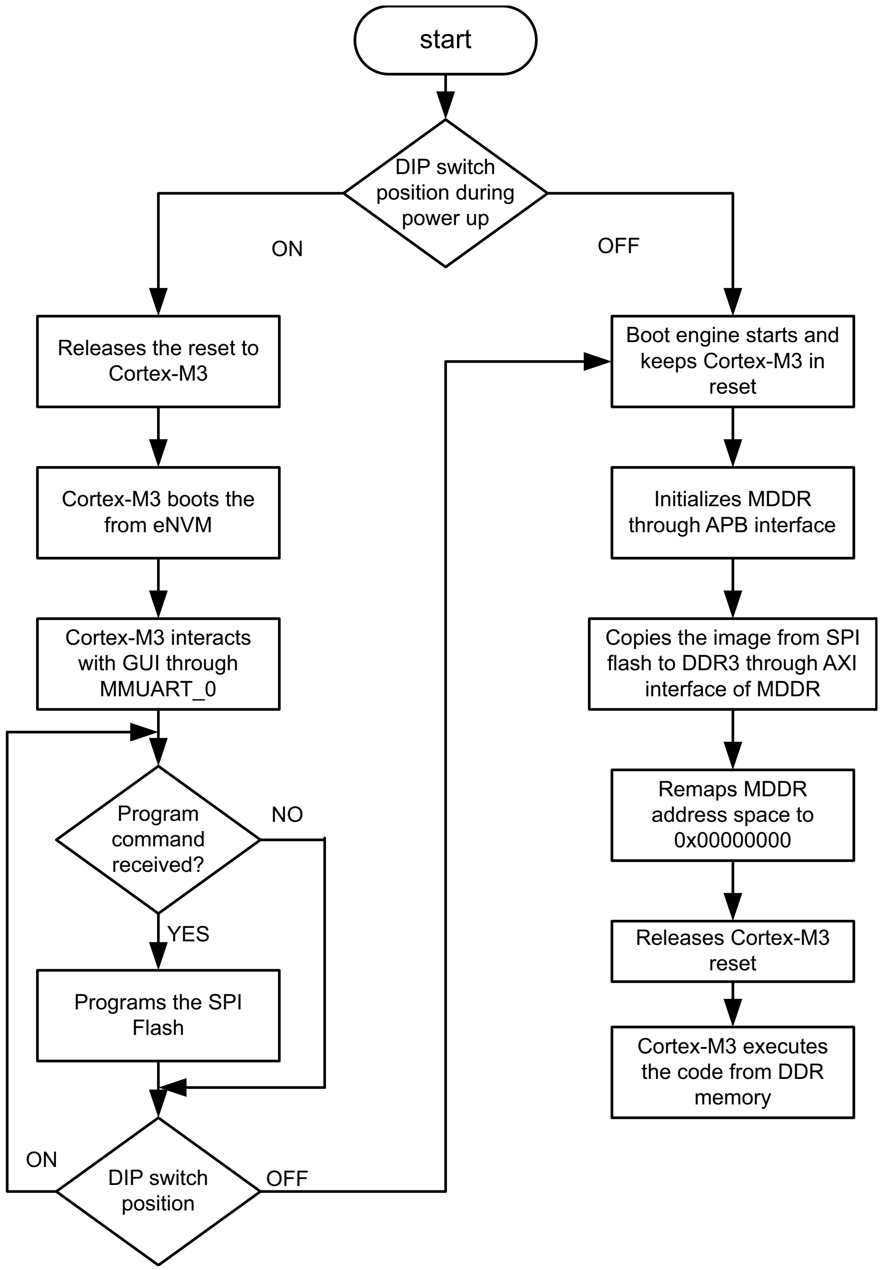


Figure 6 • **Design Flow for Hardware Boot Engine Method**



2.3.3 Creating Target Application Image for DDR Memory

An image that can be executed from the DDR memory is required to run the demo. Use the "production-execute-in-place-externalDDR.ld" linker description file that is included in the design files to build the application image. The linker description file defines the DDR memory starting address as 0x00000000 since the bootloader/boot engine performs the DDR memory remapping from 0xA0000000 to 0x00000000. The linker script creates an application image with instructions, data, and BSS sections in memory whose starting address is 0x00000000. A simple light-emitting diode (LED) blinking, timer and switch based interrupt generation application image file is provided for this demo.

2.3.4 SPI Flash Loader

The SPI flash loader is implemented to load the on-board SPI flash memory with the executable target application image from the host PC through the MMUART_0 interface. The Cortex-M3 processor makes a buffer for the data coming over the MMUART_0 interface and initiates the peripheral DMA (PDMA) to write the buffered data into SPI flash through the MSS_SPI0.

2.4 Running the Demo

The demo shows how to load the application image in the SPI flash and execute that application image from external DDR memories. It provides an example application image "sample_image_DDR3.bin". This image shows the welcome messages and timer interrupt message on the serial console and blinks LED1 to LED8 on the SmartFusion2 Advanced Development Kit. To see the GPIO interrupt messages on the serial console, press **SW2** or **SW3** switch.

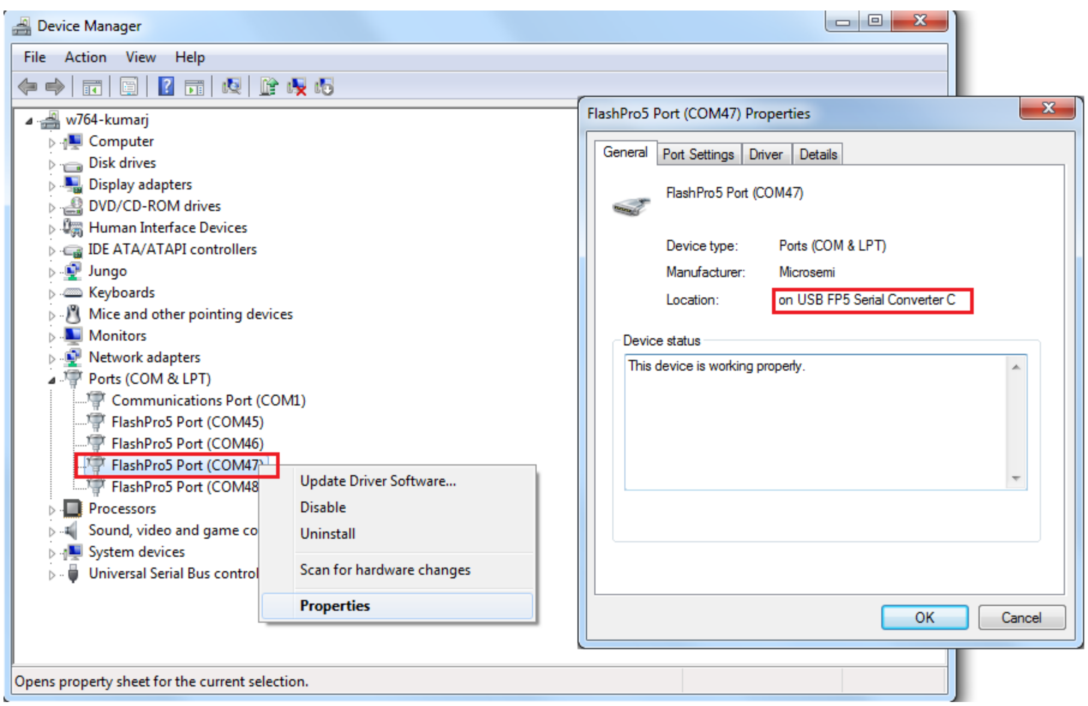
2.5 Setting Up the Demo Design

The following steps describe how to setup the demo for SmartFusion2 Advanced Development Kit board:

1. Connect the Host PC to the J33 Connector using the USB A to mini-B cable. The USB to UART bridge drivers are automatically detected. Verify if the detection is made in the device manager as shown in Figure 7.
2. If USB drivers are not detected automatically, install the USB driver.
3. For serial terminal communication through the FTDI mini USB cable, install the FTDI D2XX driver. Download the drivers and installation guide from:

   http://www.microsemi.com/soc/documents/CDM_2.08.24_WHQL_Certified.zip.

Figure 7 • **USB to UART Bridge Drivers**



4. Connect the jumpers on the SmartFusion2 Advanced Development Kit board, as shown in Table 2.

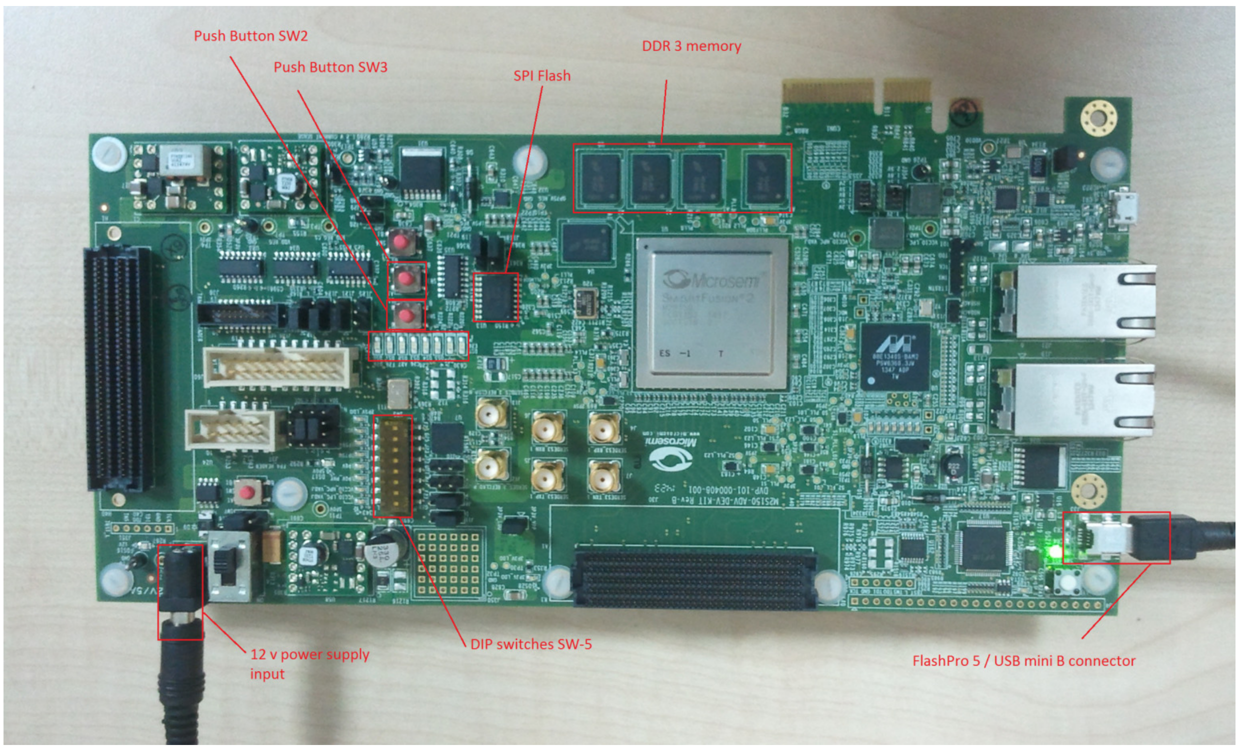
Caution: Switch **OFF** the power supply switch, **SW7** while connecting the jumpers.

Table 2 • **SmartFusion2 Advanced Development Kit Jumper Settings**

Jumper	Pin (From)	Pin (To)	Comments
J116, J353, J354, J54	1	2	These are the default jumper settings of the Advanced Development Kit Board. Make sure these jumpers are set accordingly.
J123	2	3	
J124, J121, J32	1	2	JTAG programming through FTDI
J118, J119	1	2	Programming SPI Flash

5. In the SmartFusion2 Advanced Development Kit, connect the power supply to the J42 connector. Figure 8. shows the board setup for running the code shadowing from SPI flash to DDR3 demo on the SmartFusion2 Advanced Development Kit.

Figure 8 • **SmartFusion2 Advanced Development Kit Setup**

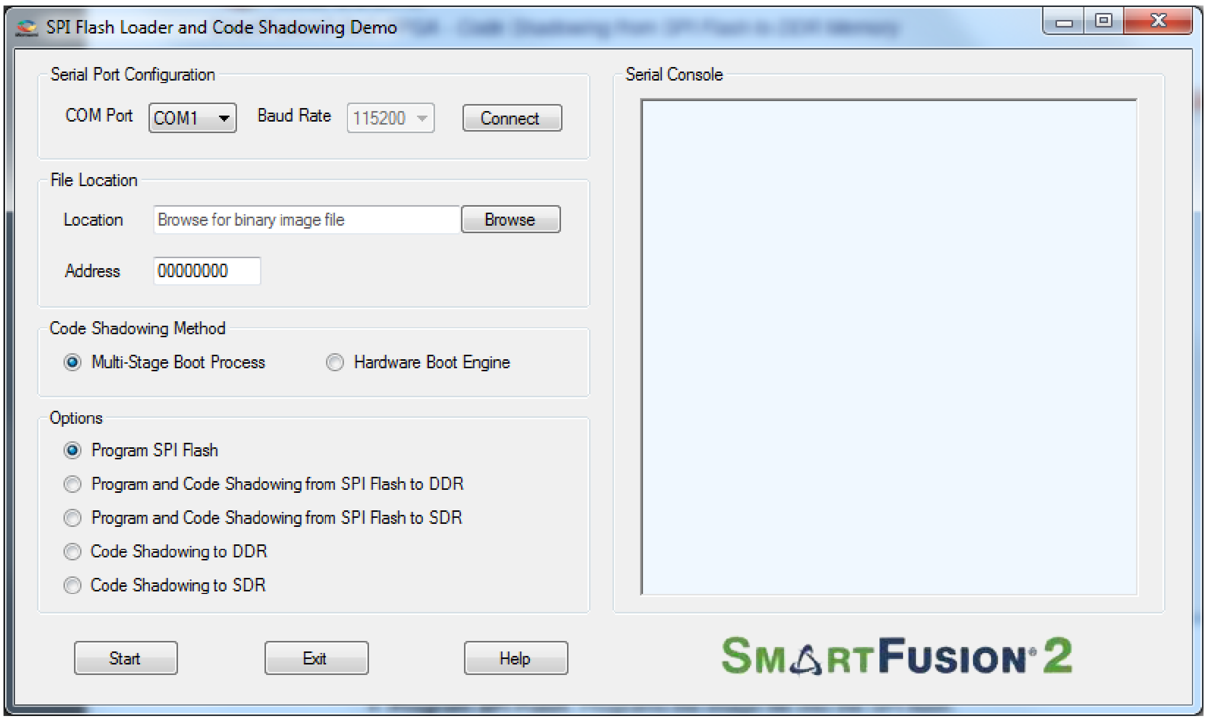


2.5.1 SPI Flash Loader and Code Shadowing Demo GUI

The GUI is required to run the code shadowing demo. SPI Flash Loader and Code Shadowing Demo GUI is a simple graphic user interface that runs on the host PC to program the SPI flash and runs the code shadowing demo on the SmartFusion2 Advanced Development Kit. UART is a communication protocol between the host PC and SmartFusion2 Advanced Development Kit. It also provides the Serial Console section to print the debug messages received from the application over the UART interface.

Figure 9. shows the SPI Flash Loader and Code Shadowing Demo Window.

Figure 9 • **SPI Flash Loader and Code Shadowing Demo Window**



The GUI supports the following features:

- **Program SPI Flash**: Programs the image file into the SPI flash.
- **Program and Code Shadowing from SPI Flash to DDR**: Programs the image file into SPI flash, copies it to the DDR memory, and boots the image from the DDR memory.
- **Program and Code Shadowing from SPI Flash to SDR**: Programs the image file into SPI flash, copies it to the SDR memory, and boots the image from the SDR memory.
- **Code Shadowing to DDR**: Copies the existing image file from SPI flash to the DDR memory and boots the image from the DDR memory.
- **Code Shadowing to SDR**: Copies the existing image file from SPI flash to the SDR memory and boots the image from the SDR memory. Click **Help** for more information on the GUI.

2.5.2 Running the Demo Design for Multi-Stage Boot Process Method

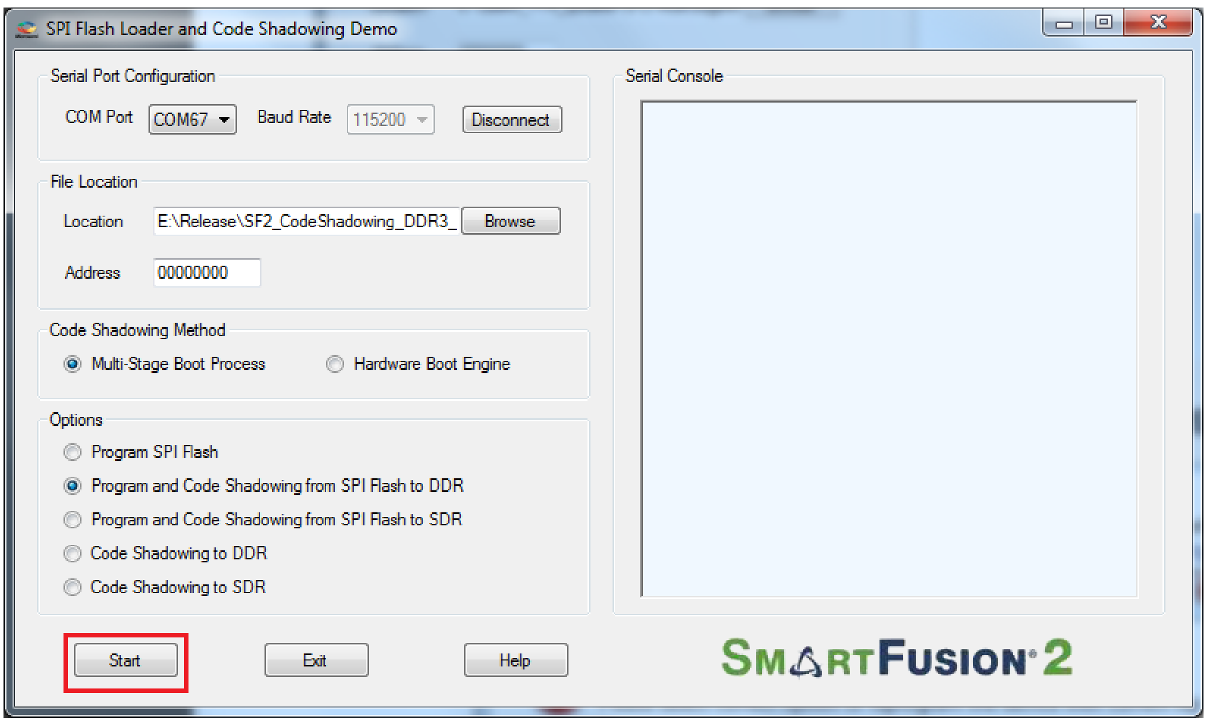
The following steps describe how to run the demo design for multi-stage boot process method:

1. Switch ON the power supply switch, SW7.
2. Program the SmarFusion2 SoC FPGA device with the programming file provided in the design files (*SF2_CodeShadowing_DDR3_DF\Programming Files\MultiStageBoot_meothod\CodeShadowing_top.stp* using the FlashPro design software).
3. Launch the **SPI Flash Loader and Code Shadowing Demo** GUI executable file available in the design files (*SF2_CodeShadowing_DDR3_DF\GUI Executable\SF2_FlashLoader.exe*).
4. Select the appropriate COM port (to which the USB Serial drivers are pointed) from the **COM Port** drop-down list.
5. Click **Connect**. After establishing the connection, **Connect** changes to **Disconnect**.
6. Click **Browse** to select the example target executable image file provided with the design files (*SF2_CodeShadowing_DDR3_DF/Sample Application Images/sample_image_DDR3.bin*).

Note: To generate the application image bin file, see "Appendix: Generating Executable Bin File" on page 25.

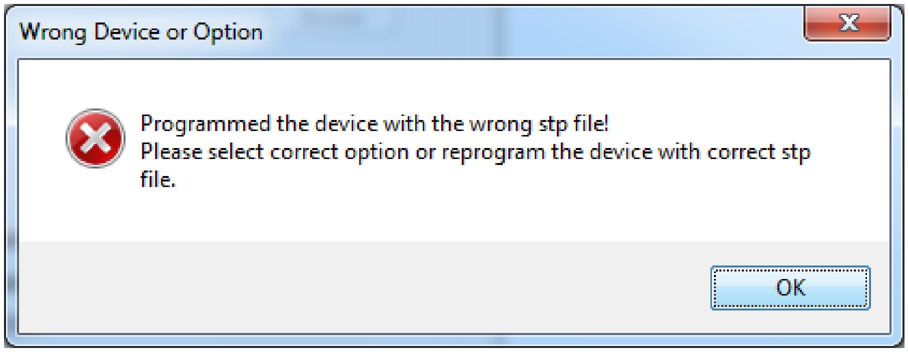
7. Keep the starting address of the SPI flash memory as default at 0x00000000.
8. Select the **Program and Code Shadowing from SPI Flash to DDR** option.
9. Click **Start** as shown in Figure 10 to load the executable image into SPI flash and code shadowing from DDR memory.

Figure 10 • **Starting the Demo**



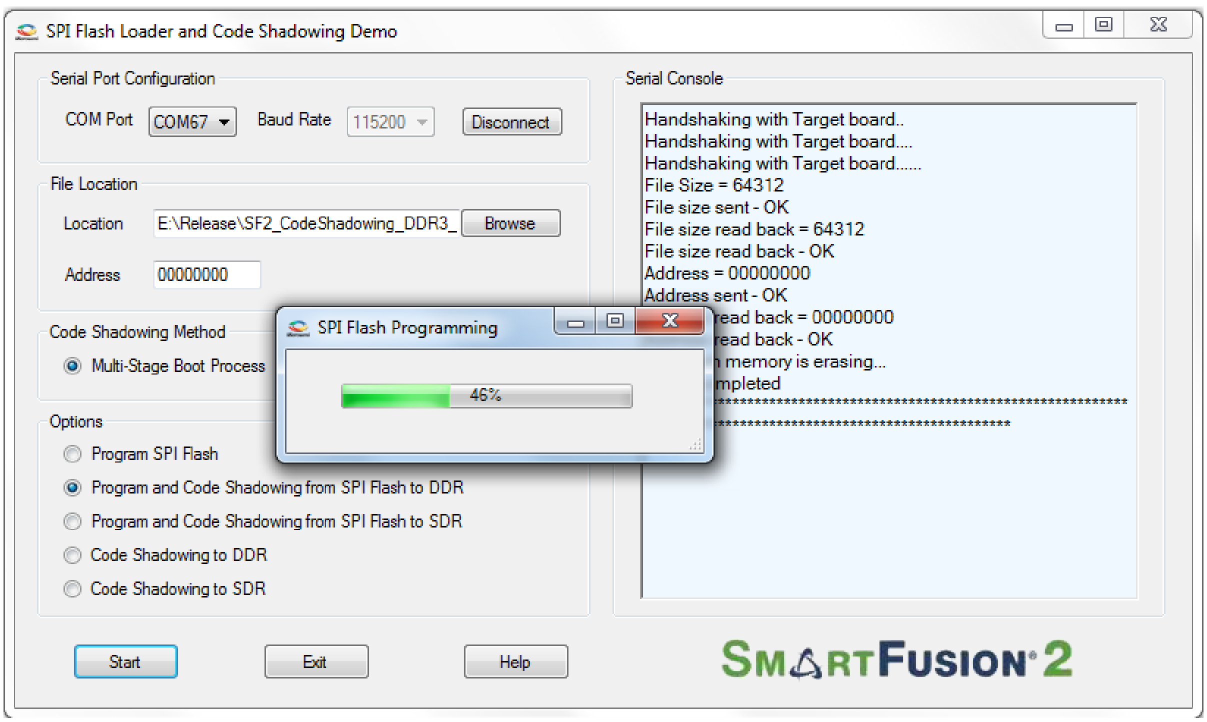
10. If the SmartFusion2 SoC FPGA device is programmed with a STAPL file in which MDDR is not configured for DDR memory then it shows an error message, as shown in Figure 11.

Figure 11 • **Wrong Device or Option Message**



11. The Serial Console section on the GUI shows the debug messages and starts programming SPI flash on successfully erasing the SPI flash. Figure 12 shows the status of SPI flash writing

Figure 12 • **Flash Loading**



12. On programming the SPI flash successfully, the bootloader running on SmartFusion2 SoC FPGA copies the application image from SPI flash to the DDR memory and boots the application image. If the provided image **sample_image_DDR3.bin** is selected, the serial console shows the welcome messages, switch interrupt and timer interrupt messages as shown in Figure 13 on page 18 and Figure 14 on page 18. A running LED pattern is displayed on LED1 to LED8 on the SmartFusion2 Advanced Development Kit.
13. Press **SW2** and **SW3** switches to see interrupt messages on serial console.

Figure 13 • **Running the Target Application Image from DDR3 Memory**

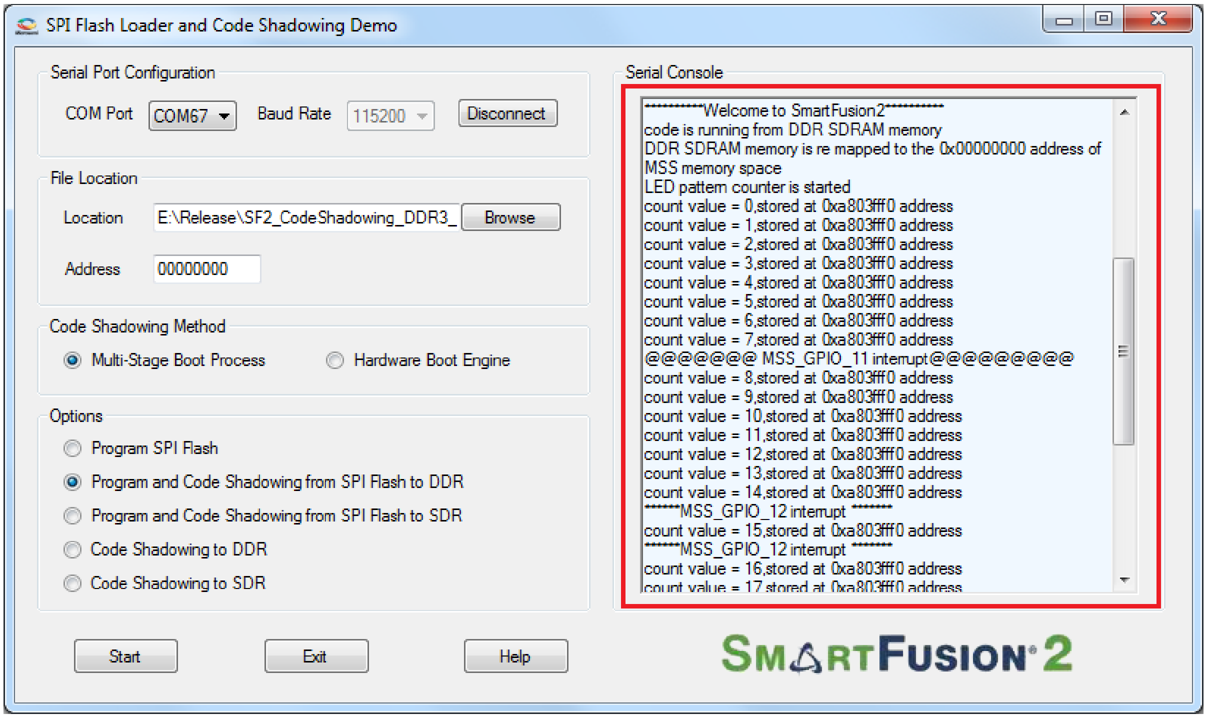
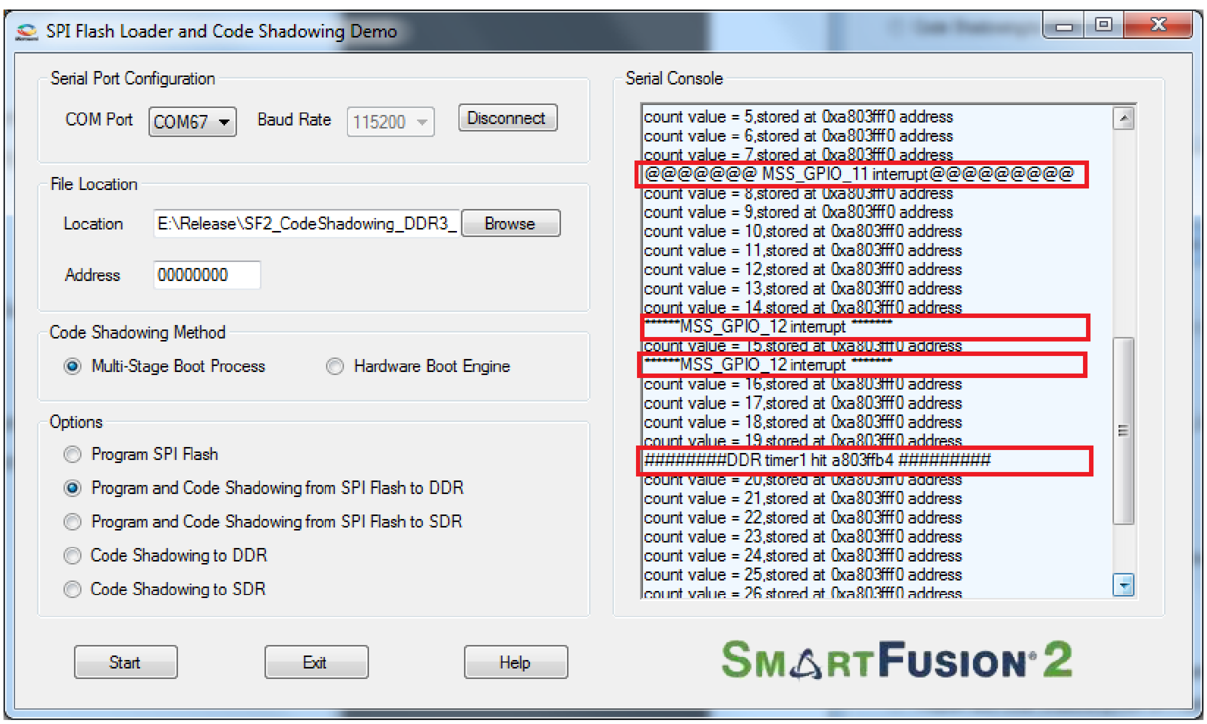


Figure 14 • **Timer and Interrupt Messages in Serial Console**



2.5.3 Running the Hardware Boot Engine Method Design

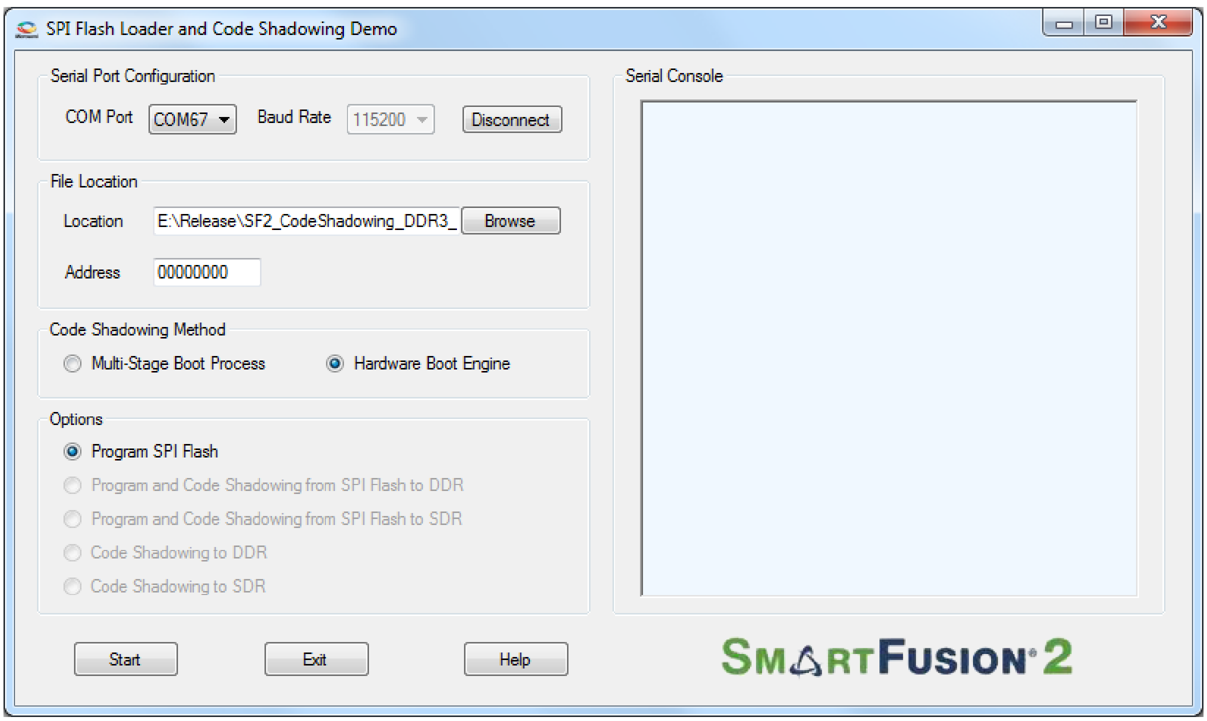
The following steps describe how to run the hardware boot engine method design:

1. Switch ON the power supply switch, SW7.
2. Program the SmarFusion2 SoC FPGA device with the programming file provided in the design files (*SF2_CodeShadowing_DDR3_DF\Programming Files\HWBootEngine_method\CodeShadowing_Fabric.stp* using the FlashPro design software).
3. To program the SPI Flash make DIP switch SW5-1 to **ON** position. This selection makes to boot Cortex-M3 from eNVM. Press **SW6** to reset the SmartFusion2 device.
4. Launch the **SPI Flash Loader and Code Shadowing Demo** GUI executable file available in the design files (*SF2_CodeShadowing_DDR3_DF\GUI Executable\SF2_FlashLoader.exe*).
5. Select the appropriate COM port (to which the USB Serial drivers are pointed) from the COM Port drop-down list.
6. Click **Connect**. After establishing the connection, **Connect** changes to **Disconnect**.
7. Click **Browse** to select the example target executable image file provided with the design files (*SF2_CodeShadowing_DDR3_DF/Sample Application Images/sample_image_DDR3.bin*).

Note: To generate the application image bin file, see "Appendix: Generating Executable Bin File" on page 25.

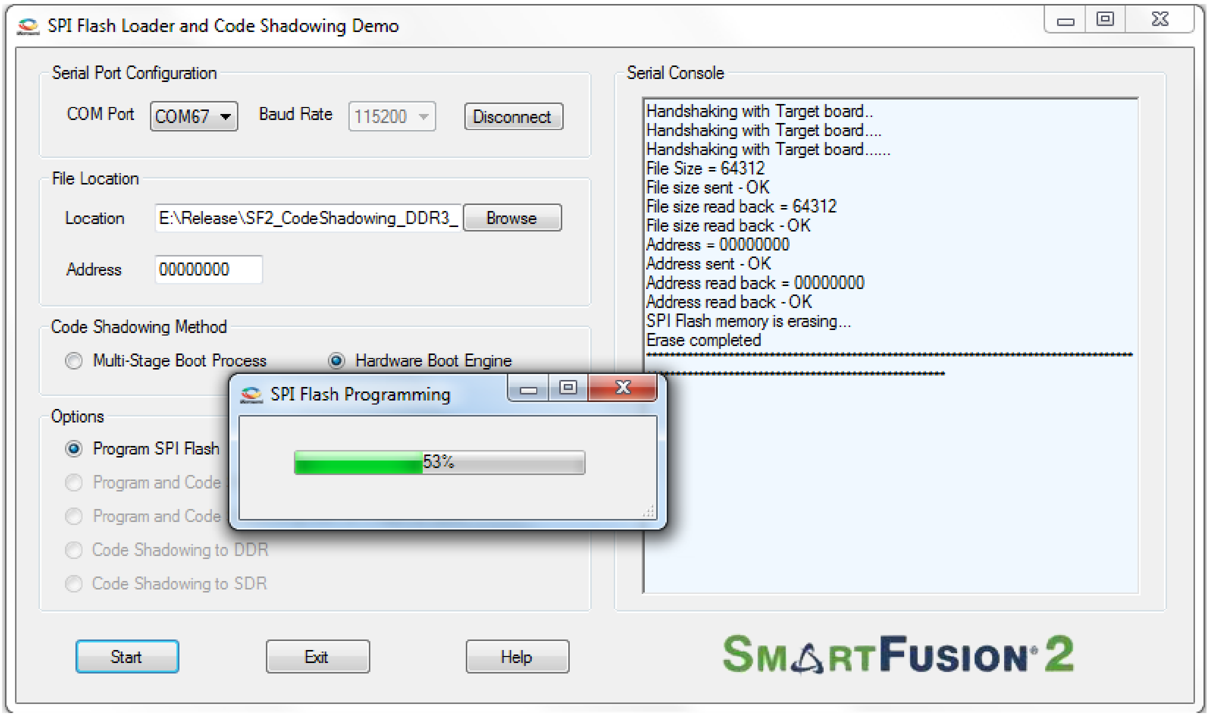
8. Select **Hardware Boot Engine** option in **Code Shadowing Method**.
9. Select the **Program SPI Flash** option from **Options** menu.
10. Click **Start**, as shown in Figure 15 to load the executable image into SPI flash.

Figure 15 • **Starting the Demo**



11. The Serial Console section on the GUI shows the debug messages and the status of SPI flash writing, as shown in Figure 16.

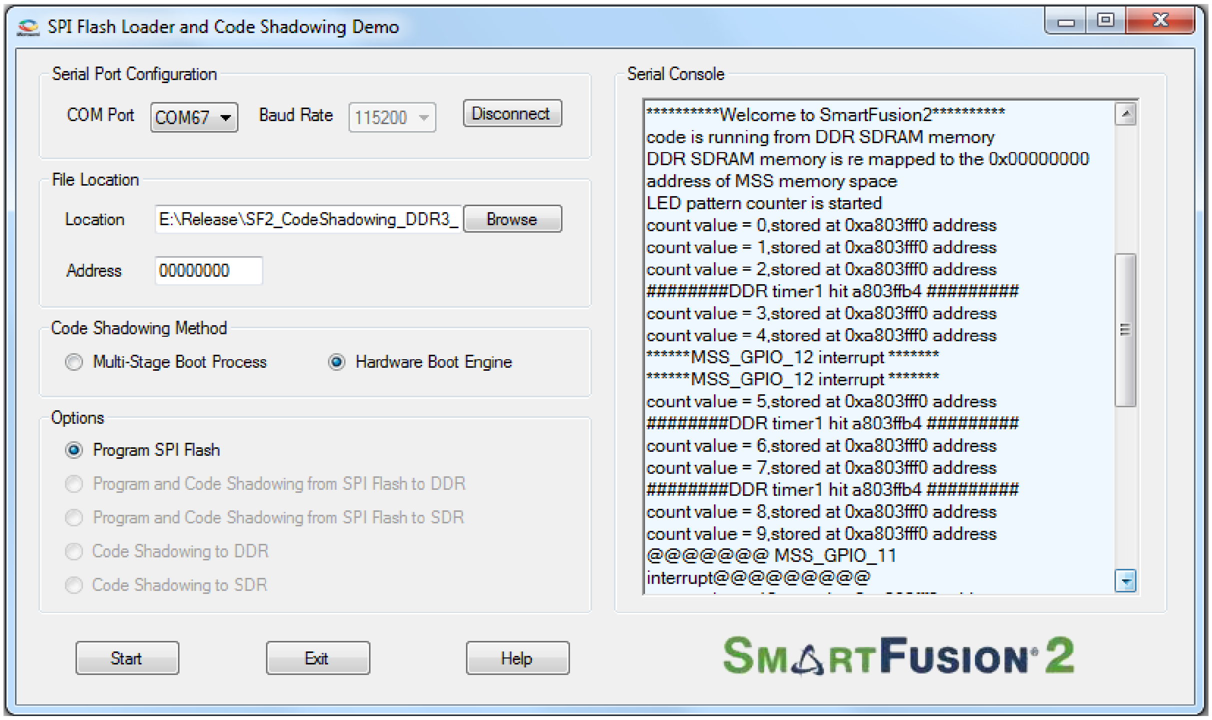
Figure 16 • **Flash Loading**



12. After programming the SPI flash successfully, change DIP switch SW5-1 to **OFF** position. This selection makes to boot the Cortex-M3 processor from DDR memory.

13. Press **SW6** to reset the SmartFusion2 device. The boot engine copies the application image from SPI flash to the DDR memory and releases reset to Cortex-M3, which boots the application image from DDR memory. If the provided image "sample_image_DDR3.bin" is loaded to SPI flash, the serial console shows the welcome messages, switch interrupt (press **SW2** or **SW3**) and timer interrupt messages as shown in Figure 17 and a running LED pattern is displayed on LED1 to LED8 on the SmartFusion2 Advanced Development Kit.

Figure 17 • **Running the Target Application Image from DDR3 Memory**



2.6 Conclusion

This demo 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 SPI flash memory device. It also shows two methods of code shadowing implementation on the SmartFusion2 device.

3 Appendix: DDR3 Configurations

The following figures show the DDR3 configuration settings.

Figure 18 • **General DDR Configuration Settings**

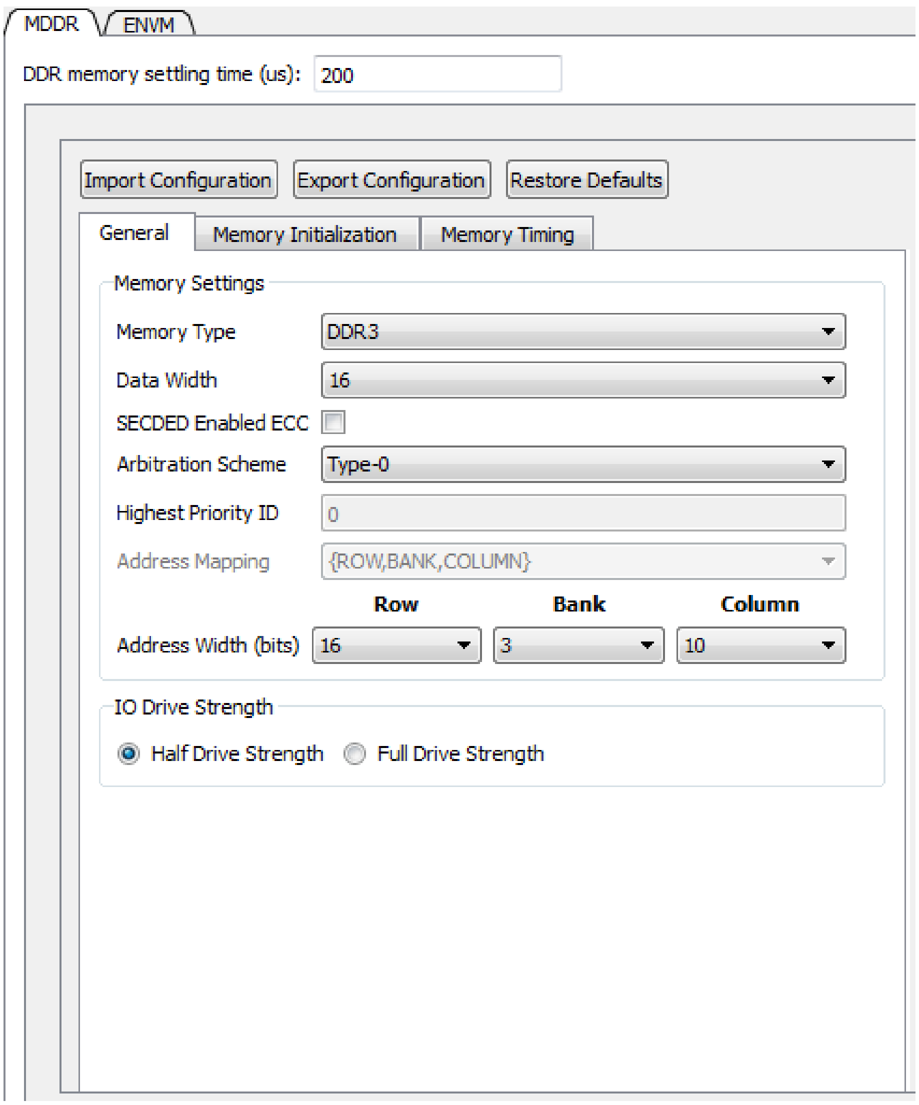


Figure 19 • **DDR Memory Initialization Settings**

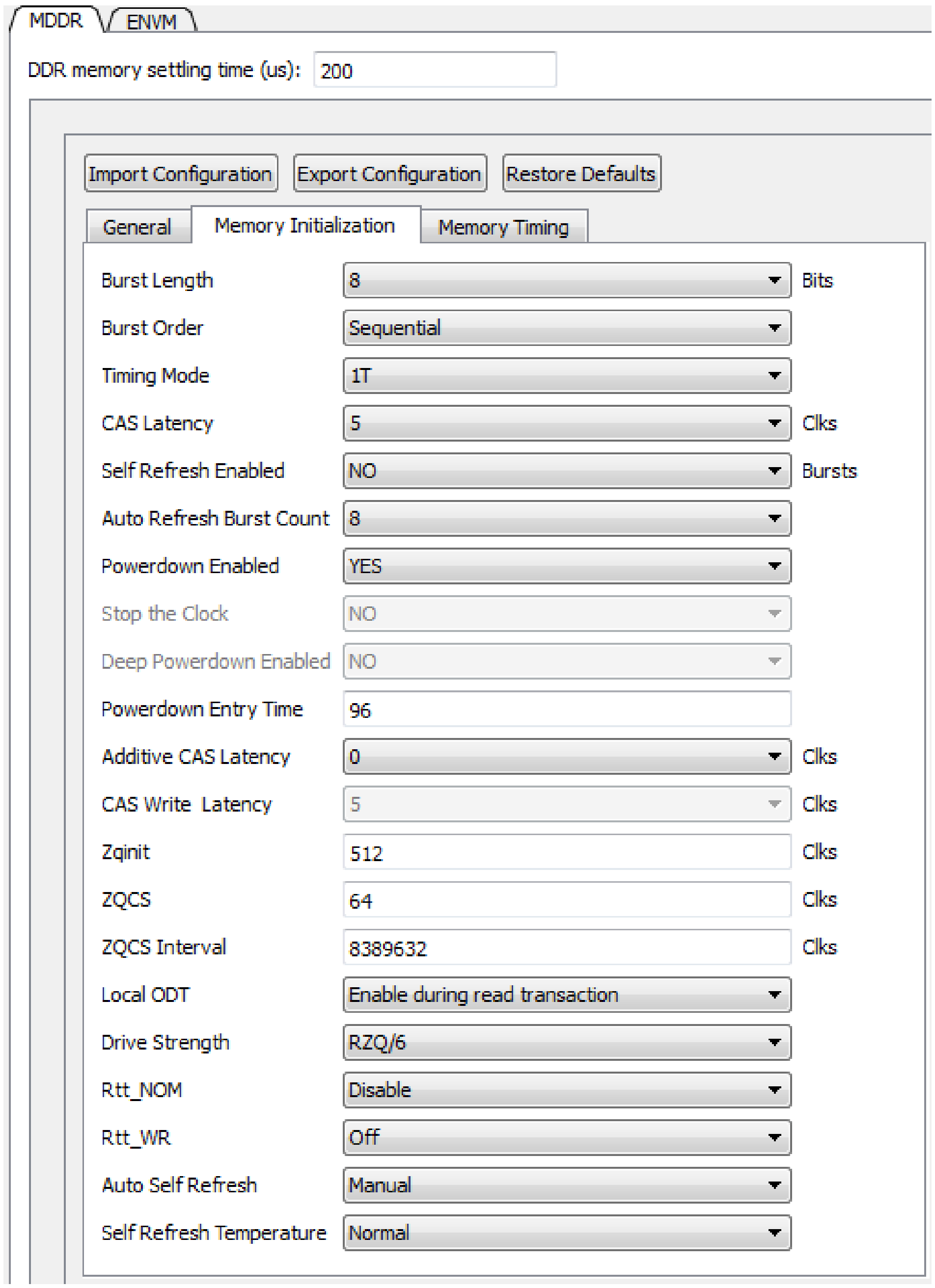
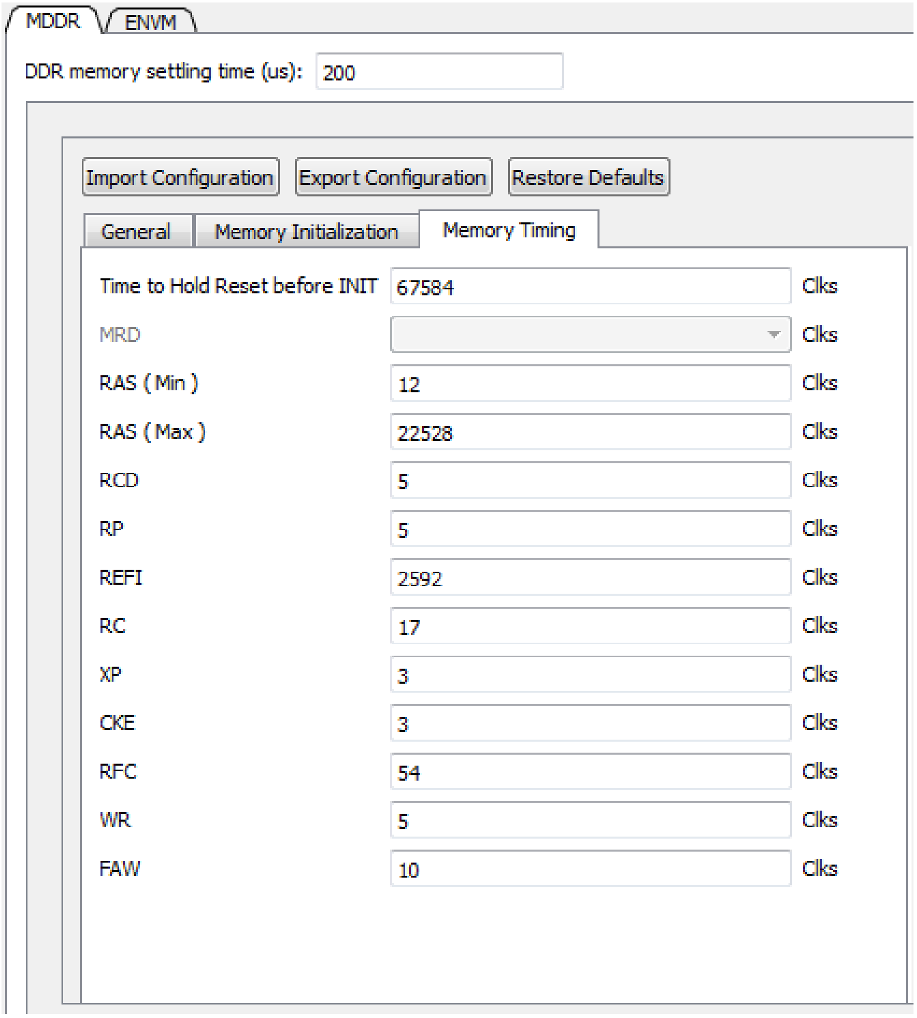


Figure 20 • **DDR Memory Timing Settings**

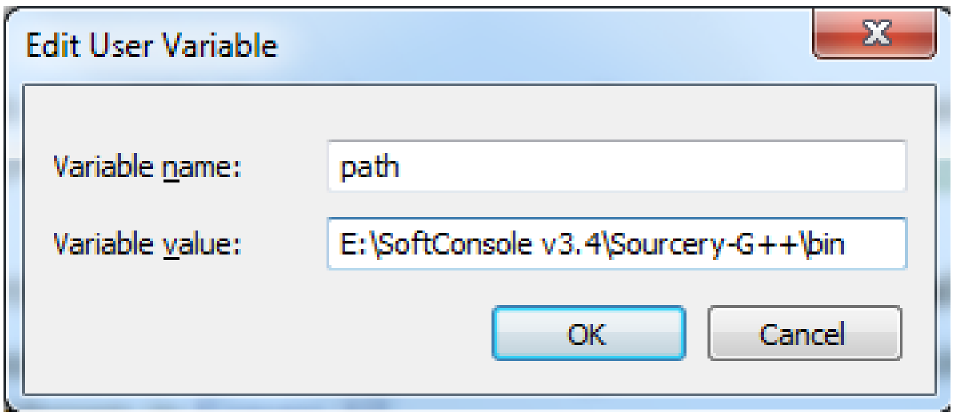


4 Appendix: Generating Executable Bin File

The executable bin file is required to program the SPI flash for running the code shadowing demo. To generate the executable bin file from "sample_image_DDR3" SoftConsole, perform the following steps:

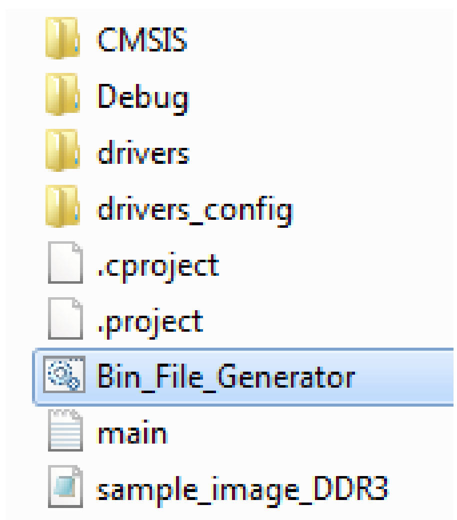
1. Build the SoftConsole project with the linkerscript **production-execute-in-place-externalDDR**.
2. Add the SoftConsole installation path, for example, C:\Microsemi\Libero_v11.7\SoftConsole\Sourcery-G++\bin, to the 'Environment Variables' as shown in Figure 21.

Figure 21 • **Adding SoftConsole Installation Path**



3. Double-click the batch file **Bin-File-Generator.bat** located at: SoftConsole/CodeShadowing_MSS_CM3/Sample_image_DDR3 folder, as shown in Figure 22.

Figure 22 • **Bin File Generator**



4. The **Bin-File-Generator** creates **sample_image_DDR3.bin** file.

5 Revision History

The following table shows important changes made in this document for each revision.

Revision	Changes
Revision 7 (March 2016)	Updated the document for Libero SoC v11.7 software release (SAR 77816).
Revision 6 (October 2015)	Updated the document for Libero SoC v11.6 software release (SAR 72424).
Revision 5 (September 2014)	Updated the document for Libero SoC v11.4 software release (SAR 60592).
Revision 4 (May 2014)	Updated the document for Libero SoC 11.3 software release (SAR 56851).
Revision 3 (December 2013)	Updated the document for Libero SoC v11.2 software release (SAR 53019).
Revision 2 (May 2013)	Updated the document for Libero SoC v11.0 software release (SAR 47552).
Revision 1 (March 2013)	Updated the document for Libero SoC v11.0 beta SP1 software release (SAR 45068).

6 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.

6.1 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

6.2 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.

6.3 Technical Support

For Microsemi SoC Products Support, visit http://www.microsemi.com/products/fpga-soc/design-support/fpga-soc-support.

6.4 Website

You can browse a variety of technical and non-technical information on the Microsemi SoC Products Group home page, at http://www.microsemi.com/products/fpga-soc/fpga-and-soc.

6.5 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.

6.5.1 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.

6.5.2 My Cases

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

6.5.3 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. Visit About Us for sales office listings and corporate contacts.

6.6 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@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.

Microsemi Corporate Headquarters
One Enterprise, Aliso Viejo,
CA 92656 USA

Within the USA: +1 (800) 713-4113
Outside 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 & 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; enterprise storage and communication solutions, security technologies and scalable anti-tamper products; Ethernet solutions; Power-over-Ethernet ICs and midspans; as well as custom design capabilities and services. Microsemi is headquartered in Aliso Viejo, Calif, and has approximately 4,800 employees globally. Learn more at **www.microsemi.com**.

50200386-7/3.16