

Accessing Serial Flash Memory Using SPI Interface - Libero SoC v11.7 and Keil uVision Flow for SmartFusion2

TU0548 Tutorial



Power Matters.™

Contents

1	Preface	5
1.1	Purpose	5
1.2	Intended Audience	5
1.3	References	5
2	Accessing Serial Flash Memory using SPI Interface - Libero SoC v11.7 and Keil uVision Flow for SmartFusion2	6
2.1	Introduction	6
2.2	Design Requirements	7
2.2.1	Project Files	7
2.3	Design Overview	8
2.4	Step 1: Creating a Libero SoC Project	9
2.4.1	Launching Libero SoC	9
2.4.2	Connecting Components in SPI_Flash_0 SmartDesign	16
2.5	Step 2: Generating the Program File	17
2.6	Step 3: Programming the SmartFusion2 Security Evaluation Board Using FlashPro	18
2.7	Step 4: Configuring and Generating Firmware	19
2.8	Step 5: Building the Software Application Using Keil uVision 5 IDE	21
2.9	Step 6: Configuring Serial Terminal Emulation Program	35
2.10	Step 7: Connecting the ULINK-ME to the Board and PC	36
2.11	Step 8: Debugging the Application Project using Keil uVision 5	38
2.12	Conclusion	45
3	Appendix: Board Setup for Debugging from Keil uVision	46
4	Appendix: Board Setup for Programming the Tutorial	47
5	Appendix: SmartFusion2 Security Evaluation Kit Board Jumper Locations	48
6	Revision History	49
7	Product Support	50
7.1	Customer Service	50
7.2	Customer Technical Support Center	50
7.3	Technical Support	50
7.4	Website	50
7.5	Contacting the Customer Technical Support Center	50
7.5.1	Email	50
7.5.2	My Cases	50
7.5.3	Outside the U.S.	51
7.6	ITAR Technical Support	51

Figures

Figure 1.	SPI Flash Interfacing Block Diagram	8
Figure 2.	Libero SoC Project Manager	9
Figure 3.	Project Details Page	10
Figure 4.	Device Selection Page	11
Figure 5.	Device Settings Page	11
Figure 6.	Design Template Window	12
Figure 7.	System Builder Window	12
Figure 8.	System Builder – Device Features Page	13
Figure 9.	System Builder – Peripherals Page	14
Figure 10.	System Builder – Clocks Page	15
Figure 11.	SPI_Flash SmartDesign	16
Figure 12.	SPI_Flash SmartDesign	16
Figure 13.	Generate Component	16
Figure 14.	Log Window	17
Figure 15.	Generate Bitstream	17
Figure 16.	Run Program Action	18
Figure 17.	Configuring Firmware	18
Figure 18.	Export Firmware Dialog Box	19
Figure 19.	Keil Homepage	20
Figure 20.	uVision Workspace	21
Figure 21.	uVision Workspace main.c file	22
Figure 22.	uVision Workspace Window - Add winbondflash SPI Driver Files	23
Figure 23.	Cortex-M3_SRAM Settings	24
Figure 24.	Target Options	25
Figure 25.	Target Options-Adding Symbols	26
Figure 26.	Build HW Platform Window	27
Figure 27.	Set as Active Project	28
Figure 28.	Cortex-M3_SRAM Settings	29
Figure 29.	Target Options	30
Figure 30.	Target Options - Scatter File	31
Figure 31.	Target Options - Utilities Settings	32
Figure 32.	Build Application Window	33
Figure 33.	Build Output	33
Figure 34.	Device Manager Window	34
Figure 35.	ULINK-ME Connections	35
Figure 36.	ULINK-ME Debugger	36
Figure 37.	Selecting Start/Stop Debug Session	37
Figure 38.	Debug Menu	38
Figure 39.	Selecting Run from the Debug Menu	39
Figure 40.	HyperTerminal Window	39
Figure 41.	HyperTerminal Window - Option 1	40
Figure 42.	HyperTerminal Window - Option 2	40
Figure 43.	Disassembly Window	41
Figure 44.	Values of the Cortex-M3 Internal Registers	42
Figure 45.	Keil uVision Workbench - Stop Debug Option	43
Figure 46.	Keil uVision Workbench - Step Level Debugging	44
Figure 47.	SmartFusion2 Security Evaluation Kit in Debug Mode using Keil uVision	45
Figure 48.	SmartFusion2 Security Evaluation Kit in Programming Mode	46
Figure 49.	SmartFusion2 Security Evaluation Kit Board Jumper Locations	47

Tables

Table 1.	Design Requirements	7
Table 2.	SmartFusion2 Security Evaluation Kit Jumper Settings	17

1 Preface

1.1 Purpose

This tutorial provides steps to create a Libero® system-on-chip (SoC) software design using the System Builder. It describes how to build, debug, and run Keil uVision application. It also provides a simple design to access the SPI flash.

1.2 Intended Audience

This tutorial is intended for:

- FPGA designers
- Embedded designers
- System-level designers

1.3 References

The following documents are referred in this user tutorial:

- *TU0546: SoftConsole v4.0 and Libero SoC v11.7 Tutorial*
- *TU0547: Accessing Serial Flash Memory Using SPI Interface - Libero SoC and IAR Embedded Workbench Flow Tutorial for SmartFusion2*

2 Accessing Serial Flash Memory using SPI Interface - Libero SoC v11.7 and Keil uVision Flow for SmartFusion2

2.1 Introduction

The Libero SoC software generates firmware projects using Keil, SoftConsole, and IAR tools. This tutorial describes the process to build a Keil uVision application that can be implemented and validated using the SmartFusion®2 system-on-chip (SoC) field programmable gate array (FPGA) Security Evaluation Kit.

The same firmware project can be built using the IAR and softconsole tools as well. Following are the respective references:

- [*TU0546: SoftConsole v4.0 and Libero SoC v11.7 Tutorial*](#)
- [*TU0547: Accessing Serial Flash Memory Using SPI Interface - Libero SoC and IAR Embedded Workbench Flow Tutorial for SmartFusion2*](#)

This tutorial describes the following:

- Step 1: Creating a Libero SoC Project
- Step 2: Generating the Program File
- Step 3: Programming the SmartFusion2 Security Evaluation Board Using FlashPro
- Step 4: Configuring and Generating Firmware
- Step 5: Building the Software Application Using Keil uVision 5 IDE
- Step 6: Configuring Serial Terminal Emulation Program
- Step 7: Connecting the ULINK-ME to the Board and PC
- Step 8: Debugging the Application Project using Keil uVision 5

2.2 Design Requirements

Table 1 lists the design requirements of Keil uVision flow.

Table 1 • Design Requirements

Design Requirements	Description
Hardware Requirements	
SmartFusion2 Security Evaluation Kit: - FlashPro4 programmer - USB A to Mini-B cable 12 V Adapter	Rev D or later
Keil debugger	–
Host PC or Laptop	Any 64-bit Windows Operating System
Software Requirements	
Libero SoC	v11.7
Keil uVision	v5
FlashPro programming software	v11.7
Host PC Drivers	USB to UART drivers

2.2.1 Project Files

The design files for this tutorial can be downloaded from the Microsemi website:

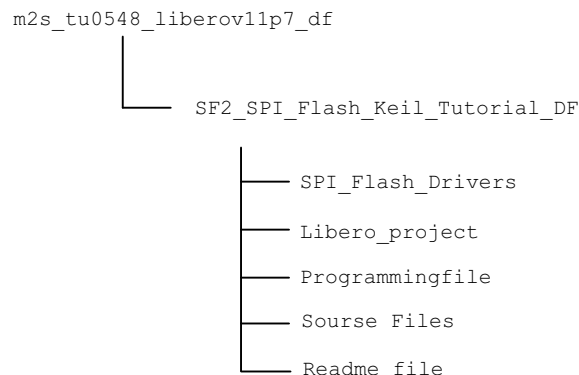
http://soc.microsemi.com/download/rsc/?f=m2s_tu0548_liberov11p7_df

The design files include:

- LiberoProject
- Programmingfile
- Source Files
- SPI_Flash_Drivers
- Readme file

Figure 1 shows the top-level structure of the design files. For further details, refer to the `Readme.txt` file.

Figure 1 • Design Files Top-Level Structure



2.3 Design Overview

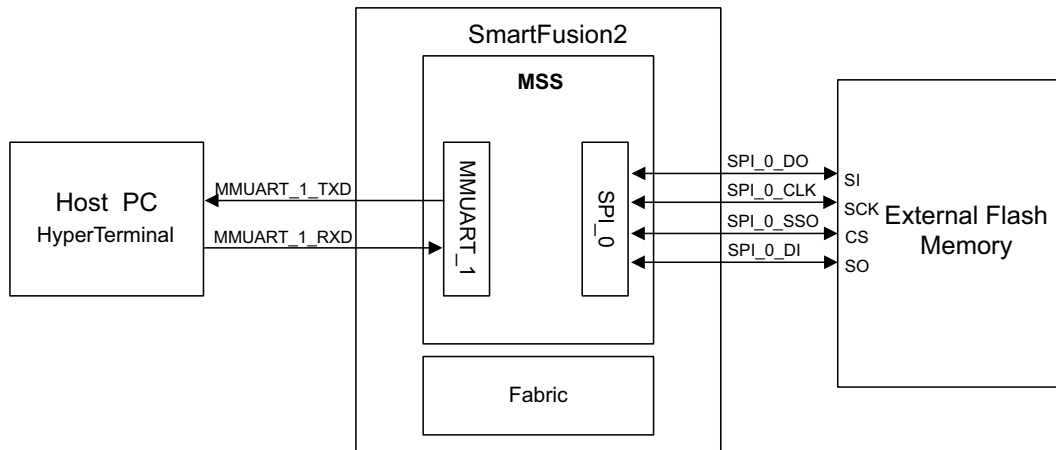
This design example demonstrates the execution of basic read and write operations on the SPI flash present on the SmartFusion2 Security Evaluation Kit board. This kit has a built-in winbond SPI flash memory W25Q64FVSSIG, which is connected to the SmartFusion2 microcontroller subsystem (MSS) through the dedicated MSS SPI_0 interface.

Read and write data information is displayed using HyperTerminal, which communicates to the SmartFusion2 MSS using the MMUART_1 interface.

Refer to the [UG0331: SmartFusion2 Microcontroller Subsystem User Guide](#) for more information on SPI.

Figure 2 shows interfacing the external SPI flash to MSS SPI_0.

Figure 2 • SPI Flash Interfacing Block Diagram



2.4 Step 1: Creating a Libero SoC Project

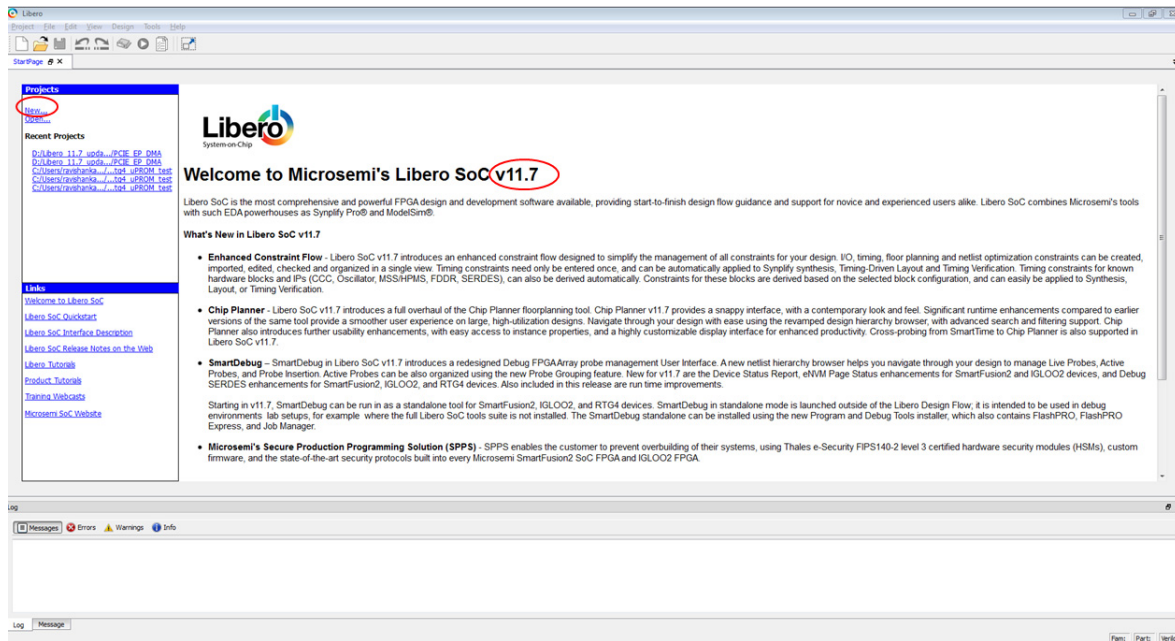
This section describes how to create a Libero SoC project.

2.4.1 Launching Libero SoC

The following steps describe how to launch Libero SoC:

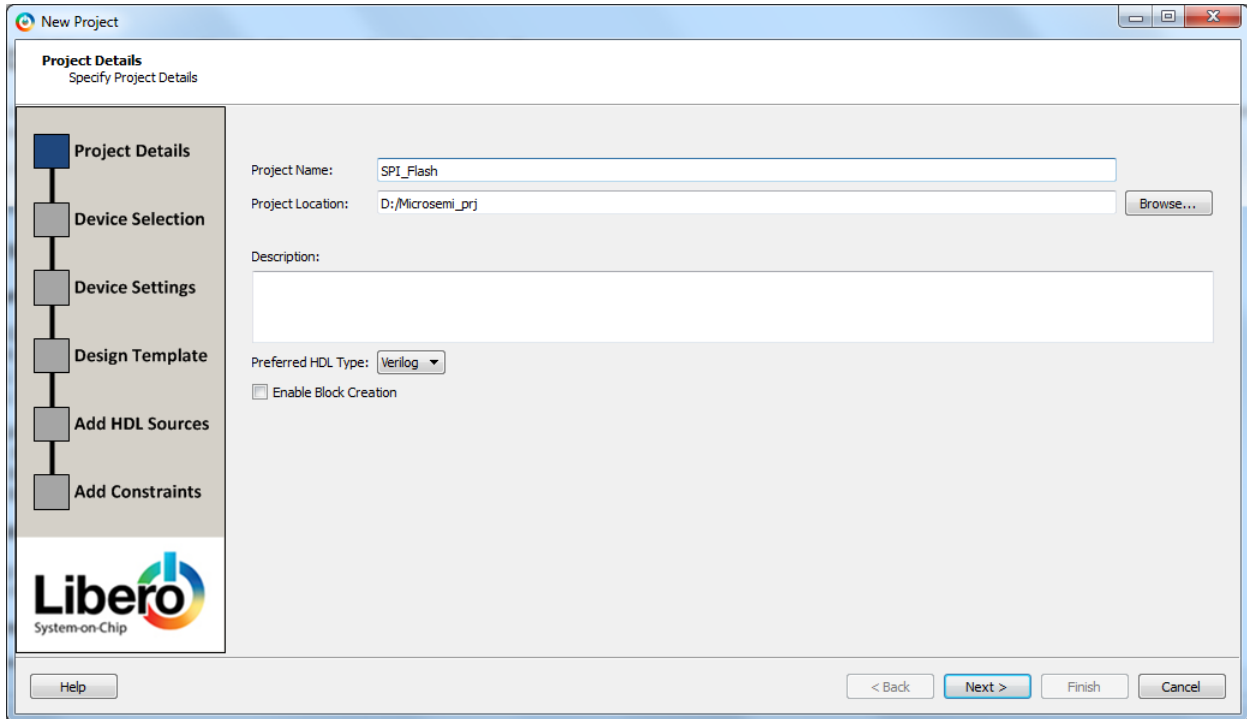
1. Choose **Start > Programs > Microsemi Libero SoC v11.7 > Libero SoC v11.7**, or double-click the shortcut on desktop to open the Libero SoC v11.7 Project Manager.
2. Create a new project using one of the following options:
 - Select **New** on the **Start Page** tab, as shown in Figure 3.
 - Click **Project > New Project** from the Libero SoC menu.

Figure 3 • Libero SoC Project Manager



3. Enter the following information in the **Project Details** page, as shown in Figure 4.
 - **Project Name:** SPI_Flash
 - **Project Location:** Select an appropriate location (For example, *D:/Microsemi_prj*)
 - **Preferred HDL Type:** Verilog
 - **Enable Block Creation:** Unchecked

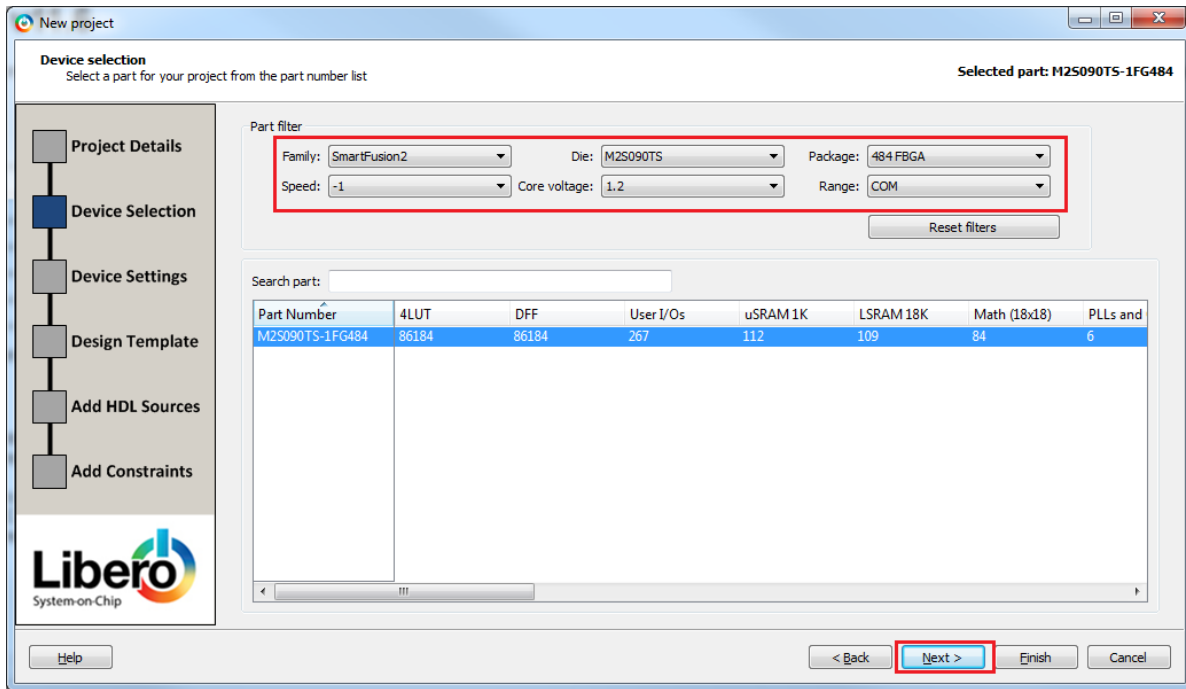
Figure 4 • Project Details Page



The screenshot shows the 'New Project' dialog box with the 'Project Details' tab selected. The 'Project Name' field contains 'SPI_Flash' and the 'Project Location' field contains 'D:/Microsemi_prj'. The 'Preferred HDL Type' is set to 'Verilog' and the 'Enable Block Creation' checkbox is unchecked. The sidebar on the left shows the project setup steps, with 'Project Details' being the current step. The bottom of the dialog has a 'Help' button and navigation buttons: '< Back', 'Next >', 'Finish', and 'Cancel'.

4. Click **Next**. The **Device Selection** page is displayed, as shown in Figure 5 on page 11. Select the following values from the drop-down list:
 - **Family:** SmartFusion2
 - **Die:** M2S090TS
 - **Package:** 484FBGA
 - **Speed:** -1
 - **Core Voltage:** 1.2
 - **Range:** COM

Figure 5 • Device Selection Page



Device selection
Select a part for your project from the part number list

Selected part: M2S090TS-1FG484

Part filter

Family: SmartFusion2 Die: M2S090TS Package: 484 FBGA
Speed: -1 Core voltage: 1.2 Range: COM

Reset filters

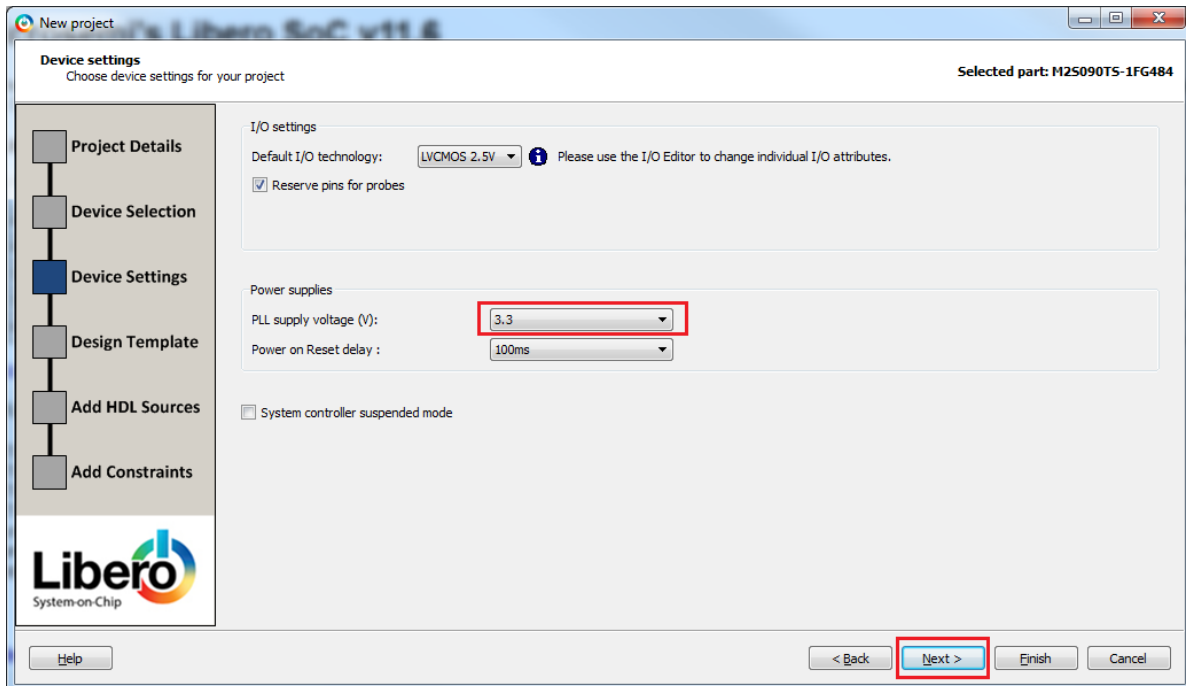
Search part:

Part Number	4LUT	DFF	User I/Os	uSRAM 1K	LSRAM 18K	Math (18x18)	PLLs and
M2S090TS-1FG484	86184	86184	267	112	109	84	6

Help < Back **Next >** Finish Cancel

5. Click **Next**. The **Device Settings** page is displayed.
6. Select **PLL supply voltage (V)** as **3.3**, as shown in Figure 6 and click **Next**.

Figure 6 • Device Settings Page



Device settings
Choose device settings for your project

Selected part: M2S090TS-1FG484

I/O settings

Default I/O technology: LVCMOS 2.5V Please use the I/O Editor to change individual I/O attributes.

☒ Reserve pins for probes

Power supplies

PLL supply voltage (V): 3.3

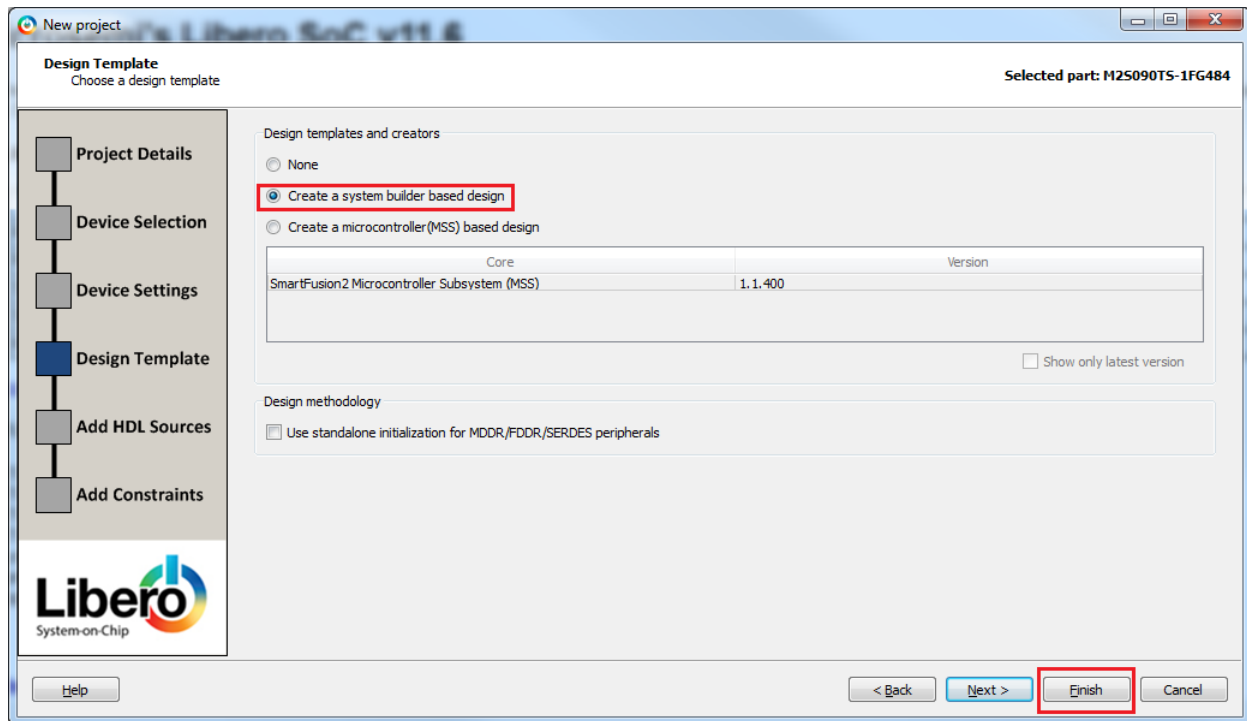
Power on Reset delay : 100ms

☐ System controller suspended mode

Help < Back **Next >** Finish Cancel

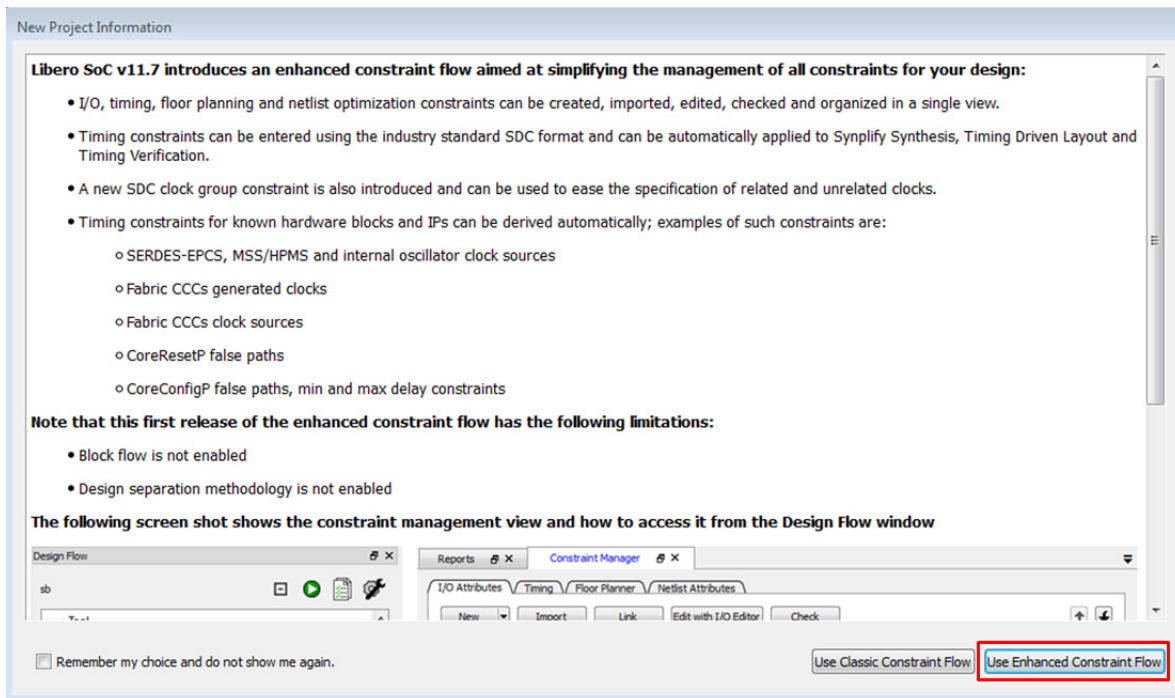
- Click **Next**. The **Design Template** page is displayed, as shown in Figure 7. Under **Design Template and Creators**, click **Create a system builder based design**.

Figure 7 • Design Template Window



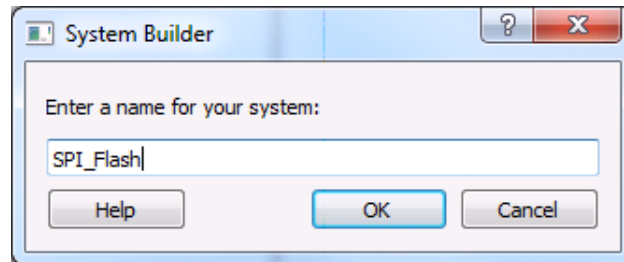
- Click **Finish**. This displays the **New Project Information** window. Select **Use Enhanced Constraint Flow**, as shown in Figure 8.

Figure 8 • New Project Information



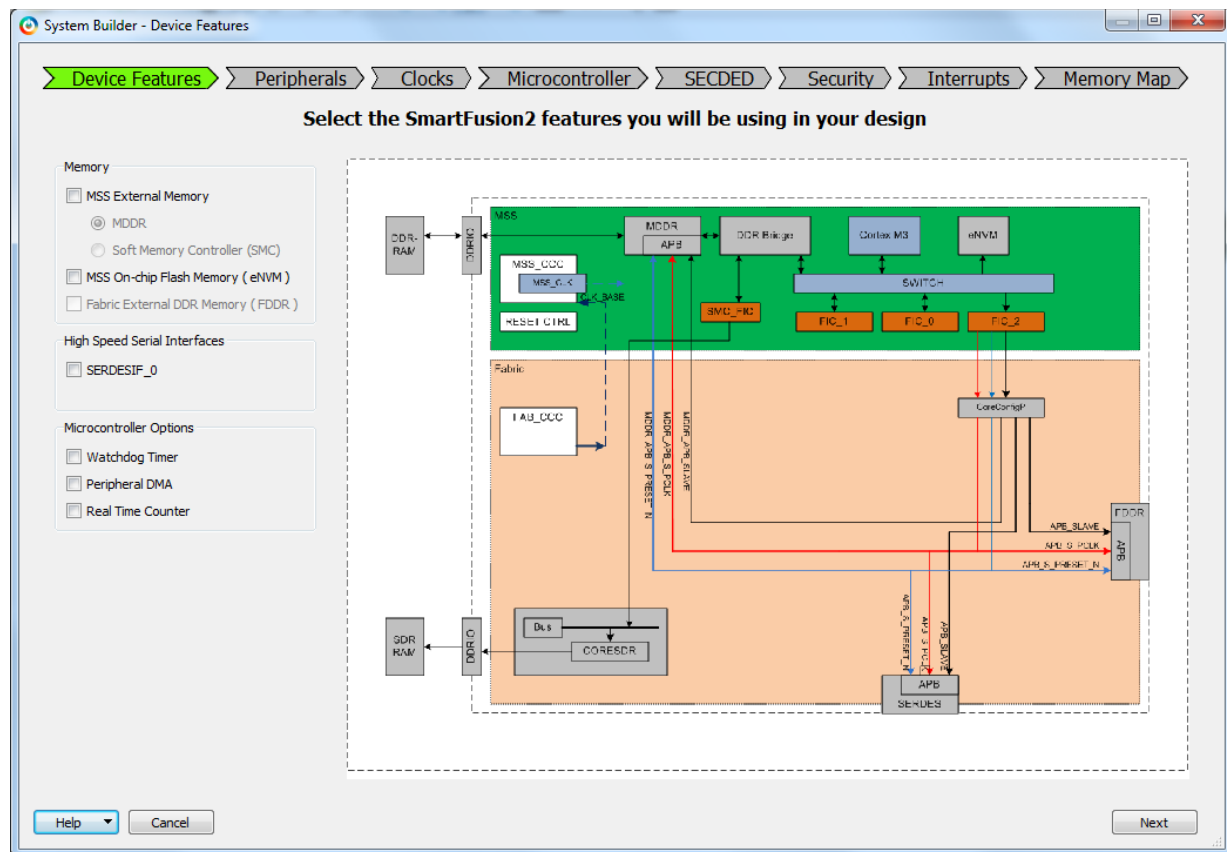
9. The **System Builder** window is displayed. Enter a name for your system as **SPI_Flash** and click **OK**, as shown in [Figure 9](#).
- Note:** System Builder is a graphical design wizard. It creates a design based on high-level design specifications by taking the user through a set of high-level questions to create the intended system.

Figure 9 • System Builder Window



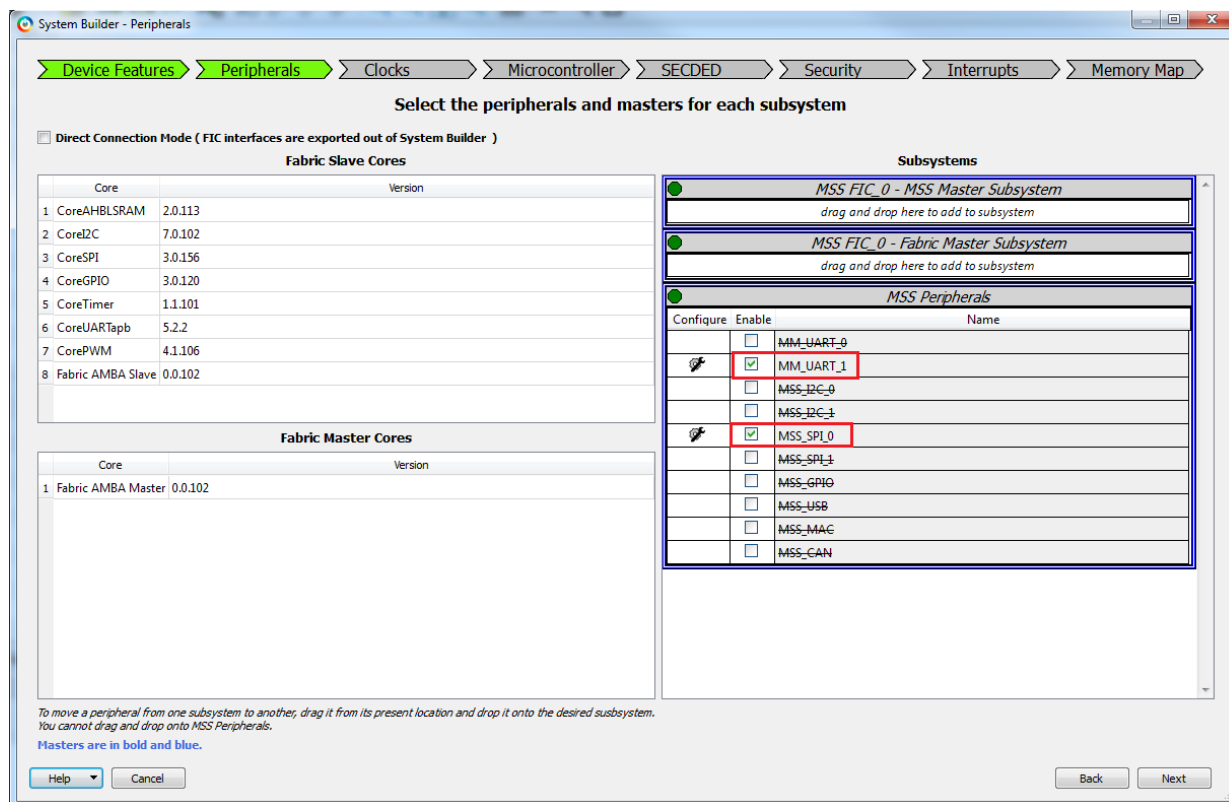
[Figure 10](#) shows the **System Builder – Device Features** page.

Figure 10 • System Builder – Device Features Page



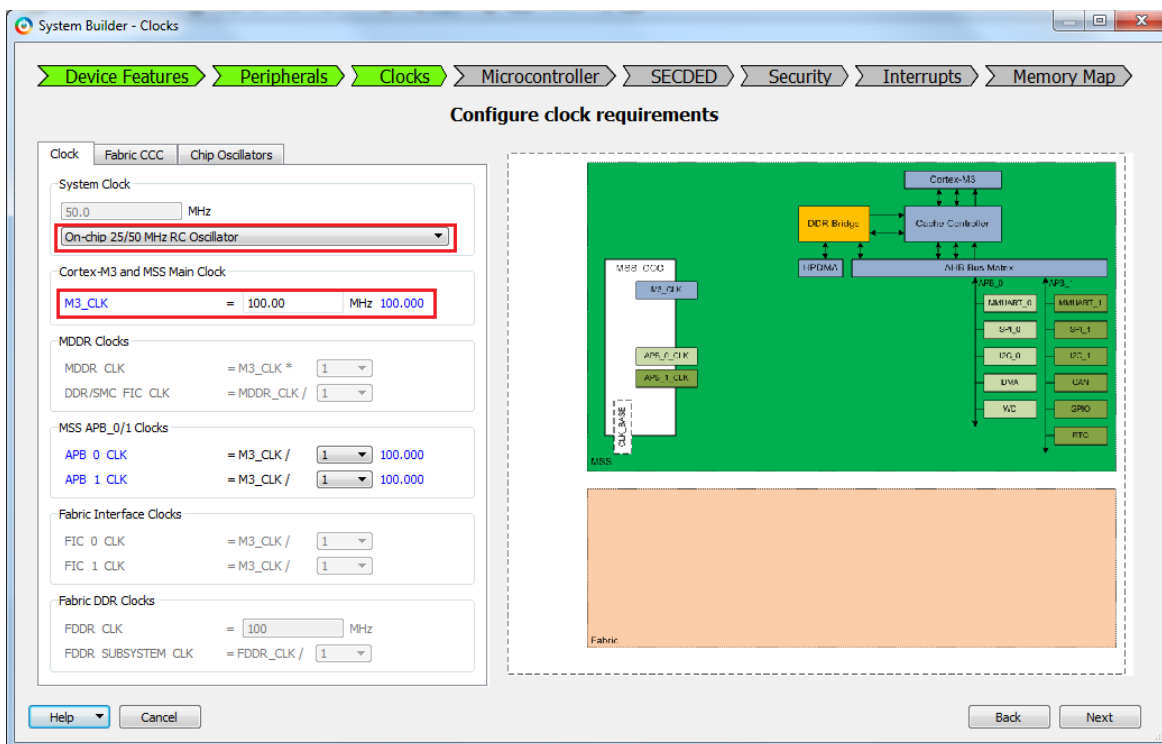
10. Click **Next**. The **System Builder - Peripherals** page is displayed, as shown in Figure 11.
11. Under the **MSS Peripherals** section, clear all the check boxes except **MM_UART_1** and **MSS_SPI_0**, as shown in Figure 11.

Figure 11 • System Builder – Peripherals Page



12. Click **Next**. The **System Builder - Clocks** page is displayed, as shown in Figure 12.
13. In the **System Builder - Clocks** page (refer Figure 12):
 - Select **System Clock** frequency as **50 MHz** and clock source as **On-chip 25/50 MHz RC Oscillator**
 - Select **M3_CLK** as **100 MHz**
 - Select **APB_0_CLK** and **APB_1_CLK** frequency as **M3_CLK/1**

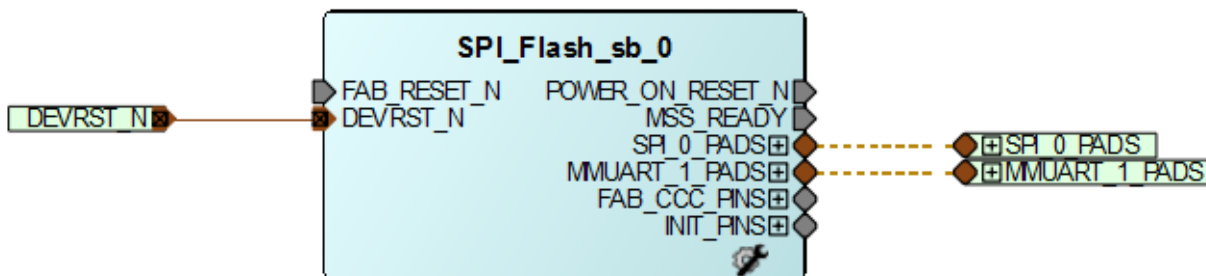
Figure 12 • System Builder – Clocks Page



14. Click **Next**. The **System Builder - Microcontroller** page is displayed. Do not change the default selections.
15. Click **Next**. The **System Builder - SECEDED** page is displayed. Do not change the default selections.
16. Click **Next**. The **System Builder - Security** page is displayed. Do not change the default selections.
17. Click **Next**. The **System Builder - Interrupts** page is displayed. Do not change the default selections.
18. Click **Next**. The **System Builder - Memory Map** page is displayed. Do not change the default selections.
19. Click **Finish**.

20. Select **File > Save** to save **SPI_Flash**. Select the **SPI_Flash** tab on the Smart Design canvas, as shown in Figure 13.

Figure 13 • SPI_Flash SmartDesign



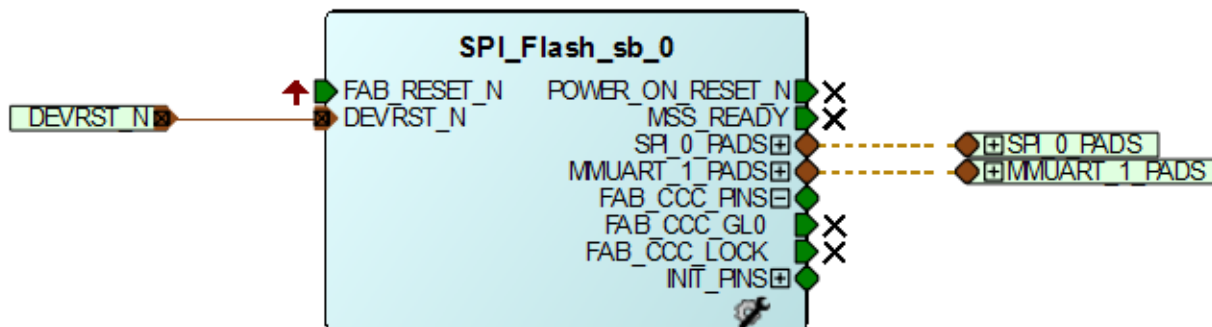
2.4.2 Connecting Components in SPI_Flash_0 SmartDesign

The following steps describe how to connect the components in the **SPI_Flash** SmartDesign:

1. Right-click **POWER_ON_RESET_N** and select **Mark Unused**.
2. Right-click **MSS_READY** and select **Mark Unused**.
3. Expand **INIT_PINS**, right-click **INIT_DONE** and select **Mark Unused**.
4. Expand **FAB_CCC_PINS**, right-click **FAB_CCC_GL0** and select **Mark Unused**.
5. Right-click **FAB_CCC_LOCK** and select **Mark Unused**.
6. Right-click **FAB_RESET_N** and select **Tie High**.
7. Click **File > Save**.

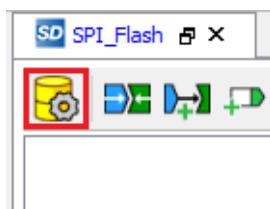
The SPI_Flash design is displayed, as shown in Figure 14.

Figure 14 • SPI_Flash SmartDesign



8. Generate the SPI_Flash SmartDesign by clicking **SmartDesign > Generate Component** or by clicking **Generate Component** on the SmartDesign toolbar, as shown in Figure 15.

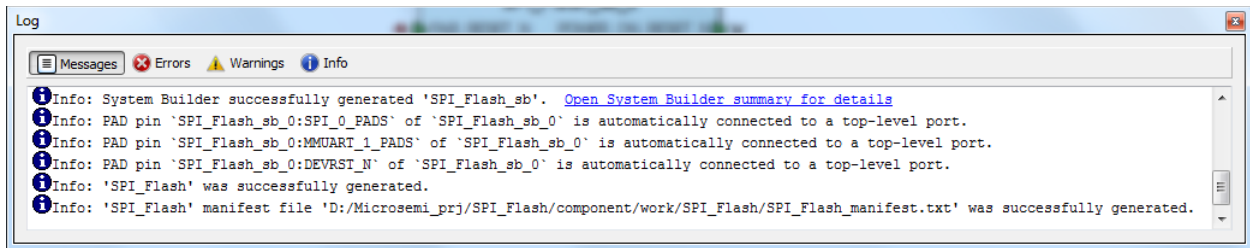
Figure 15 • Generate Component



After successful generation of all the components, the following message is displayed on the log window, as shown in Figure 16.

Info: 'SPI_Flash' was successfully generated.

Figure 16 • Log Window



2.5 Step 2: Generating the Program File

The following step describe how to generate the program file:

1. Double-click **Manage Constraints** and click **Derive Constraints** option under **Timing** tab to generate SDC file for root module. Click **Yes** to associate SDC file to synthesis, place and route and timing verification stages, as shown in the Figure 17, Figure 18 on page 18, and Figure 19 on page 18.

Figure 17 • Manage Constraints

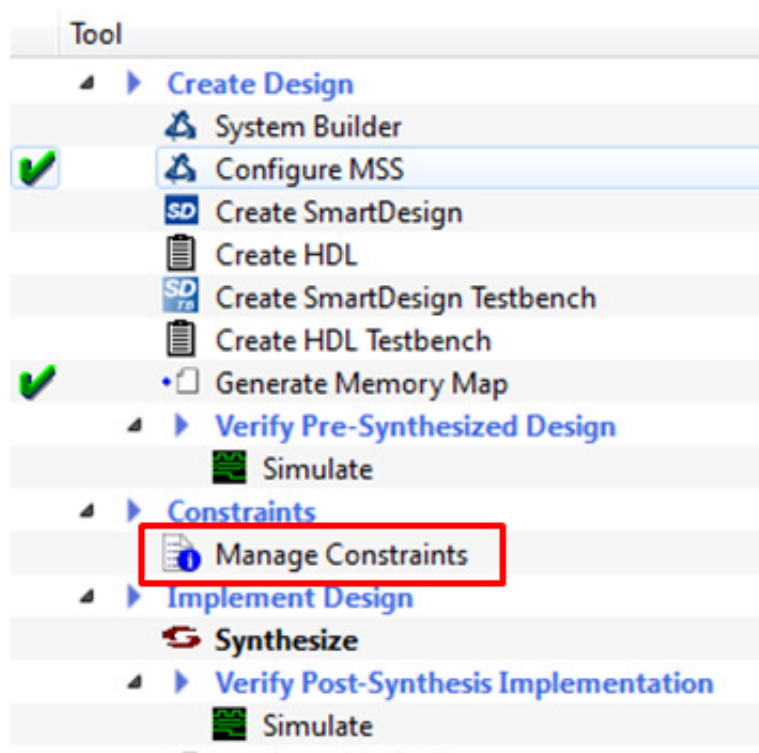


Figure 18 • Derive Constraints

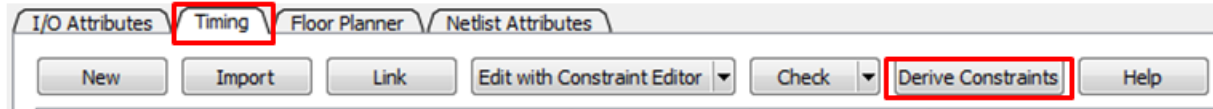
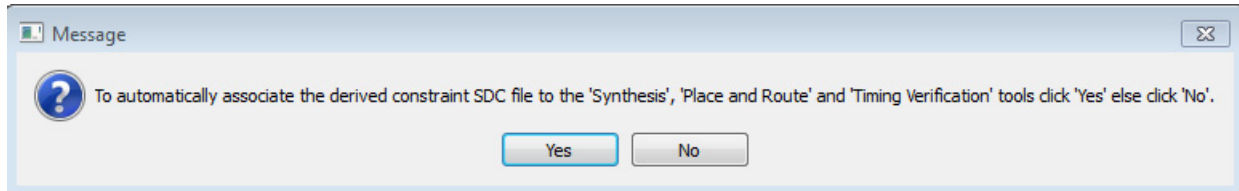


Figure 19 • Associate SDC File



2. Click Generate Bitstream to generate the programming file, as shown in Figure 20.

Figure 20 • Generate Bitstream



2.6 Step 3: Programming the SmartFusion2 Security Evaluation Board Using FlashPro

The following steps describe how to program the SmartFusion2 Security Evaluation Kit board using FlashPro:

1. Connect the FlashPro4 programmer to the J5 connector of the SmartFusion2 Security Evaluation Kit board.
2. Connect the jumpers on the SmartFusion2 Security Evaluation Kit board, as listed in Table 2. For more information on jumper locations, refer to the "[Appendix: SmartFusion2 Security Evaluation Kit Board Jumper Locations](#)" on page 48.

CAUTION: Ensure that the power supply switch, **SW7** is switched OFF while connecting the jumpers on the SmartFusion2 Security Evaluation Kit.

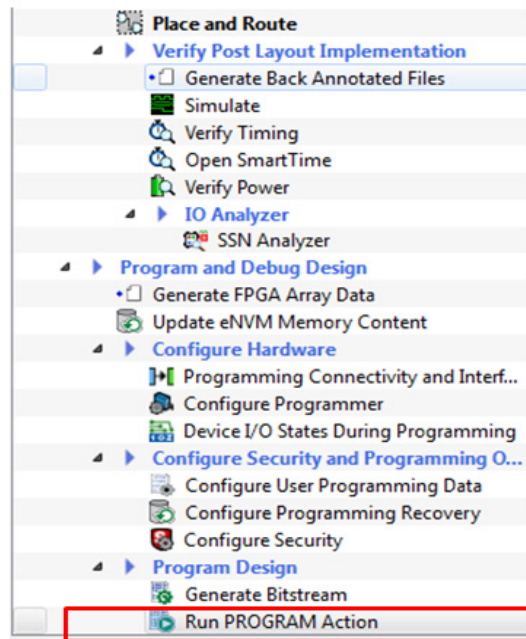
Table 2 • SmartFusion2 Security Evaluation Kit Jumper Settings

Jumper Number	Pin (From)	Pin (To)	Comments
J22, J23, J24, J8, J3	1	2	These are the default jumper settings of the SmartFusion2 Security Evaluation Kit board. Ensure that these jumpers are set accordingly.

3. Connect the power supply to the J6 connector.
Switch **ON** the power supply switch, **SW7**. Refer to "[Appendix: Board Setup for Programming the Tutorial](#)" on page 47 for information on the board setup for running the tutorial.

- To program the SmartFusion2 device, double-click **Run PROGRAM Action** in the **Design Flow** tab, as shown in Figure 21.

Figure 21 • Run Program Action



2.7 Step 4: Configuring and Generating Firmware

The Design Firmware window displays compatible firmware drivers based on peripherals configured in the design. Following drivers are used in this tutorial:

- CMSIS
- MMUART
- SPI

To generate the required drivers:

- Double-click on **Configure Firmware Cores** in **Handoff design for Firmware Development** in **Design Flow** window.
- Clear all the drivers check boxes, except **SmartFusion2_CMSIS_0**, **SmartFusion2_MSS_MMUART_Driver_0**, and **SmartFusion2_MSS_SPI_Driver_0**, as shown in Figure 22.

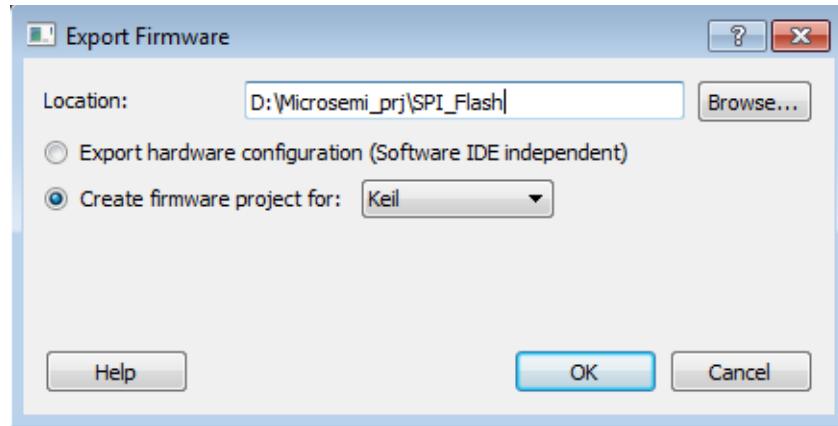
Note: Select the latest version of the drivers.

Figure 22 • Configuring Firmware

Generate	Instance Name	Core Type	Version	Compatible Hardware Instance
☒	SmartFusion2_CMSIS_0	SmartFusion2_CMSIS	2.3.105	SPI_Flash_sb_MSS
☐	SmartFusion2_MSS_HPDMADriver_0	SmartFusion2_MSS_HPDMADriver	2.2.100	SPI_Flash_sb_MSS
☒	SmartFusion2_MSS_MMUART_Driver_0	SmartFusion2_MSS_MMUART_Driver	2.1.100	SPI_Flash_sb_MSS:MMUART_1
☐	SmartFusion2_MSS_NVM_Driver_0	SmartFusion2_MSS_NVM_Driver	2.4.100	SPI_Flash_sb_MSS
☒	SmartFusion2_MSS_SPI_Driver_0	SmartFusion2_MSS_SPI_Driver	2.7.100	SPI_Flash_sb_MSS:SPI_0
☐	SmartFusion2_MSS_System_Services_Driver_0	SmartFusion2_MSS_System_Services_Driver	2.7.100	SPI_Flash_sb_MSS
☐	SmartFusion2_MSS_Timer_Driver_0	SmartFusion2_MSS_Timer_Driver	2.2.100	SPI_Flash_sb_MSS

3. Double-click on **Export Firmware** in **Handoff design for Firmware Development** in **Design Flow** window.
Export Firmware dialog box is displayed, as shown in [Figure 23](#).

Figure 23 • Export Firmware Dialog Box



4. In the **Export Firmware** dialog box:
 - Select **Create firmware project for**.
 - Select **Keil** from the drop-down list.
5. Click **OK**. The successful firmware generation window is displayed.

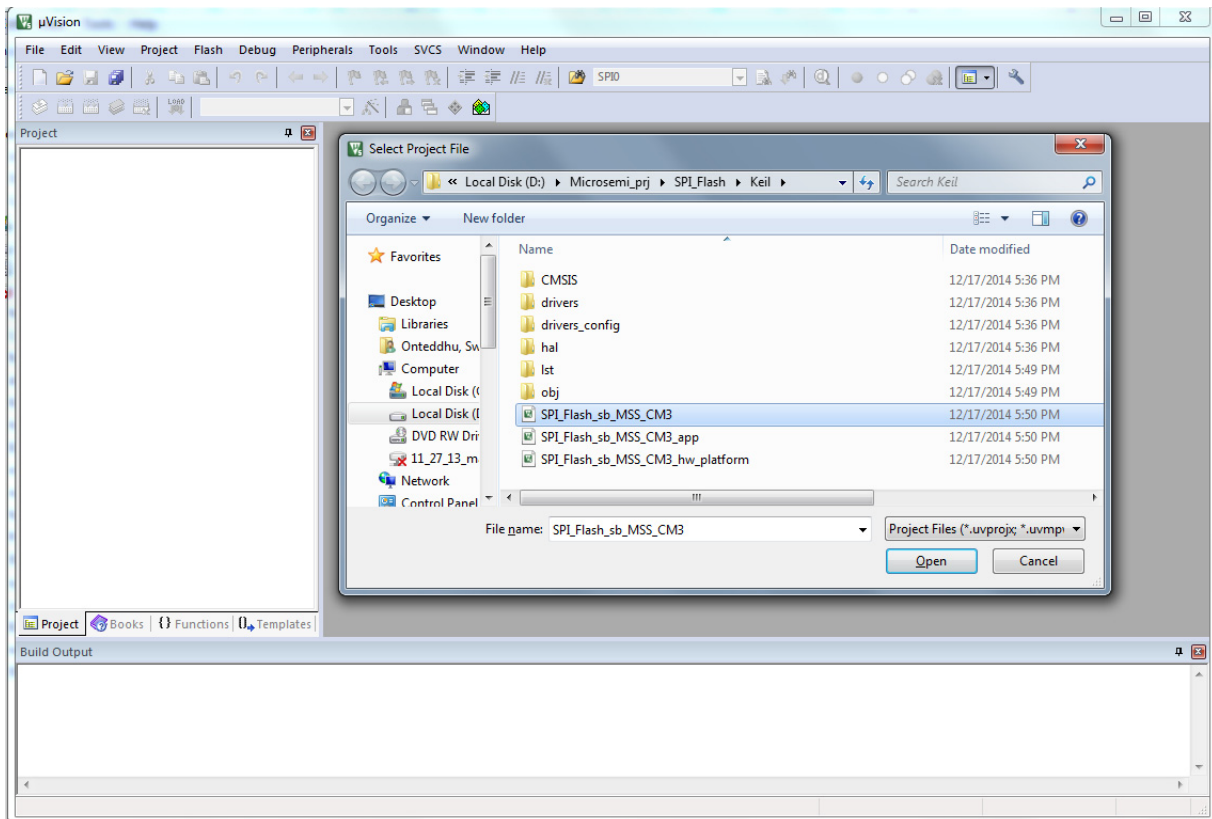
The SmartFusion2 Security Evaluation Kit is ready for running and debugging the Keil application through ULINK-ME Debugger.

2.8 Step 5: Building the Software Application Using Keil uVision 5 IDE

The following steps describe how to build a software application using Keil uVision 5 IDE:

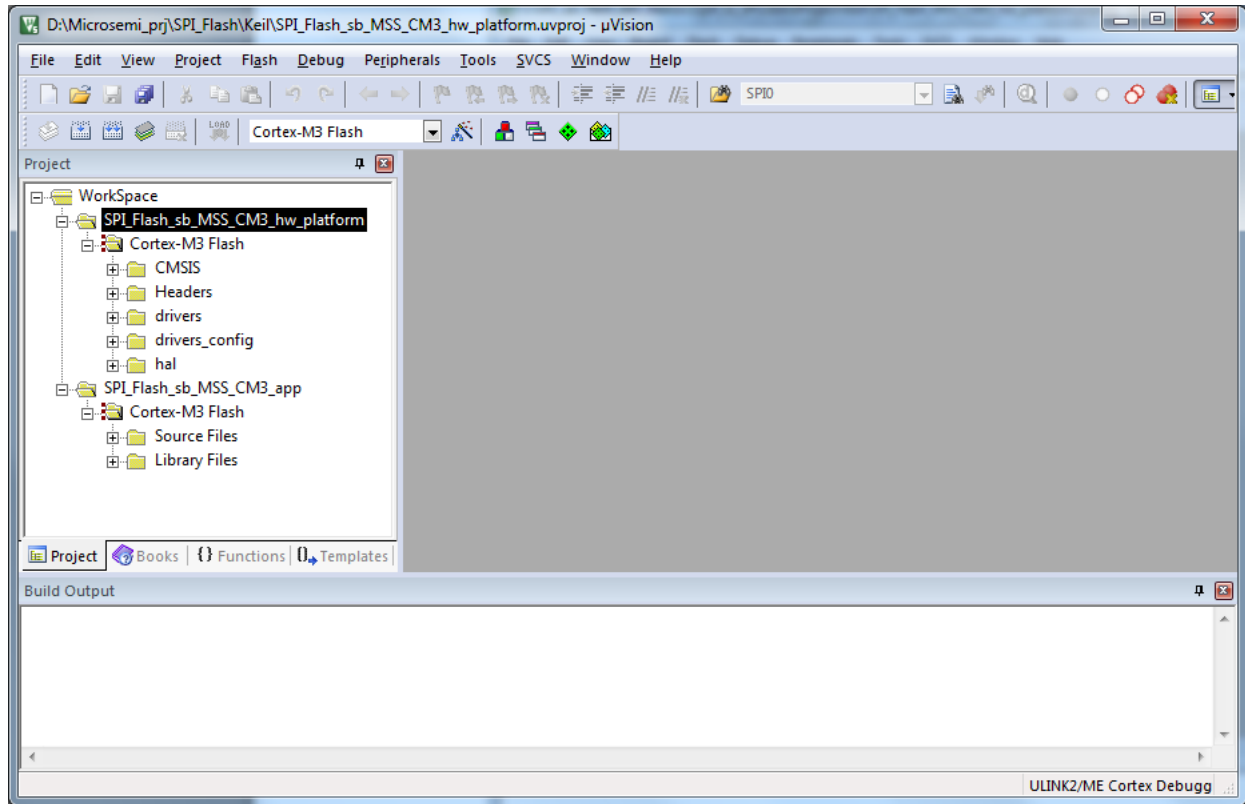
1. Launch the Keil IDE. Open the Keil project by double-clicking **SPI_Flash_sb_MSS_CM3** Keil project, as shown in Figure 24.

Figure 24 • Keil Homepage



The Keil workspace is displayed, as shown in Figure 25.

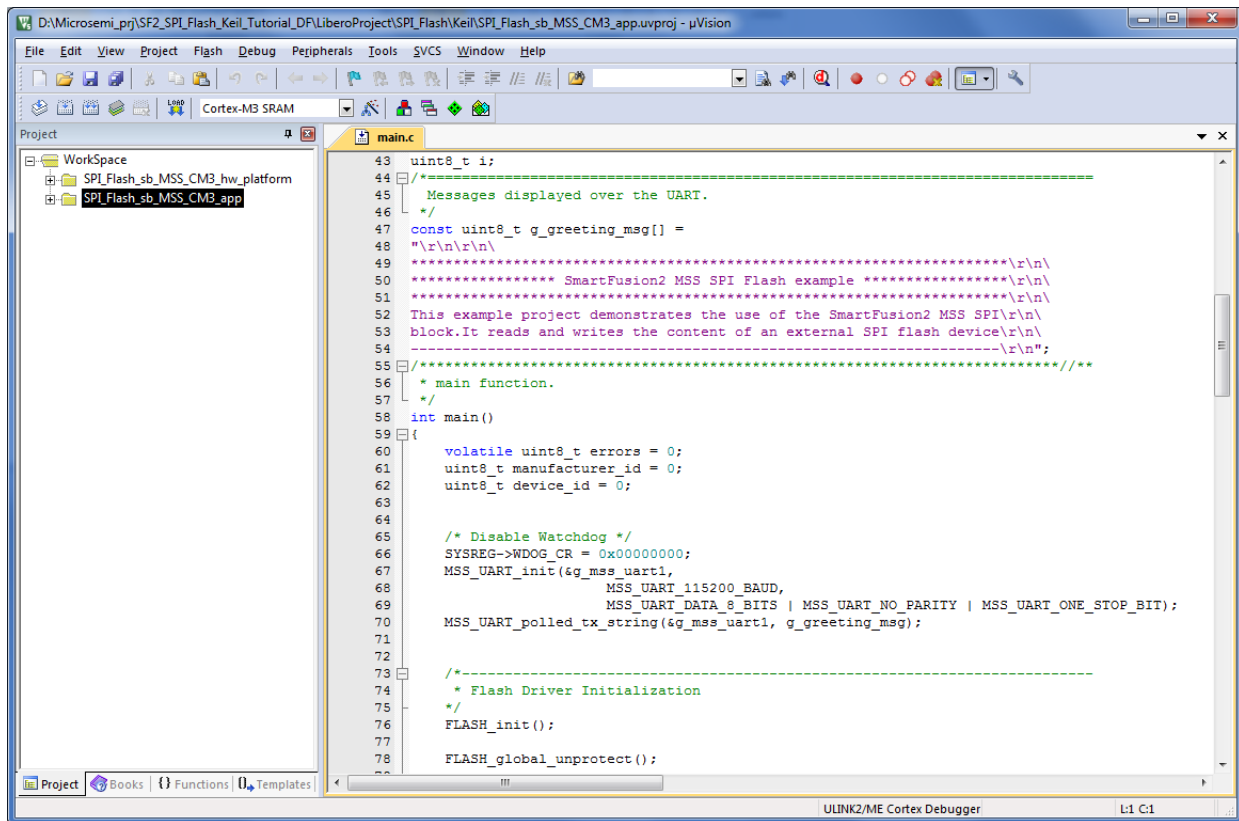
Figure 25 • uVision Workspace



2. Browse to the `main.c` file, location in the design files folder:
`<download_folder>/SF2_SPI_Flash_Keil_Tutorial_DF\SourceFiles`.
3. Copy the `main.c` file and replace the existing `main.c` file under `SPI_Flash_sb_MSS_CM3_app` project in the uVision workspace.

The uVision window displays the `main.c` file, as shown in Figure 26.

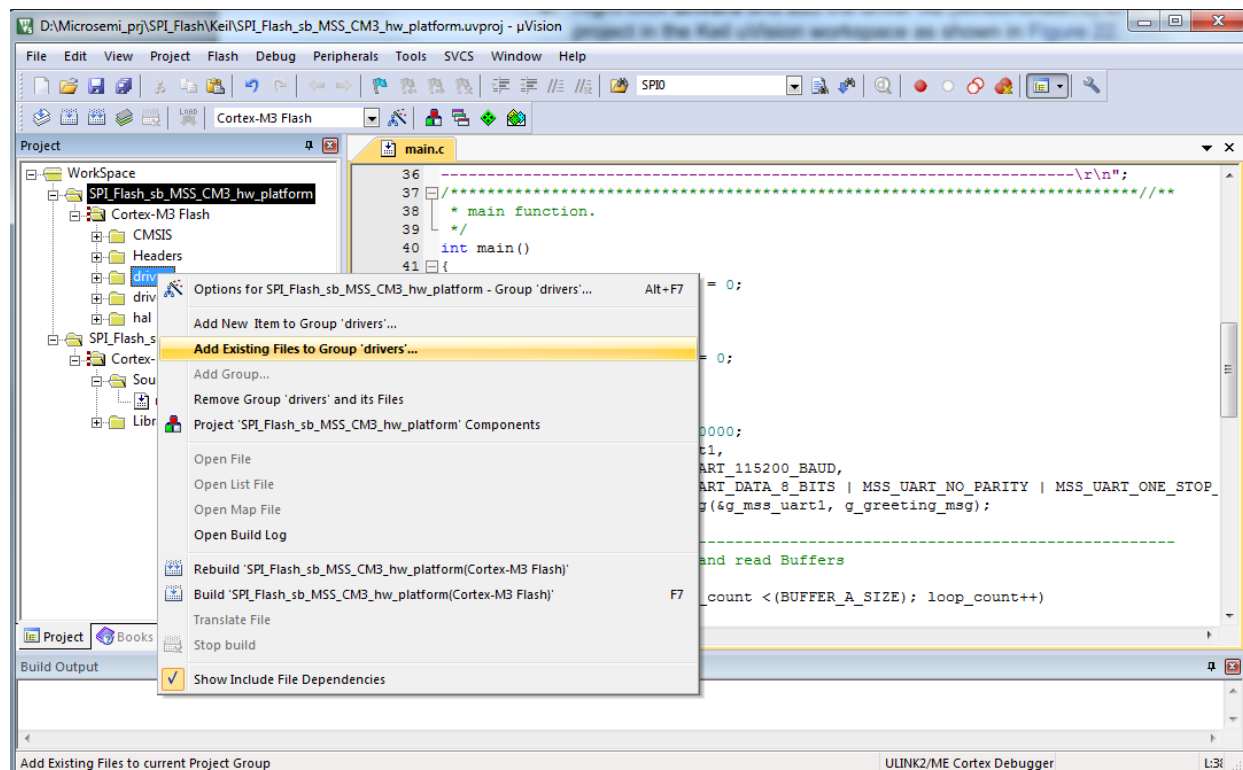
Figure 26 • uVision Workspace `main.c` file



4. winbondflash drivers are not included in the Libero generated uVision workspace. To include the drivers in the uVision workspace, browse to the location of the winbondflash drivers in the design files folder:
<download_folder>\SF2_SPI_Flash_Keil_Tutorial_DF\SPI_Flash_Drivers.
5. Copy the **winbond flash** folder to the drivers folder of SFI_Flash_sb_MSS_CM3_hw_platform project in the uVision workspace.

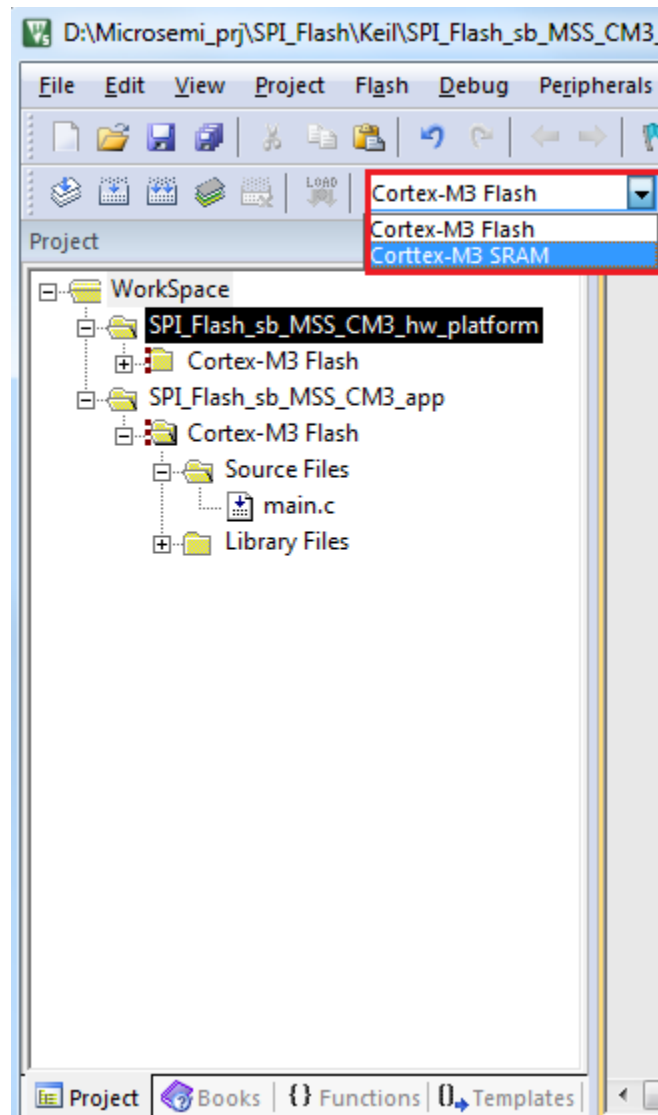
- Right-click and add the driver file (winbondflash.c) to SPI_Flash_sb_MSS_CM3_hw_platform project in the Keil uVision workspace, as shown in Figure 27.

Figure 27 • uVision Workspace Window - Add winbondflash SPI Driver Files



7. Change **SPI_Flash_sb_MSS_CM3_hw_platform** debug mode to **Cortex-M3_SRAM** by selecting **Cortex-M3_SRAM** from the drop-down list, as shown in Figure 28.

Figure 28 • Cortex-M3_SRAM Settings



This tutorial uses `printf` statements to display memory read data. Redirection of the output of `printf()` to a UART is enabled by adding the **MICROSEMI_STDIO_THRU_UART** symbol.

Follow the steps to add the MICROSEMI_STUDIO_THRU_UART symbol:

- a. Right-click **Cortex - M3 SRAM** under **SPI_Flash_sb_MSS_CM3_hw_platform** and click **Options for SPI_Flash_sb_MSS_CM3_hw_platform - Target Cortex - M3 SRAM**.

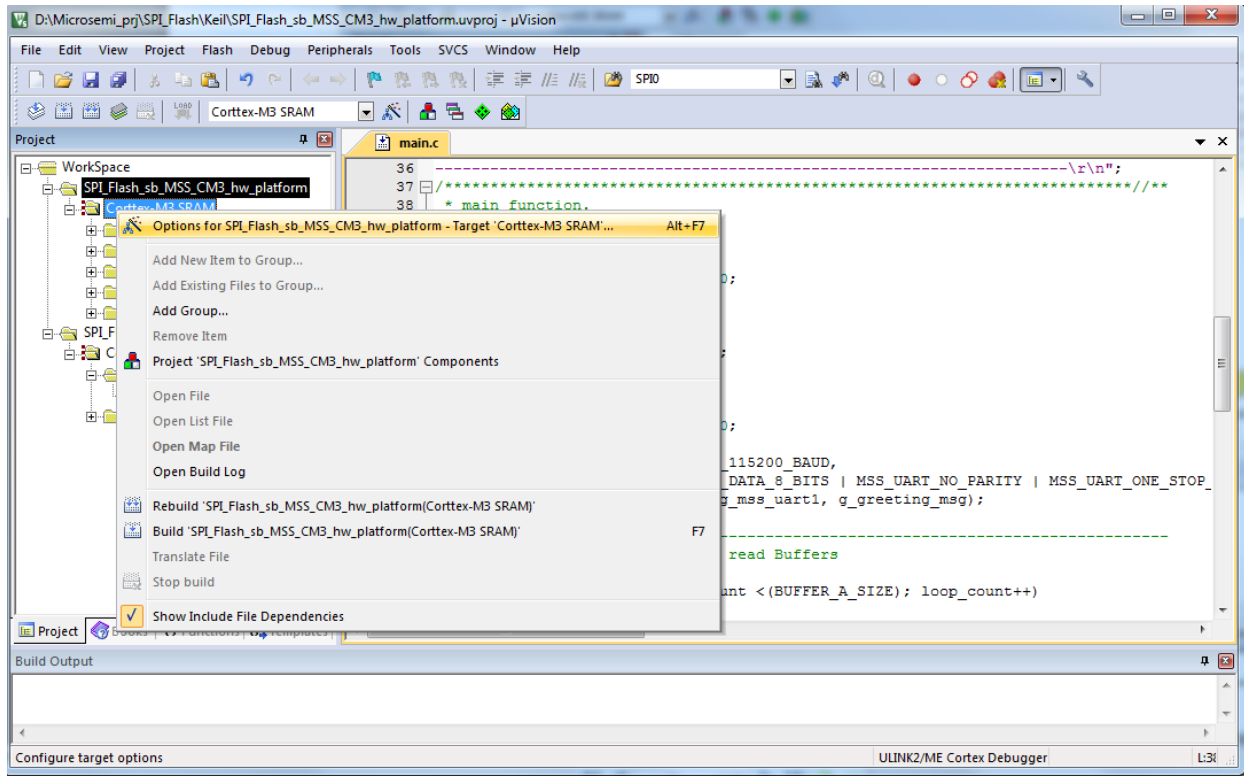
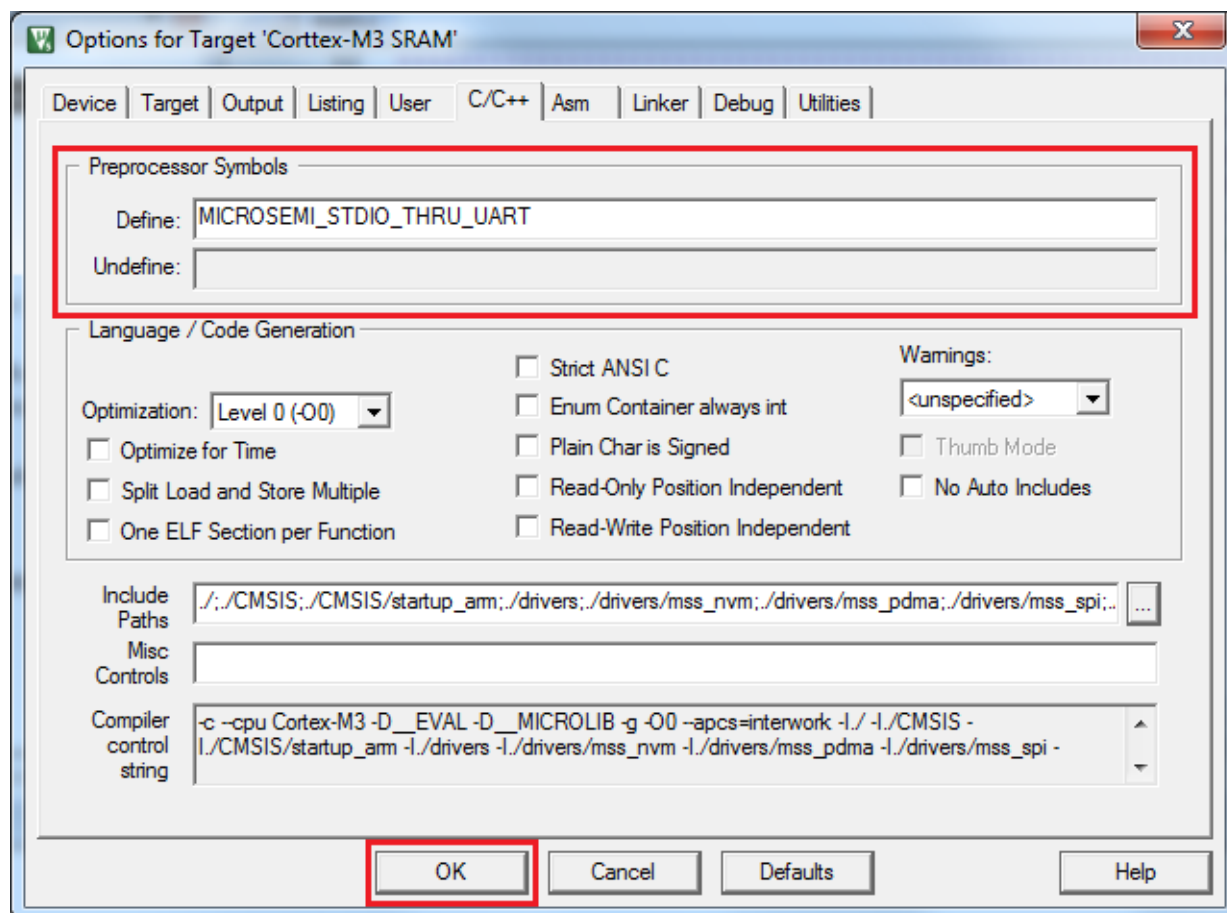


Figure 29 • Target Options

- b. Go to **C/C++** tab and enter **MICROSEMI_STDIO_THRU_UART** at **Define** under Preprocessor Symbols, as shown in Figure 30.

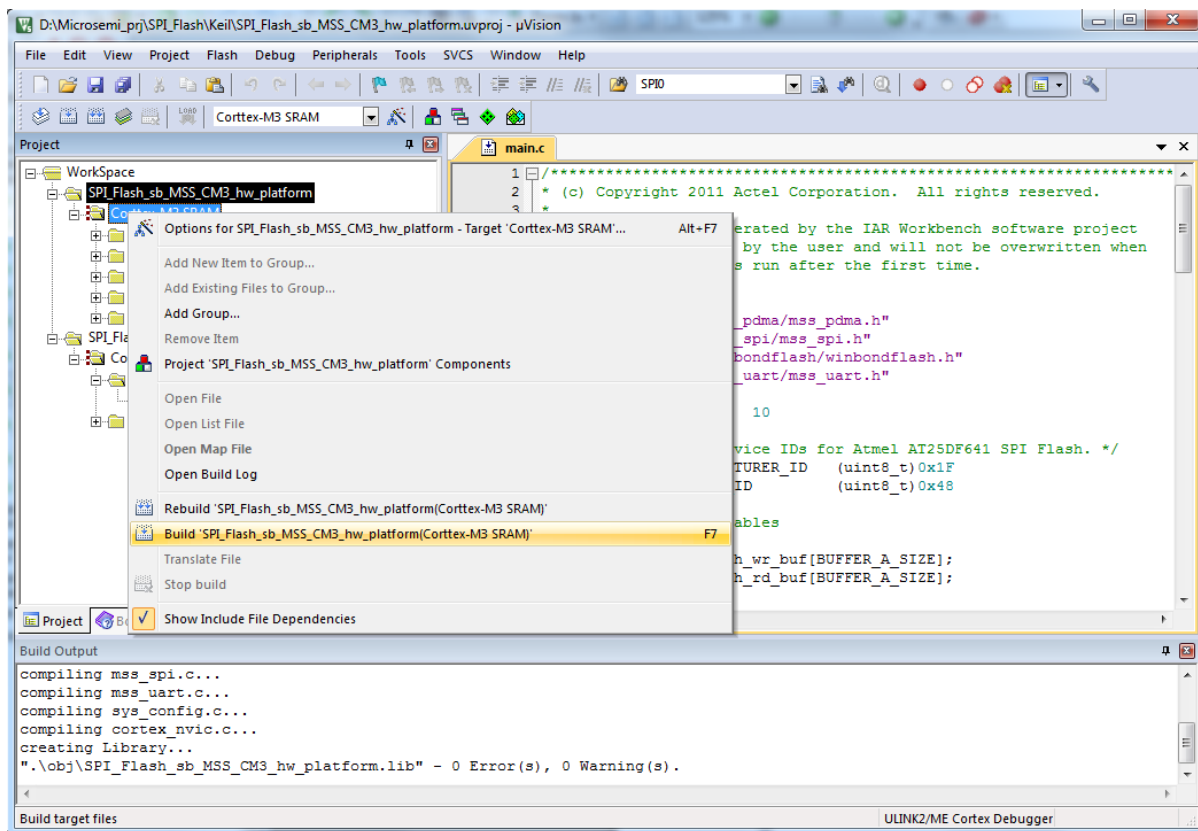
Figure 30 • Target Options-Adding Symbols



- c. Click **OK**.

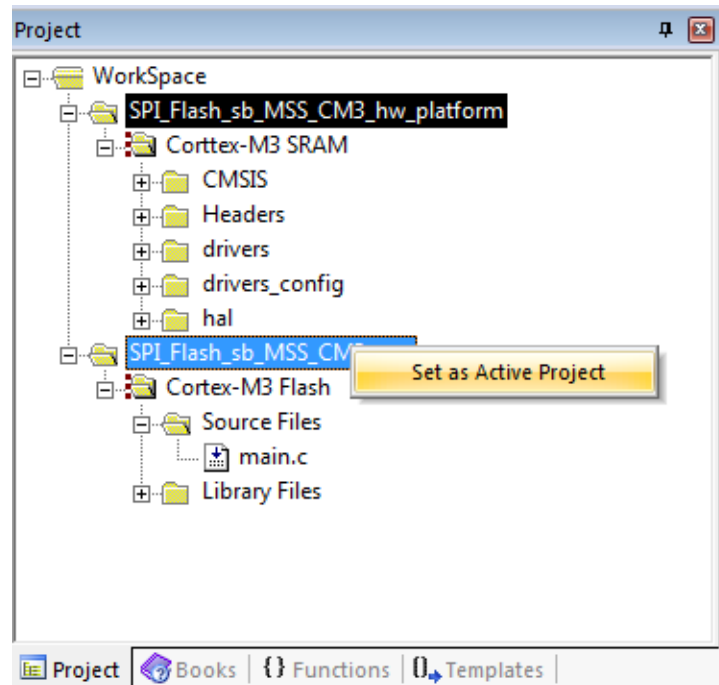
- Right-click **Cortex-CM3_SRAM** under **SPI_Flash_sb_MSS_CM3_hw_platform** and select **Build SPI_Flash_sb_MSS_CM3_hw_platform (Cortex-CM3 SRAM)**, as shown in Figure 31.

Figure 31 • Build HW Platform Window



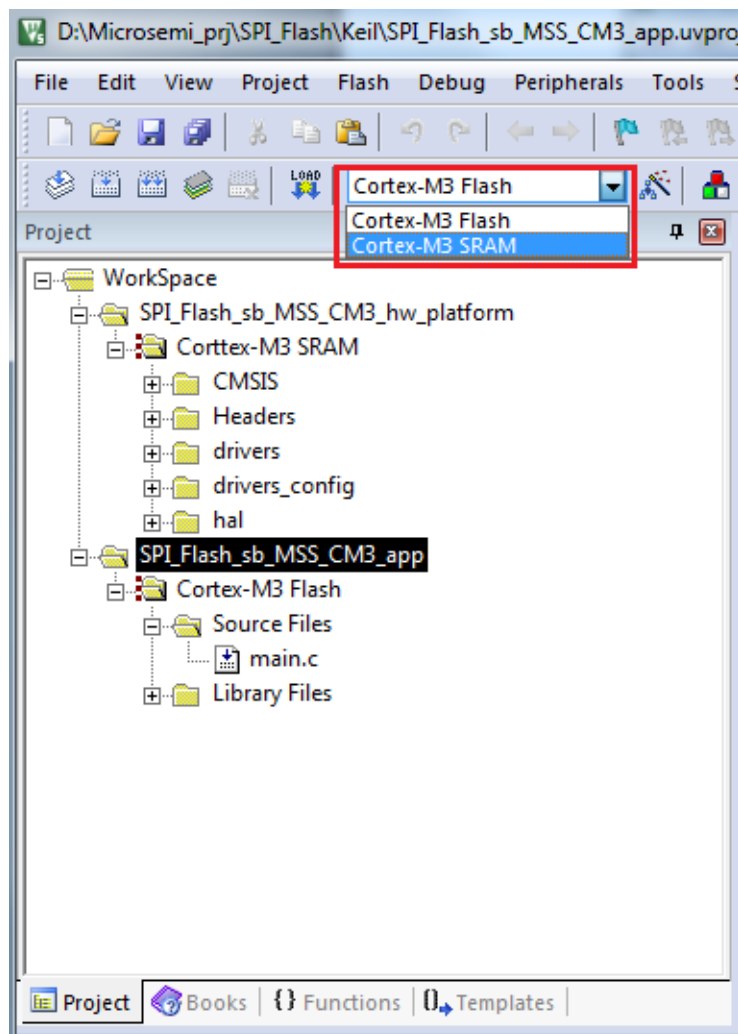
9. Right-click **SPI_Flash_sb_MSS_CM3_app** and select **Set as Active Project**.

Figure 32 • Set as Active Project



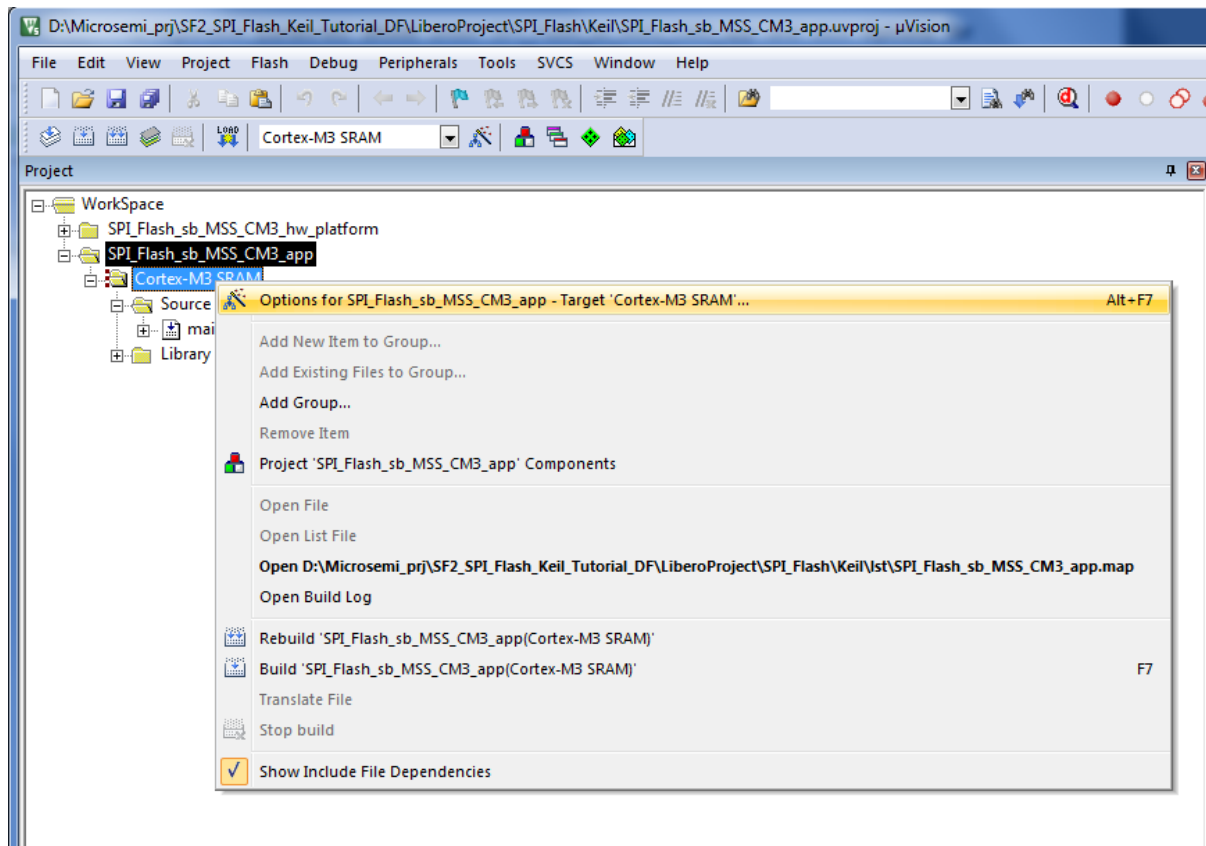
10. Change **SPI_Flash_sb_MSS_CM3_app** debug mode to **Cortex-M3_SRAM** by selecting **Cortex-M3_SRAM** from the drop-down list, as shown in Figure 33.

Figure 33 • Cortex-M3_SRAM Settings



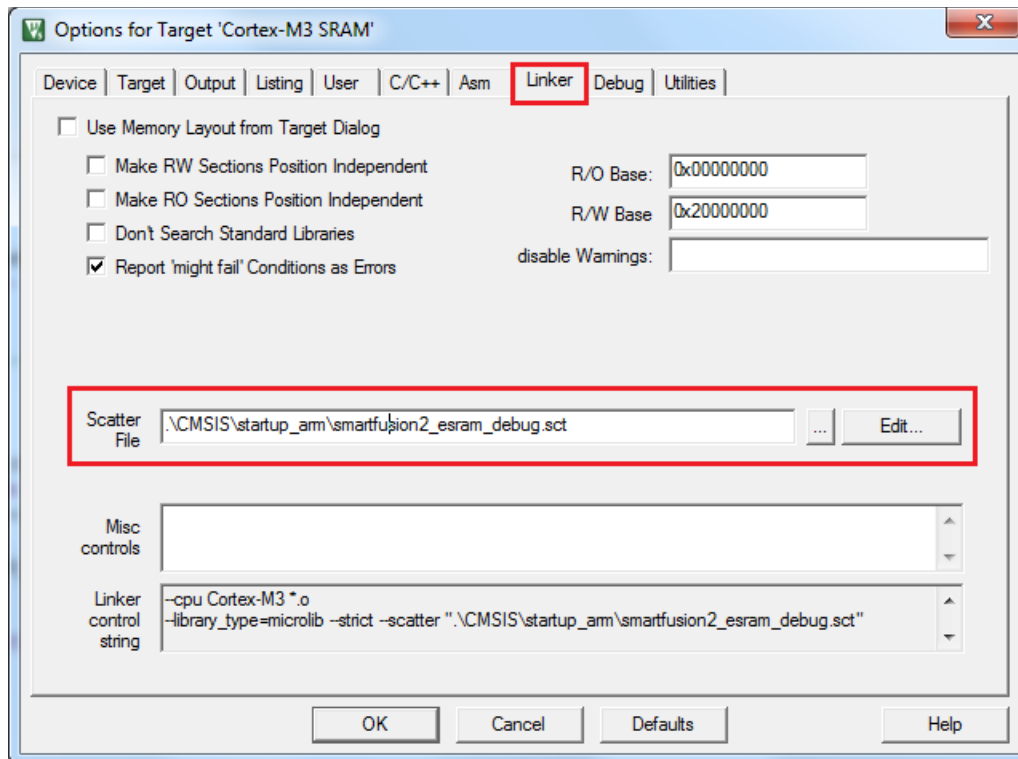
11. Right-click **Cortex-M3 SRAM** under **SPI_Flash_sb_MSS_CM3_app** and click **Options for project**.

Figure 34 • Target Options



12. Click the **Linker** tab and navigate to the
`SF2_SPI_Flash_Keil_Tutorial_DF\LiberoProject\Keil\CMSIS\startup_arm` folder to select the
Scatter File as `smartfusion2_esram_debug.sct`, as shown in Figure 35.

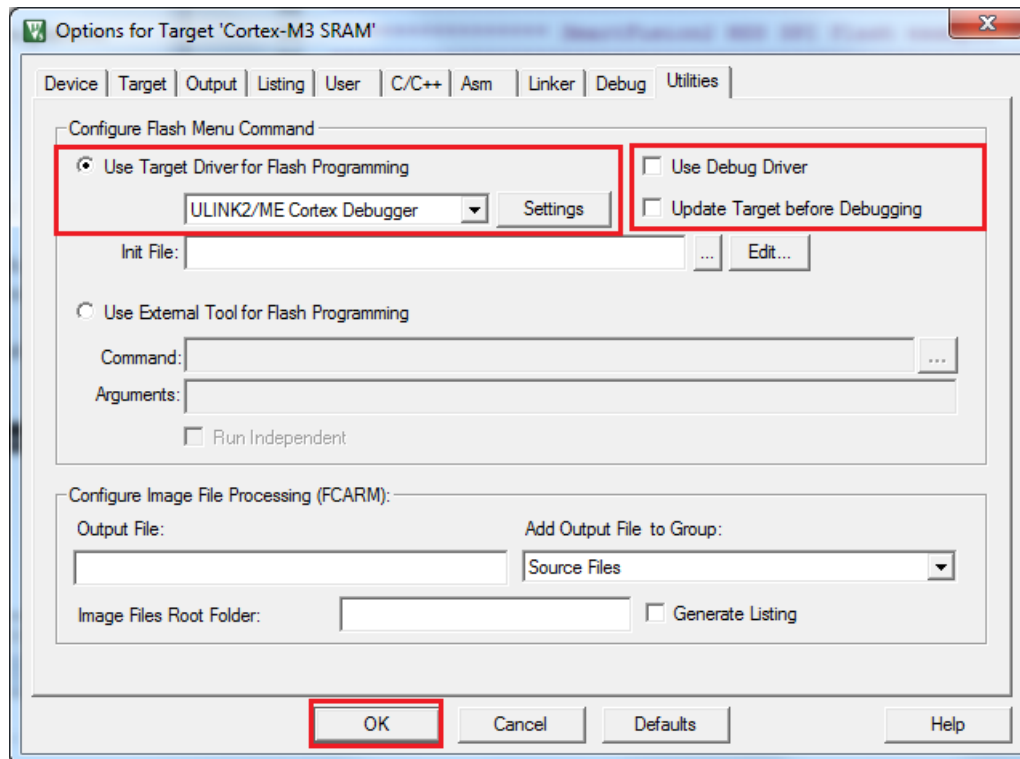
Figure 35 • Target Options - Scatter File



13. Click the **Utilities** tab and clear **Use Debug Driver** and **Update Target before Debugging** check boxes.

14. Select **ULINK2/ME Cortex Debugger** from the drop-down list and click **OK**, as shown in [Figure 36](#).

Figure 36 • Target Options - Utilities Settings



15. Right-click **Cortex-M3 SRAM** under **SPI_Flash_sb_MSS_CM3_app** and select **Build SPI_Flash_sb_MSS_CM3_app (Cortex-M3 SRAM)**, as shown in Figure 37. It compiles all of the source files and links the object files into an AXF file to debug. Ensure that there are no errors. Correct syntax errors, if any and rebuild if necessary.

Figure 37 • Build Application Window

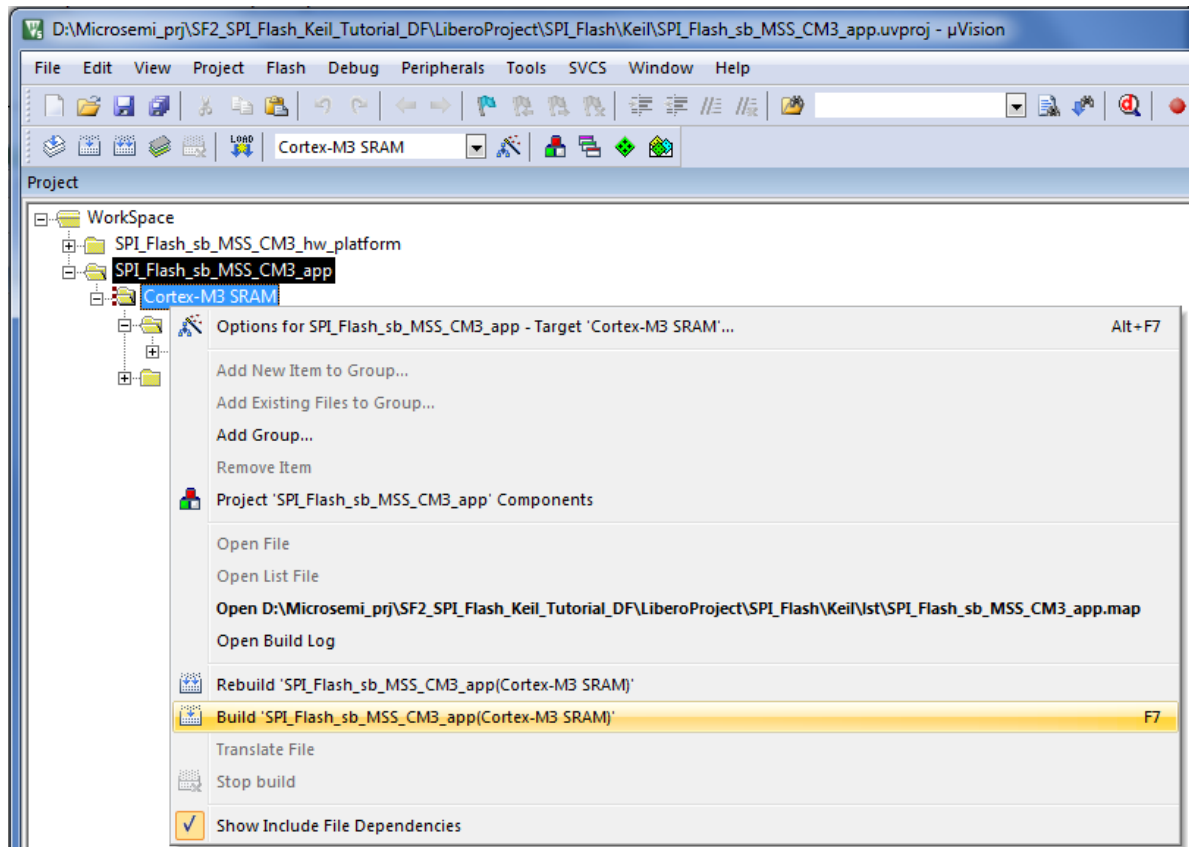
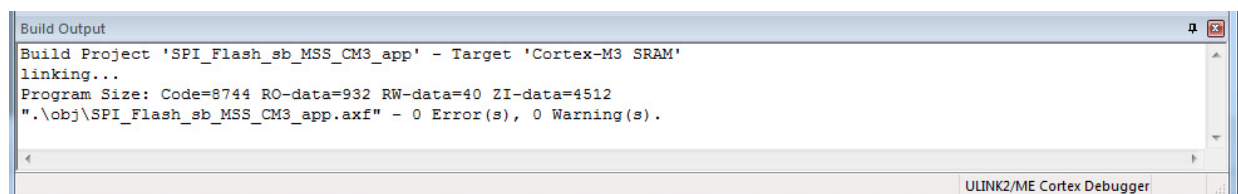


Figure 38 displays the messages in the console after the build.

Figure 38 • Build Output

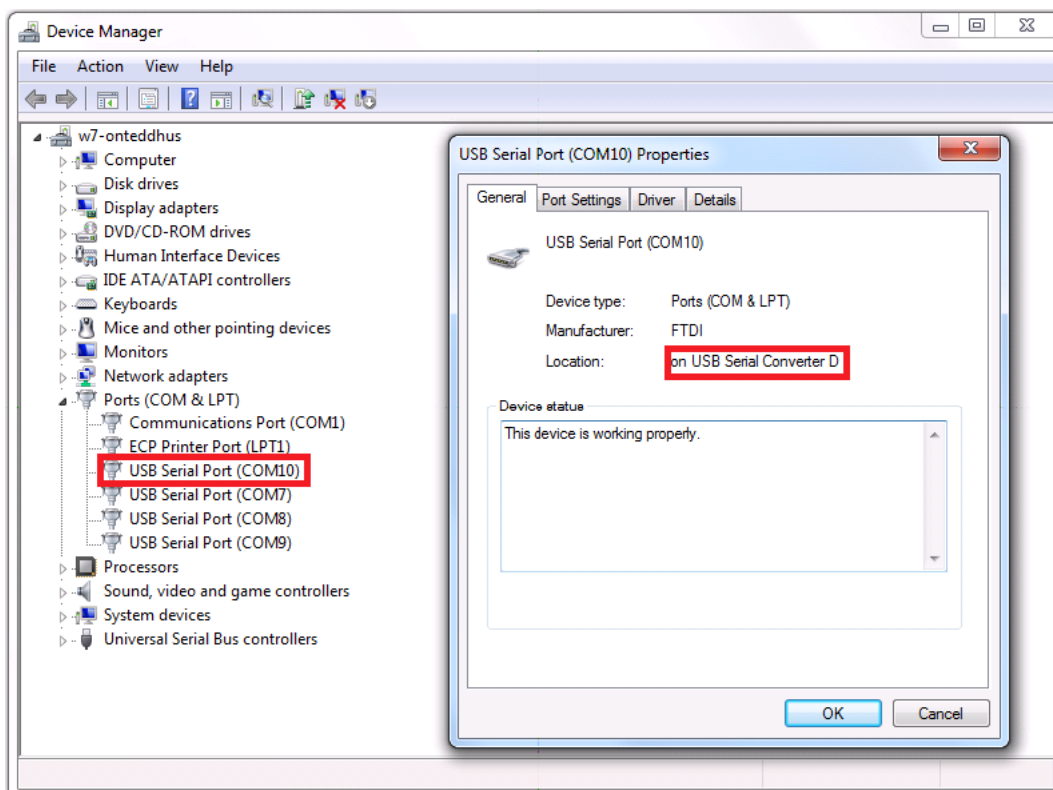


2.9 Step 6: Configuring Serial Terminal Emulation Program

The following steps describe how to configure serial terminal emulation program:

1. Install the USB driver. For serial terminal communication through the FTDI mini USB cable, install the FTDI D2XX driver. Download the drivers and the installation guide from:
www.microsemi.com/soc/documents/CDM_2.08.24_WHQL_Certified.zip
2. Connect the host PC to the J18 connector using the USB Mini-B cable. The USB to UART bridge drivers are automatically detected. Of the four COM ports, select the one with Location as **on USB Serial Converter D**. Figure 39 shows an example **Device Manager** window.

Figure 39 • Device Manager Window



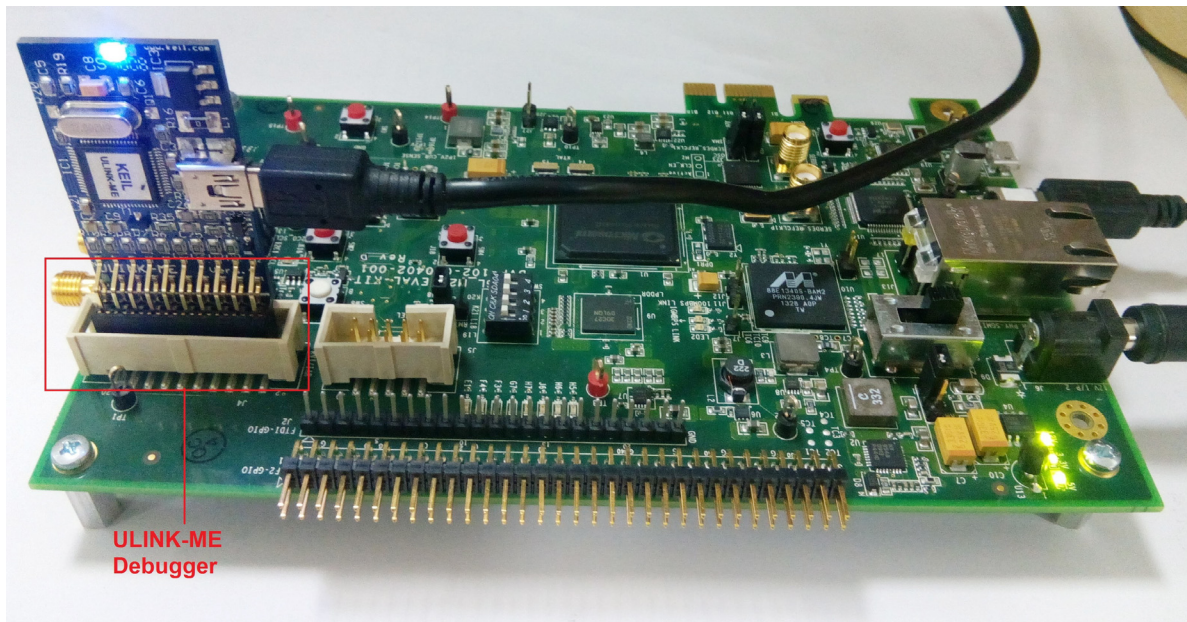
3. Start the HyperTerminal session. If the HyperTerminal program is not available in the computer, any free serial terminal emulation program such as PuTTY or TeraTerm can be used. Refer to the [Configuring Serial Terminal Emulation Programs Tutorial](#) for configuring the HyperTerminal, TeraTerm, or PuTTY.
The HyperTerminal settings are as follows:
 - 115200 baud rate
 - 8 data bits
 - 1 stop bit
 - No parity
 - No flow control

2.10 Step 7: Connecting the ULINK-ME to the Board and PC

The following steps describe the connection between the SmartFusion2 Security Evaluation Kit board, ULINK-ME, and host PC. Use the appropriate settings for the board that is in use.

1. Connect Pin 2 and Pin 3 on the jumper J8 on the SmartFusion2 Security Evaluation Kit board.
2. Connect the USB A-Mini B cable between the host PC and the SmartFusion2 Security Evaluation Kit board. This is used to display the HyperTerminal communications.
3. Verify that the ULINK-ME debugger is connected to the SmartFusion2 Security Evaluation Kit board RVI Header, as shown in [Figure 40](#) and also to the host PC through a USB A-Mini B cable. The ULINK-ME adapter has one LED that indicates connection status in the following ways:
 - Blinking slowly indicates that ULINK-ME is ready to communicate with the debugger.
 - Blinking speedily indicates that the target board is executing the program under debugger control.
 - Remaining **ON** during debugging indicates that the debugger has halted the target board.
 - Remaining **ON** during download indicates that target download and verification is in progress.
4. Switch **ON** the power supply switch, SW7.

Figure 40 • ULINK-ME Connections



Refer to "Appendix: Board Setup for Debugging from Keil uVision" on page 46 for information on the board setup for running the tutorial.

Figure 41 • ULINK-ME Debugger

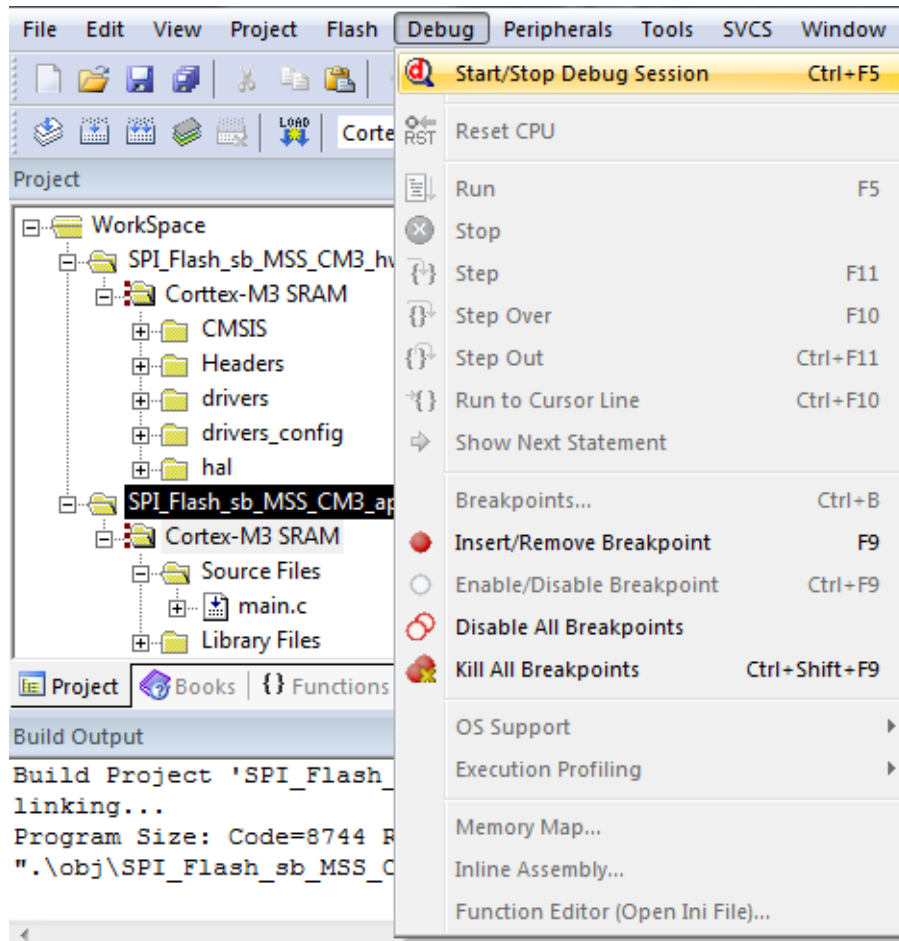


2.11 Step 8: Debugging the Application Project using Keil uVision 5

The following steps describe how to debug the application project using Keil uVision:

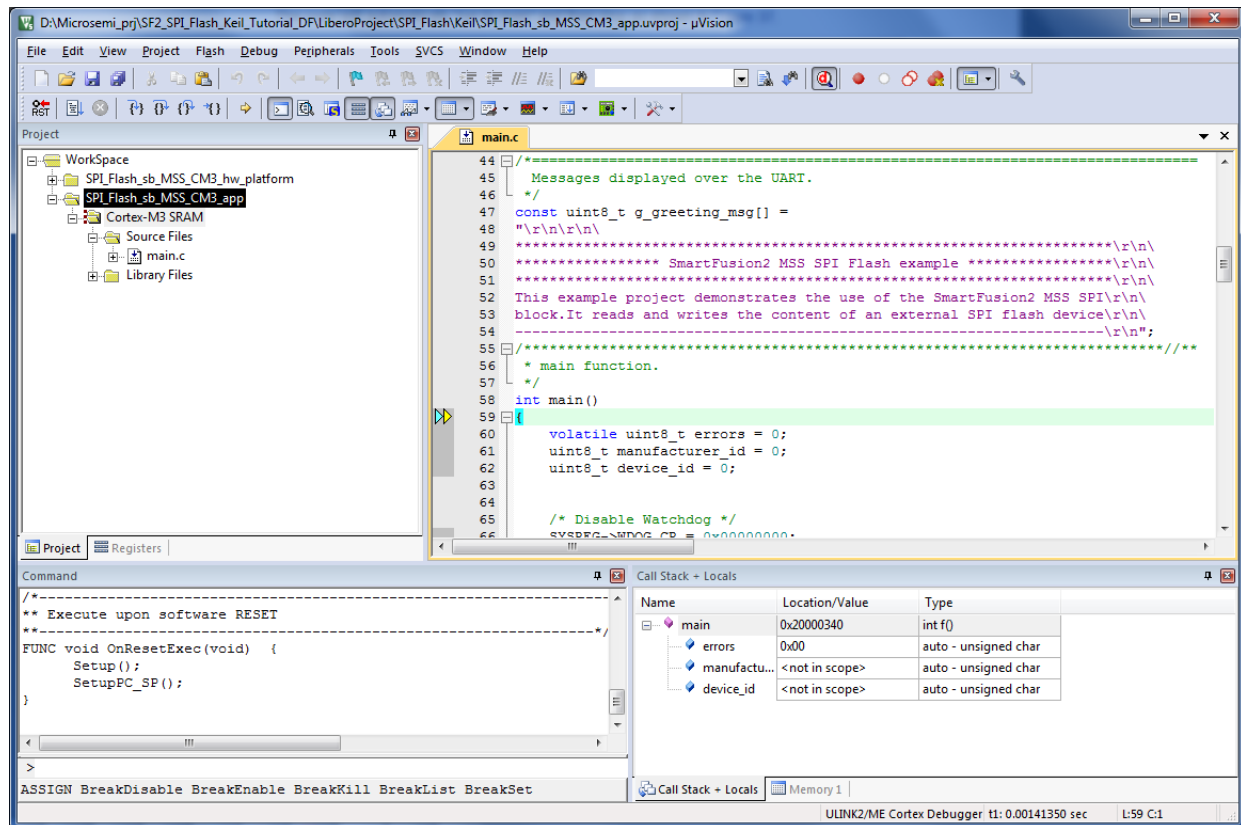
1. Select **Start/Stop Debug Session** from the **Debug** menu in the uVision window to run it through the debug hardware, as shown in Figure 42. The processor code is downloaded to the SmartFusion2 eSRAM.

Figure 42 • Selecting Start/Stop Debug Session



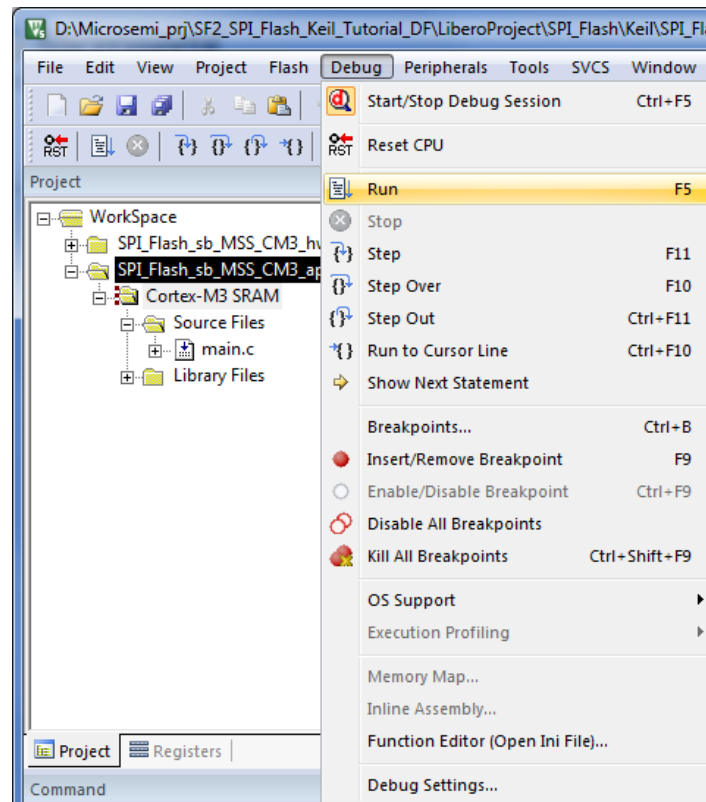
The code automatically runs in the `main.c` file, as shown in Figure 43.

Figure 43 • Debug Menu



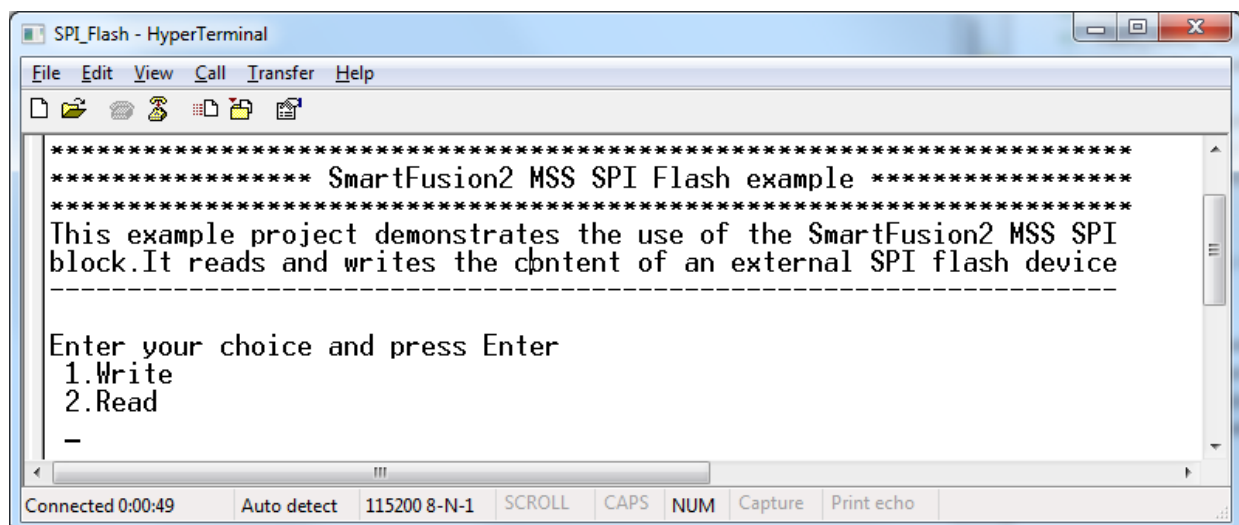
2. Click **Run** from the **Debug** menu, as shown in Figure 44.

Figure 44 • Selecting Run from the Debug Menu



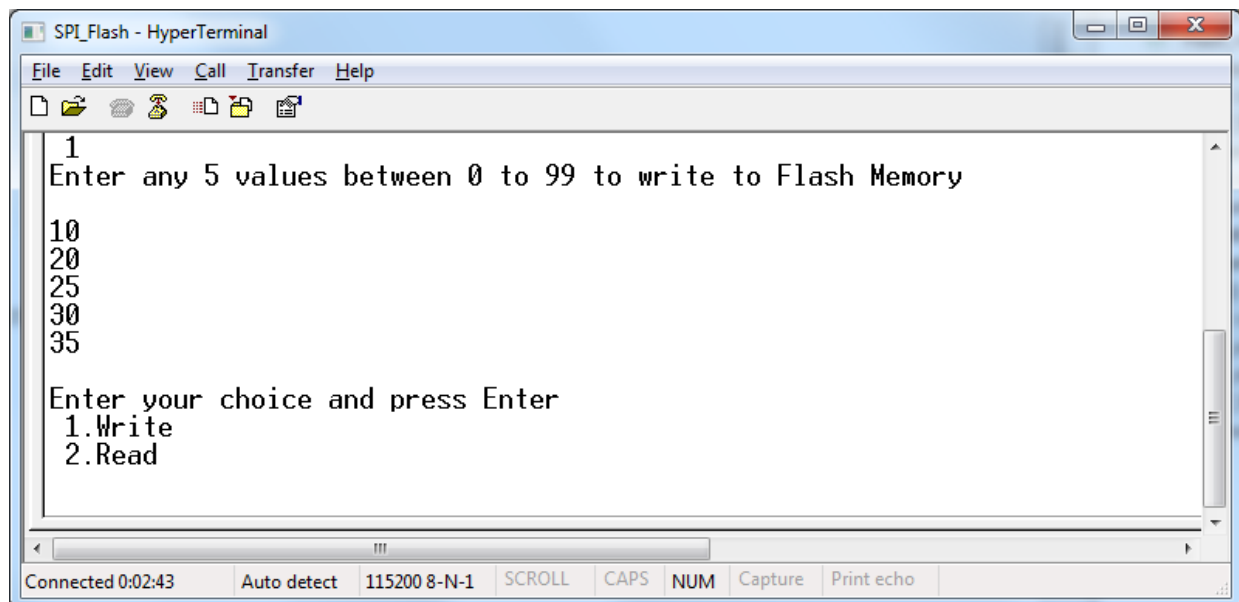
On successful operation, the HyperTerminal window displays a message, as shown in Figure 45.

Figure 45 • HyperTerminal Window



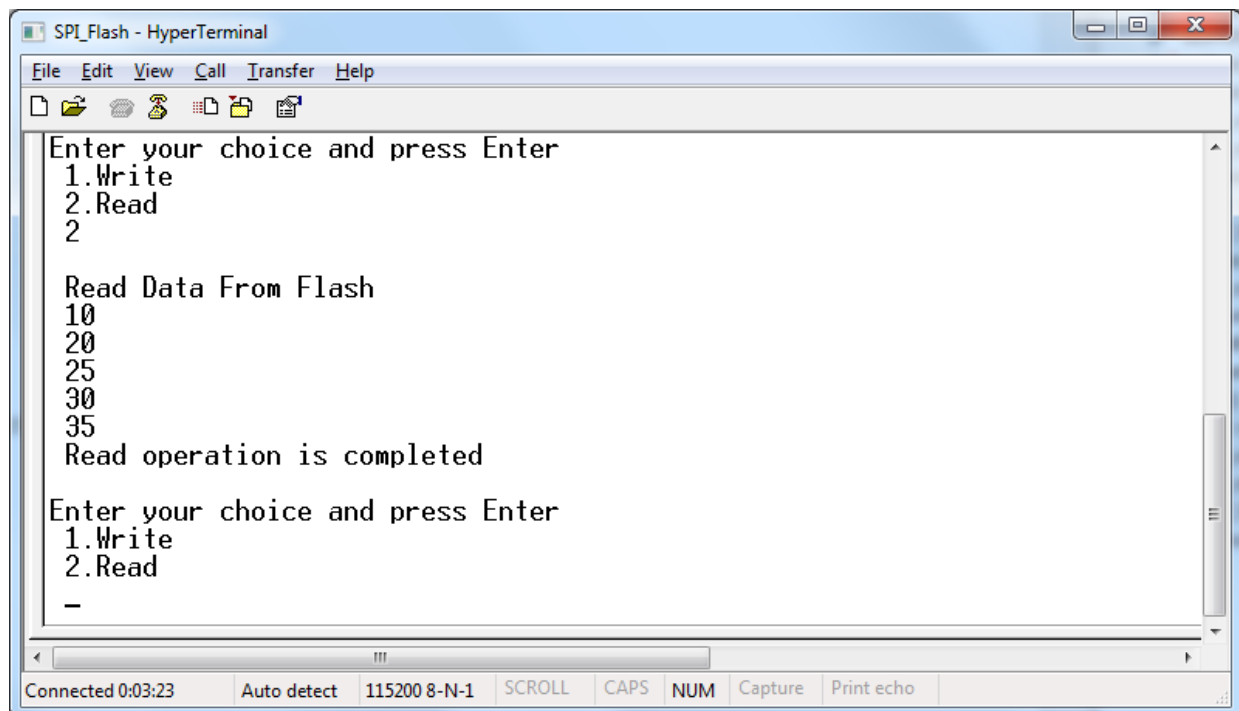
3. Select option 1 and enter values to write to the SPI Flash Memory, as shown in Figure 46.

Figure 46 • HyperTerminal Window - Option 1



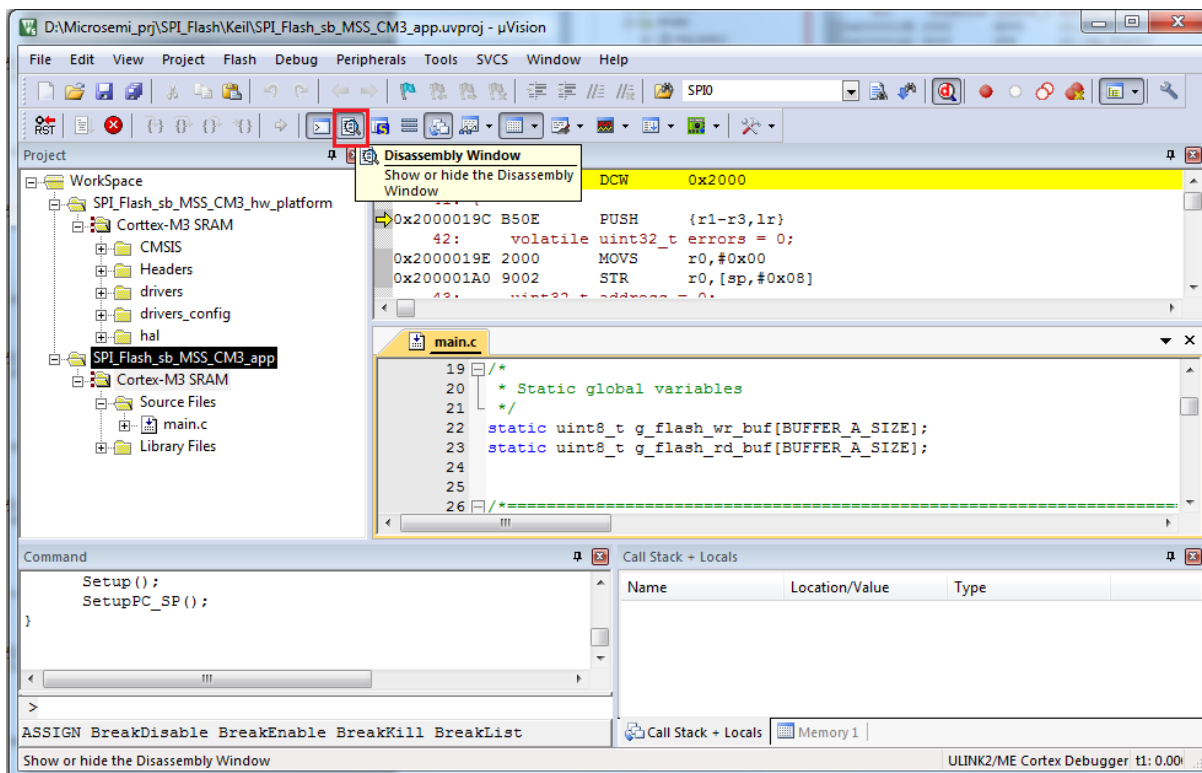
4. Select option 2 to read data from SPI Flash Memory, as shown in Figure 47.

Figure 47 • HyperTerminal Window - Option 2



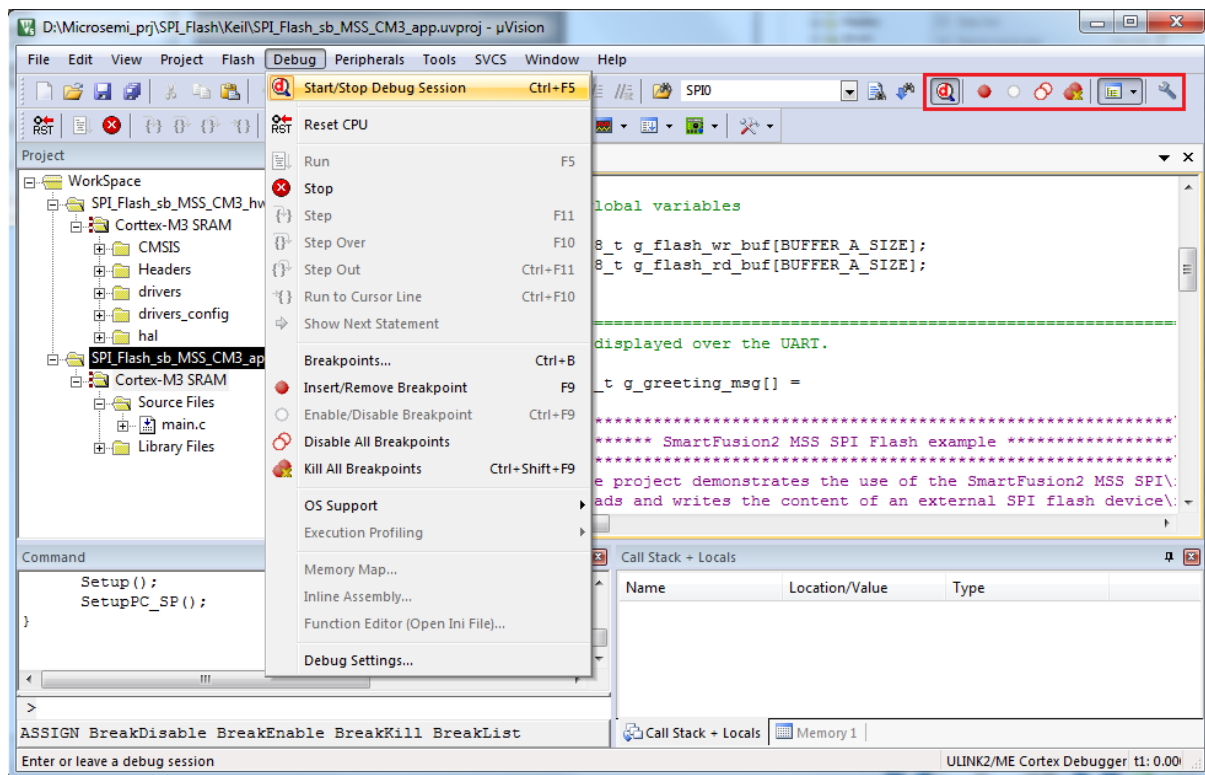
The **Disassembly** window is displayed in the middle of the **Debug** section, as shown in Figure 48. If not, click the **Disassembly** icon to display the **Disassembly** section.

Figure 48 • Disassembly Window



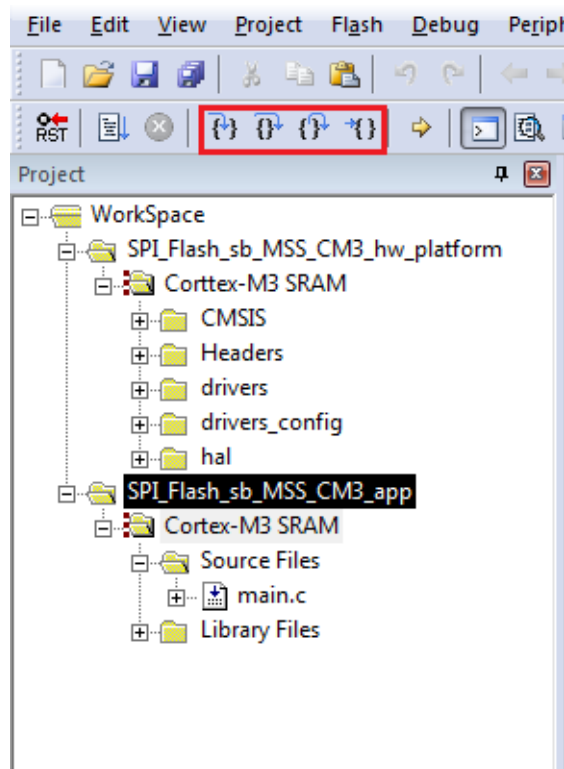
6. When the debug process is finished, terminate execution of the code by choosing **Debug > Start/Stop Debug Session**, as shown in Figure 50.

Figure 50 • Keil uVision Workbench - Stop Debug Option



7. The Step Level Debugging can be performed before running the application using **Run**. These can be accessed from the Debug menu or on the Keil uVision workbench, as shown in [Figure 51](#).

Figure 51 • Keil uVision Workbench - Step Level Debugging



- Source code can be single-stepped by selecting from the Debug menu **Debug > Step**, **Debug > Step Over**, **Debug > Step Out** or by selecting the respective options from the Keil uVision workbench, as shown in [Figure 51](#). Observe the changes in the source code window and disassembly section. Performing a step over provides an option for stepping over functions. The entire function is run, but there is no need to single-step through each instruction contained in the function.
 - Select **Debug > Step Out** to exit the instruction in stepping mode.
8. Add breakpoints from the **Debug** menu in workbench to force the code to halt, start Debug session, and then single-step and observe the instruction sequence.
 9. Close uVision using **File > Exit**.
 10. Close the HyperTerminal using **File > Exit**.

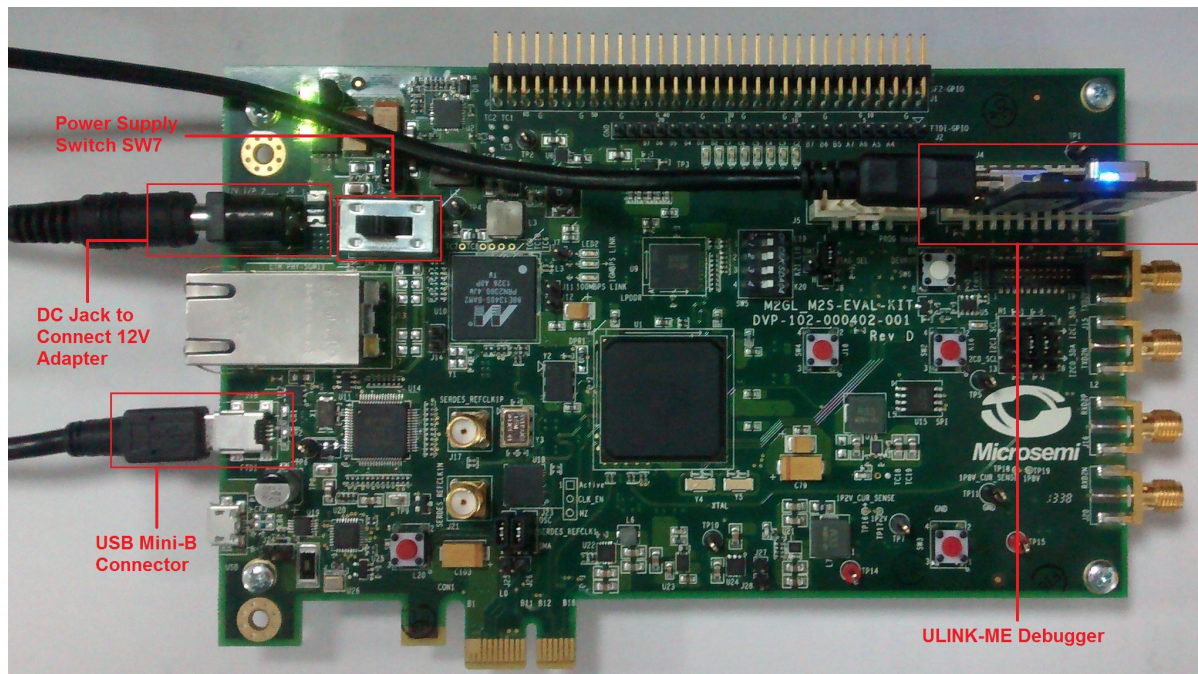
2.12 Conclusion

This tutorial provides steps to create a Libero SoC software design using the System Builder. It describes how to build, debug, and run Keil uVision application. It also provides a simple design to access the SPI flash.

3 Appendix: Board Setup for Debugging from Keil uVision

Figure 52 shows the board setup for debugging the Keil uVision on the SmartFusion2 Security Evaluation Kit board.

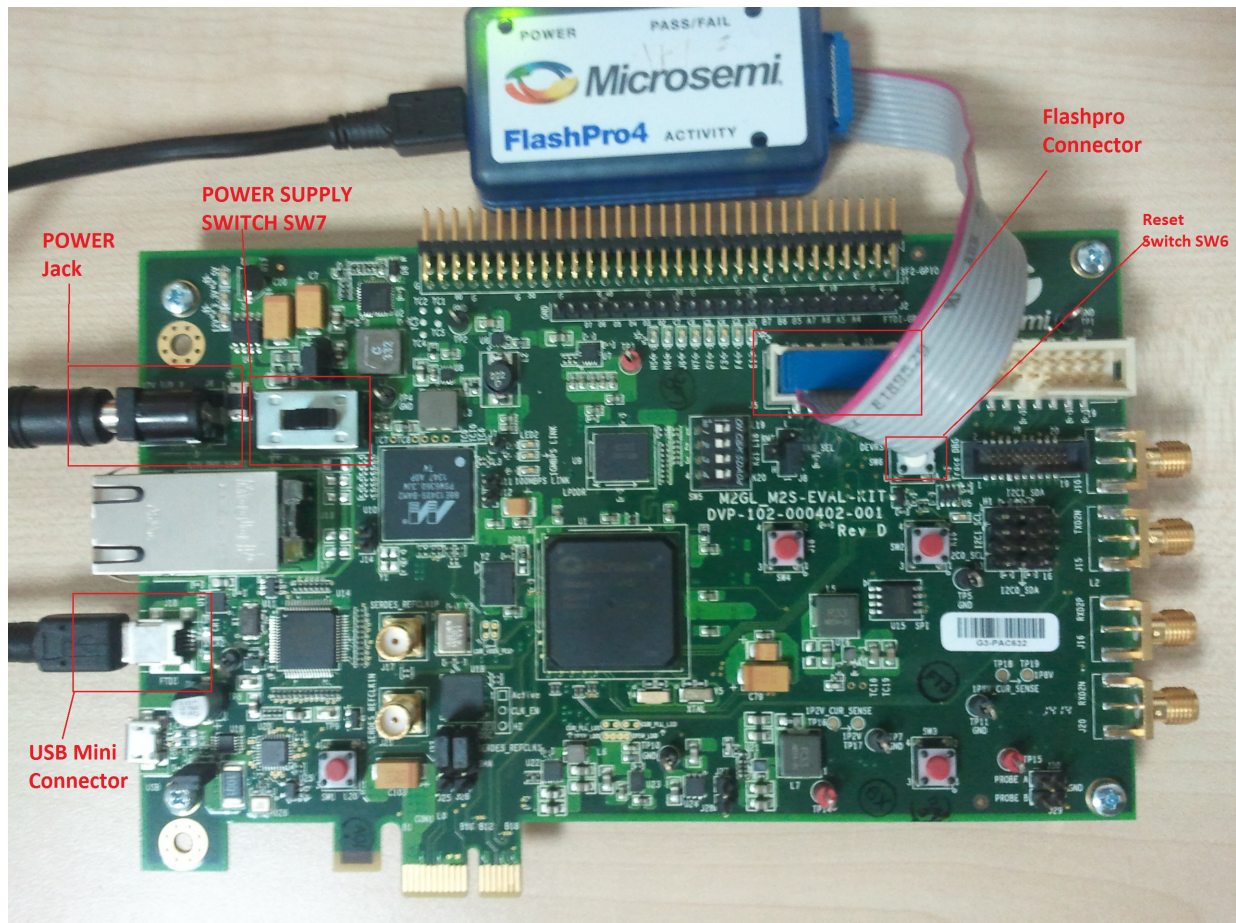
Figure 52 • SmartFusion2 Security Evaluation Kit in Debug Mode using Keil uVision



4 Appendix: Board Setup for Programming the Tutorial

Figure 53 shows the board setup for running the tutorial on the SmartFusion2 Security Evaluation Kit board.

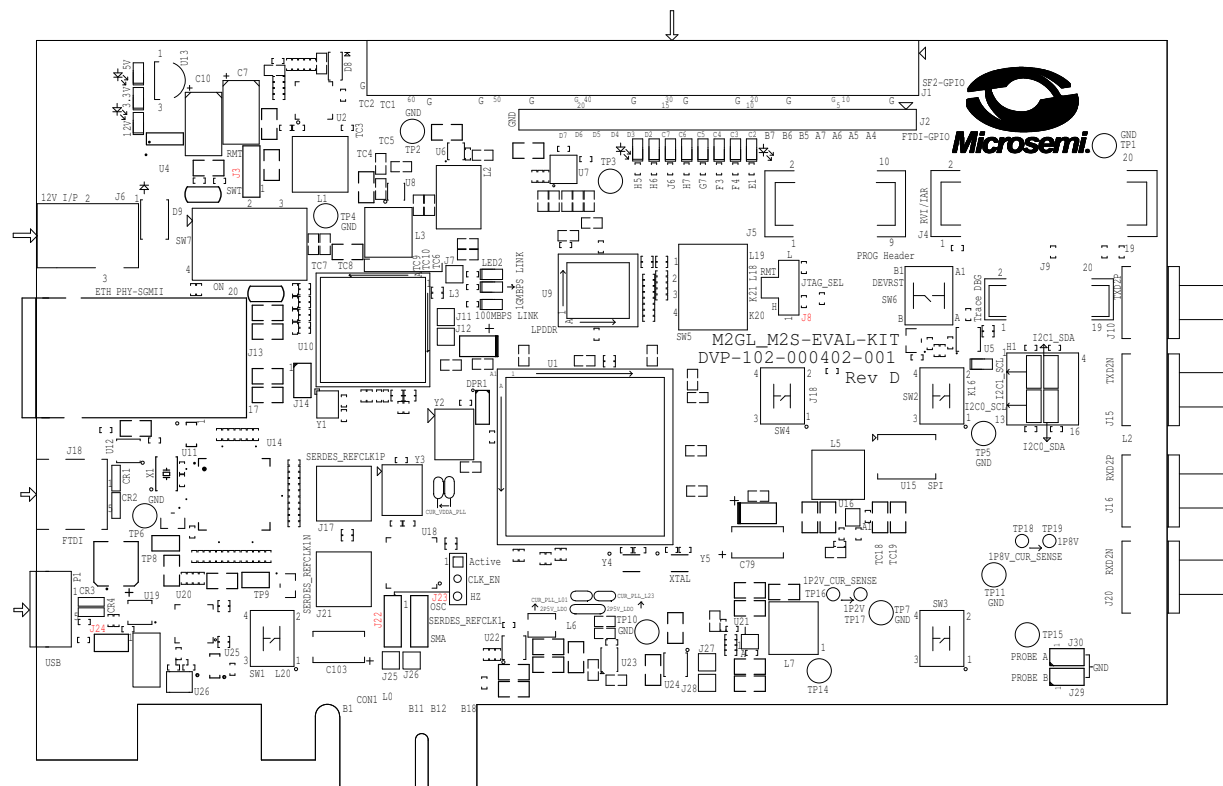
Figure 53 • SmartFusion2 Security Evaluation Kit in Programming Mode



5 Appendix: SmartFusion2 Security Evaluation Kit Board Jumper Locations

Figure 54 shows the jumper locations on the SmartFusion2 Security Evaluation Kit board.

Figure 54 • SmartFusion2 Security Evaluation Kit Board Jumper Locations



Notes:

- Jumpers highlighted in red (J22, J23, J24, J3, J8) are set by default.
- The location of the jumpers in [Figure 1](#) are searchable.

6 Revision History

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

Revision	Changes
Revision 5 (March 2016)	Updated the document for Libero SoC v11.7 software release changes (SAR 77347).
Revision 4 (October 2015)	Updated the document for Libero SoC v11.6 software release changes (SAR 72567).
Revision 3 (March 2015)	Updated the document for Libero SoC v11.5 software release (SAR 64189).
Revision 2 (November 2014)	Updated the document for Libero SoC v11.4 software release (SAR 61938).
Revision 1 (April 2014)	Initial release.

7 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.

7.1 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

7.2 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.

7.3 Technical Support

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

7.4 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>.

7.5 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.

7.5.1 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.

7.5.2 My Cases

Microsemi SoC Products Group customers may submit and track technical cases online by going to [My Cases](#).

7.5.3 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](#).

7.6 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

© 2016 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.

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.

Microsemi makes no warranty, representation, or guarantee regarding the information contained herein or the suitability of its products and services for any particular purpose, nor does Microsemi assume any liability whatsoever arising out of the application or use of any product or circuit. The products sold hereunder and any other products sold by Microsemi have been subject to limited testing and should not be used in conjunction with mission-critical equipment or applications. Any performance specifications are believed to be reliable but are not verified, and Buyer must conduct and complete all performance and other testing of the products, alone and together with, or installed in, any end-products. Buyer shall not rely on any data and performance specifications or parameters provided by Microsemi. It is the Buyer's responsibility to independently determine suitability of any products and to test and verify the same. The information provided by Microsemi hereunder is provided "as is, where is" and with all faults, and the entire risk associated with such information is entirely with the Buyer. Microsemi does not grant, explicitly or implicitly, to any party any patent rights, licenses, or any other IP rights, whether with regard to such information itself or anything described by such information. Information provided in this document is proprietary to Microsemi, and Microsemi reserves the right to make any changes to the information in this document or to any products and services at any time without notice.