TU0310
Tutorial
Interfacing User Logic with the Microcontroller Subsystem

Microsemi
a Microchip company

Microsemi 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
Email: sales.support@microsemi.com
www.microsemi.com

About Microsemi

Microsemi, a wholly owned subsidiary of Microchip Technology Inc. (Nasdaq: MCHP), offers a comprehensive portfolio of semiconductor and system solutions for aerospace & defense, communications, data center 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. Learn more at www.microsemi.com.

Contents

Figures

Tables

1 Revision History

The revision history describes the changes that were implemented in the document. The changes are listed by revision, starting with the most current publication.

1.1 Revision 14.0

The following is a summary of the changes made in this revision.

- Updated the document for Libero SoC v2021.1.
- Removed the references to Libero version numbers.

1.2 Revision 13.0

Updated the document for Libero v11.7 software release changes (SAR 76666).

1.3 Revision 12.0

Updated Figure 1 and Figure 43 (SAR 73892).

1.4 Revision 11.0

Updated the document for Libero v11.6 software release changes (SAR 72067).

1.5 Revision 10.0

Updated the document for Libero v11.5 software release (SAR 64506).

1.6 Revision 9.0

Updated the document for Libero v11.4 software release (SAR 59820).

1.7 Revision 8.0

Updated the document for Libero v11.3 software release (SAR 56454).

1.8 Revision 7.0

Updated the document (SAR 54212).

1.9 Revision 6.0

Updated the document for Libero version 11.2 (SAR 52904).

1.10 Revision 5.0

Updated the document for 11.0 production SW release (SAR 47302).

1.11 Revision 4.0

Updated the document for Libero 11.0 Beta SP1 software release (SAR 44868).

1.12 Revision 3.0

Updated the document for Libero 11.0 Beta SPA software release (SAR 42904).

1.13 Revision 2.0

Updated the document for Libero 11.0 Beta launch (SAR 41696).

1.14 Revision 1.0

Updated the document for LCP2 software release (SAR 38954).

1.15 Revision 1.0

Initial release.

2 Interfacing User Logic with the Microcontroller Subsystem

2.1 Introduction

This tutorial describes how to interface and handle communication between user logic in the Field Programmable Gate Array (FPGA) fabric and the SmartFusion®2 Microcontroller Subsystem (MSS). It also explains the Microsemi Libero® System-on-Chip (SoC) design software tool flow for designing applications for the SmartFusion2 System-on-Chip (SoC) FPGA family of devices.

A SmartFusion2 device has two fabric interface controllers (FIC_0 and FIC_1) as a part of the MSS. These FIC blocks provide a means of interfacing from the SmartFusion2 MSS AHB-Lite (AHBL) bus to user masters or user slaves in the FPGA fabric. Each FIC block performs an AHBL to AHBL or AHBL to APB3 bridging function between the AHB Bus Matrix and AHBL or APB3 bus in the FPGA fabric. Each FIC block provides two bus interfaces between the MSS and FPGA fabric. The first one is mastered by the MSS and has slaves in the FPGA fabric; the second one has a master in the fabric and slaves in the MSS. The bus interfaces to the FPGA fabric can be either 32-bit AHBL or 32-bit APB type. The FIC block provides registered bridging between the MSS AHBL interface and the FPGA fabric AHBL/APB circuitry to run at frequency ratios of 1:1, 2:1, 4:1, 8:1, 16:1, or 32:1. In AHB-Lite configuration, a bypass mode is provided, in which signals to and from the fabric are not registered and hence requires fewer clock cycles to complete each transaction. SmartFusion2 FIC has six memory regions. You can allocate a memory region to a particular FIC that is either to FIC_0 or FIC_1. Each memory region has a predefined memory map. For more information on FIC blocks in the Fabric Interface Controller chapter, refer to the *UG0331: SmartFusion2 Microcontroller Subsystem User Guide*.

This tutorial covers the following:

1. Creating a project for a SmartFusion2 SoC FPGA using the Microsemi Libero SoC toolset.
2. Using the SmartFusion2 System Builder to Configure MSS and generate a System Builder Component.
3. Configuring fabric interface controllers (FIC_0) to interface user logic in the fabric with the MSS.
4. Using on-chip oscillators and fabric CCC (FAB_CCC) for generating system clocks.
5. Writing a simple bus functional model (BFM) script for simulating the design.
6. Verifying the design by running BFM commands.
7. Generating the programming file to program the SmartFusion2 device.
8. Opening the project in SoftConsole from Libero SoC and writing the application code.
9. Validating the application design on the SmartFusion2 board.

2.2 Design Requirements

Table 1 • **Design Requirements**

Requirement	Version
Operating system	64 bit Windows 7 and 10
Hardware	
• SmartFusion2 Security Evaluation Kit Board • FlashPro4 programmer	Rev D or later
USB Cables	–
Host PC or Laptop	–
Software	
FlashPro Express	**Note:** Refer to the `readme.txt` file provided in the design files for the software versions used with this reference design.
Libero SoC	
SoftConsole	
Host PC Drivers	*USB Drivers*

Note: Libero SmartDesign and configuration screen shots shown in this guide are for illustration purpose only. Open the Libero design to see the latest updates.

2.3 Prerequisites

Before you begin:

Download and install Libero SoC (as indicated in the website for this design) on the host PC from the following location.

https://www.microsemi.com/product-directory/design-resources/1750-libero-soc

2.4 Design Files

You can download the associated design files for this tutorial from the Microsemi website:
http://soc.microsemi.com/download/rsc/?f=m2s_tu0310_df

The design files include the following:

- Source
- Solution
- Programming File
- Readme file

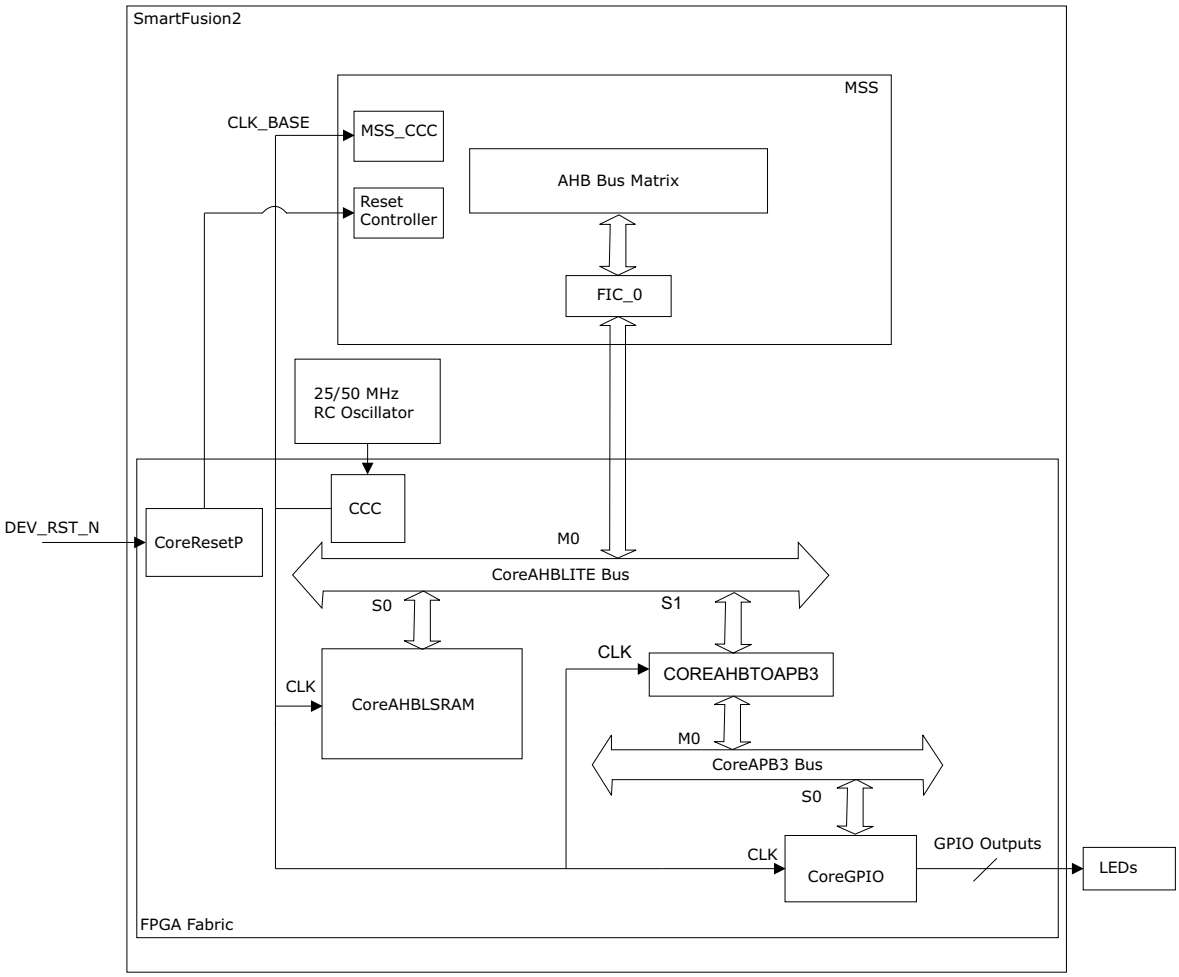
Refer to the `Readme.txt` file provided in the design files for the complete directory structure.

2.5 Design Description

The design uses the SmartFusion2 MSS block, one CCC block, an on-chip 25/50 MHz RC oscillator, and two different slaves in the FPGA fabric. The MSS FIC_0 is configured for the AHBL master interface is connected to the slaves CoreAHBLSRAM and CoreGPIO using the CoreAHBLite, CoreAHBTOAPB, and CoreAPB3 bus interfaces. Figure 1 shows the block diagram of the design. The ARM® Cortex®-M3 processor or any other MSS master can access these slaves through the FIC blocks. In this design, you can:

- Verify the bus read and write to the fabric peripherals from the MSS side using BFM models.
- Perform read and write to the CoreAHBLSRAM memory, configure the CoreGPIO block, and set GPIO outputs using a BFM script.
- Validate the bus read and write to the CoreAHBLSRAM, set the GPIO to blink the LEDs on the SmartFusion2 Security Evaluation Kit board.

Figure 1 • **Block Diagram of the Design**



2.5.1 Design Steps

The major steps to run this tutorial are as follows:

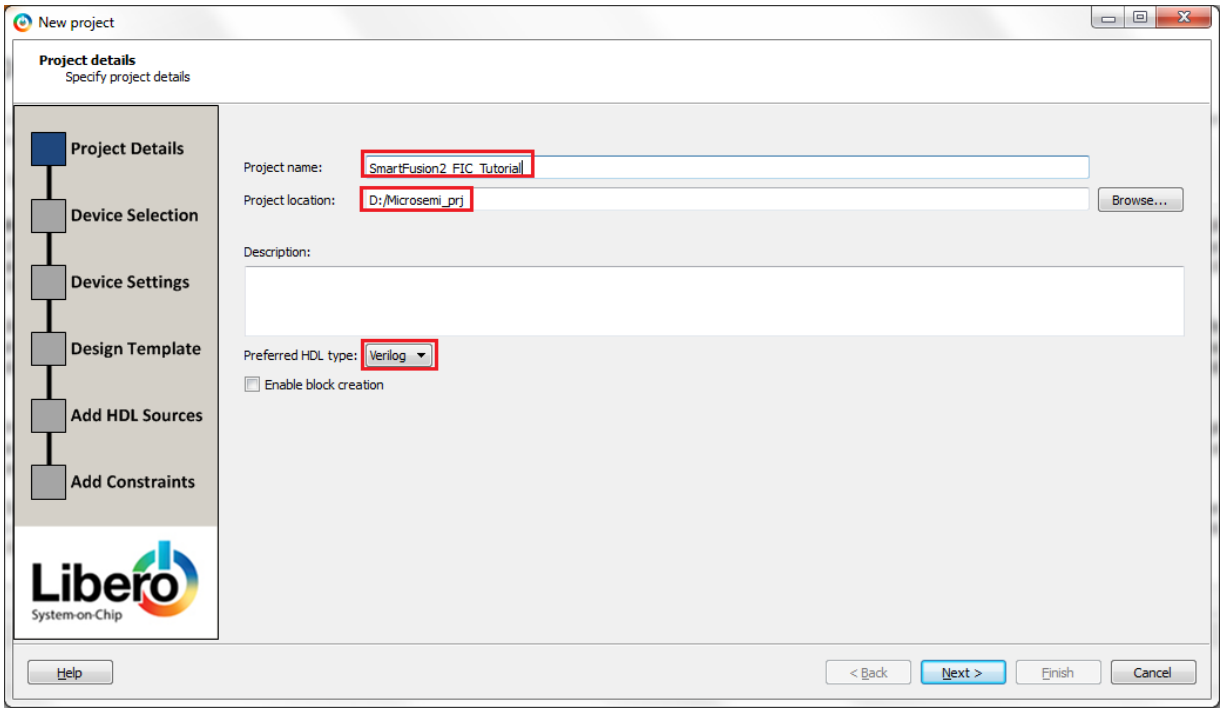
1. Creating a new Libero SoC project for the SmartFusion2 device.
2. Using the SmartFusion2 System Builder to configure the FIC blocks and clock.
3. Writing user BFM script to simulate a design.
4. Simulating the design using BFM Models and ModelSim.
5. Generate a programming file to program the SmartFusion2 SoC FPGA device.
6. Open the software project in SoftConsole and write the application program.
7. Run the design on the SmartFusion2 Security Evaluation Kit board.

2.6 Step 1: Creating a New Libero SoC Project

The following steps describe how to create a new Libero SoC project:

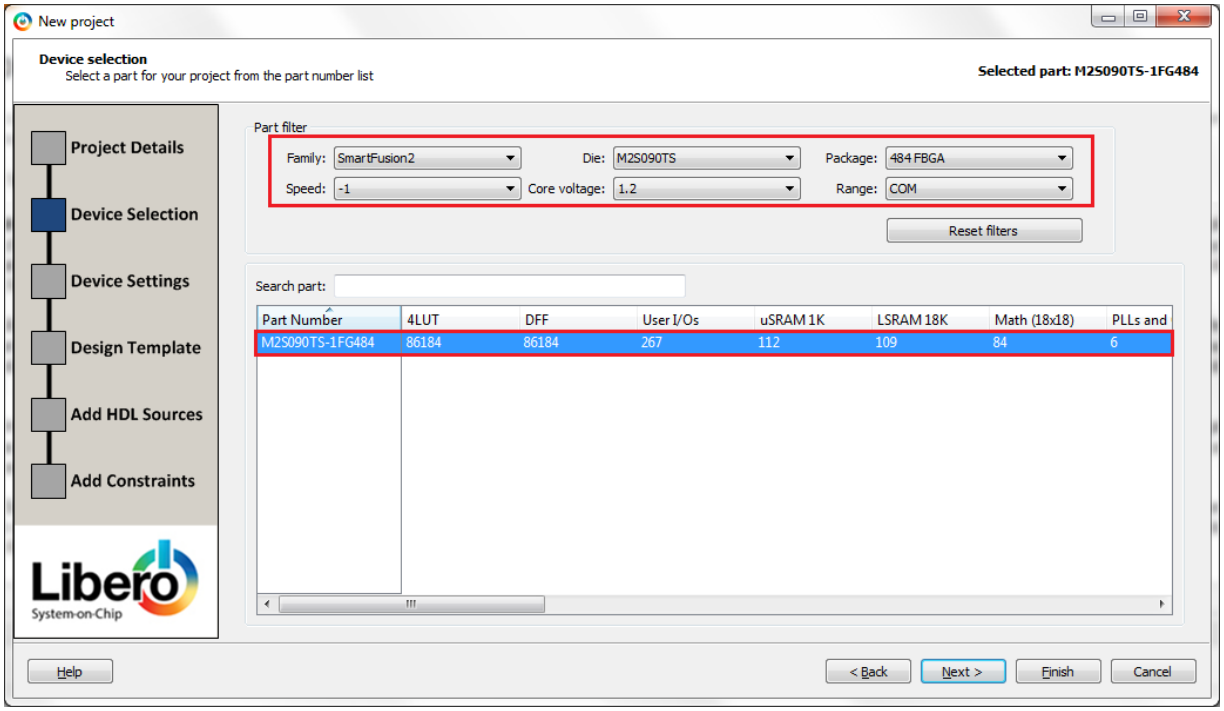
1. Open Libero SoC design software (**Start > Programs > Microsemi Libero SoC v(x.x) > Libero SoC v(x.x)**) or click the Libero SoC shortcut available on your desktop. The version number of the Libero SoC design software depends on the version that is installed on your PC. You can use either **v(x.x)** or the later.
2. Select **New Project** from the **Project** menu.
3. Enter the following **New Project** information, as shown in Figure 2, and click **Next**.
    - **Project name**: top
    - **Project location**: Select an appropriate location (for example, *D:/Microsemi_prj*)
    - **Preferred HDL type**: Verilog

Figure 2 • **Libero SoC New Project Dialog Box**



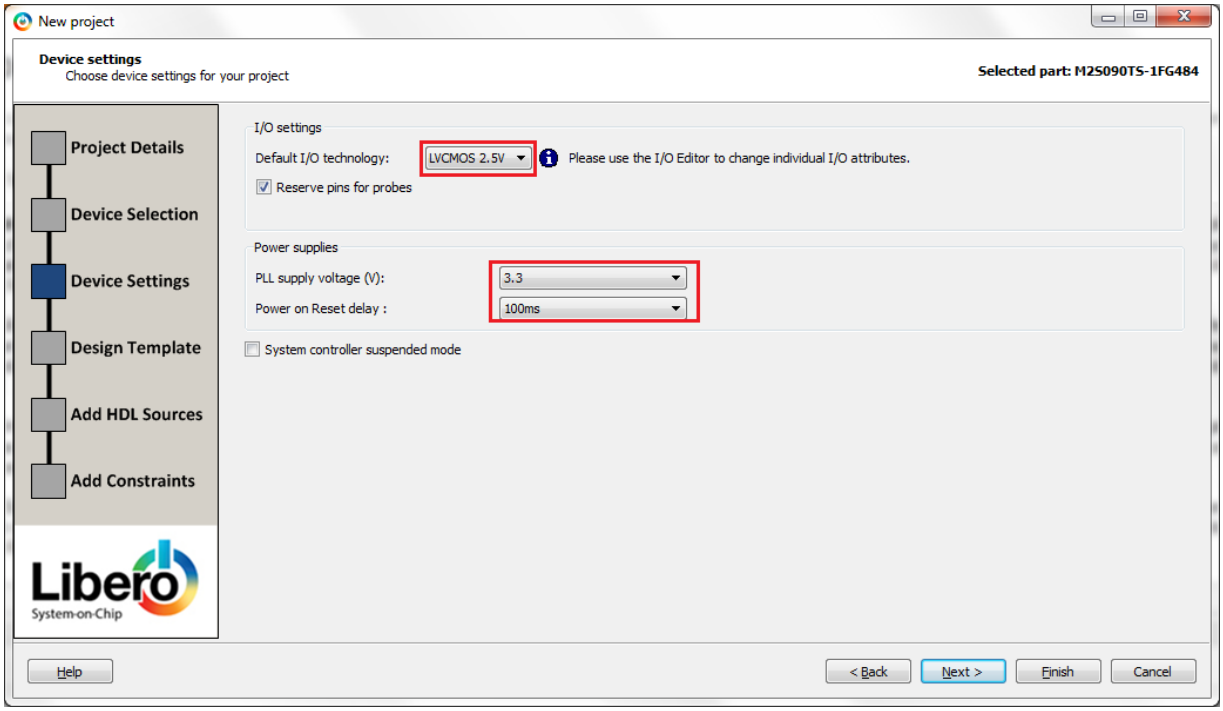
4. Select the following values from the drop-down lists for **Device Selection**, as shown in Figure 3, page 7.
    - **Family**: SmartFusion2
    - **Die**: M2S090TS
    - **Package**: 484 FBGA
    - **Speed**: -1
    - **Core voltage**: 1.2
    - **Range**: COM

Figure 3 • **New Project - Device Selection**



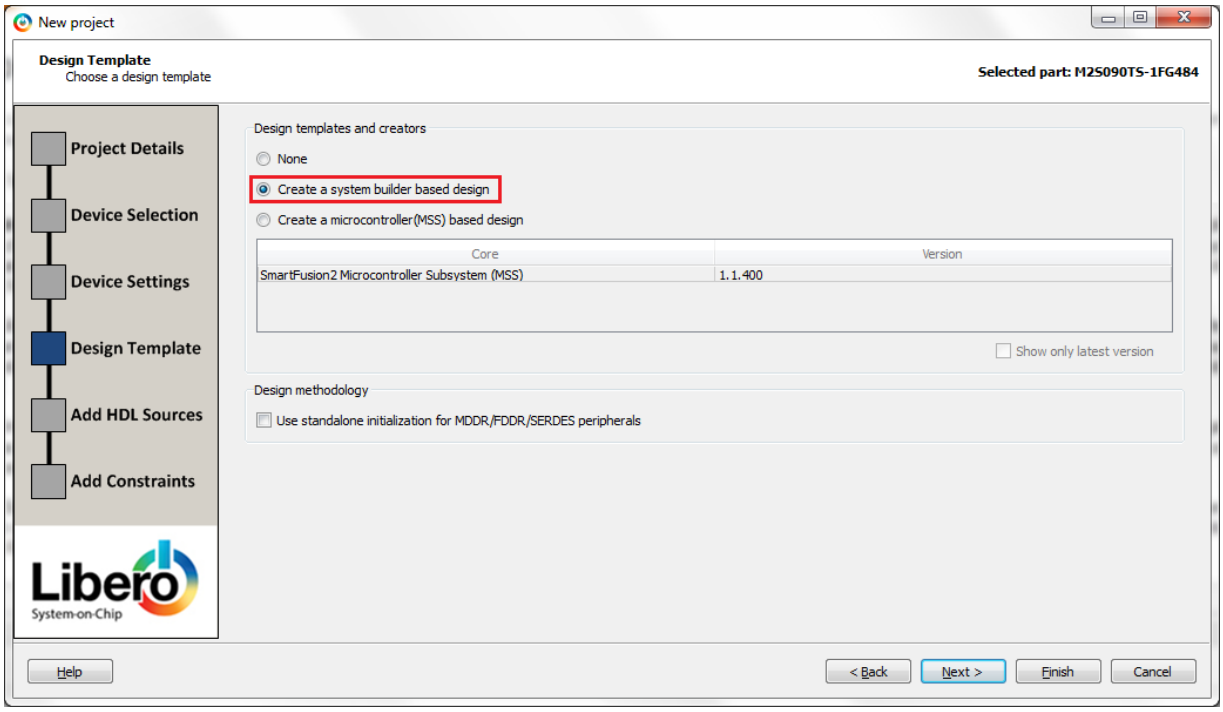
5. Click **Next**.
6. Select the information for **Device settings** as shown in Figure 4 and click **Next**.

Figure 4 • **New Project - Device Settings**



7. Select **Create a system builder based design** under **Design templates and creators,** as shown in Figure 5.

Figure 5 • **New Project - Design Template**

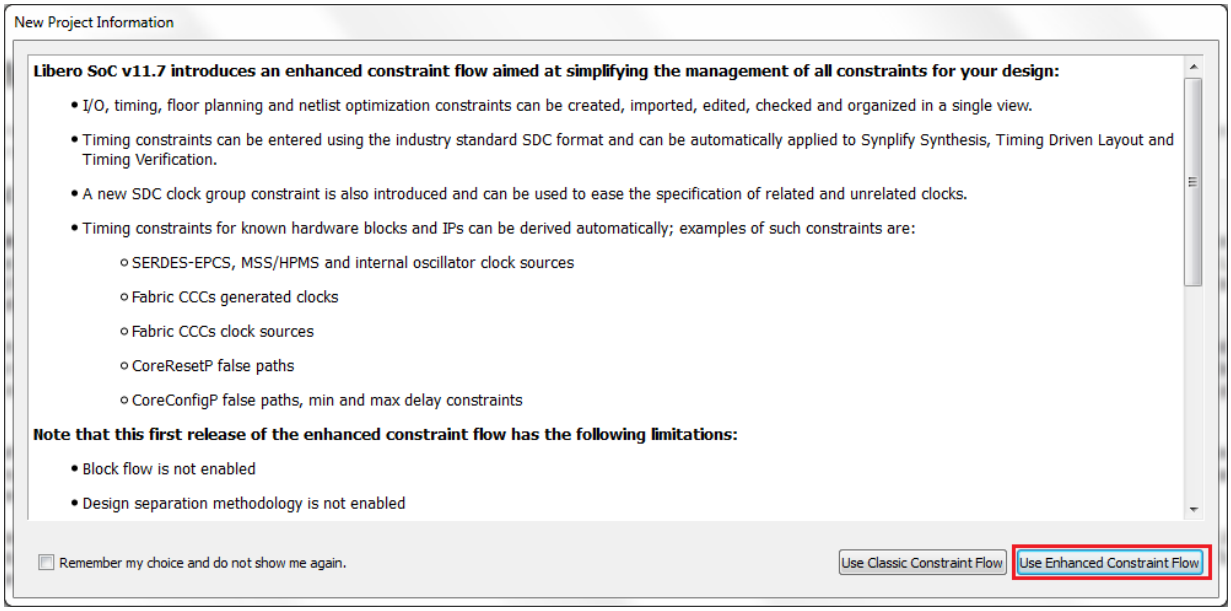


8. Click **Finish**.

   The **New Project Information** window is displayed, as shown in Figure 6. This tutorial uses the enhanced constraints flow of Libero v(x.x), which simplifies the management of all constraints (I/O, timing, floor planning, and netlist optimization constraints).

9. Select **Use Enhanced Constraints Flow**, as shown in Figure 6.

Figure 6 • **Using Enhanced Constraints Flow**

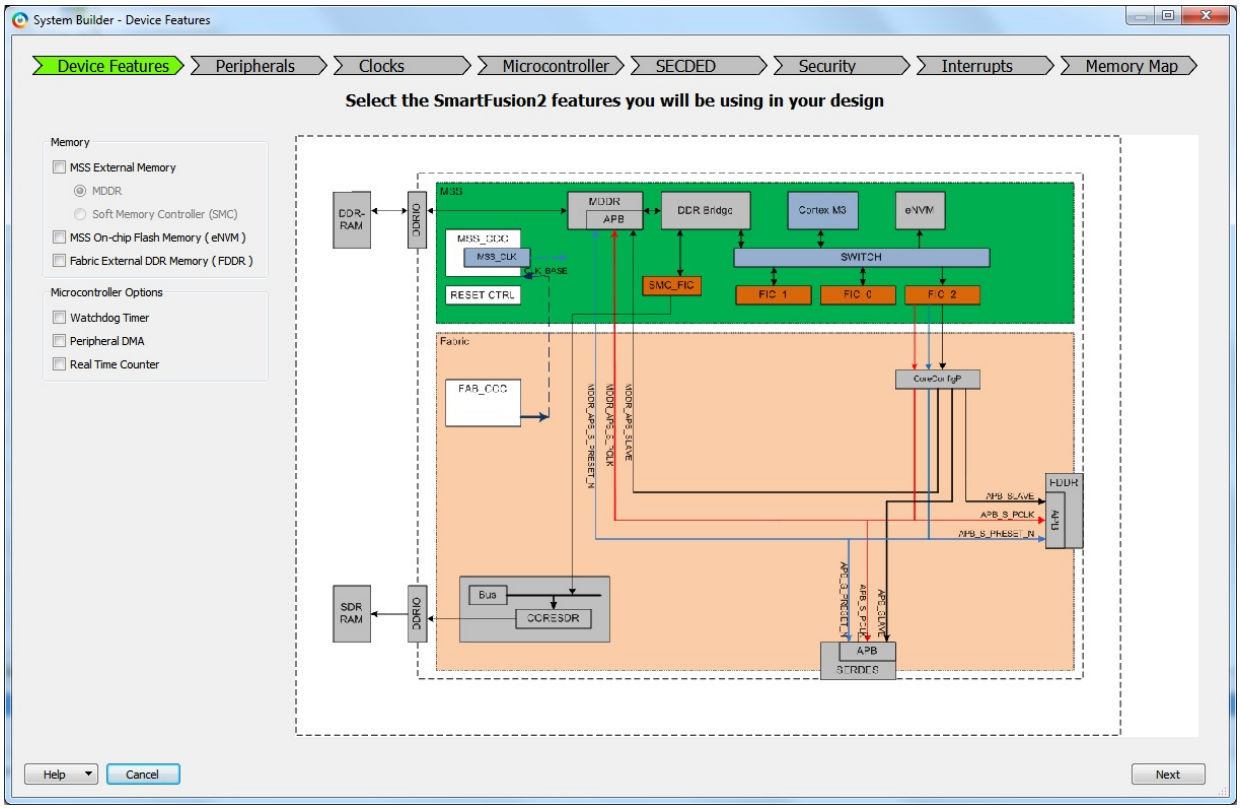


10. After selecting the Use System Builder, enter the name of the system in the Enter a name for your system dialog box, as shown in Figure 7.

Figure 7 • **Libero SoC Project Window**

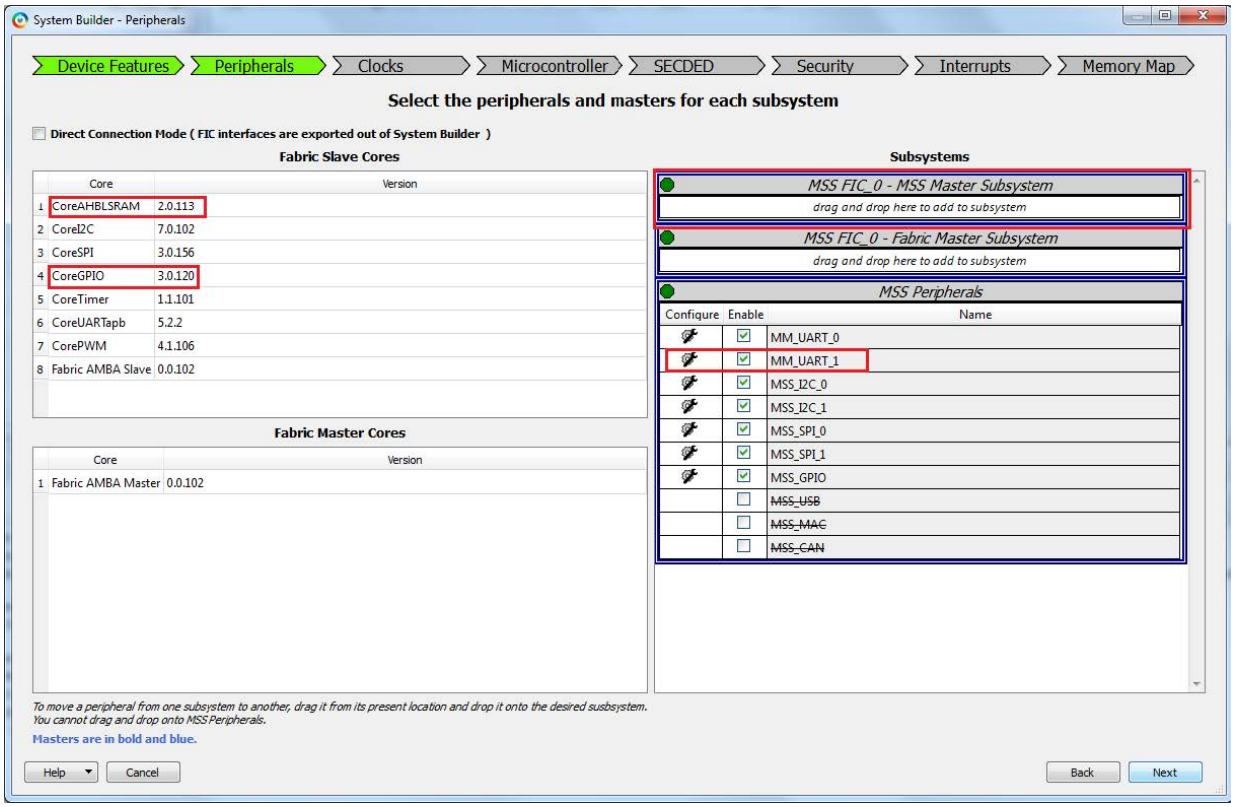
11. Enter **top** as the name of the system and click **OK**. The **System Builder** window is displayed, as shown in Figure 8.

Figure 8 • **SmartFusion2 System Builder Device Features**



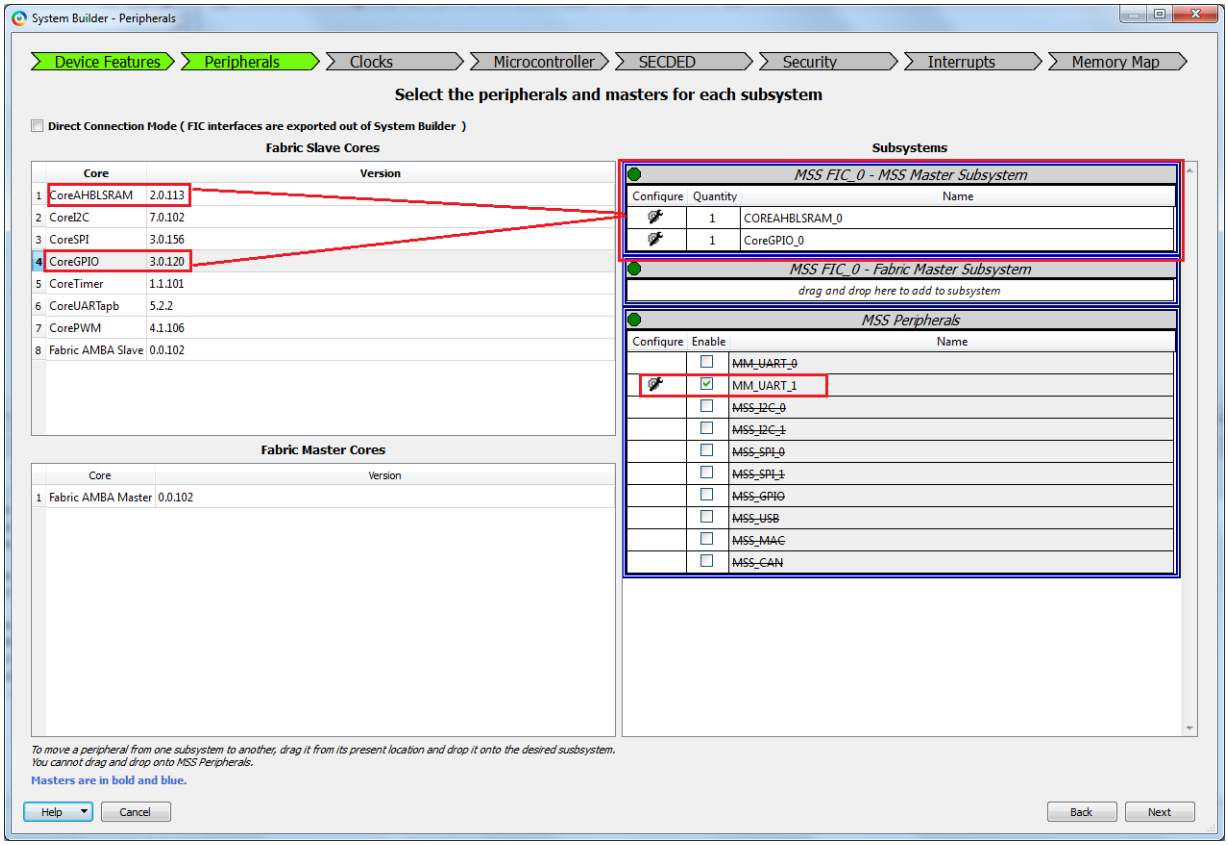
12. Select **Next**. **System Builder- Peripherals** page is displayed, as shown in Figure 9 and Figure 10, page 11. This tutorial uses the MSS MMUART peripherals.

Figure 9 • **SmartFusion2 System Builder Peripherals (M2S090TS Device)**



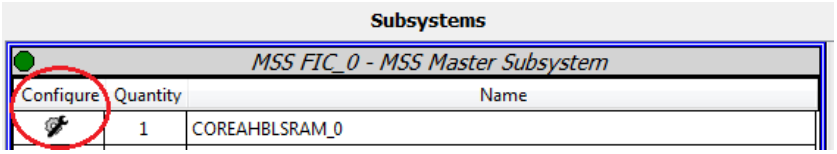
13. Select **MM_UART_1** for SmartFusion2 Security Evaluation Kit (**M2S090TS** device), uncheck all the other peripherals, as shown in Figure 10, page 11 and Figure 15, page 11.
14. Drag the **CoreAHBLSRAM** and **CoreGPIO** IPs to **MSS FIC_0 - MSS Master Subsystem** for M2S090TS device, as shown in Figure 10, page 11. This tutorial uses the CoreAHBLSRAM and CoreGPIO IPs.

Figure 10 • **SmartFusion2 System Builder MSS Peripherals (M2S090TS Device)**



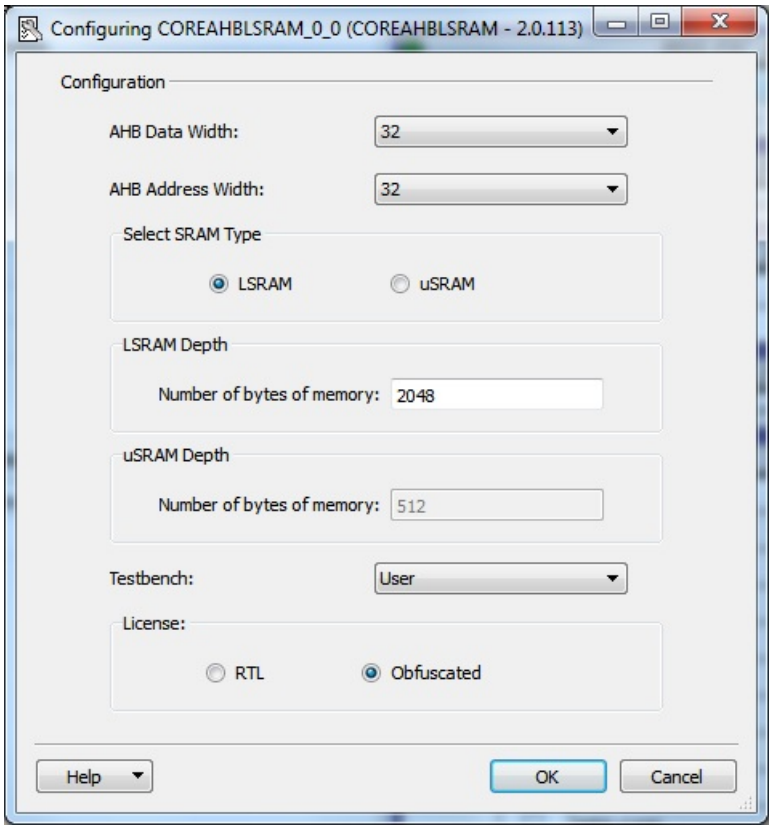
15. Configure **COREAHBLSRAM_0** by clicking the **Configure** icon, as shown in Figure 11.

Figure 11 • **CoreAHBLSRAM Configuration**



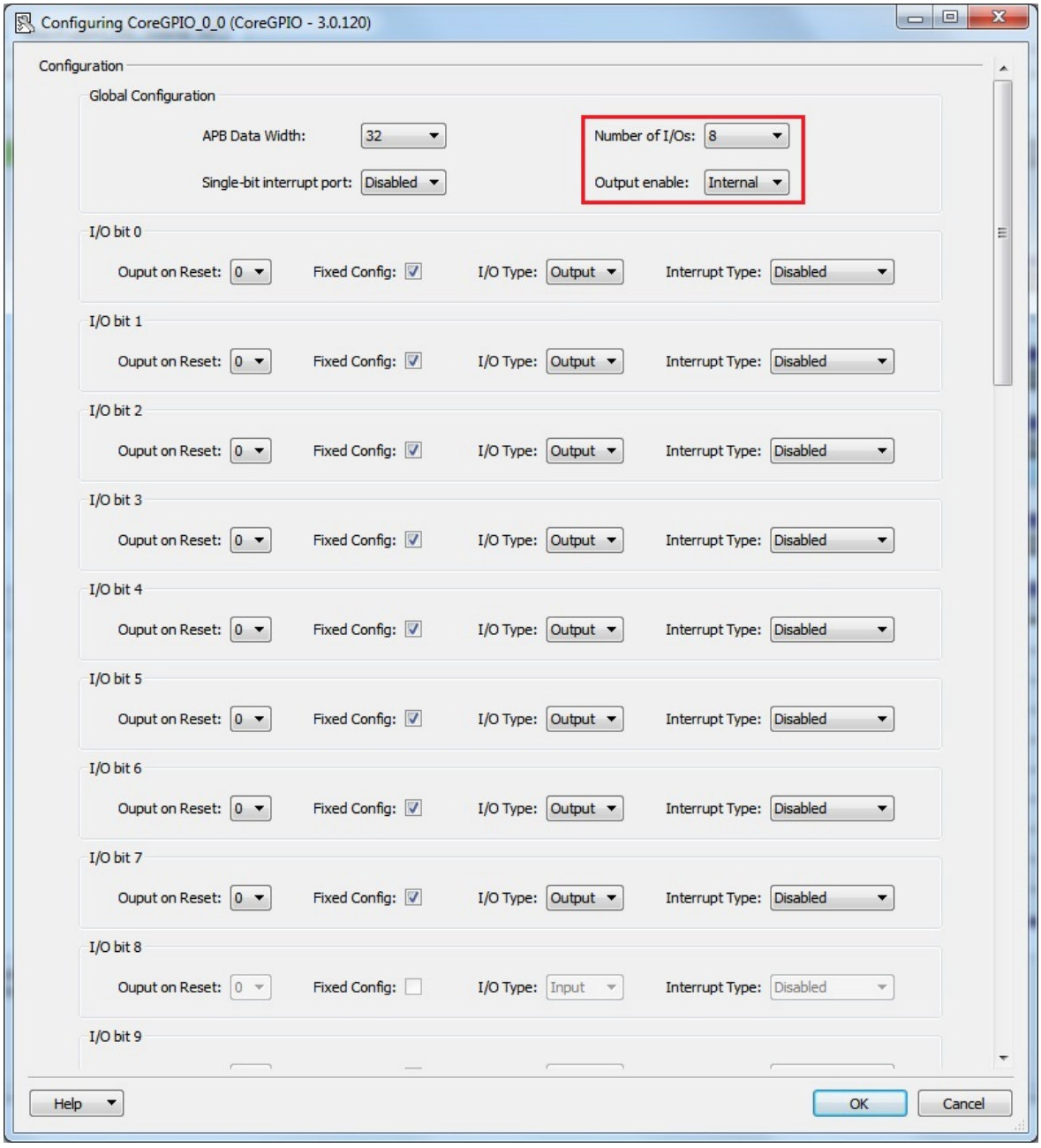
Use the settings, as shown in Figure 12.

Figure 12 • **CoreAHBLSRAM Configuration**



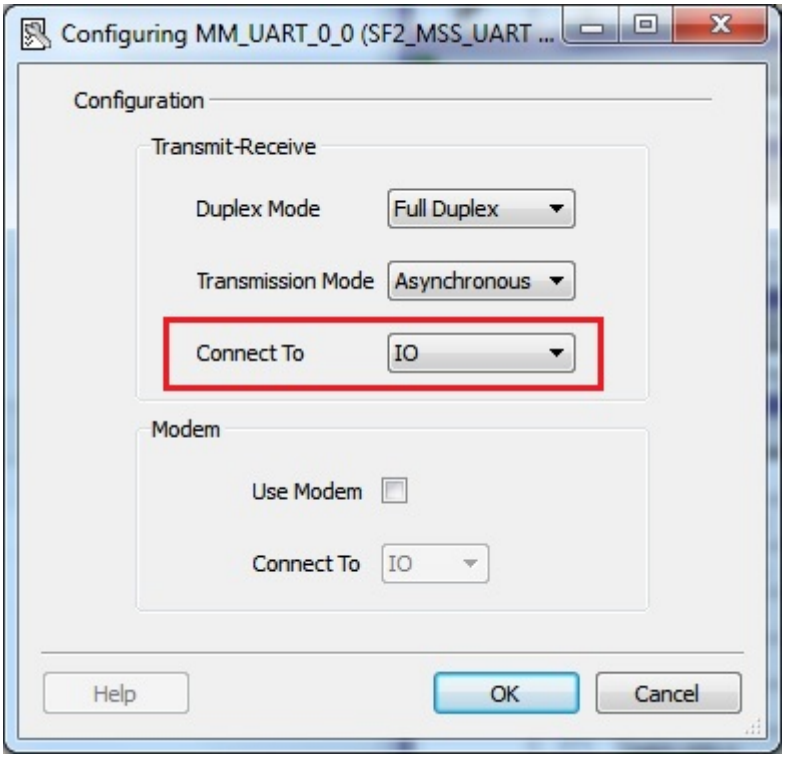
16. Click **OK** after completion of COREAHBLSRAM configuration.
17. Click **CoreGPIO** Configure icon and use the following settings for the SmartFusion2 Security Evaluation Kit board as shown in Figure 13, page 13, and keep the rest at default states
    - **Number of I/Os**: 8 - For SmartFusion2 Security Evaluation Kit board
    - **Output enable**: Internal
    - **Fixed Config**: Select the check box
    - **I/O Type**: Output

Figure 13 • **CoreGPIO Configuration**



18. Click **OK** after completion of CoreGPIO configuration.
19. Double-click the **MM_UART_1** configure icon for the M2S090TS device.
20. Select **IO** from the **Connect To** drop-down list and retain the default settings, as shown in Figure 14.

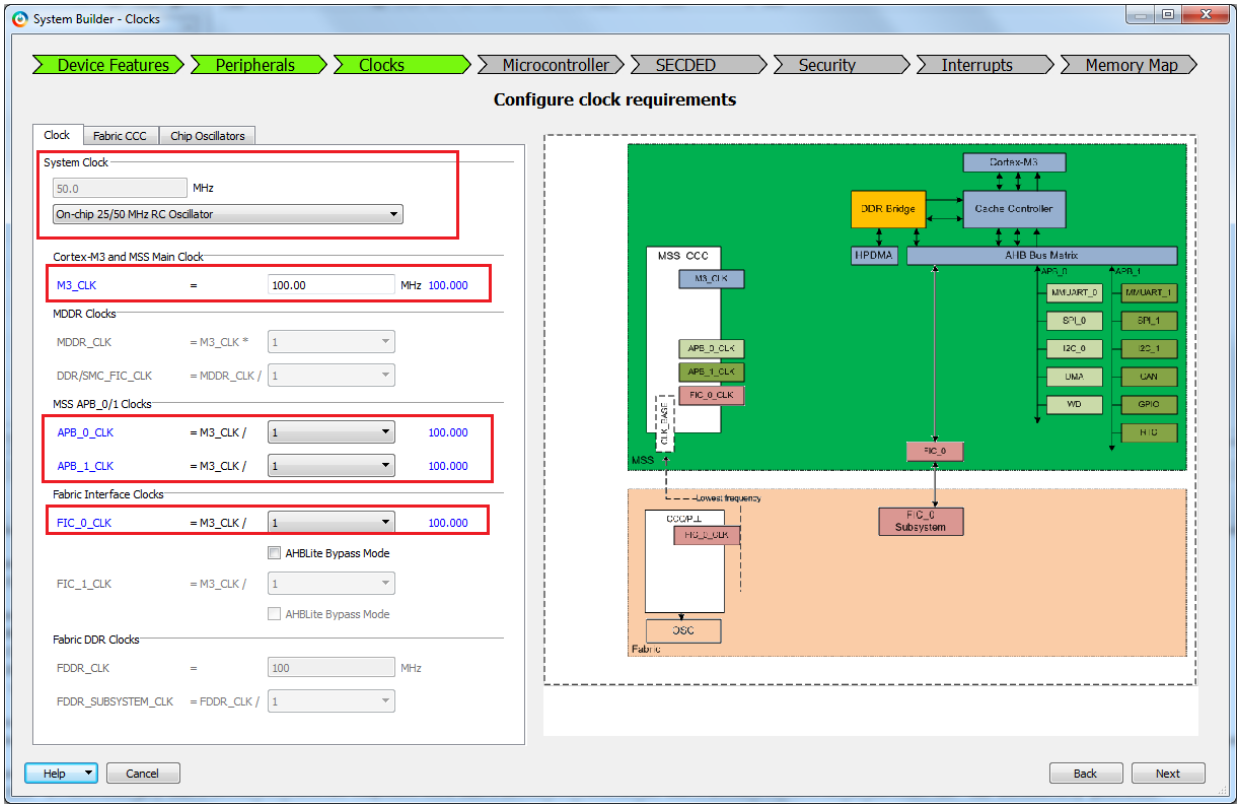
Figure 14 • **MM_UART Configuration**



21. Click **OK**.

22. Select **Next**. **System Builder- Clocks** page is displayed, as shown in Figure 15. Select the following options:
    - **System Clock**: Set it to On-chip 25/50 MHz RC Oscillator from the drop-down list.
    - **M3_CLK**: 100 MHz
    - **MSS APB_0/1 Clocks**: 100 MHz
    - **Fabric Interface Clocks**: 100 MHz

Figure 15 • **SmartFusion2 System Builder Clocks**



23. Click **Next**, the System Builder - **Microcontroller Options** page is displayed.
    - Leave all the Default Selections.
24. Click **Next**, the System Builder - **SECDED Options** page is displayed.
    - Leave all the Default Selections.
25. Click **Next**, the System Builder - **Interrupts Options** page is displayed.
    - Leave all the Default Selections.
26. Click **Next**, the System Builder - **Memory Map Options** page is displayed.
    - Leave all the Default Selections.

Figure 16 shows the address map for AHBL peripherals.

Figure 16 • **SmartFusion2 System Builder CoreAHBLite Address Map (M2S090T Device)**

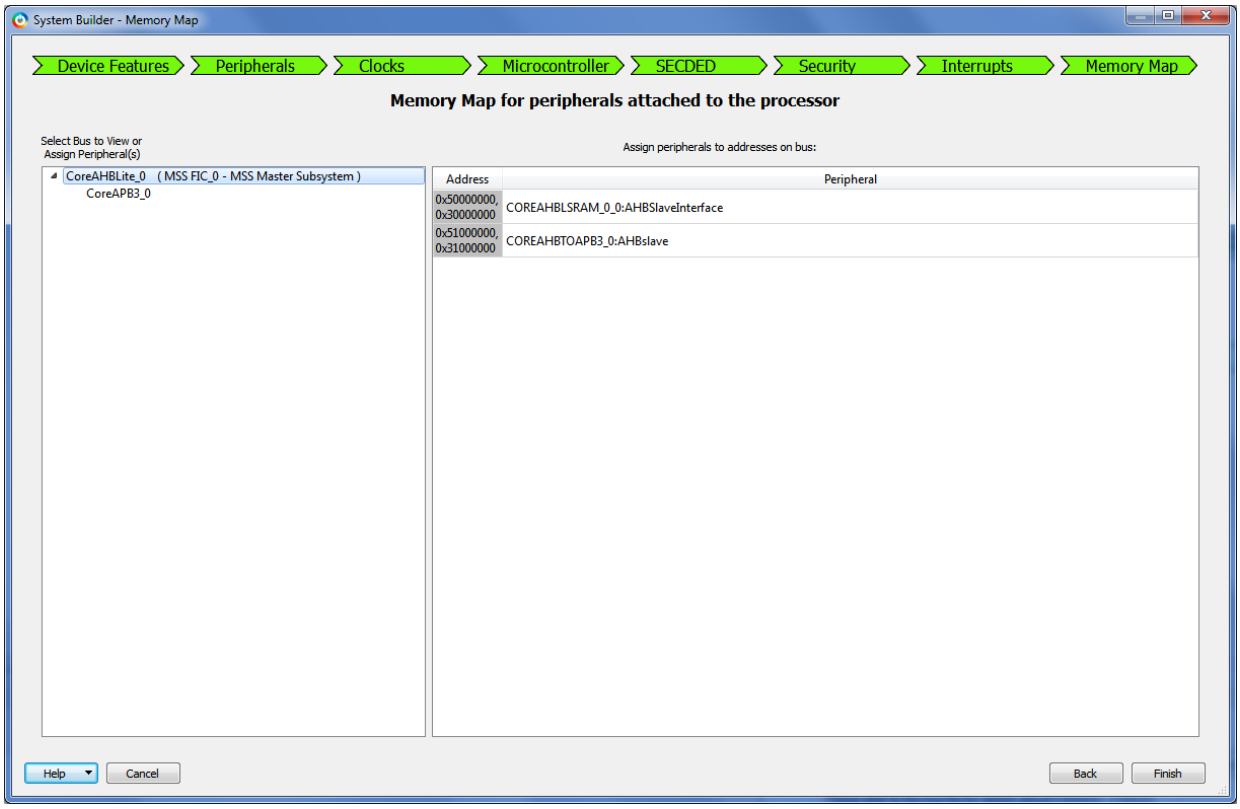
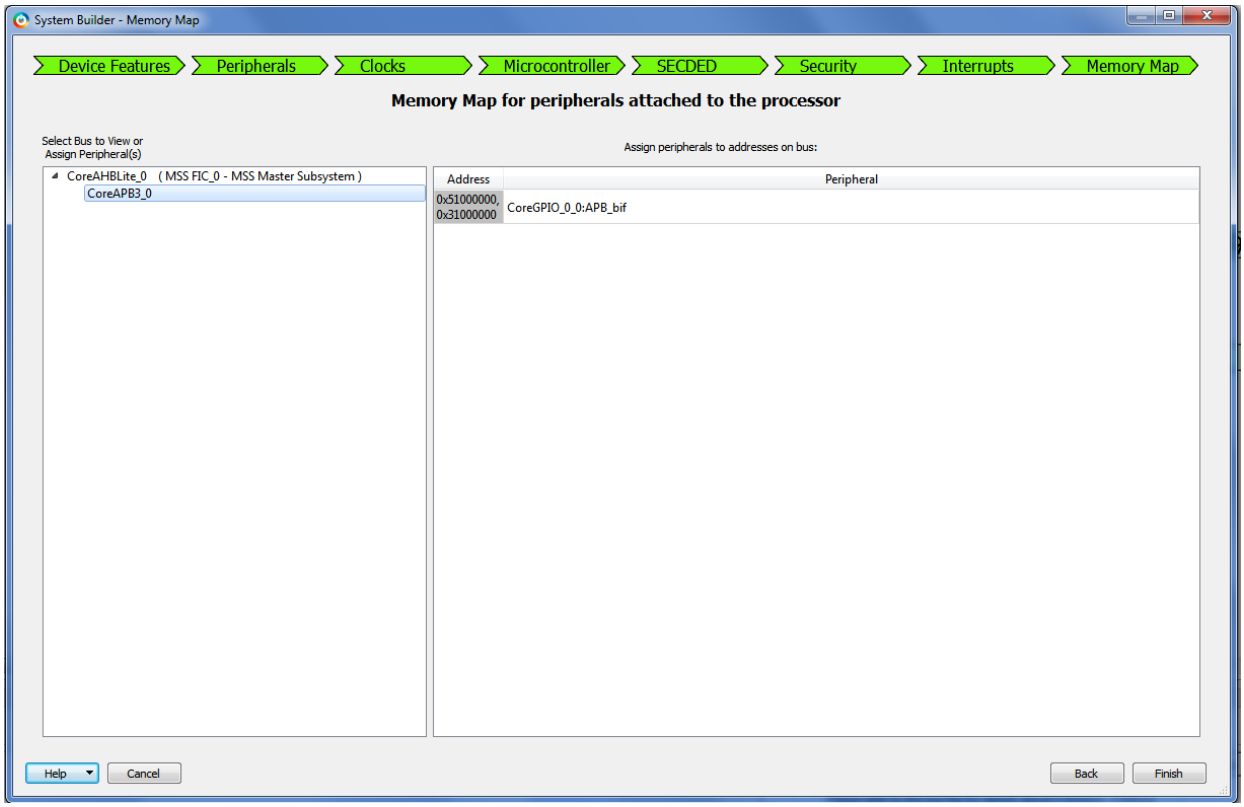


Figure 17 shows the address map for APB3 peripheral.

Figure 17 • **SmartFusion2 System Builder CoreAPB Address Map**

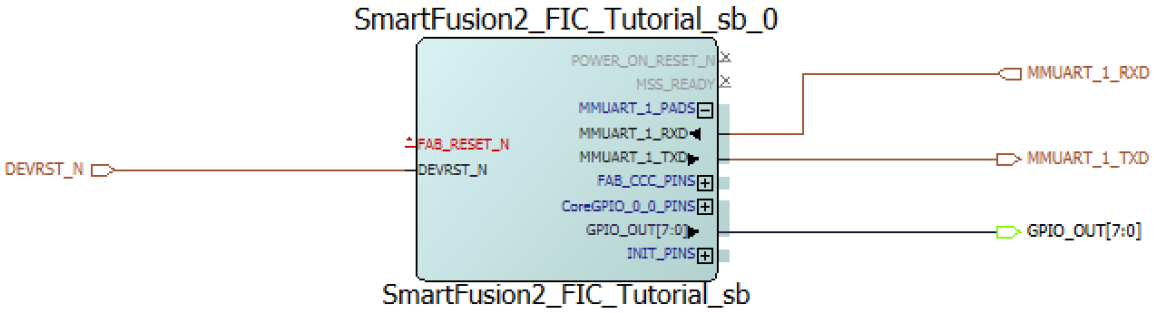


27. Click **Finish**.

The System Builder generates a system based on the selected options.

The System Builder block is created and added to the Libero SoC project, as shown in Figure 18.

Figure 18 • **SmartFusion2 System Builder Component**

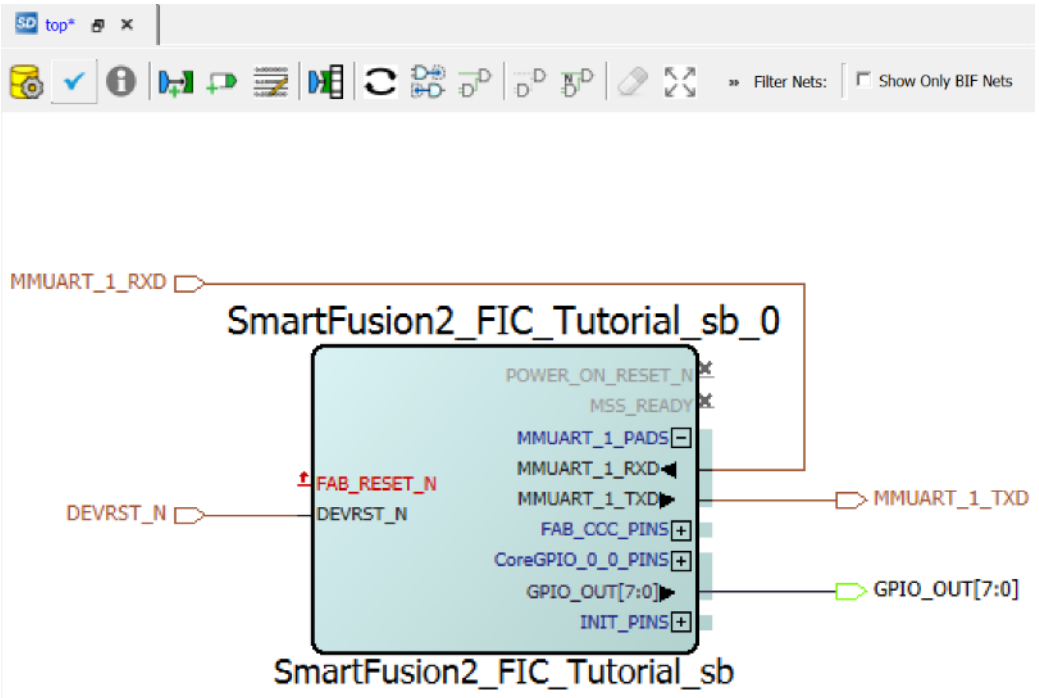


To initialize a user design in the SmartFusion2 devices, Microsemi provides a CoreResetP soft Reset Controller IP. The CoreResetP IP handles a sequence of reset signals in the SmartFusion2 devices. The CoreResetP does automatically be instantiated and connected by the System Builder. Open the System Builder component in the Smart Design canvas to view how these blocks are connected.

28. Connect the pins as follows:
- Right-click **FAB_RESET_N** and select **Tie High**.
- To select POWER_ON_RESET_N and MSS_READY pins, hold CTRL key, select pins, right-click and select **Mark Unused**.
- Expand **INIT_PINS**, right-click **INIT_DONE** and select **Mark Unused**.
- Expand **FAB_CCC_PINS**, right-click **FAB_CCC_GL0** and **FAB_CCC_LOCK,** and select **Mark Unused**.
- Expand **CoreGPIO_0_0_PINS**.
  - Mark the **INT[7:0] PINS** as unused by right-clicking and selecting **Mark Unused**.
  - Tie the **GPIO_IN[7:0]** to high by right-clicking and selecting **Tie High**.
  - Promote the **GPIO_OUT[7:0]** to the top by right-clicking and selecting **Promote to Top Level**.

After connecting the pins, the System Builder block is displayed, as shown in Figure 19.

Figure 19 • **SmartFusion2 System Builder Block**

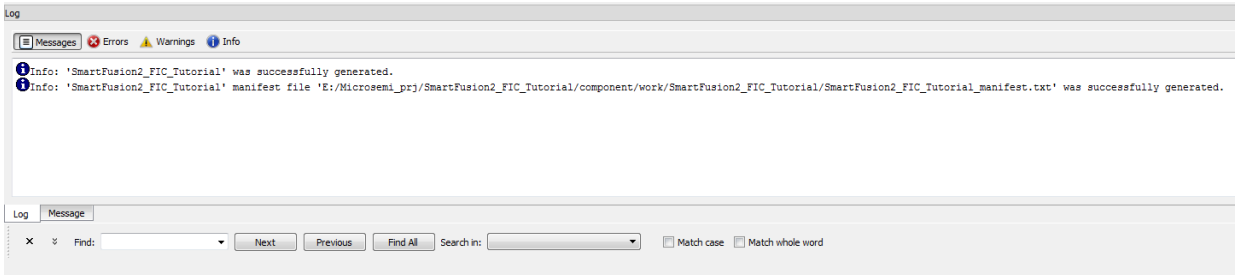


29. Click **Generate Component** icon on the SmartDesign toolbar or right-click on the canvas and select **Generate Component**.

Figure 20 • **Generate Component**

After successful generation of the system, the message **top was successfully generated** is displayed in the Libero SoC log window if the design is generated without any errors. The log window is displayed, as shown in Figure 21.

Figure 21 • **Log Window**



2.7 Step 2: Modifying User BFM Script for Simulation

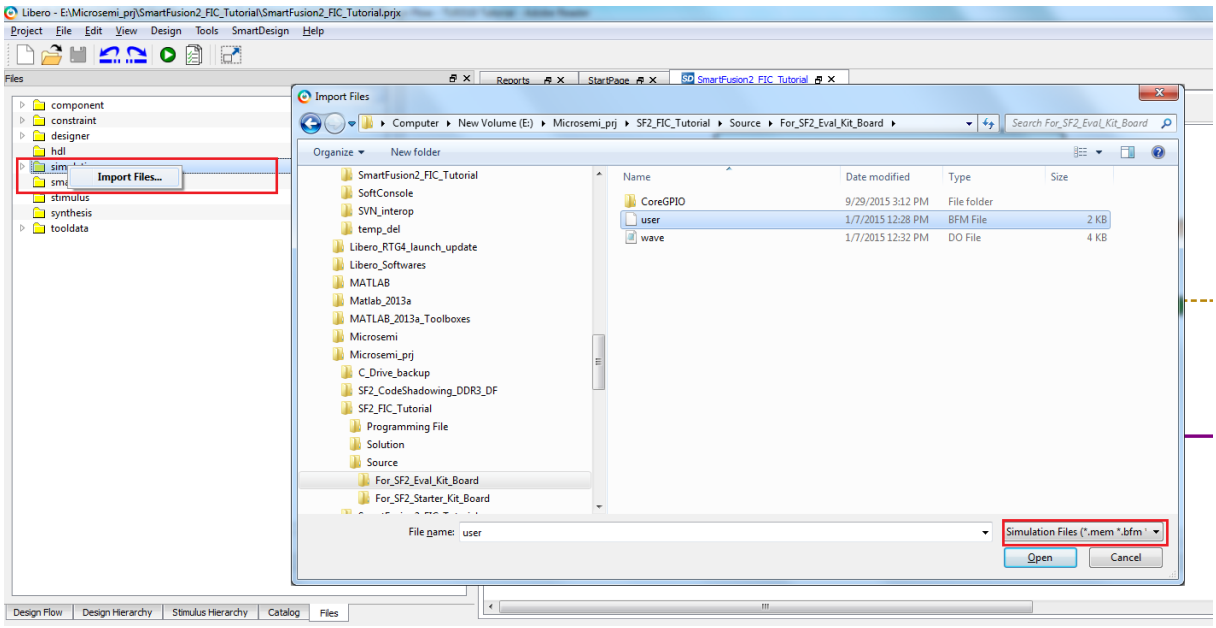
Verify the design by using the BFM master or slave model and a BFM script to drive the AHBL/APB input of the DUT. This setup allows the BFM to write or read to the AHBL/APB register set and to verify that the DUT is behaving as expected.

This step explains adding BFM commands to the `user.bfm` file to perform design simulation. For more information on BFM commands refer to the *CoreAMBA BFM User Guide*. The `user.bfm` file is created by Libero SoC Design software and is available in the simulation folder of the project files.

Note: Download the project files. Refer to the Design Files, page 4.

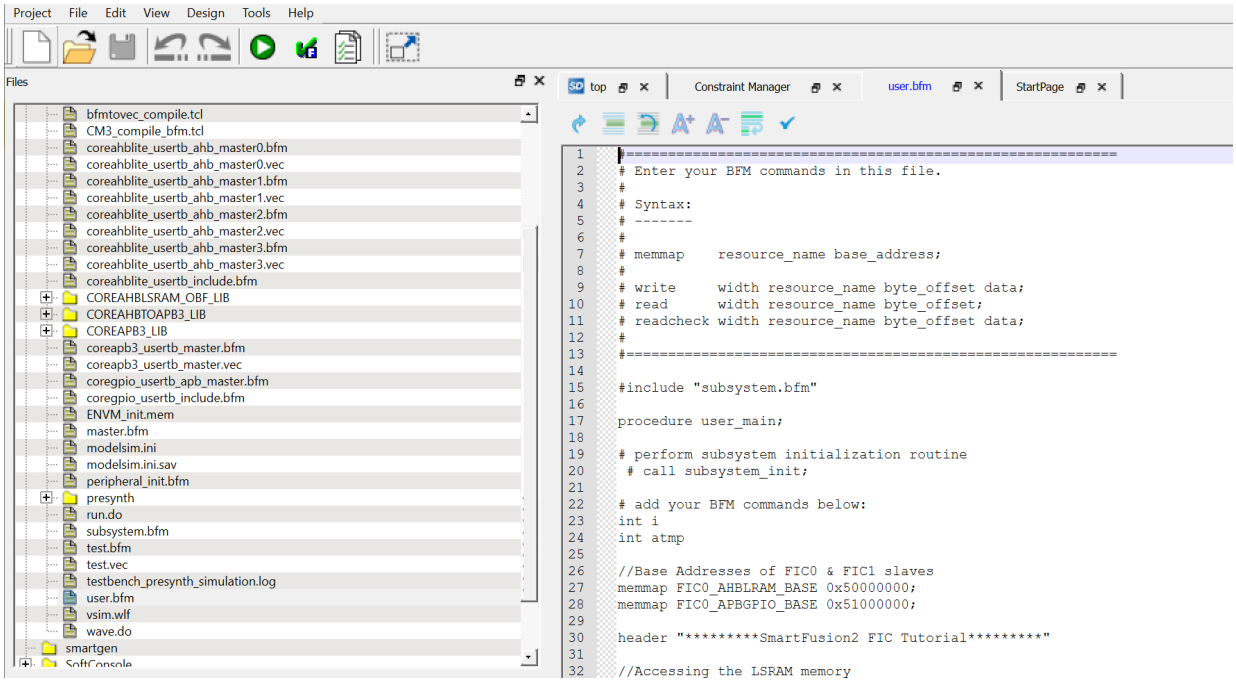
1. For the SmartFusion2 Security Evaluation Kit board, right-click the simulation under project files and select import files to import the `user.bfm` file which is located in downloaded design files (*\m2s_tu310_df\Source\user.bfm*) as shown in Figure 22, page 19 or select **Files > Import > Others** to import the `user.bfm` file.
2. Click **Yes to all** to replace the existing `user.bfm` file.

Figure 22 • **Import bfm file**



3. After importing, double-click the `user.bfm` file under simulation folder. This opens the `user.bfm` file as a new tab in the project window, as shown in Figure 23.

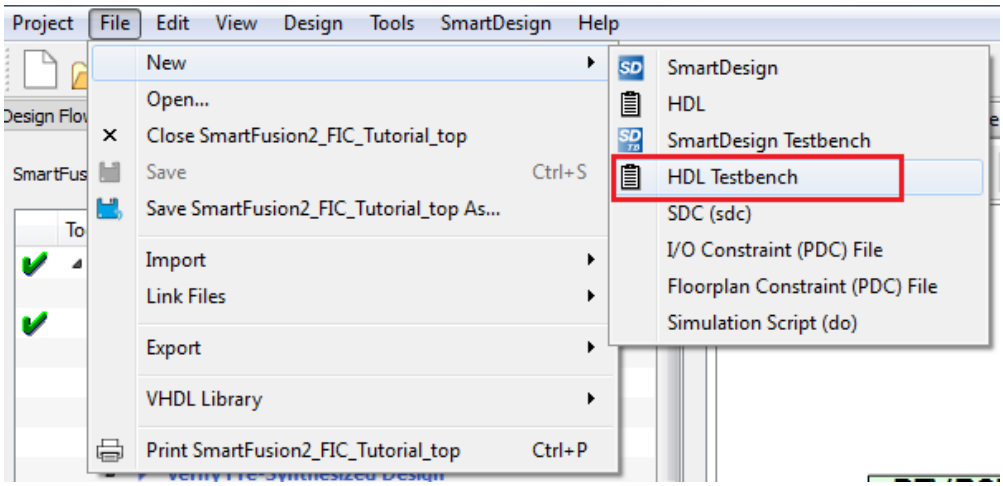
Figure 23 • **user.bfm file**



2.7.1 Generating Testbench

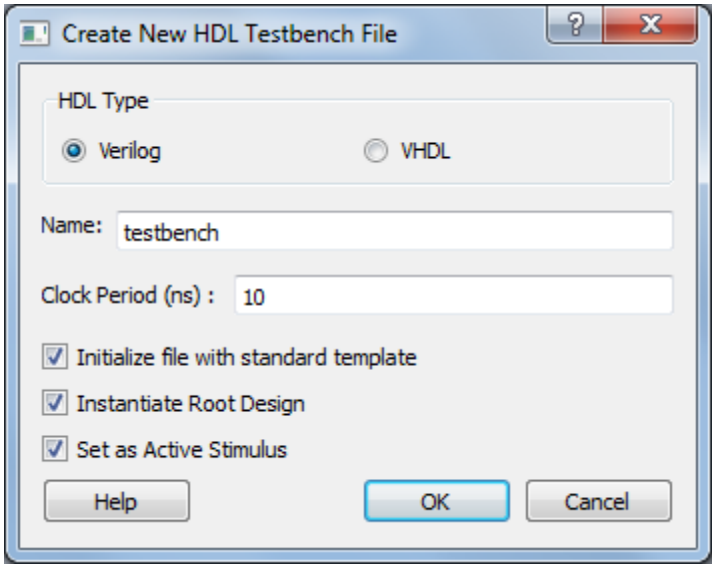
1. From the **File** menu, select **New > HDL Testbench,** as shown in Figure 24.

Figure 24 • **HDL Testbench**



The **Create New HDL Testbench File** dialog box is displayed.

Figure 25 • **Create New HDL Testbench File**



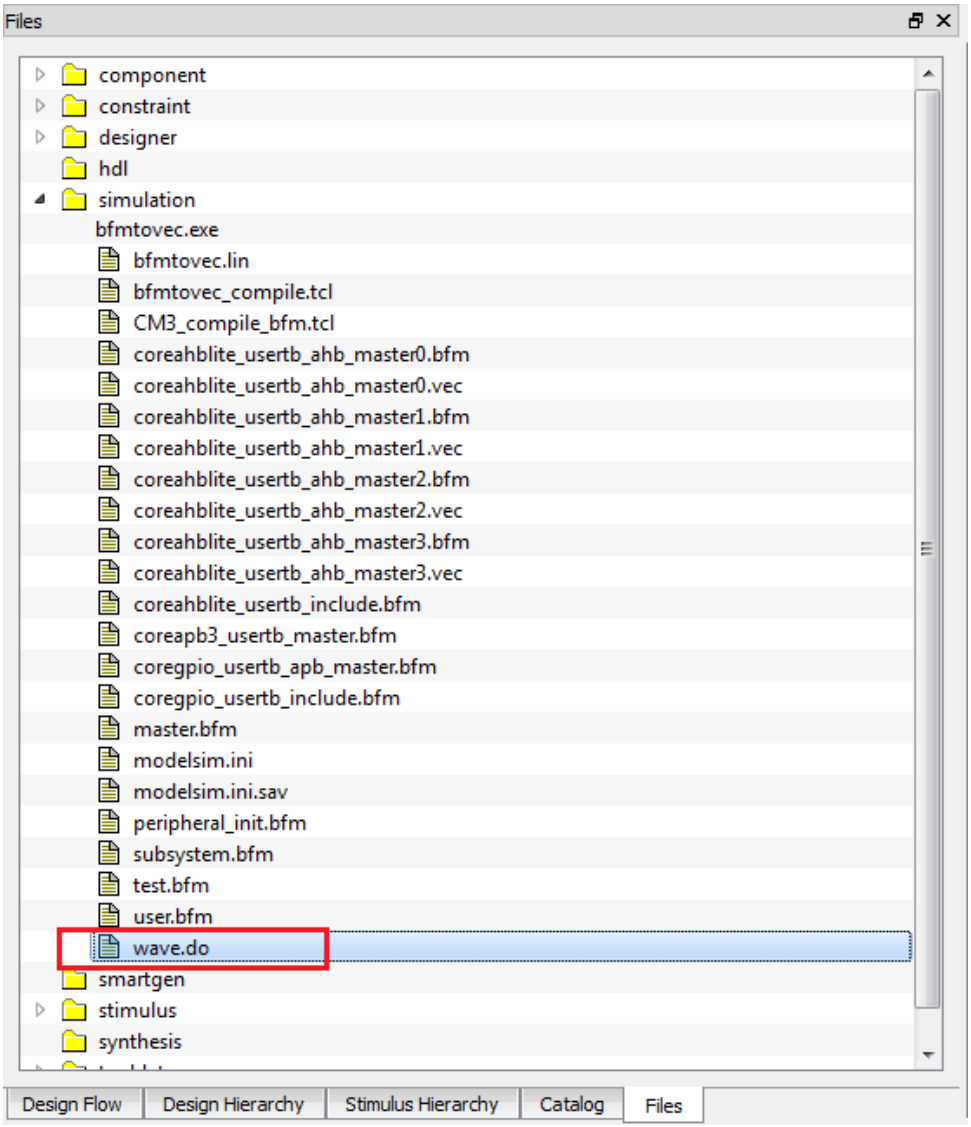
2. Select **HDL Type** as **Verilog** or **VHDL**.
3. Enter **Name** as testbench in the text box and retain the default settings.
4. Enter **Clock Period (ns)** as 10.
5. Click **OK**.

2.8 Step 3: Simulating Design Using BFM Models

This section describes how to use the testbench and BFM script file to simulate the design.

1. Add the `wave.do` file to the top design simulation folder by clicking **File > Import > Others**.
2. Browse to the `wave.do` file location in the design files folder: *\m2s_tu310_df\Source*. Figure 26 shows the `wave.do` file under simulation folder in the Files window.

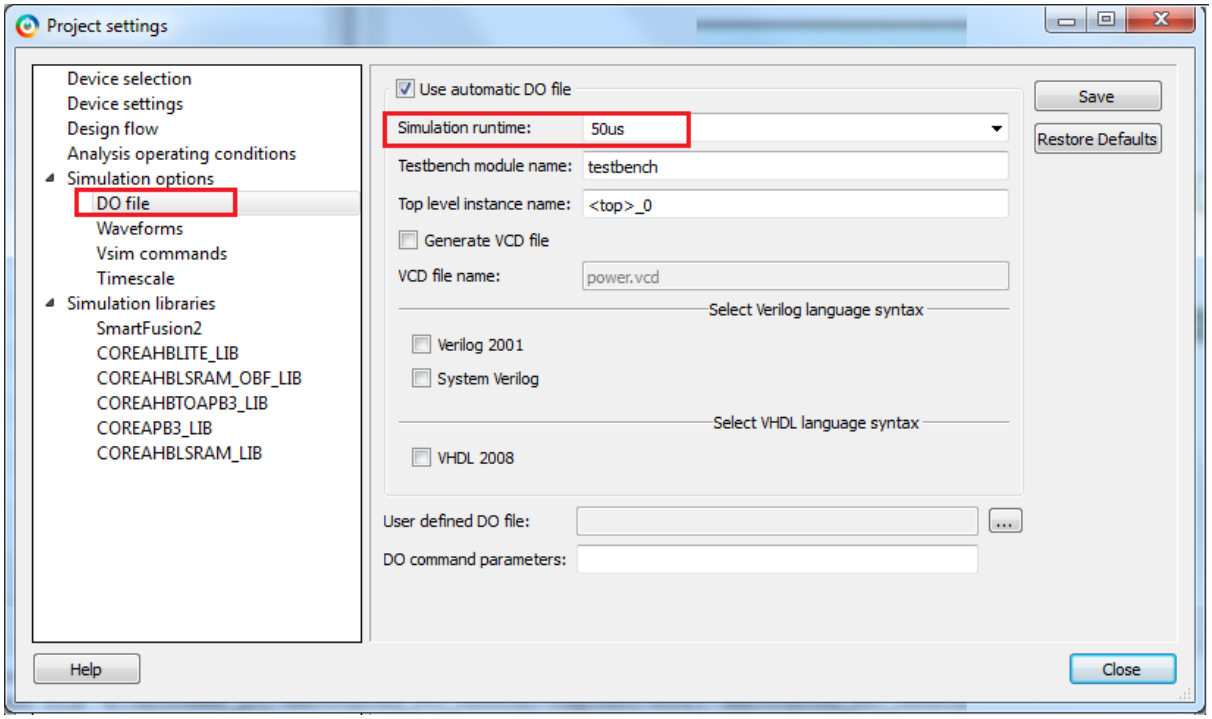
Figure 26 • **wave-do File**



3. Set up the simulation environment as follows:

Select **Project > Project settings**. On the Project Settings window, under **Simulation options**, select **DO file** to change the simulation run time, enter **50 µs** in the **Simulation runtime** field, as shown in Figure 27, page 23.

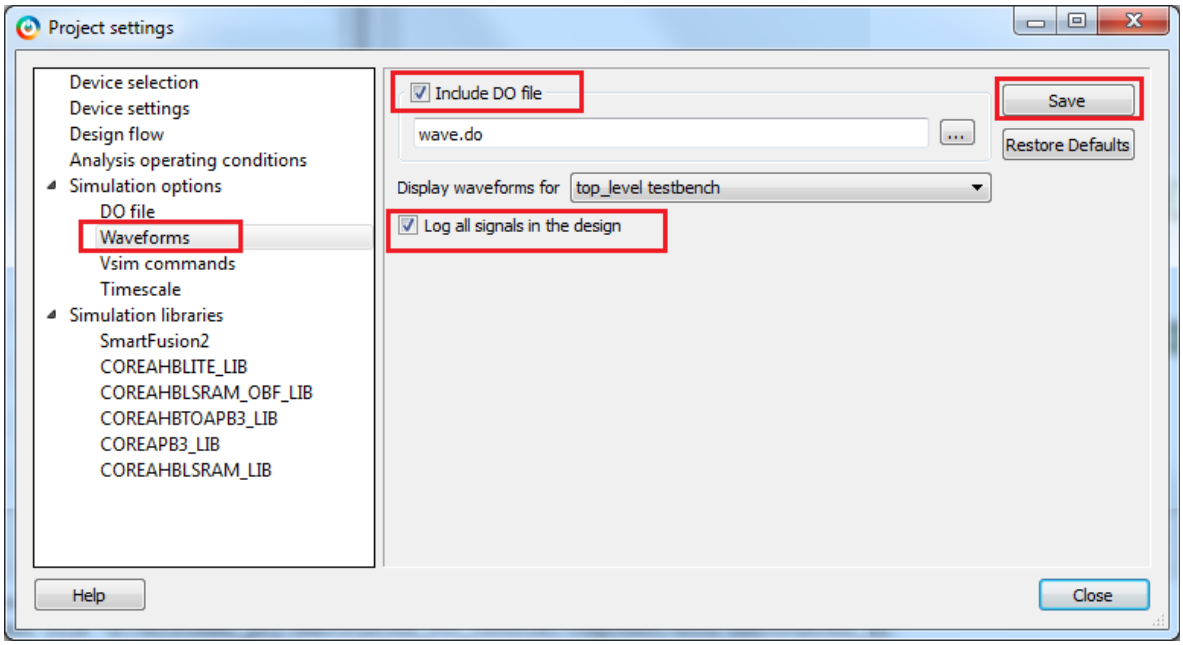
Figure 27 • **Project Settings – Do File**



4. Save the **Do file** configuration. This can be done by clicking the **Save**.
5. Select **Waveforms** under **Simulation options,** as shown in Figure 28:
   - Select **Include Do file**.
   - Select **Log all signals in the design** check box.
   - Select **Save** when prompted to save the changes.
   - Click **Close** to close the Project settings dialog box.

Note: You can also add ports or signals of interest in the ModelSim software.

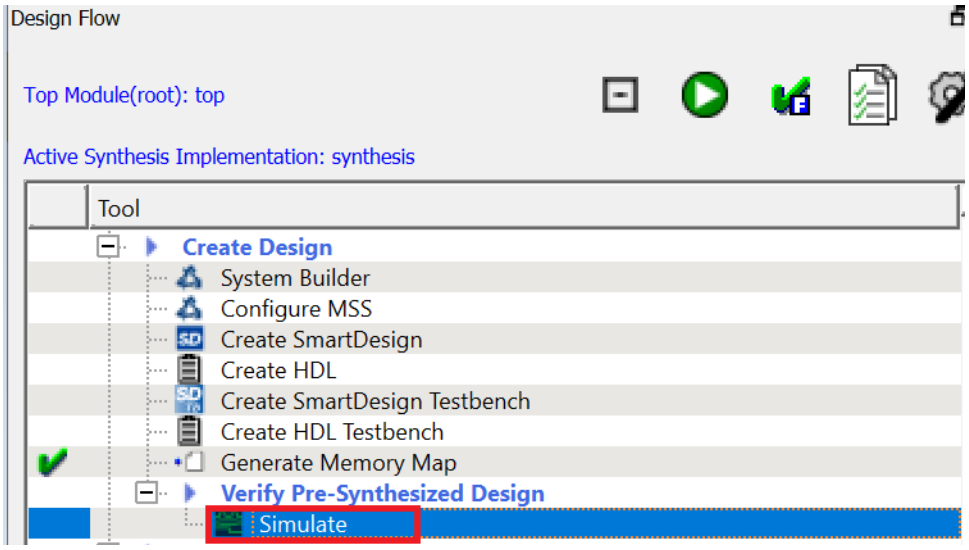
Figure 28 • **Project Settings – Waveforms**



6. Select the **Design Flow** tab in the project window.

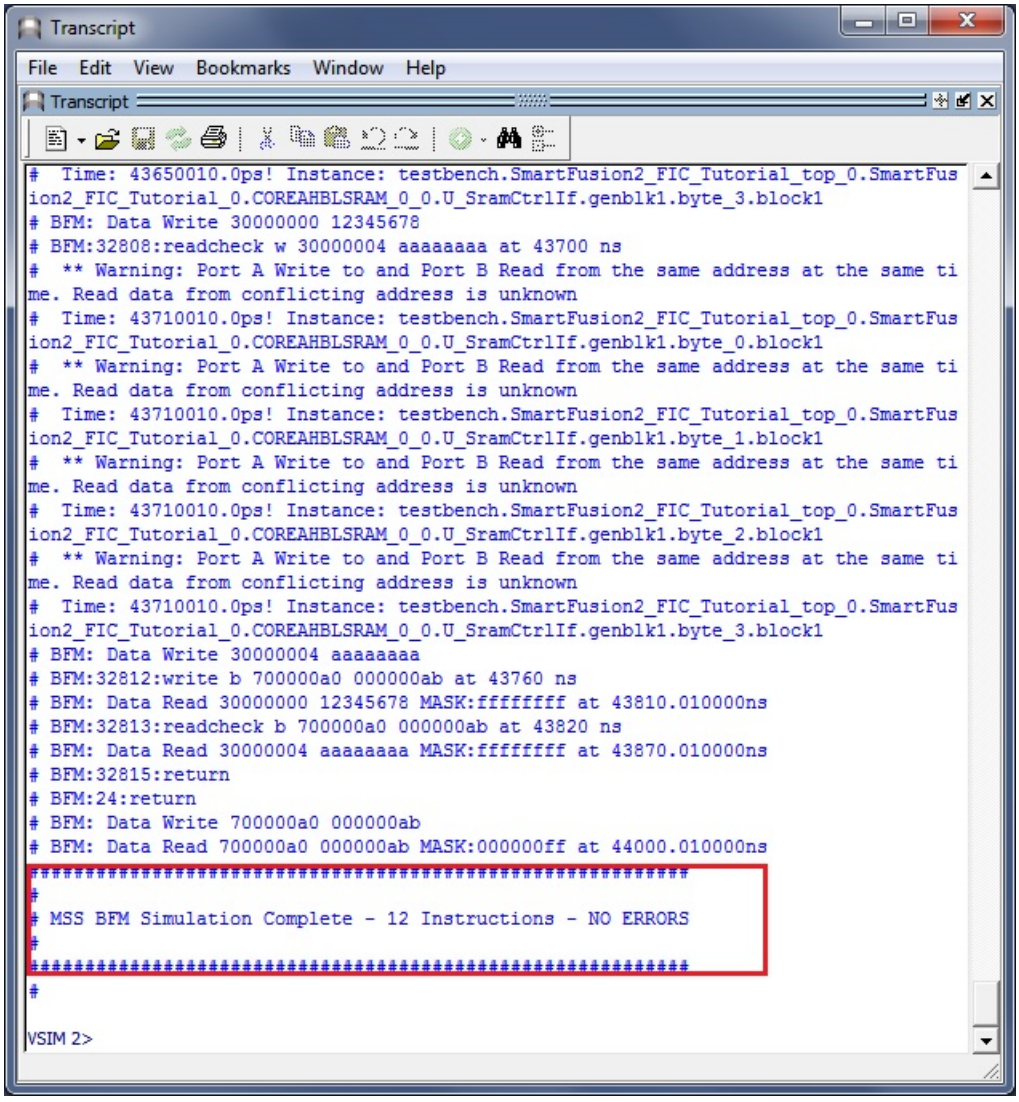
7. Expand the **Verify Pre-Synthesized Design**, as shown in Figure 29. Double-click **Simulate** to invoke ModelSim. After invoking ModelSim, the design is loaded. Alternatively, right-click the **Simulate** and select **Open Interactively**.

Figure 29 • **Design Flow – Verify Pre-Synthesized Design**



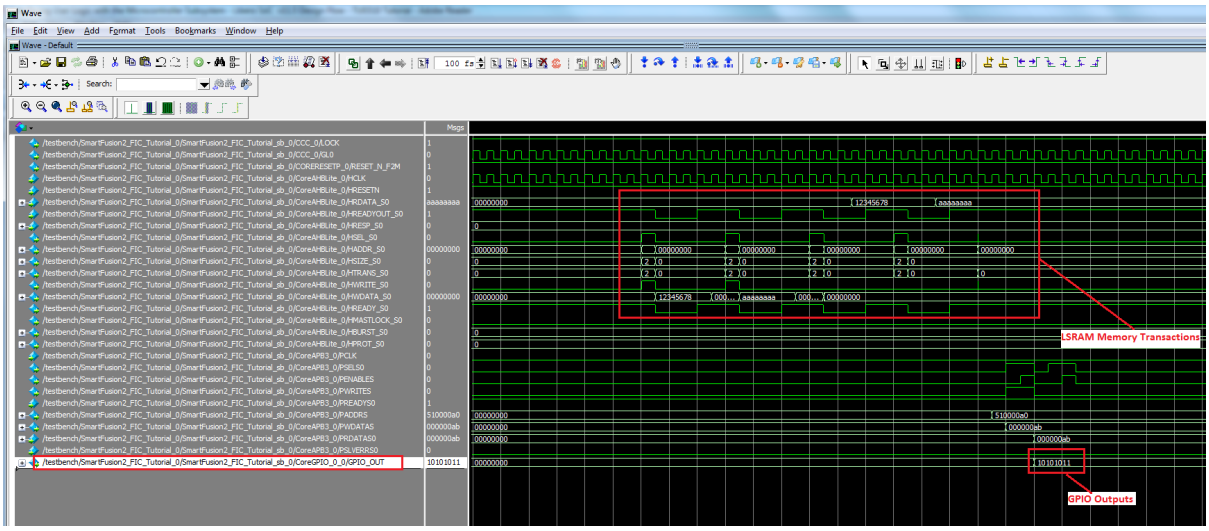
8. Maximize the **ModelSim Transcript** window to see the BFM commands execution. Ensure that there are no errors. Figure 30 shows the ModelSim Transcript window.

Figure 30 • **ModelSim Transcript Window – BFM Commands**



9. After successful BFM simulation, observe the ModelSim waveform window for the read and write bus transactions to the fabric peripherals, as shown in Figure 31. Notice the result of GPIO configuration BFM commands in GPIO states.

Figure 31 • **Design Simulation Waveforms**



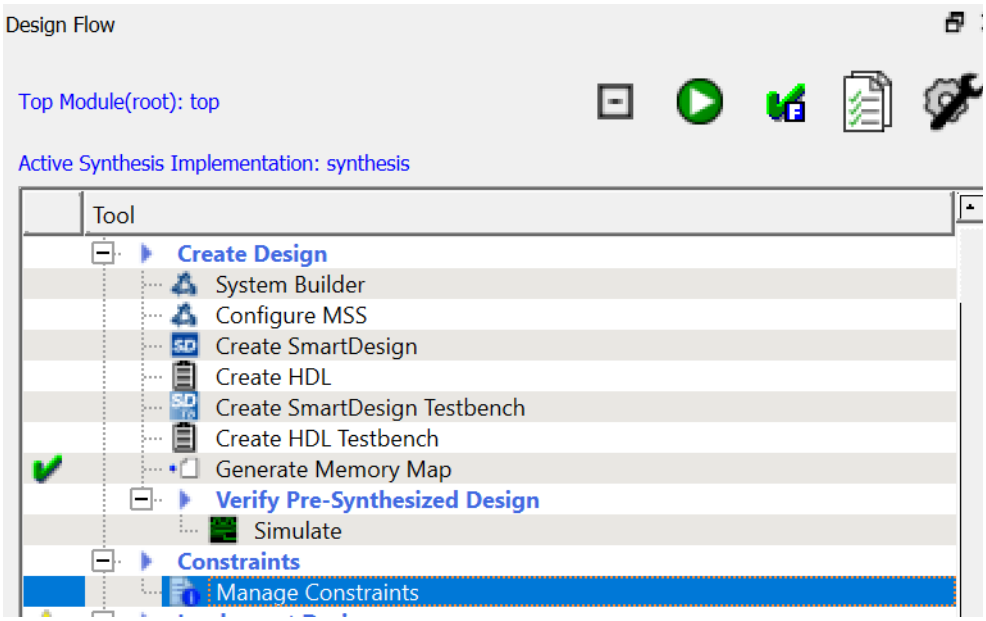
10. Quit the ModelSim simulator by selecting **File > Quit**.

2.9 Step 4: Generating Programming File

The following steps describe how to generate a program file:

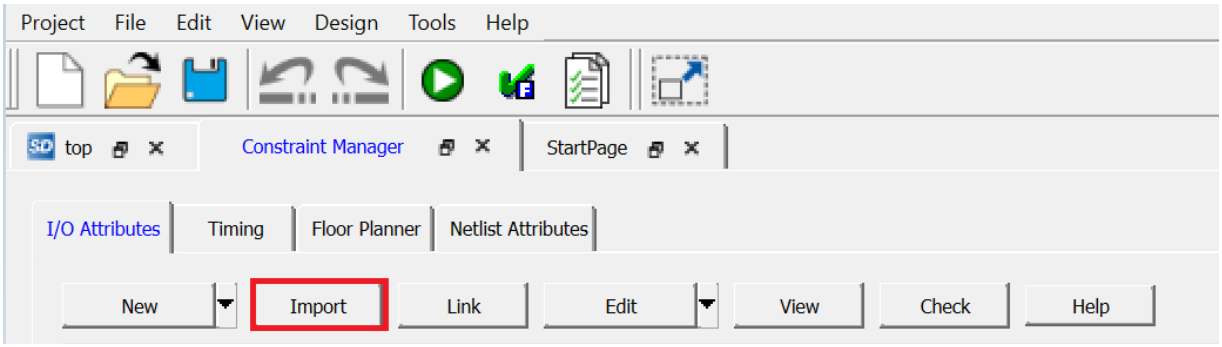
1. Expand **Constraints**, as shown in Figure 32. Double-click **Manage Constraints** to add the I/O and Timing constraints.

Figure 32 • **I/O Constraints**



2. To add the I/O constraints, click **Import** as shown in Figure 33.

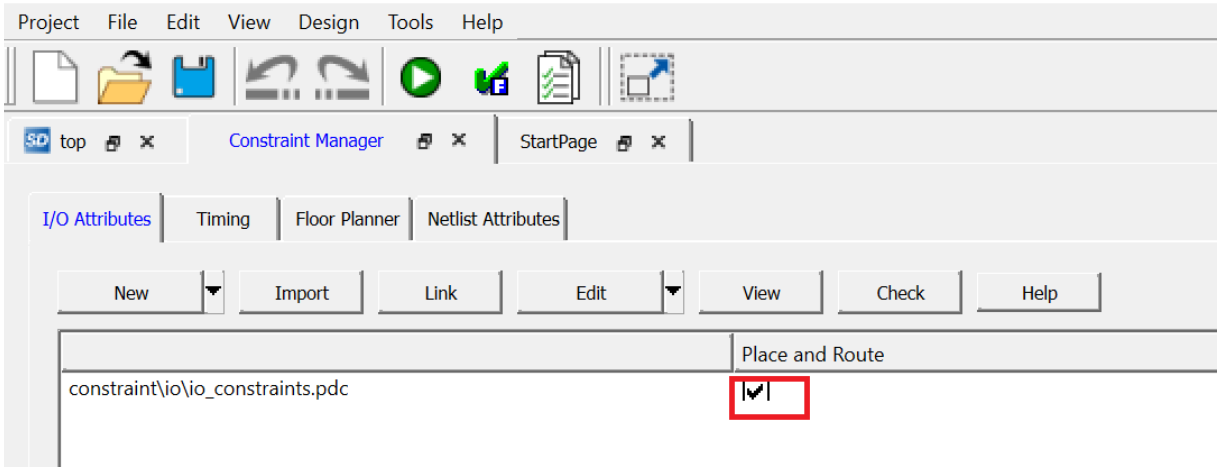
Figure 33 • **Importing I/O Constraints**



3. Browse the top.io.pdc file, and select the check box as shown in to select the constraint for place and route, the pdc file is located at:

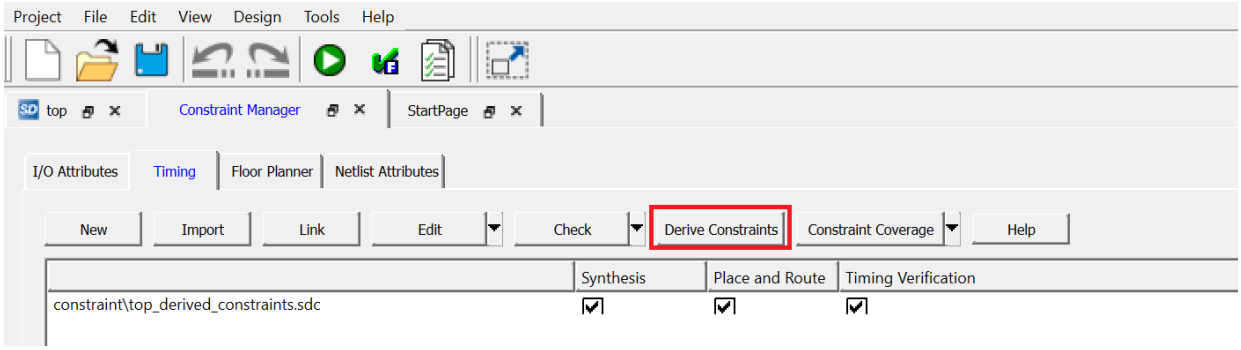
   SF2_FIC_Tutorial\Source\For_SF2_Eval_Kit_Board.

Figure 34 • **Managing I/O Constraints**



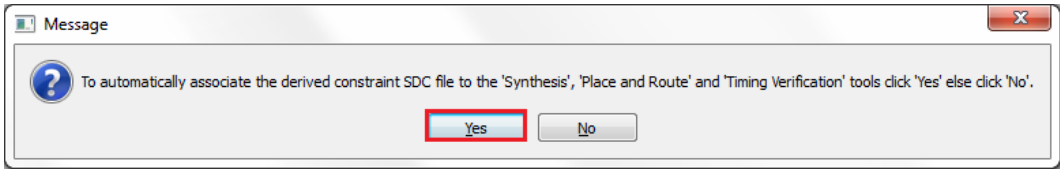
4. Select the **Timing** tab in the **Constraint Manager** and click **Derive Constraints,** as shown in Figure 35 to generate the timing constraints for the design.

Figure 35 • **Derive Constraints**



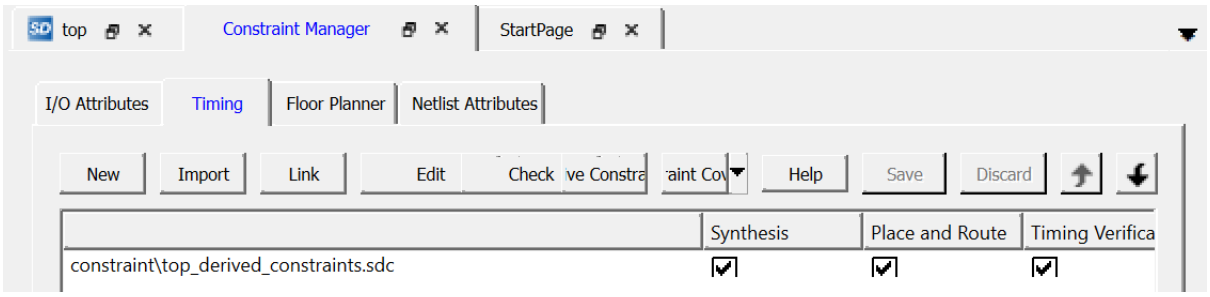
5. Click **Yes,** as shown in Figure 36.

Figure 36 • **Selecting Timing Constraint for Synthesis, Place and Route, and Timing Verification**



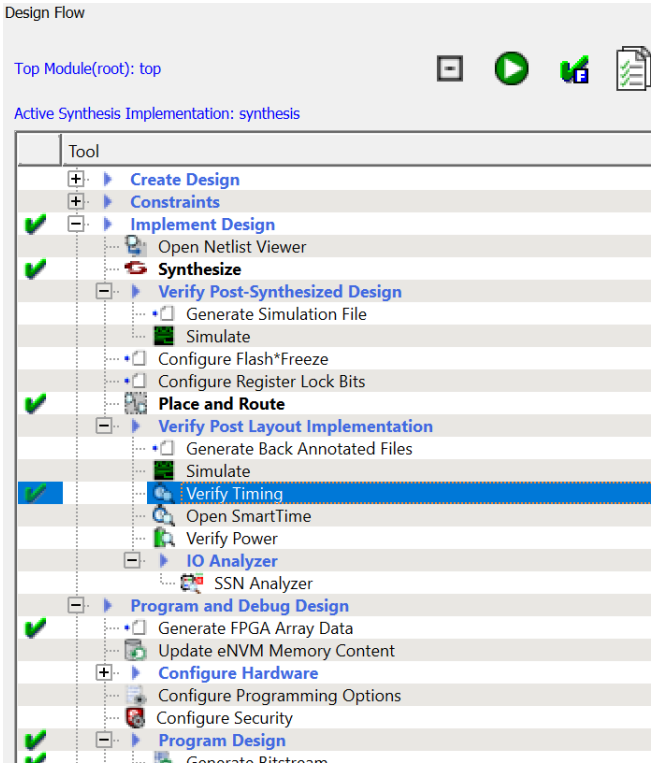
6. The Constraint Manager is shown in Figure 37.

Figure 37 • **Constraint Manager**



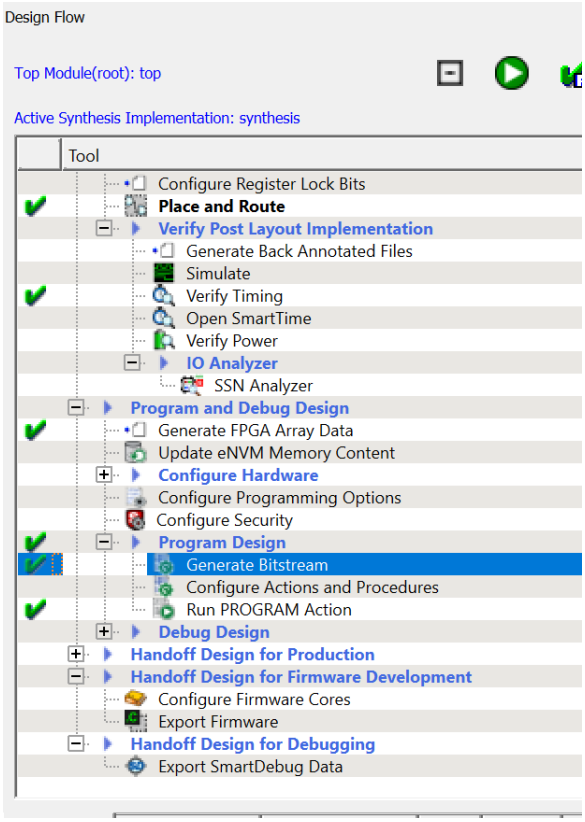
7. Under **Verify Post Layout Implementation**, double-click **Verify Timing,** as shown in Figure 38 to verify the design timing.

Figure 38 • **Verify Timing**



8. Click **Generate Bitstream** as shown in Figure 39 to generate the programming file.

Figure 39 • **Generate Bitstream**



2.10 Step 5: Jumper Settings for the SmartFusion2 Board

Use the following details to ensure the correct jumper setting.

2.10.1 Jumper Settings for SmartFusion2 Security Evaluation Kit Board

Connect the jumpers on the SmartFusion2 Security Evaluation Kit Board, as shown in Table 2. Switch OFF the power supply switch while connecting the jumper.

Table 2 • **Jumper Settings for SmartFusion2 Security Evaluation Kit Board**

Jumper	Pin (from)	Pin (to)
J3, J8	1 (default)	2

2.10.2 Programming the Device

1. Program the SmartFusion2 Security Evaluation kit board with the job file provided as part of the design files using FlashPro Express software, refer to Appendix: Programming the Device Using FlashPro Express, page 44.
2. Double-click the **Run PROGRAM Action** under **Program Design** in the **Design Flow** window as shown in Figure 40 to program the SmartFusion2 SoC FPGA device.

Figure 40 • **Run PROGRAM Action**

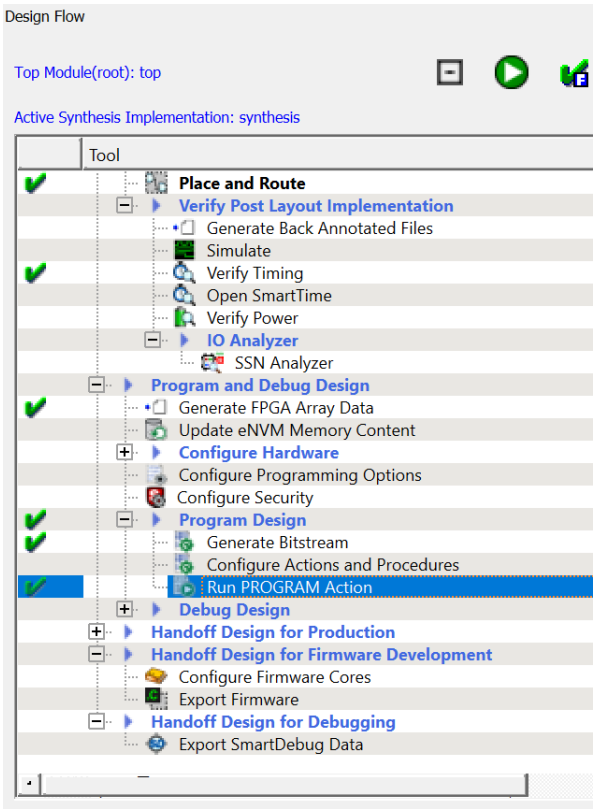
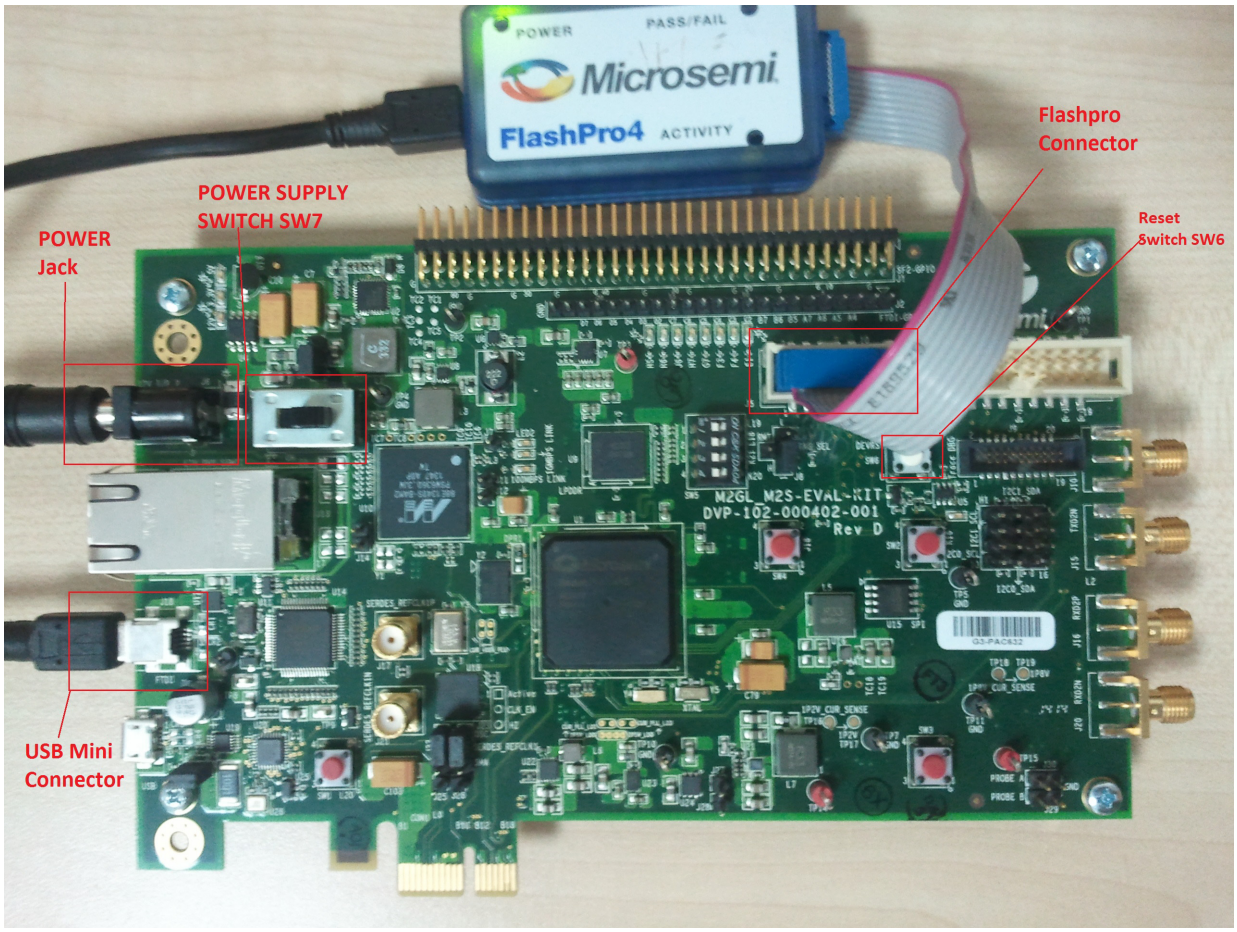


Figure 41 shows the board setup for running the application design on the SmartFusion2 Security Evaluation Kit board.

Figure 41 • **SmartFusion2 Security Evaluation Kit Setup**



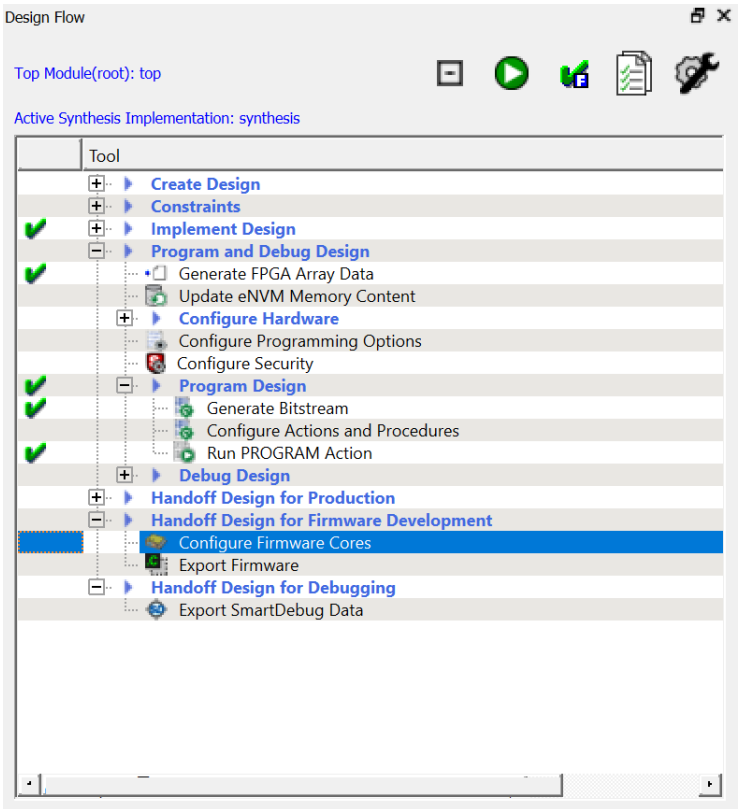
Note: Do not interrupt the programming sequence; it may damage the device or the programmer. If you face any problems, contact Microsemi Tech Support at *soc_tech@microsemi.com*.

2.11 Step 6: Building the Software Application through SoftConsole

The following steps describe how to build the software application via SoftConsole:

1. In the **Design Flow** window, double-click **Configure Firmware Cores** under **Handoff Design For Firmware Development** as shown in Figure 42, to select the firmware drivers for software application.

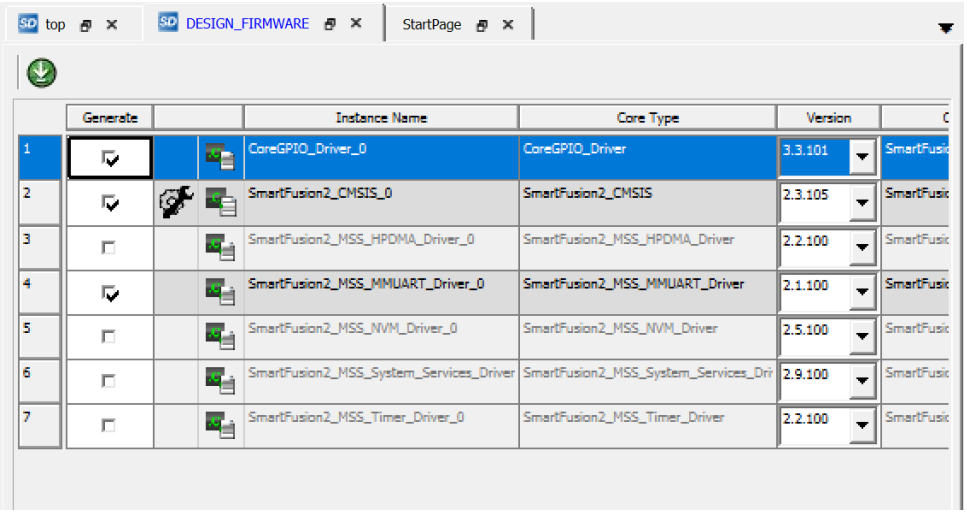
Figure 42 • **Handoff Design for Firmware Development**



2. The **Design Firmware** window displays compatible firmware drivers based on the peripherals configured in the design. The following drivers are used in this tutorial:
   - CMSIS
   - CoreGPIO
   - MMUART
3. In the **DESIGN_FIRMWARE** tab, clear all the driver check boxes, except CMSIS, CoreGPIO and MMUART as shown in Figure 43, page 33.

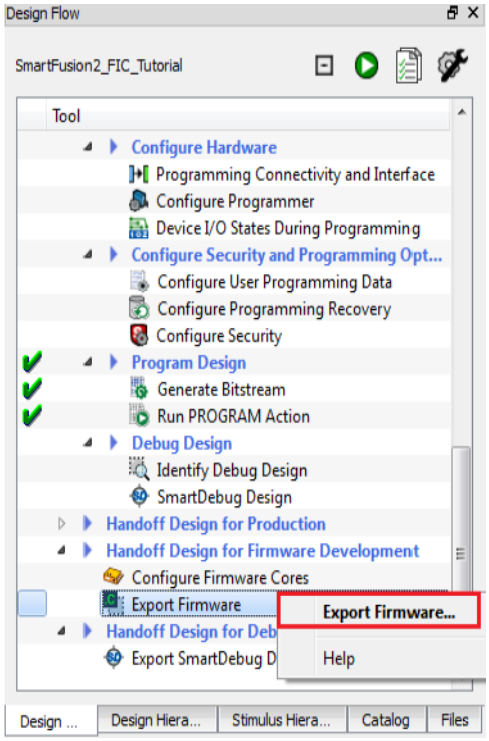
Note: Select the latest version of the drivers.

Figure 43 • **Design Firmware**



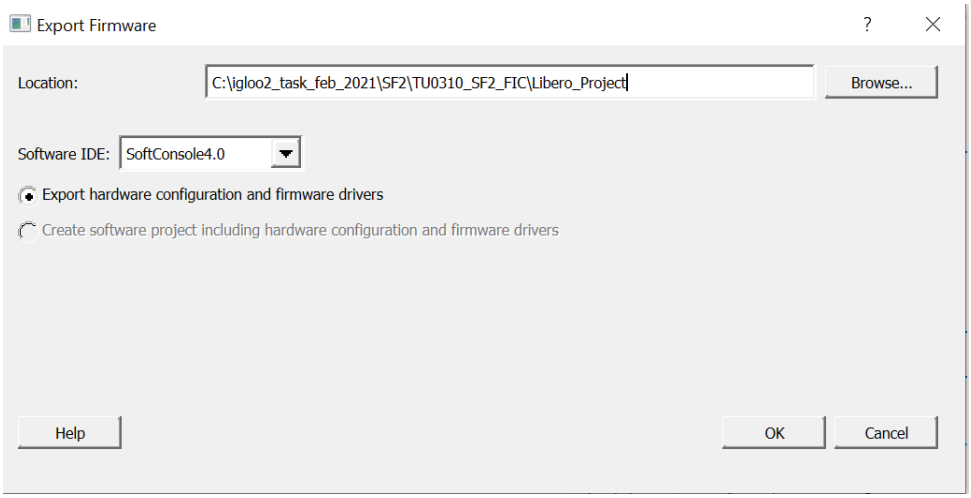
4. Click **Download all firmware** button highlighted in red to download the latest version of drivers for peripherals as shown in Figure 43. Close the **DESIGN_FIRMWARE** tab.
5. In the **Design Flow** window, select **Handoff Design for Firmware Development > Export Firmware**.
6. Right-click and select **Export Firmware**… as shown in Figure 44.

Figure 44 • **Select Export Firmware**



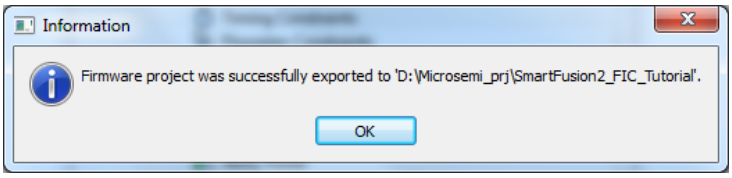
7. The **Export Firmware** dialog box is displayed as shown in Figure 45, page 34. Enter the following information in the **Export Firmware** dialog box:
   - Browse to the Location such as `<C:\ or D:\Microsemi_prj\SmartFusion2_FIC_Tutorial>`.
   - Select the **Create project** check box and select **SoftConsole** v(x.x) from the drop-down list.

Figure 45 • **Export Firmware**



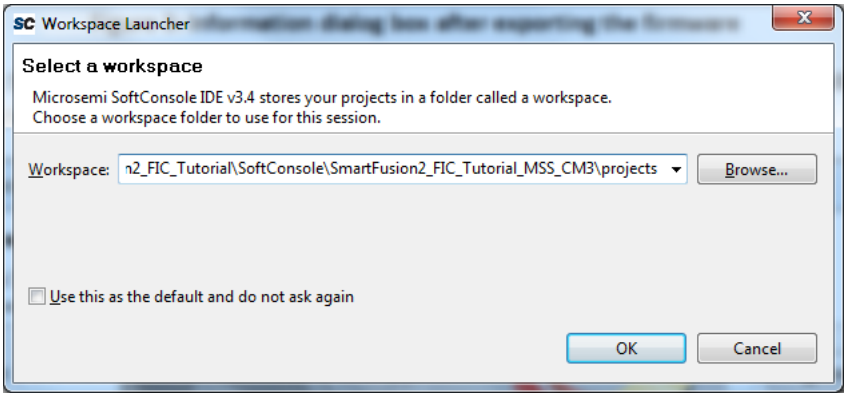
8. Click **OK**. The **Information** dialog box is displayed, as shown in Figure 46.

Figure 46 • **Export Firmware-Information**



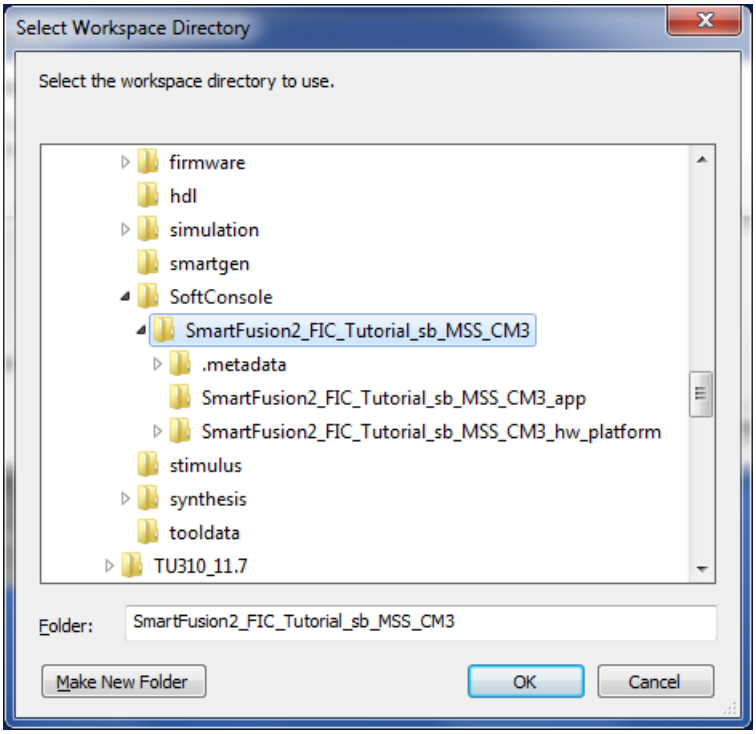
9. Click **OK**.
10. Click **Start** > **Programs** > **Microsemi SoftConsole v(x.x)** > **Microsemi SoftConsole IDE v(x.x)** or double-click the shortcut icon on your desktop. The **SoftConsole Workspace Launcher** is displayed, as shown in Figure 47.

Figure 47 • **SoftConsole Workspace Launcher**



11. Navigate to the **SoftConsole folder** and select **projects** folder as shown in Figure 48.

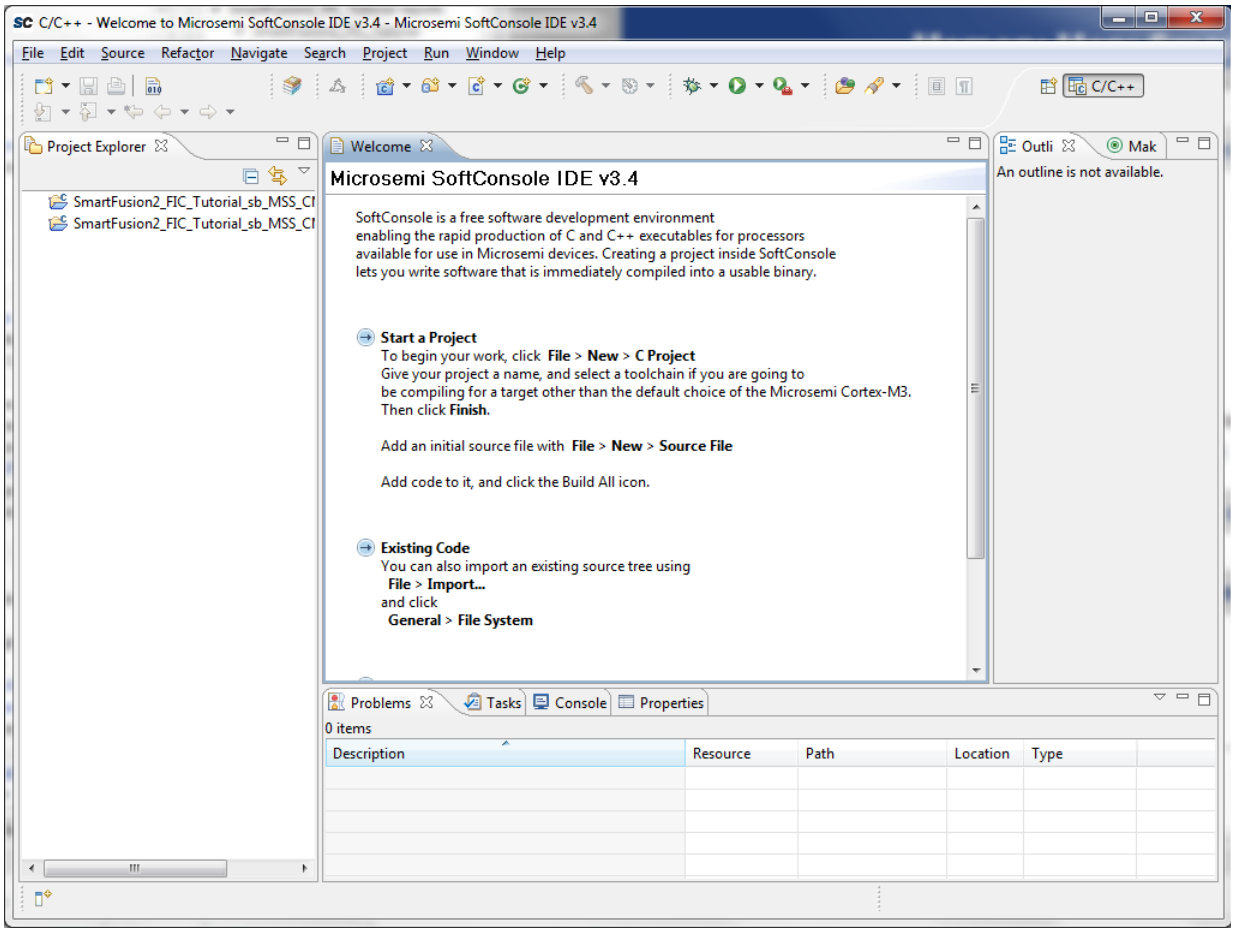
Figure 48 • **Select Workspace Directory**



12. Click **OK**.

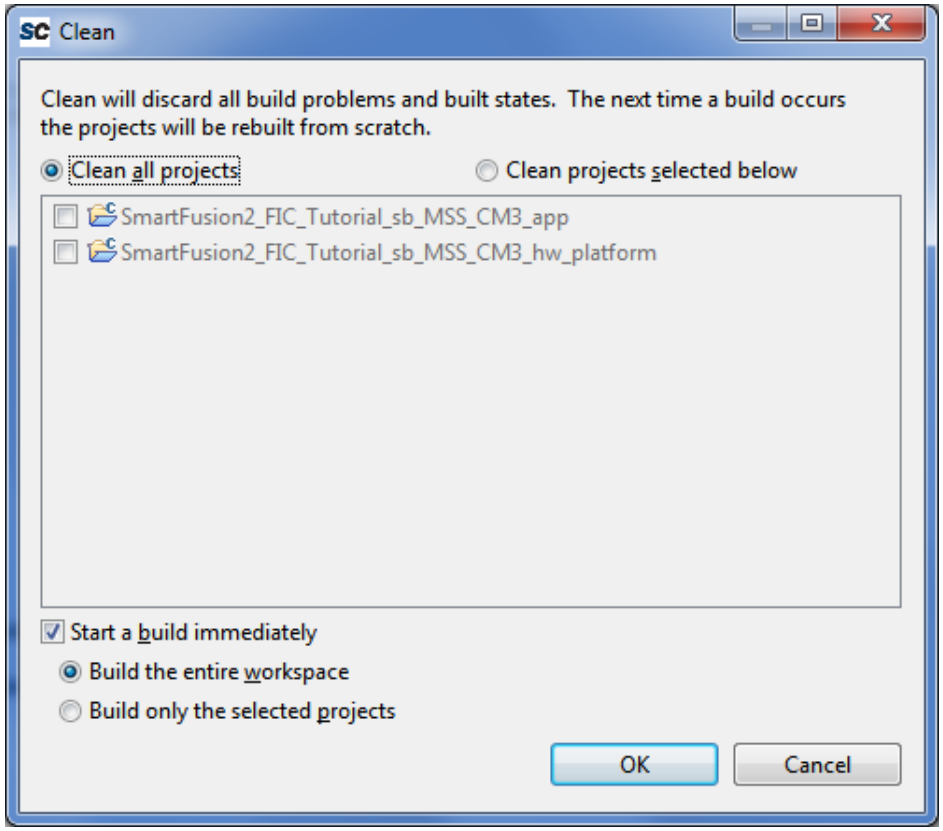
The **SoftConsole IDE** window is displayed, as shown in Figure 49.

Figure 49 • **SoftConsole IDE**



13. Go to the source folder in the downloaded design files folder, copy the code from the `Source_eval.c` file. In SoftConsole editor under **SmartFusion2_FIC_Tutorial_MSS_CM3_app** project, place the copied code in the `main.c` file and delete the existing code.
14. Select **Project > Clean** to perform a clean build. Accept the default settings in the **Clean** dialog box and click **OK**, as shown in Figure 50, page 37.

Figure 50 • **Clean and Build window**



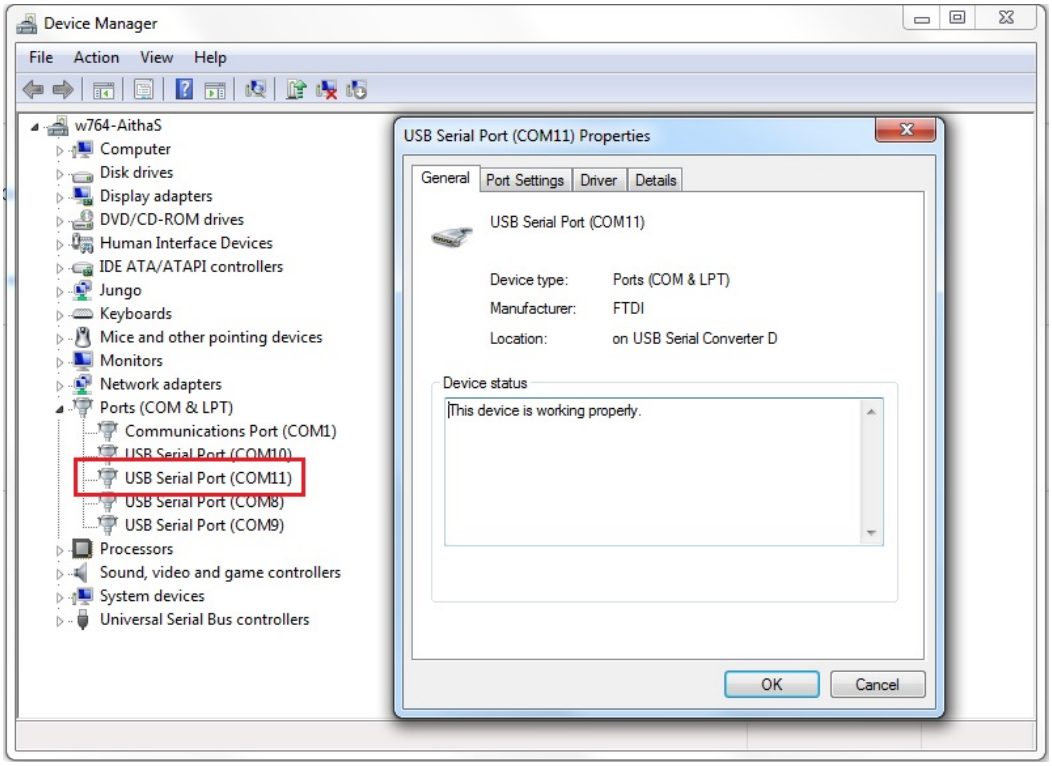
15. Ensure that there are no errors.

2.12 Step 7: Configuring the Serial Terminal Emulation Program

Before running the application program, configure the terminal emulator program on your PC. Perform the following steps to use the SmartFusion2 Security Evaluation Kit board:

1. Connect one end of the USB mini-B cable to the respective USB connector provided on the SmartFusion2 board.
2. Connect the other end of the USB cable to the host PC. Ensure that the USB to UART bridge drivers are automatically detected, as shown in Figure 51.

Figure 51 • **SmartFusion2 Security Evaluation Kit USB Serial Port Drivers**



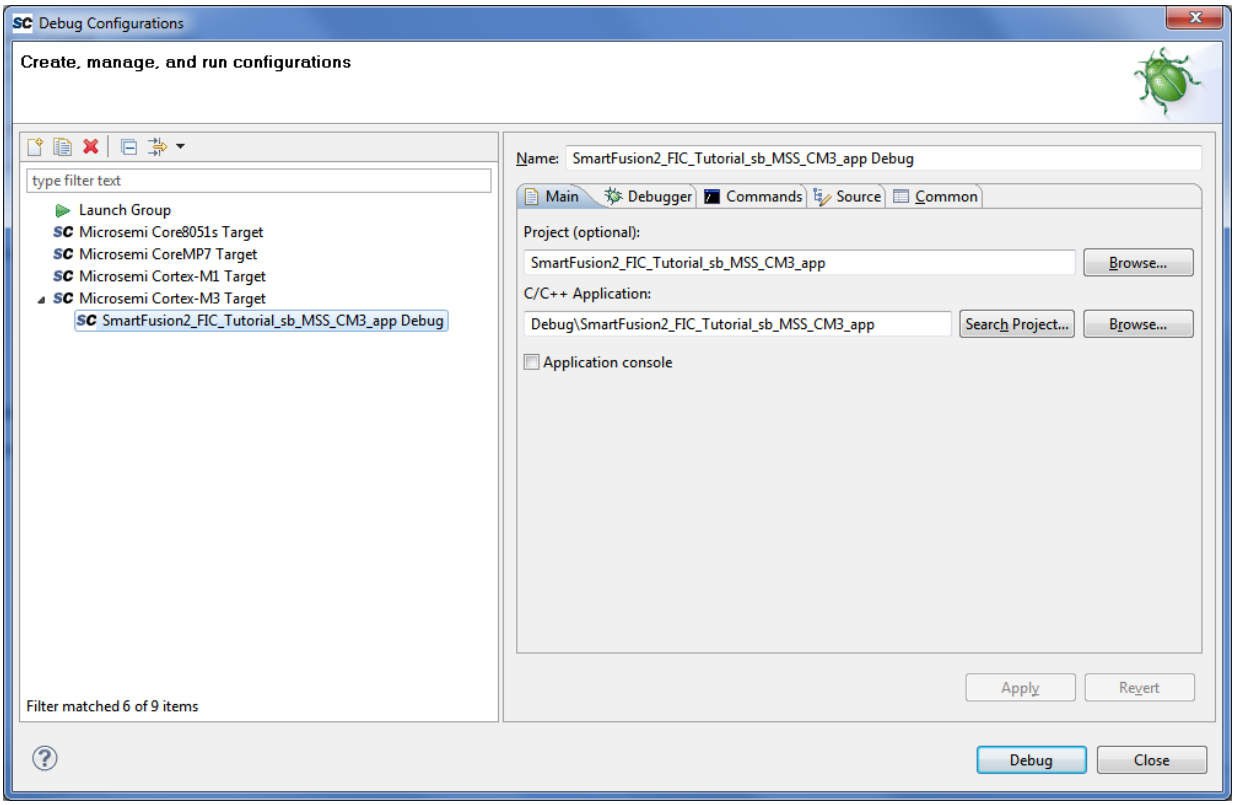
3. If USB to UART bridge drivers are not installed, download and install the drivers from *www.microsemi.com/soc/documents/CDM_2.08.24_WHQL_Certified.zip*.
4. Start a terminal emulator program with the baud rate set to 57600, 8 data bits, 1 stop bit, no parity, and no flow control. Refer to the *Configuring Serial Terminal Emulation Programs Tutorial* for configuring HyperTerminal, Tera Term, and PuTTY.

2.13 Step 8: Debugging the Application Project using SoftConsole

Use the following steps to debug the application project using SoftConsole:

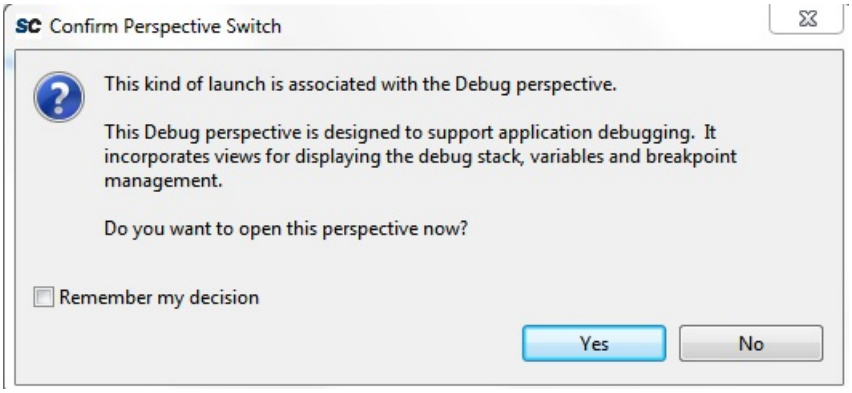
1. Select **SmartFusion2_FIC_Tutorial_sb_MSS_CM3_app Debug** in Project Explorer.
2. Select the **Debug Configurations** from the **Run** menu of the SoftConsole. The Debug dialog is displayed.
3. Double-click on **Microsemi Cortex-M3 Target** to display an image similar to Figure 52.

Figure 52 • **Debug Window**



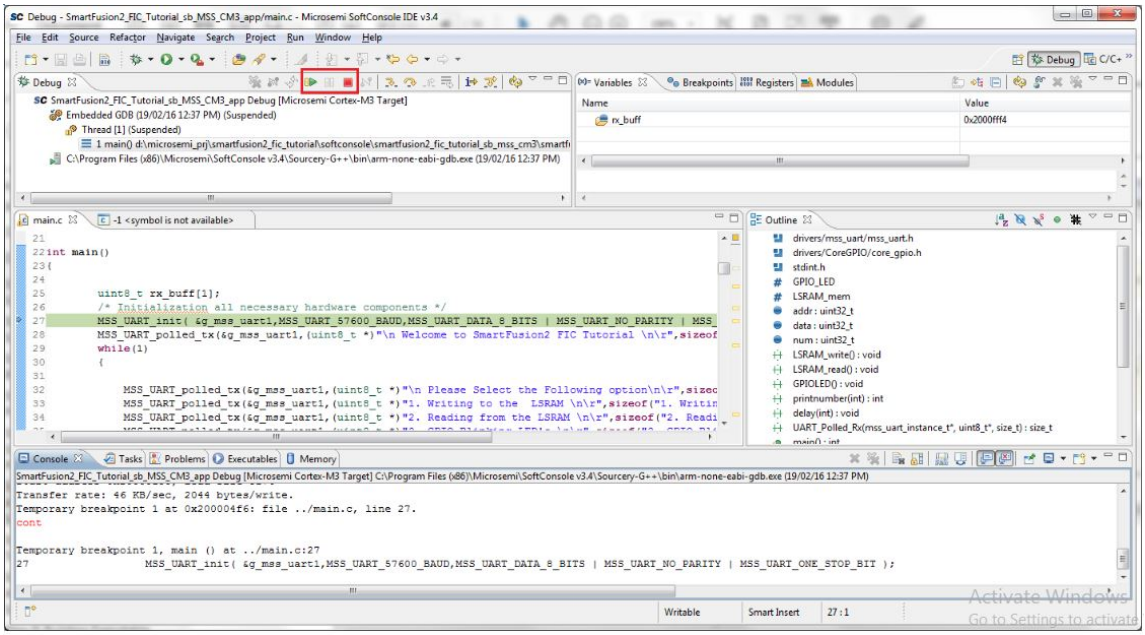
4. Confirm that the following appear on the Main tab in the Debug window:
   - **Name**: SmartFusion2_FIC_Tutorial_sb_MSS_CM3_app Debug
   - **Project (optional)**: SmartFusion2_FIC_Tutorial_sb_MSS_CM3_app
   - **C/C++ Application**: DebugSmartFusion2_FIC_Tutorial_sb_CM3_app
5. Click **Apply** and **Debug**.
6. Click **Yes**, when prompted for **Confirm Perspective Switch**. This displays the debug view mode, as shown in Figure 53, page 40.

Figure 53 • **Confirm Perspective Switch**



7. Debug Perspective is similar as shown Figure 54.

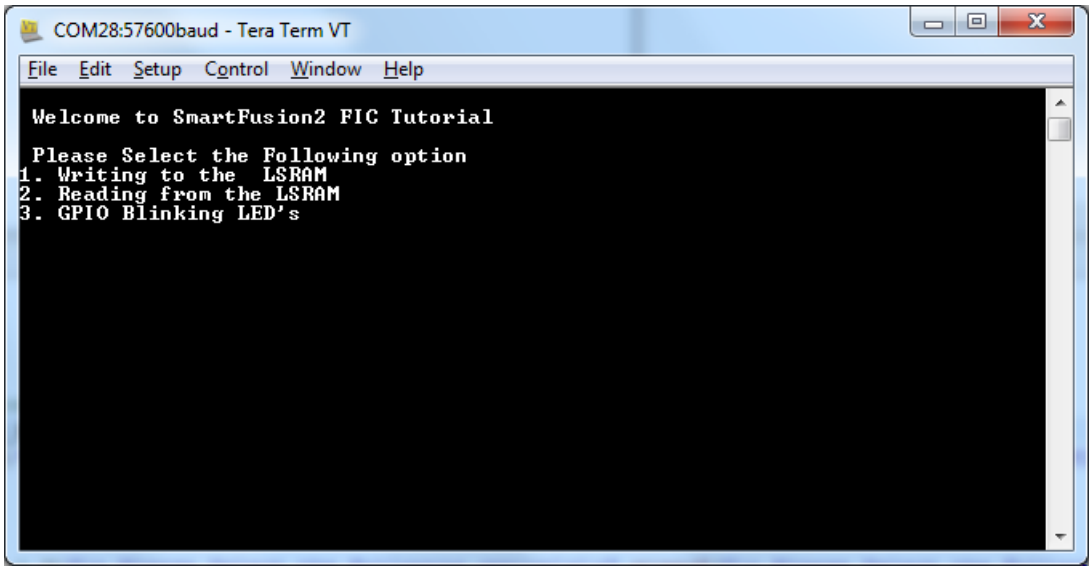
Figure 54 • **Debug Perspective**



8. Run the application by clicking **Run > Resume** or click **Run** icon on the SoftConsole toolbar.

The Application options along with the greeting message are displayed in the terminal program window, as shown in Figure 55.

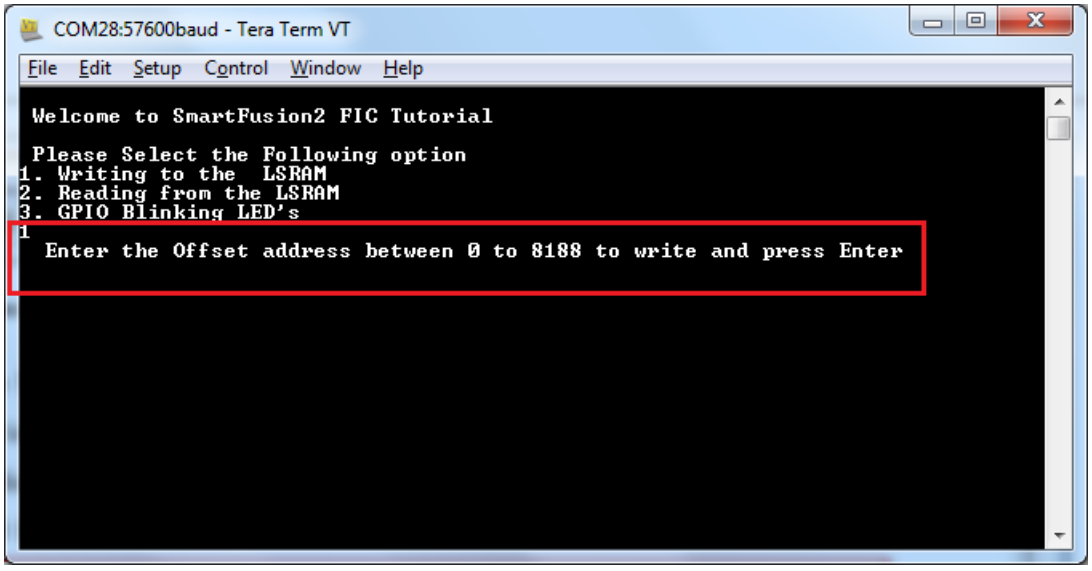
Figure 55 • **Tera Term Window**



9. Select **Writing to the LSRAM**, it prompts for **Enter the Offset address between 0 to 8188 to write and press Enter,** as shown in Figure 56.

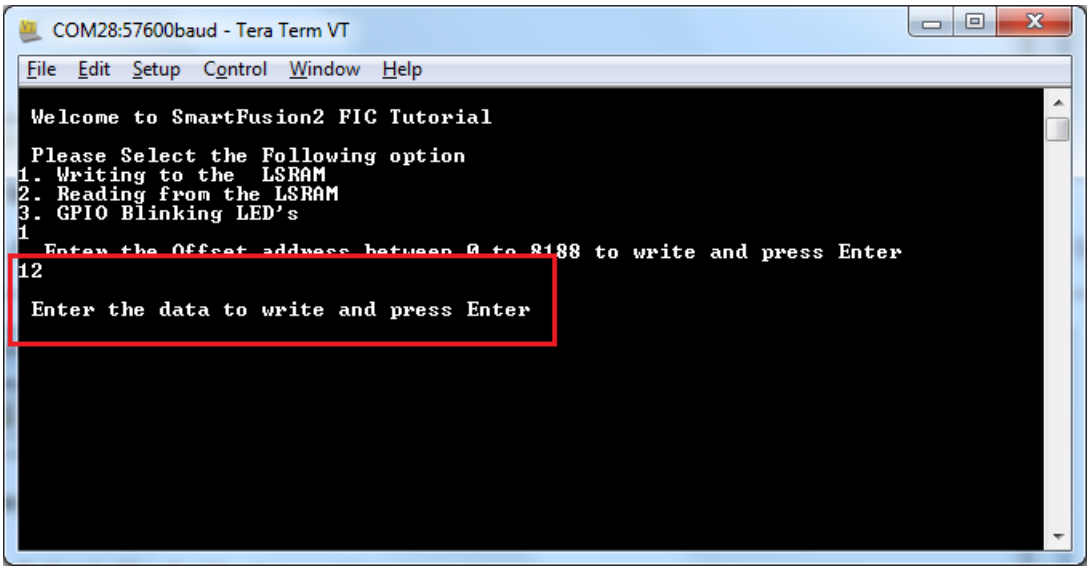
Note: In PuTTY, press CTRL+J instead of Enter.

Figure 56 • **Writing to LSRAM**



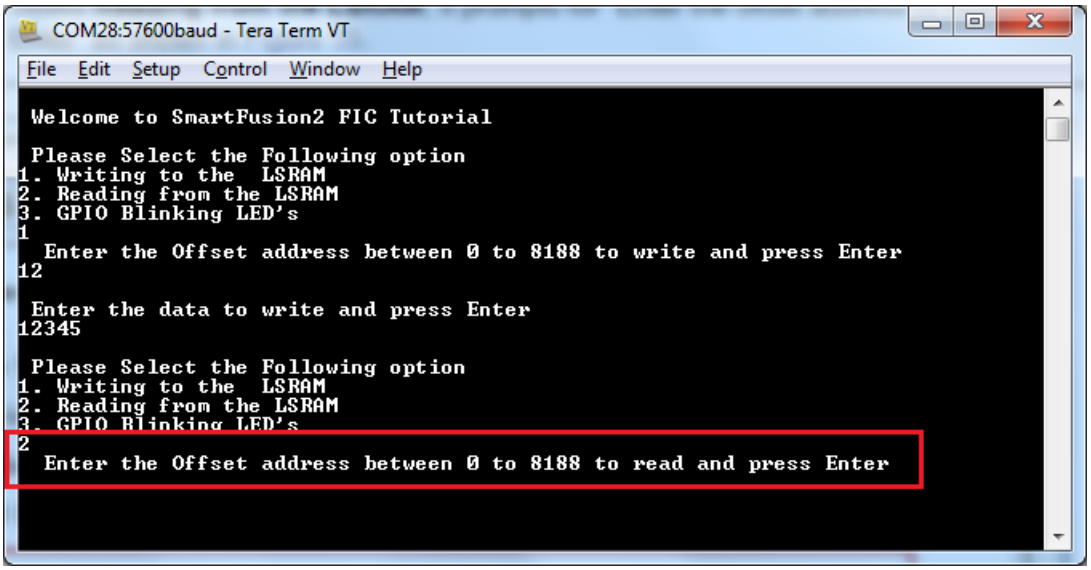
10. After Entering the offset address, it prompts for **Enter the data to write and press enter,** as shown in Figure 57.

Figure 57 • **Writing to LSRAM**



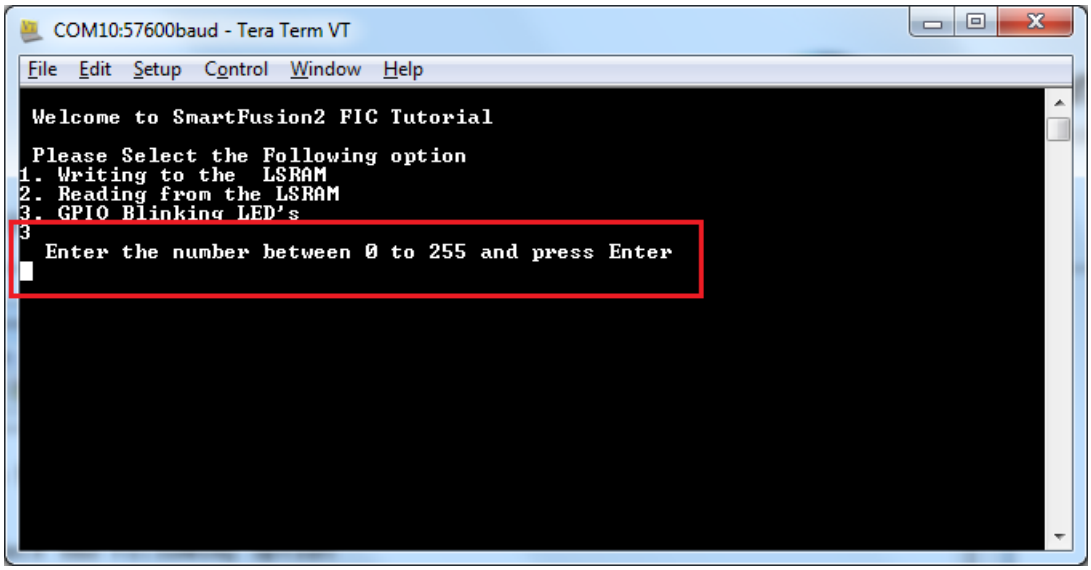
11. Select **Reading from the LSRAM**, it prompts for **Enter the Offset address between 0 to 8188 to read and press Enter,** as shown in Figure 58.

Figure 58 • **Reading from LSRAM**



12. Select **GPIO Blinking LED 's**, it prompts for **Enter the number between 0 to 255 and press Enter,** as shown in Figure 59.

Figure 59 • **Selecting GPIO LED Blinking**



2.14 Step 9: Building Executable Image in Release mode

You can build an application executable image in-release modell and load it into eNVM for executing code in eNVM of the SmartFusion2 SoC FPGA device. You can load the application executable image into eNVM with the help of the eNVM data storage client from System Builder eNVM Configurator. In release mode, you cannot use SoftConsole debugger to load the executable image into eNVM.

2.15 Conclusion

This tutorial outlined the design flow for creating a SmartFusion2 project using Libero SoC design software, configuring the SmartFusion2 MSS, interfacing fabric peripherals to the SmartFusion2 MSS using fabric interface controllers (FIC_0), simulation of the design using BFM commands and running the application design on board.

3 Appendix: Programming the Device Using FlashPro Express

This section describes how to program the SmartFusion2 device with the programming job file using FlashPro Express.

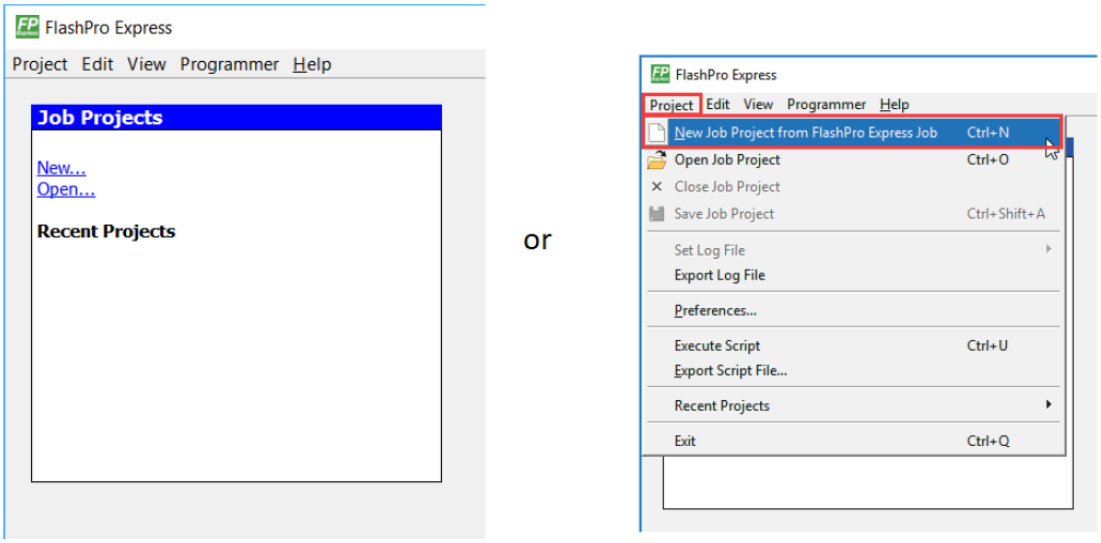
To program the device, perform the following steps:

1. Ensure that the jumper settings on the board are the same as those listed in Table 2, page 29.

Note: The power supply switch must be switched off while making the jumper connections.

2. Connect the power supply cable to the **J6** connector on the board.
3. Power **ON** the power supply switch **SW7**.
4. On the host PC, launch the **FlashPro Express** software.
5. Click **New** or select **New Job Project from FlashPro Express Job** from **Project** menu to create a new job project, as shown in Figure 60.

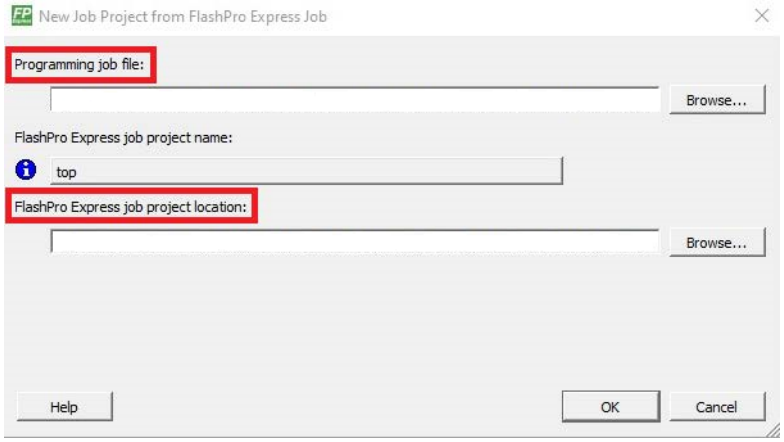
Figure 60 • **FlashPro Express Job Project**



6. Enter the following in the **New Job Project from FlashPro Express Job** dialog box:

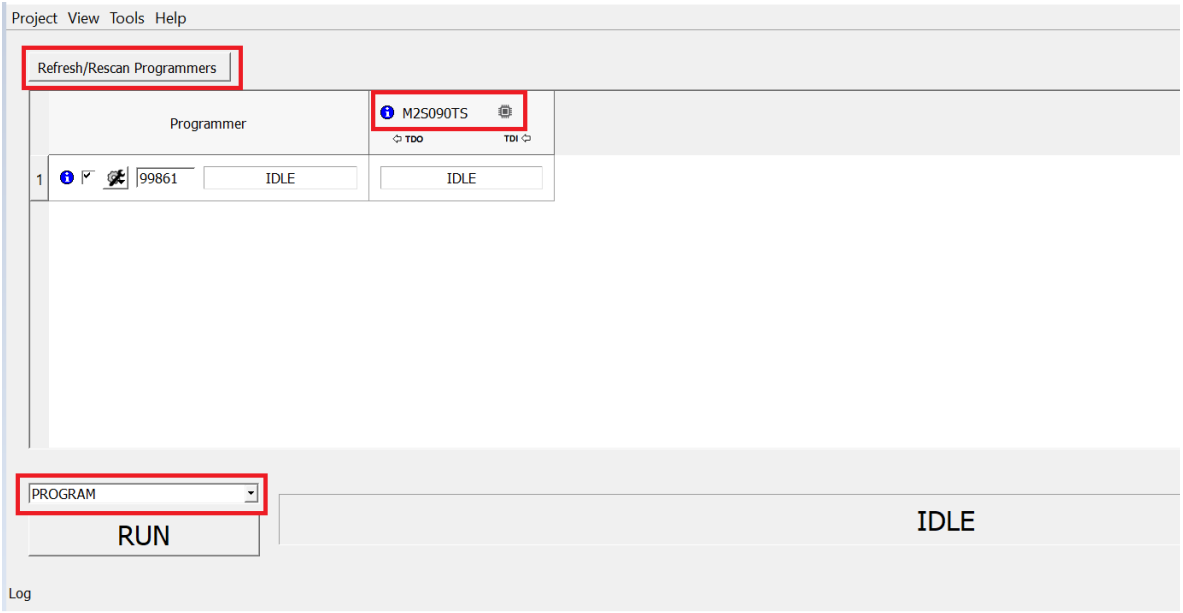
- **Programming job file:** Click **Browse**, and navigate to the location where the .job file is located and select the file. The default location is:
  <download_folder>\m2s_tu310_df\Programming_Job
- **FlashPro Express job project name:** Click **Browse** and navigate to the location where you want to save the project.

Figure 61 • **New Job Project from FlashPro Express Job**



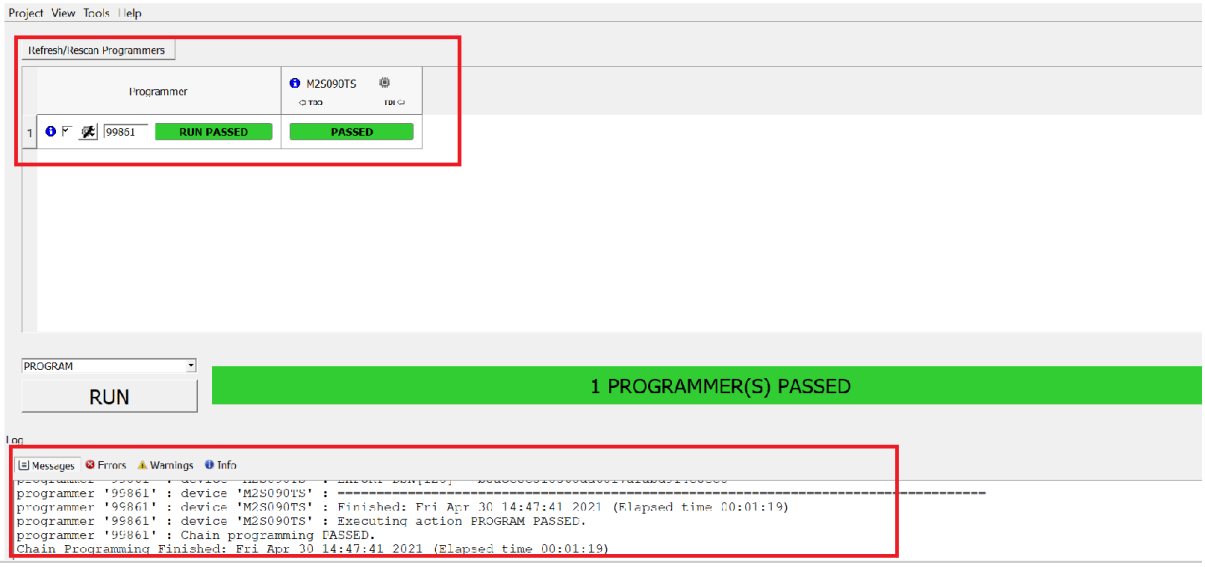
7. Click **OK**. The required programming file is selected and ready to be programmed in the device.
8. The FlashPro Express window appears as shown in Figure 62. Confirm that a programmer number appears in the Programmer field. If it does not, confirm the board connections and click **Refresh/Rescan** Programmers.

Figure 62 • **Programming the Device**



9. Click **RUN**. When the device is programmed successfully, a **RUN PASSED** status is displayed as shown in Figure 63.

Figure 63 • **FlashPro Express—RUN PASSED**



10. Close **FlashPro Express** or in the Project tab, click **Exit**.