Libero® SoC Quick Start Guide

for Software v11.4

Microsemi

Table of Contents

Introduction and Design Overview

This tutorial introduces you to the Microsemi® system-on-chip (SoC) development flow using Libero® SoC. It is a starting point for any FPGA design engineer who is new to Microsemi FPGAs, or just wants to learn more about Libero SoC. It demonstrates the basics on how to use Libero SoC and its tools to create a simple design. The sample design incorporates the Libero SoC Catalog IP core macros and hard embedded microcontroller subsystem (MSS) and guides you through synthesis, simulation and programming.

The tutorial was developed using the SmartFusion2® Evaluation Kit Board and M2S025T device. For details on the SmartFusion2 Evaluation Kit Board, refer to the SmartFusion2 Evaluation Kit User Guide. This tutorial covers the following tools and features:

- Libero SoC
  - Design flow
  - SmartDesign
  - Catalog
- Mentor Graphics® ModelSim® ME
- Synopsys Synplify Pro ME
- Microsemi Designer
  - Compile
  - Place and Route
  - I/O Attribute Editor
  - Pin Editor
  - SmartTime
- FlashPro
- SoftConsole

Tutorial Requirements

Software Requirements

This tutorial requires that the following software is installed on your computer:

- Libero SoC v11.4, which can be downloaded from http://www.microsemi.com/soc/download/software/libero/default.aspx
- FlashPro v10.0 or higher, which is installed as part of the Microsemi Libero SoC installation and can be launched from within Libero SoC or standalone.
- SoftConsole v3.4 or higher, which is installed as part of the Microsemi Libero SoC installation and can be launched from within Libero SoC or standalone.
- HyperTerminal or similar software (PuTTy or Tera Term), normally under Start > Programs > Accessories > Communications > HyperTerminal.

  Note: HyperTerminal is no longer a standard feature on newer windows installations. For other alternatives to HyperTerminal, see

  Alternatives to HyperTerminal in windows 7 | Windows Reference
- USB Drivers for USB to UART connection.

  Install the driver from

  http://www.microsemi.com/document-portal/doc_download/131593-usb-uart-driver-files

Hardware Requirements

You will need the SmartFusion2 Evaluation Kit, which consists of the following hardware:

- SmartFusion2 Evaluation Kit Board (M2S-EVAL-KIT DVP-102-000402-001 Rev.C)
- FlashPro4 JTAG Programmer
- 12V/2A Wall-Mounted Power Supply
- USB 2.0 A-male to mini-B Y-Cable for UART/power interface (up to 1A) to PC

Extracting Source Files

Download the Design Files from http://www.microsemi.com/document-portal/doc_download/134397-liberosoc-qs-df. Extract design files in the root directory of your local drive (e.g., C :\) using 7-zip.

The design files for this tutorial include:

- Timer_0 and Timer_1 HDL files - HDL source files for the Interrupt Generator block. Verilog and VHDL versions are provided.
- Main.c file - C file for running the Application in SoftConsole
- Post-layout_wave.do file - wave.do file for running post-layout simulation in ModelSim

Design Overview

SmartFusion2 cSoC FPGA devices contain a 166 MHz ARM® Cortex™-M3 processor, Ethernet MAC, DMA engine, real-time counter (RTC), embedded nonvolatile memory (eNVM), embedded SRAM (eSRAM), and FPGA fabric consisting of programmable 4-input LUTs and D Flip Flops, static random access memory (SRAM), Clock Conditioning Circuitry (CCC) with dedicated phase locked loops (PLLs) and MATH blocks. The Cortex-M3 processor includes an interrupt controller called the nested vectored interrupt controller (NVIC).

The SmartFusion2 device has 16 dedicated fabric-to-MSS interrupt lines, MSS_INT_F2M[15:0].

In addition, there are 32 General Purpose Inputs/Outputs (GPIOs) which can be configured as Inputs, Outputs or Bi-directional signals., When configured as Inputs and routed to the Fabric, the 32 GPIOs can be used as additional Fabric-to-MSS interrupt resources.

The design example (Figure 1) uses two GPIOs and one dedicated Fabric Interrupt to interrupt the MSS from the FPGA fabric. The interrupt generator block has two timer blocks, Timer0 and Timer1. The universal asynchronous receiver/transmitter (MMUART) in MSS is used for printing interrupt messages to the terminal emulator.

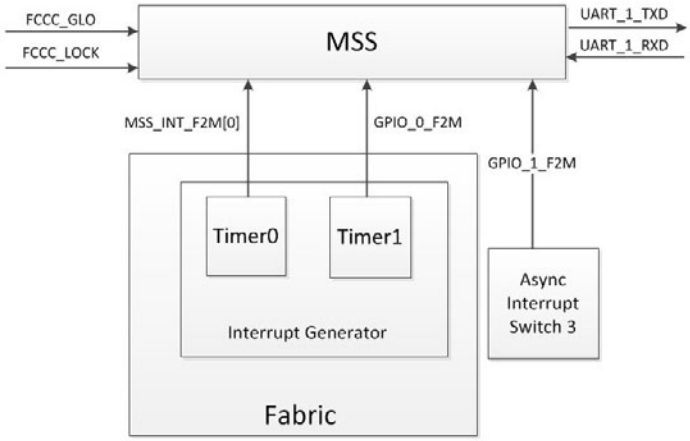


Figure 1 **• Tutorial Design Block Diagram**

The example design contains the following blocks:

- MSS
- 25/50 MHz RC Oscillator
- Fabric Clock Conditioning Circuit (CCC) with dedicated PLL
- SYSRESET to generate the System Reset signal
- Interrupt generator which consists of two counters: Timer0 and Timer1

The example design contains the following inputs and outputs:

- Inputs: Clock, Reset and External interrupt signal
- Outputs: MMUART to HyperTerminal/Tera Term

Interrupt Generator Block Description

The interrupt generator (Figure 1) has two timer blocks, Timer0 and Timer1. The two timers are implemented in the FPGA fabric and run with the same clock source. The clock conditioning circuit (CCC) generates a 10 MHz clock, which is the clock source for the timers.

Timer 0 and Timer 1 are internal synchronous interrupts. The output from Timer0 is connected to MSS_INT_F2M[0] of the MSS and the Timer1 block is connected to the GPIO_0_F2M. The third interrupt is generated asynchronously by pressing SW3 on the board. SW3 is connected to FPGA I/O pin K18, which is connected to the GPIO_1_F2M of the MSS. Figure 1 shows the block diagram of the interrupt generator.

Whenever any of the interrupts occur (Timer 0, Timer 1, or the SW3 switch interrupt), the Cortex M-3 processor executes the corresponding interrupt service routine, which sends a message to the MMUART indicating the source of the interrupt. The terminal displays the message.

This tutorial provides step-by-step instructions on how to configure the MSS and timer blocks, simulate the timer block, synthesize, place and route, and generate a programming file for the entire design. It also guides you on how to program the design on the SmartFusion2 Evaluation Kit Board and how to run the firmware applications on the Cortex M-3 using SoftConsole.

1 – Step 1 - Creating a Libero SoC Project and Configuring the SmartFusion2 Microcontroller Subsystem

1. Click **Start > Programs > Microsemi Libero SoC v11.4> Libero SoC v11.4** Libero SoC opens (Figure 1-1).

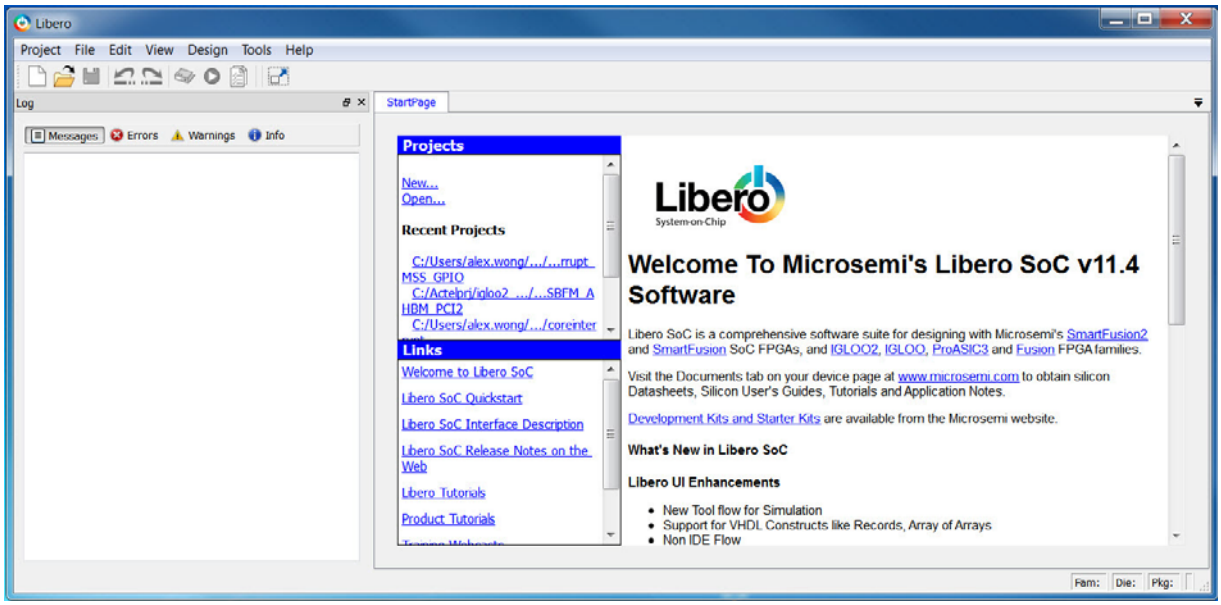


Figure 1-1 **• Libero SoC**

2. From the **Project** menu, choose **New Project**. The New Project dialog box appears (Figure 1-2).

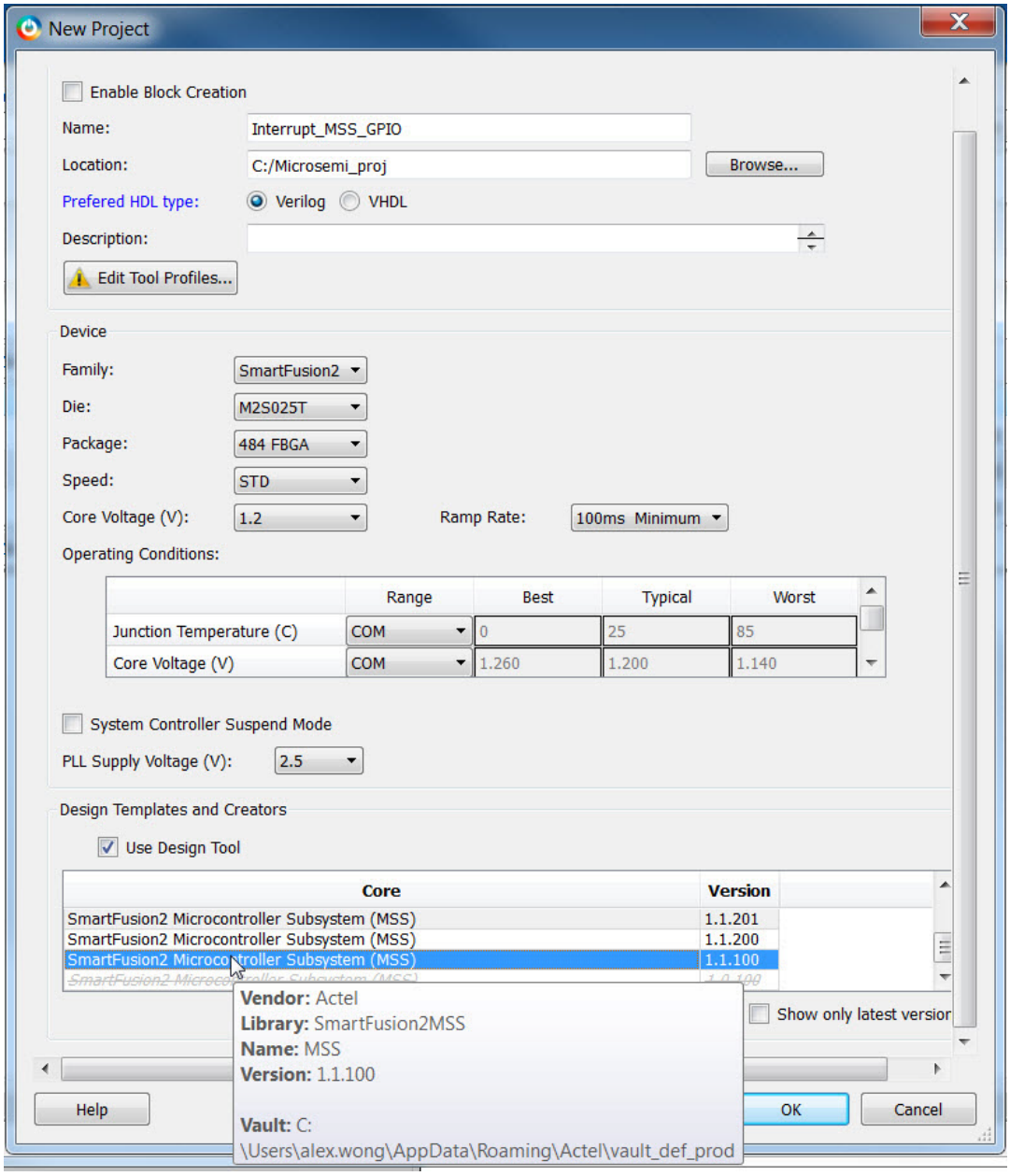


Figure 1-2 **• Libero SoC New Project Dialog Box**

3. Enter the information as shown in Figure 1-2. If a value in the dialog box is not specified below you can use the default.

- Name: Interrupt_MSS_GPIO
- Location: C:/Microsemiprj
- Preferred HDL type: VHDL (for VHDL projects) or Verilog (for Verilog projects)
- Family: SmartFusion2
- Die: M2S025T
- Package: 484 FBGA
- Speed: STD
- Core Voltage: 1.2
- Check the Use Design Tool checkbox
- Select MSS Core Version 1.1.100 or later

If the SmartFusion2 MSS text is in italics, the core is available for download but not in your IP vault. Double-click the core name to download the latest MSS core

4. Click **OK** to close the New Project dialog box.

Note: Click Yes if the software prompts you to download the MSS. This occurs when the vault does not have the selected MSS core.

Libero SoC Interface Description

The Libero SoC interface enables a push-button design flow via several tabs and an expanded work area.

- **Work window** - Displays the SmartDesign canvas, HDL editor or Report view. Click the **Maximize/Restore Work Area** button to show/hide the other interface elements.
- **Design Hierarchy tab** - Lists components and modules in your design. Use it to manage your design files.
- **Files tab** - Lists your project files by directory; use it to manage your project files directly.
- **Design Flow window** - Enables you to execute the push-button design flow or, if you prefer, open the tools interactively and specify custom settings.
- **Catalog** - Lists the cores available for use in your design. Click and drag them onto your Canvas and add them to your design.
- **HDL Templates** - Lists HDL templates for common constructs; double-click a template to copy it to the clipboard, and then paste it into your HDL to use it in your design.
- **Log window** - Lists all messages, errors, warnings, and info for the SoC tools. Click each type to filter accordingly.

2 – Step 2 - Configuring the SmartFusion2 MSS

If you completed the steps in the previous chapter, the **SmartDesign** Canvas opens with the SmartFusion2 MSS component (Figure 2-1).

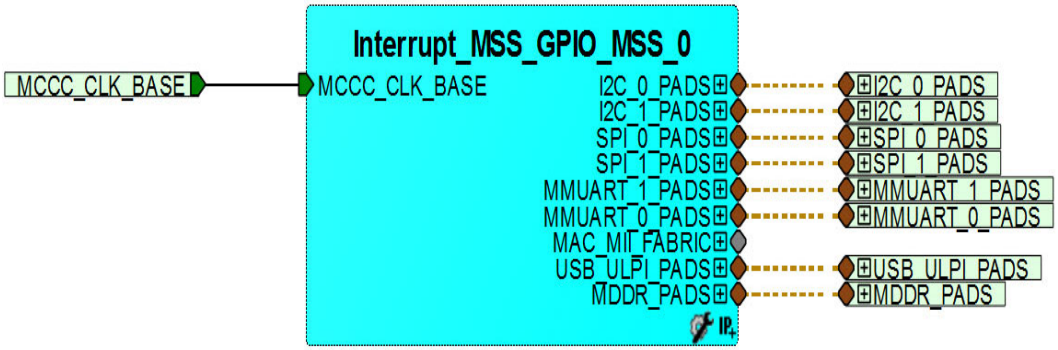


Figure 2-1 **• SmartFusion2 MSS on the Canvas**

1. Double-click **Interrupt_MSS_GPIO_MSS_0** to open the SmartFusion2 MSS configurator (Figure 2-2)

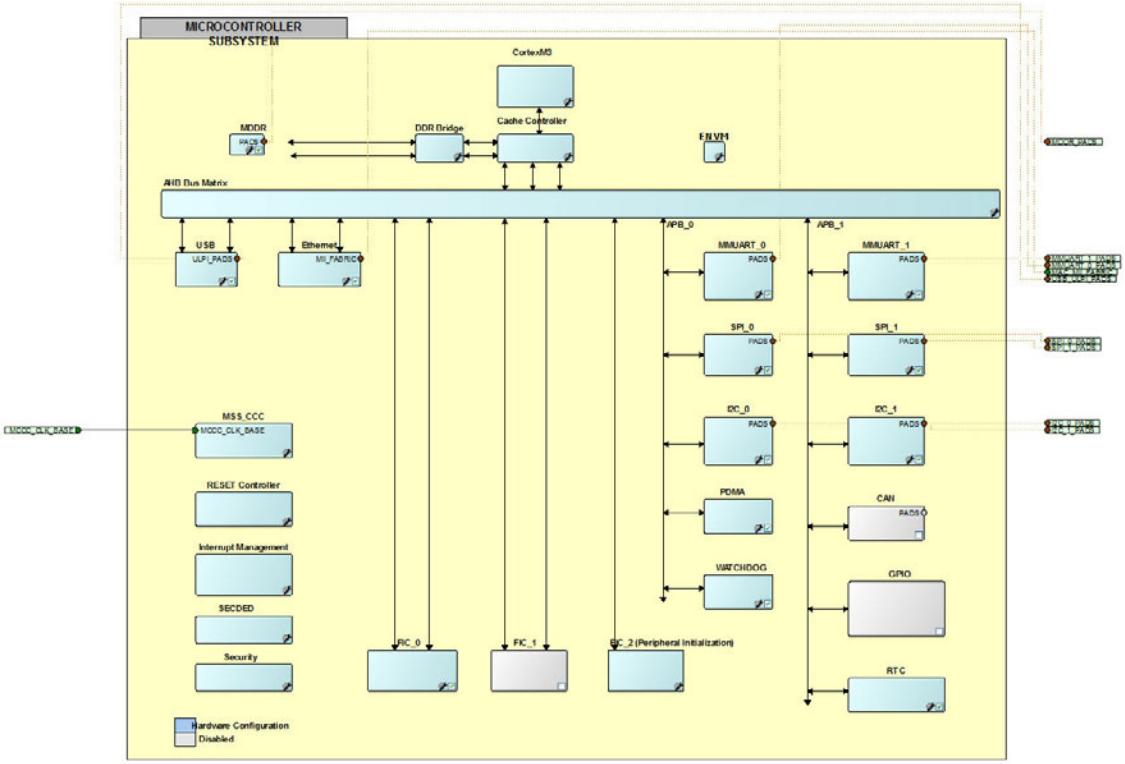


Figure 2-2 **• SmartFusion2 MSS Configurator**

Configurable MSS peripherals have a small wrench symbol and checkbox in the lower right corner (Figure 2-3). To disable a peripheral, select the peripheral, right-click and choose **Disable**, or click the checkbox. The peripheral turns grey to indicate it has been disabled.

Disabled peripherals can be enabled by repeating the procedure.

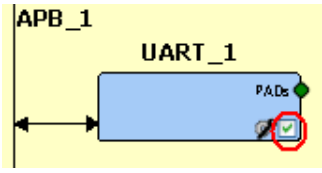


Figure 2-3 **• Disabling the Peripheral**

2. Disable the following peripherals:
   - External Memory Controller (MDDR)
   - MMUART_0
   - SPI_0
   - SPI_1
   - I2C_0
   - I2C_1
   - FIC_1
   - PDMA

- WATCHDOG
- RTC
- CAN
- USB
- Ethernet

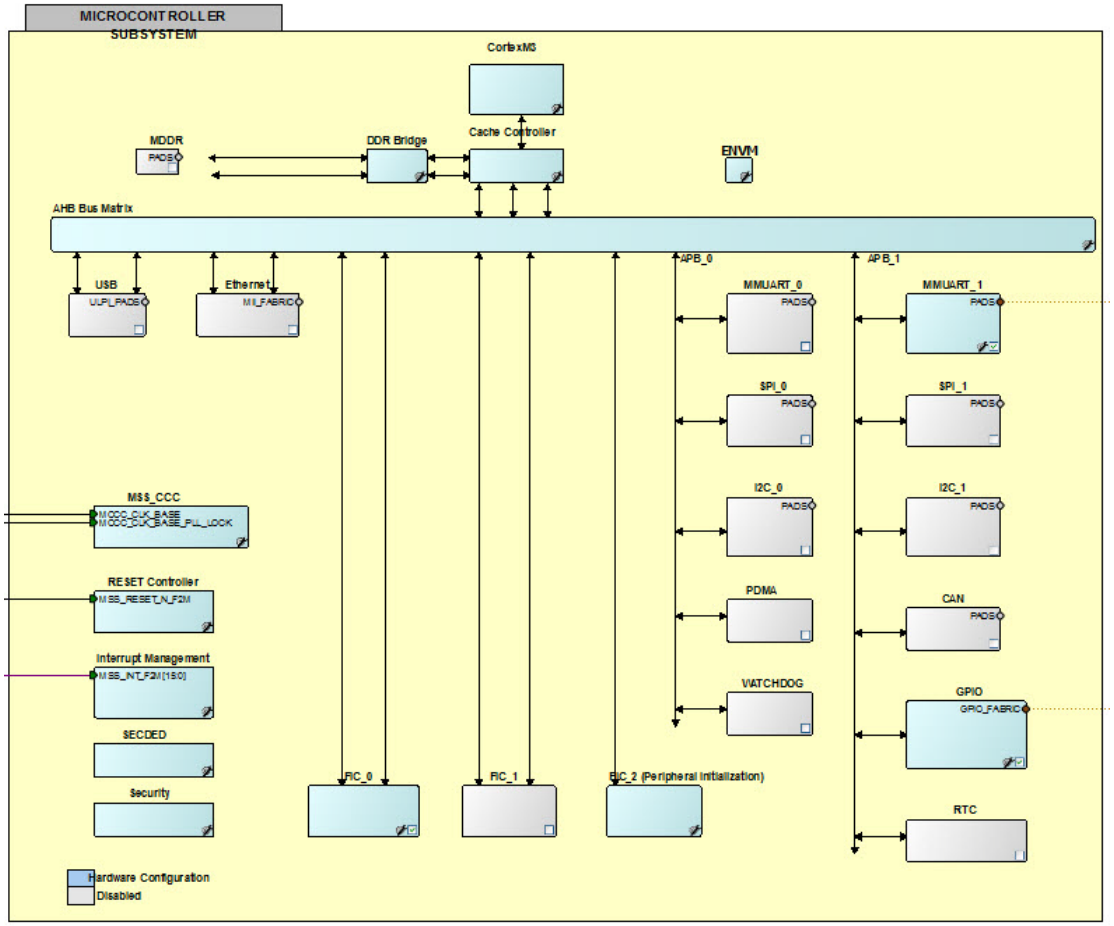


***Figure 2-4* • SmartDesign MSS in the SmartDesign Canvas**

3. Double-click the **MSS CCC** block to configure the MSS CCC (Figure 2-4).
4. Set the MSS_CLK Configurator options as follows (shown in Figure 2-5):
   - CLK_BASE: 10 MHz
   - Monitor FPGA Fabric PLL Lock (CLK_BASE_PLL_LOCK): checkbox should be checked.
   - M3_CLK: 160 MHz

   Use the default for all other settings.

Use the default for all other settings.

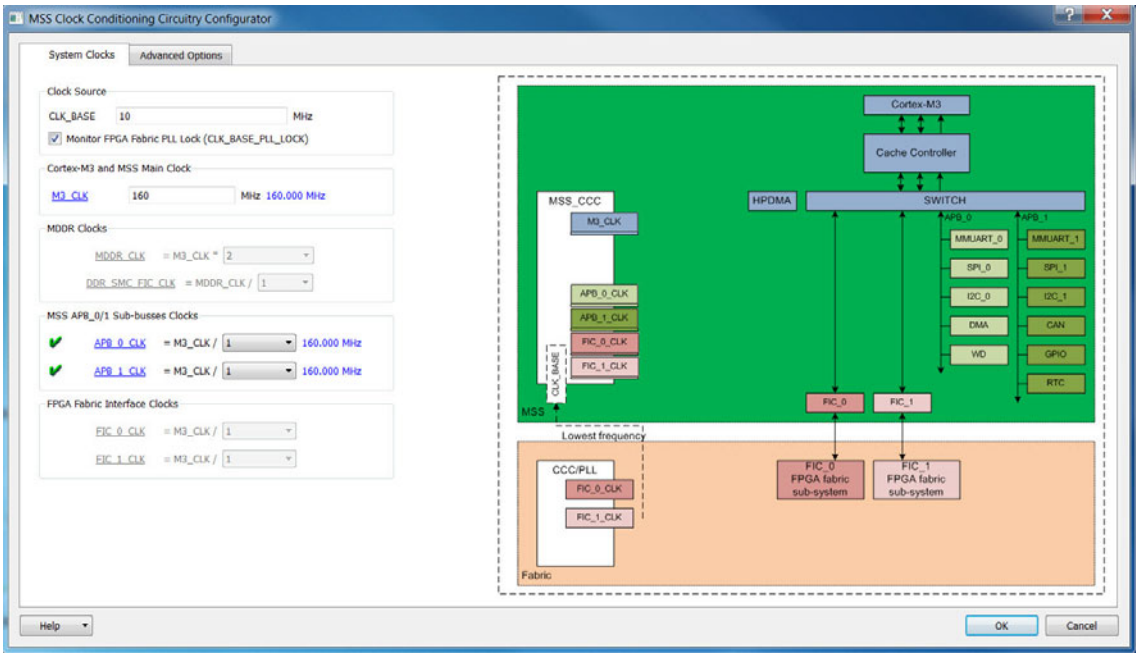


Figure 2-5 **• MSS Clock Conditioning Circuitry Configurator**

5. Click **OK** to close the MSS **Clock Conditioning Circuitry Configurator** box.
6. Double-click **Interrupt Management** to open and configure the MSS Interrupts. Check the option **Use Fabric to MSS Interrupt** (Figure 2-6). Click **OK** to continue.

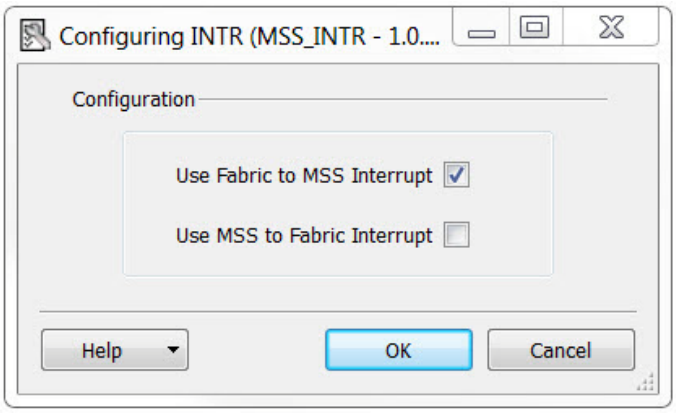


Figure 2-6 **• MSS Interrupts Management Configuration**

7. Double-click the **GPIO** block in MSS to configure it. Configure GPIO_0 and GPIO_1 as **Inputs** (as shown in Figure 2-7) and select Fabric_A from the drop-down list for the Connection. Leave the

rest of the ports at their default settings. Click **OK** to close the GPIO configuration dialog box. The F2M_GPI[1:0] ports are promoted to the top level automatically.

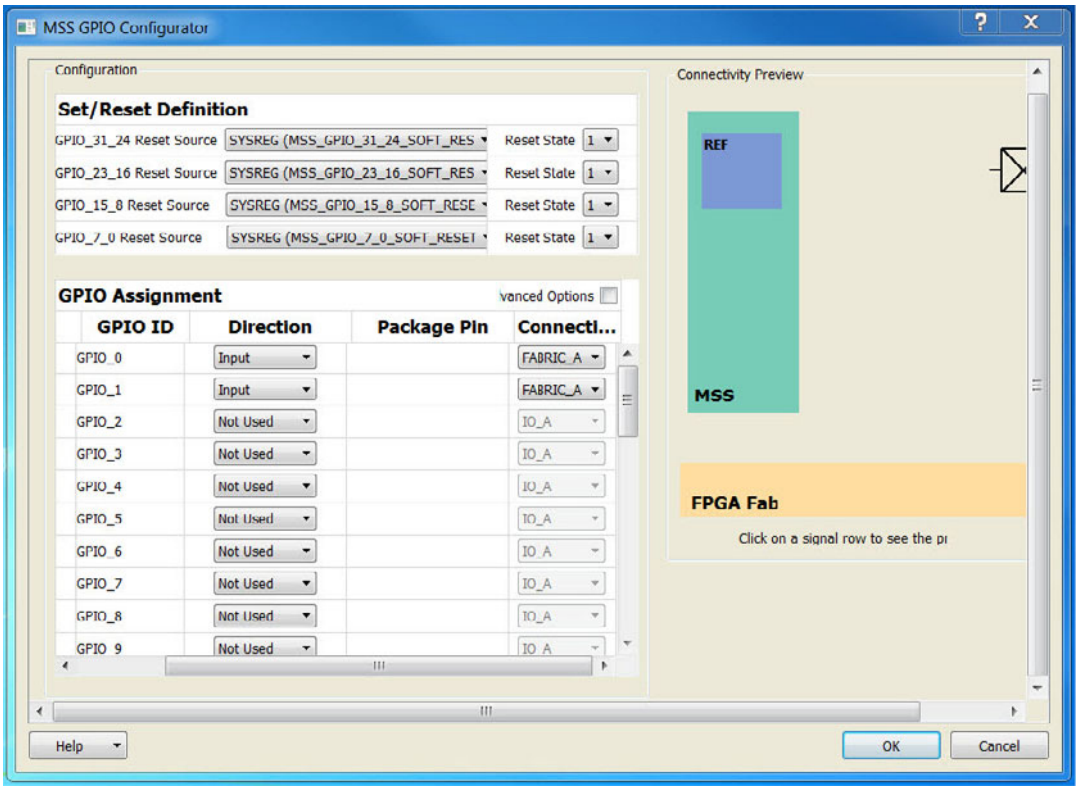


Figure 2-7 **• MSS GPIO Configurator**

8. Double-click the Reset Controller block inside the MSS to open the Reset Controller Configurator.
9. Check **Enable FPGA Fabric to MSS Reset (MSS_RESET_N_F2M)**. Uncheck the other items (Figure 2-8).

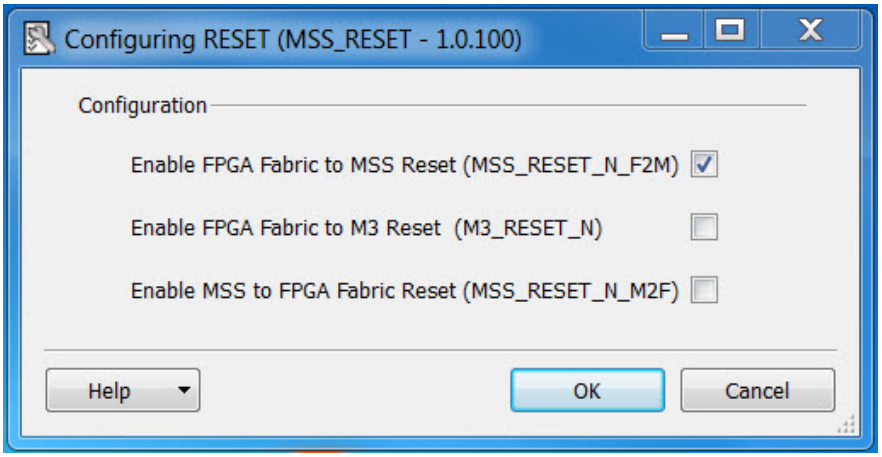


Figure 2-8 **• Reset Configurator**

10. Double-click the eNVM module available in the SmartDesign MSS Configurator to add an eNVM data storage client. The Data Storage Client stores the Cortex-M3 application code. The eNVM Configurator appears (Figure 2-9).

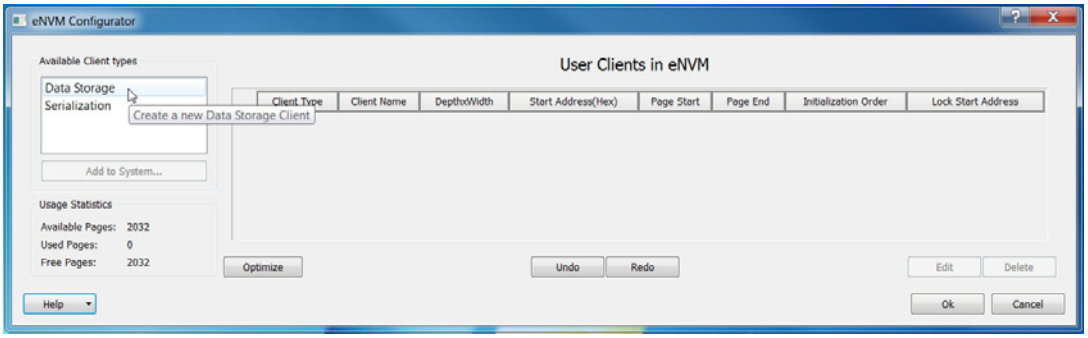


***Figure 2-9* • eNVM Configurator**

11. Double-click Data Storage under Available Client Types to add a Data Storage Client. Enter My_app in the Client name field and select the **No Content (Client is a placeholder)** radio button. This creates a placeholder in the eNVM to store the Application code.

12. Enter 32 for Size of word and 2048 for Number of Words. See Figure 2-10).

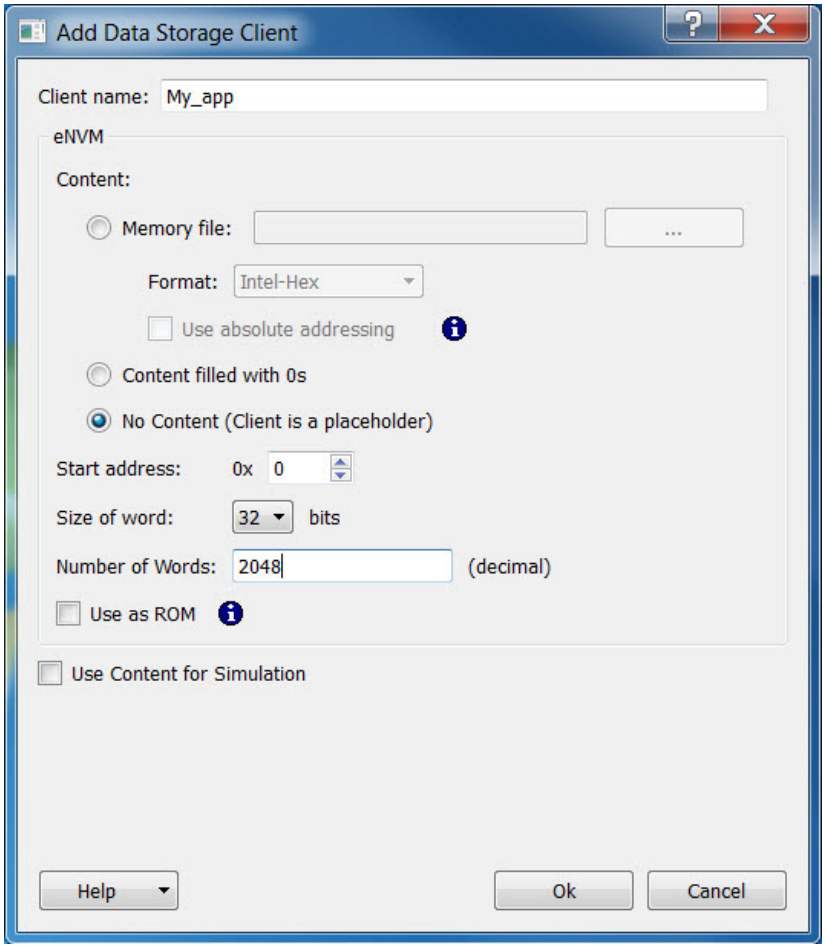


***Figure 2-10* • Add Data Storage Client**

13. Click **OK** to close the dialog box. The Data Storage Client is created with the correct depth and width (Figure 2-11).

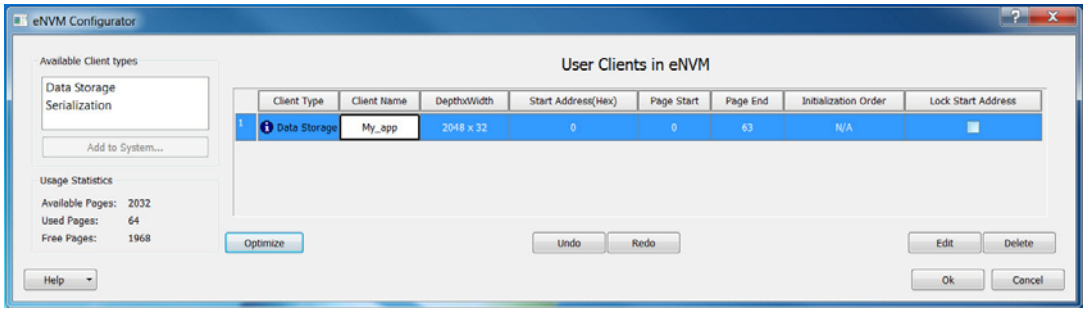


***Figure 2-11* • User Client created in eNVM**

14. Click **OK** to exit.
15. Save the Interrupt_MSS_GPIO component (**File > Save Interrupt_MSS_GPIO**).

The MSS component (Interrupt_MSS_GPIO_MSS_0) appears on the SmartDesign Canvas. The Warning symbol indicates that the port list for the SmartFusion2 component has changed.

16. Right-click and choose Update Instance(s) with Latest Component (Figure 2-12).

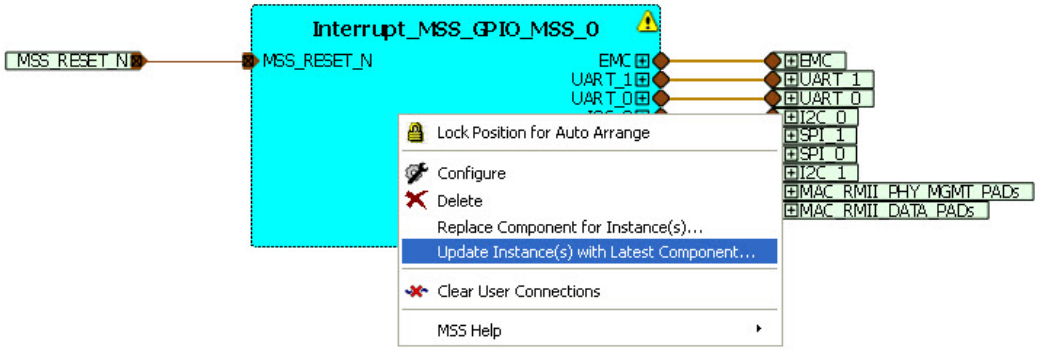


***Figure 2-12* • Update the MSS Component**

Device I/O Standard

I/O Bank 1 and I/O Bank 2 of the SmartFusion2 device are connected to 3.3 V on the Evaluation Kit Board. Set the I/O standard of the design's I/O to LVCMOS 3.3 as follows:

1. From the Project menu, choose Project Settings and then click **Device I/O Settings** (**Project > Project Settings > Device I/O Settings**).
2. From the Default I/O Technology drop-down list, choose **LVCMOS 3.3V** (Figure 2-13).

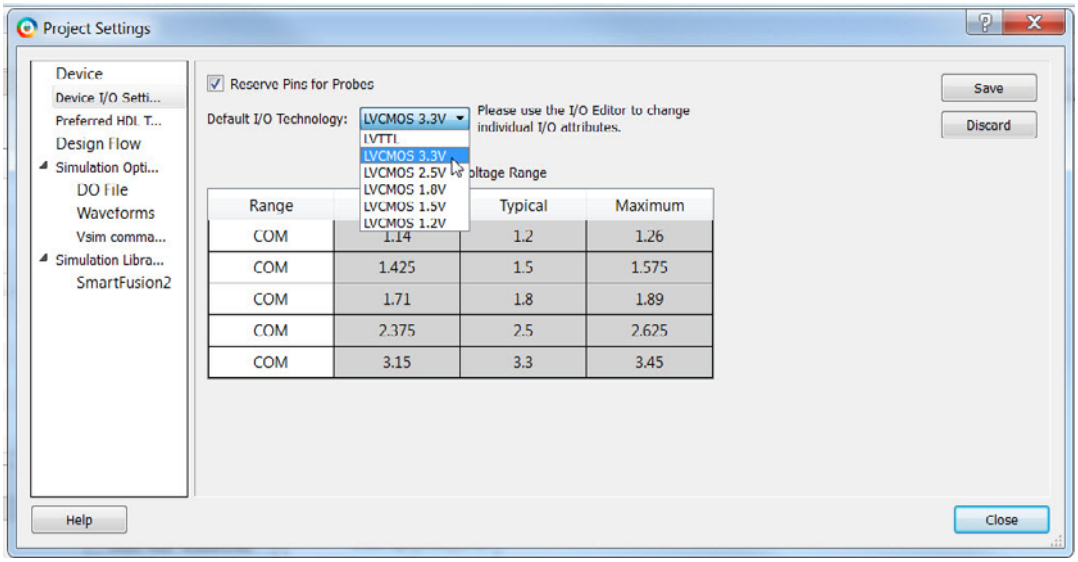


***Figure 2-13* • Device I/O Settings**

3. Click **Save** and **Close** to close the Project Settings dialog box.

3 – Step 3 - Import Timer Blocks

The two counter blocks required in the design are provided with this tutorial. Both Verilog and VHDL versions of the counters are available. The two counters serve as timers to generate synchronous interrupts to the MSS.

1. Import the Timer Block HDL (File > Import > HDL Source Files) into the Project (Figure 3-1).

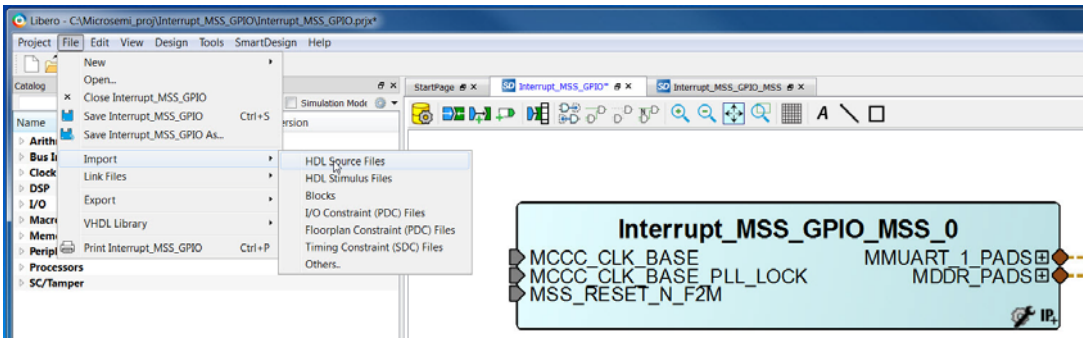


Figure 3-1 **• Import Timer Block HDL File via Import HDL Source Files**

2. Navigate to the folder location of the Verilog or VHDL timer files where you have extracted the source files for this tutorial. Select the Verilog files (Timer_0.v and Timer_1.v) or the VHDL files (Timer_0.vhd and Timer_1.vhd) to match your Preferred HDL Type in your Project Settings. Click **Open**. Timer_0 and Timer_1 appear in the Design Hierarchy after the Import (Figure 3-2).

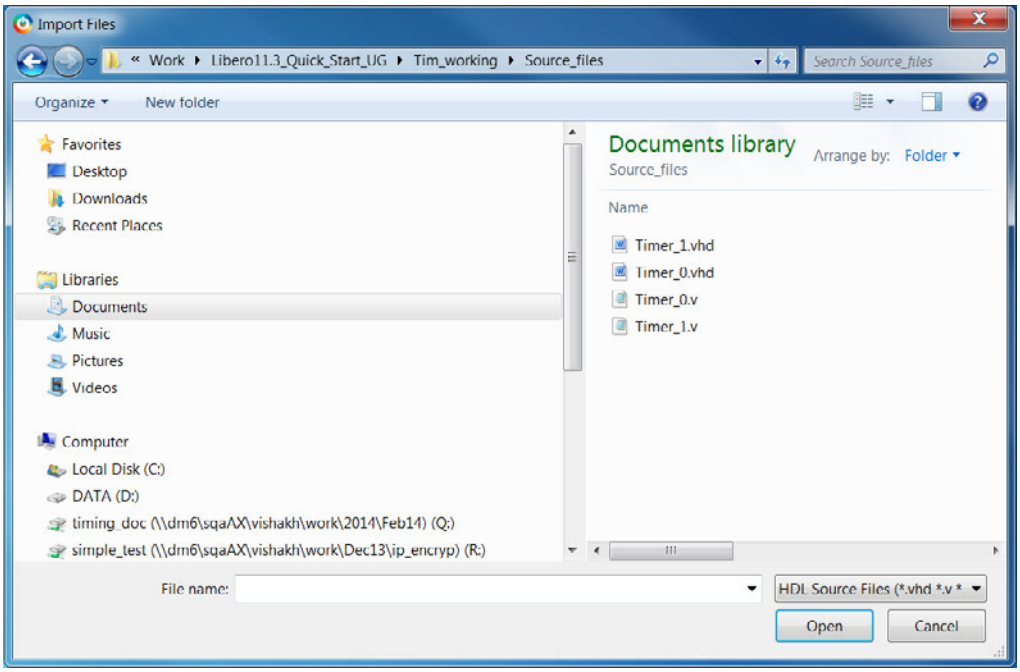


Figure 3-2 **• Timer_0 and Timer_1 HDL Source Files**

3. From the Design Hierarchy, drag and drop Timer_0 and Timer_1 HDL blocks into the SmartDesign Canvas.
4. From the Catalog, expand the Macro Library Group, and drag and drop a two-input AND gate into the SmartDesign Canvas.
5. Drag and drop SYSRESET (under the Macro Library group) into the SmartDesign Canvas.
6. From the Catalog, expand the Clock and Management group. Drag and drop the Chip Oscillator (OSC) into the SmartDesign Canvas. Double-click the OSC to open the Configurator..
7. Check the **On-chip 25/50 MHz RC Oscillator** box and select **Drives Fabric CCC(s)** (Figure 3-3).

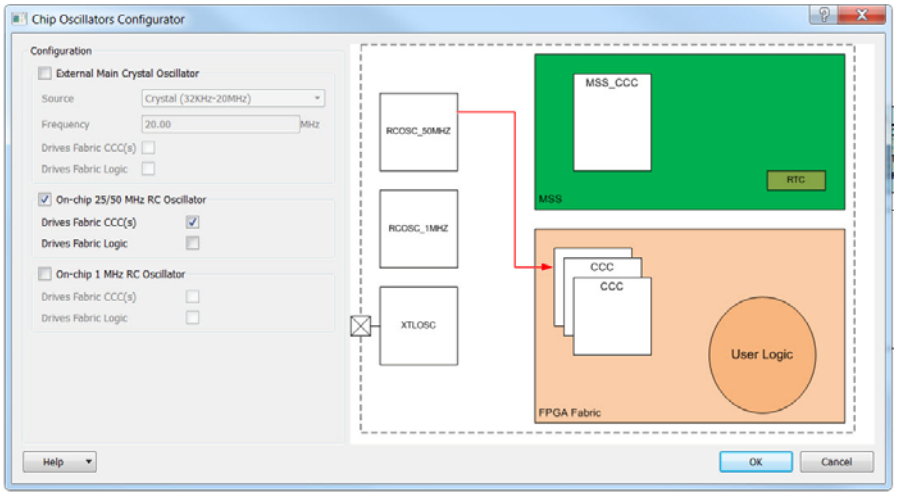


Figure 3-3 **• Oscillator (OSC) Configurator**

8. Click **OK** to close the OSC Configurator.
9. Drag and drop the Clock Conditioning Circuitry (FCCC) into the SmartDesign Canvas.
10. Double-click the FAB CCC to open the Configurator.

11. Click the **Basic** tab. For the Reference Clock, select **Oscillators** and then **25/50 MHz Oscillator** (**Oscillators** > **25/50 MHz Oscillator**). Click **OK** to close the FAB CCC Configurator (Figure 3-4).

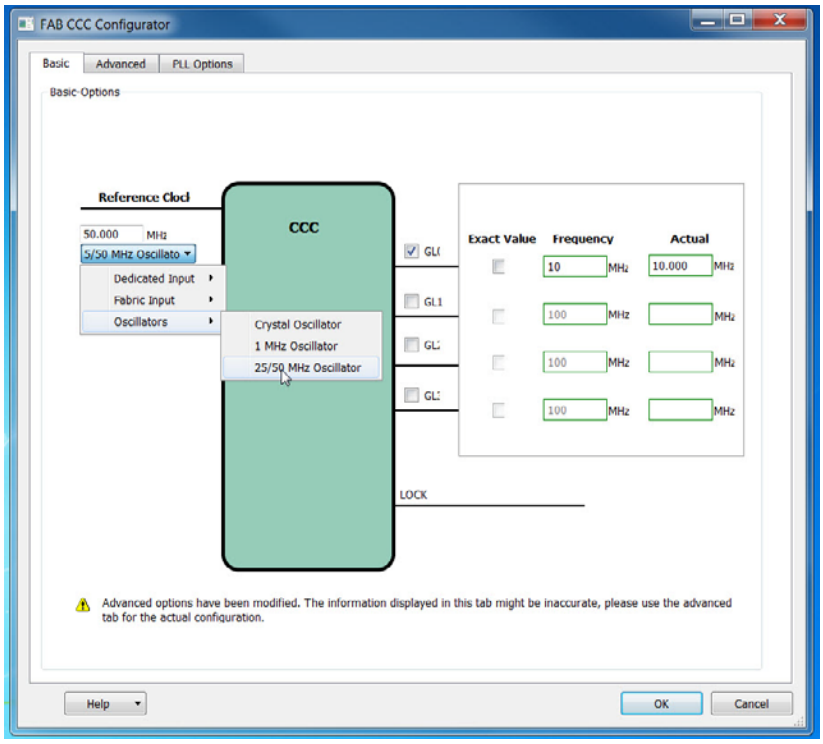


Figure 3-4 **• FCCC Configurator**

12. On the SmartDesign Canvas, hold the SHIFT key and click **RCOSC_25_50MHZ_CCC_OUT** of the Oscillator (OSC) and **RCOSC_25_50MHZ_CCC_IN** of the Clock Conditioning Circuit (FCCC) to select both ports.
13. Right-click and choose **Connect** from the drop-down menu to connect the two selected ports (Figure 3-5).

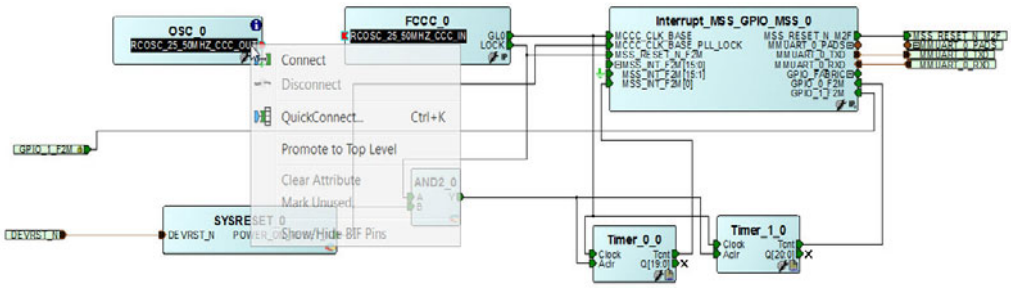


Figure 3-5 **• Making Connections in the SmartDesign Canvas**

14. Repeat the procedure to make the following connections in the SmartDesign Canvas:
    - From GLO output of Clock Conditioning Circuit to Clock input of bothTimer0 and Timer1 and MCCC_CLK_BASE of the Interrupt_MSS_GPIO_MSS_0 Block.
    - From LOCK output of the CCC to the A input of the AND gate and MSS_RESET_N_F2M Input of the Interrupt_MSS_GPIO_MSS_0 block.

- From POWER_ON_RESET_N output of SYSRESET_0 block to the B input of the AND gate and MCCC_CLK_BASE_PLL_LOCK input of Interrupt_MSS_GPIO_MSS_0 Block.
- From the Y output of the AND gate to the Aclr input of both Timer_0 and Timer_1.
- From Tcnt of Timer1 to GPIO_0_F2M of the Interrupt_MSS_GPIO_MSS_0 block.
- From Tcnt Output of Timer_0 to MSS_INT_F2M[0] of the Interrupt_MSS_GPIO_MSS_0 block.

Note: You need to slice the bus MSS_INT_F2M [15:0] first before you make the connection.

To slice the bus signal, right-click the MSS_INT_F2M[15:0] bus and choose **Edit Slice**. The Edit Slice dialog box appears.

Click the green + sign to add a slice. Enter 0 for the Left and Right columns for the first slice and enter 15 for the Left and 1 for the Right column for the second slice (as shown in Figure 3-6). Click **OK**.

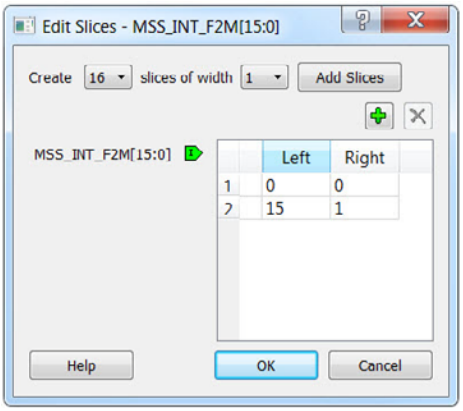


Figure 3-6 **• Edit Slice Dialog Box**

- Tie MSS_INT_F2M[15:1] to low. (Right-click > **Tie Low**).

15. Right-click and choose **Promote to Top Level**:
    - GPIO_1_F2M output of the Interrupt_MSS_GPIO_MSS_0 block
    - MSS_RESET_N_M2F output of the Interrupt_MSS_GPIO_MSS_0 block
16. Right-click and choose **Mark Unused**:
    - Q[19:0] of Timer_0
    - Q[20:0] of Timer_1

After making all the connections, the **Interrupt_MSS_GPIO** appears as shown in Figure 3-7. Pad ports and nets with dedicated pin assignments are displayed in brown.

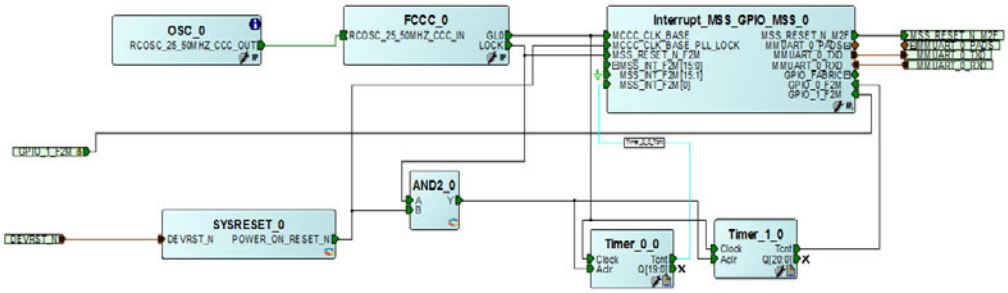


Figure 3-7 **• SmartDesign Canvas After Making Connections**

17. From the Design Flow window, right-click Configure Firmware Cores and choose **Open Interactively**.
18. In the DESIGN_FIRMWARE tab, the following required firmware drivers should have been selected for you and the checkbox under Generate checked. If the driver is greyed out and in italics, it is not in your vault. Download it into your vault. See Figure 3-8.
    - SmartFusion2_CMSIS_0 Version 2.2.101
    - SmartFusion2_MSS_GPIO_Driver_0 Version 2.0.101
    - SmartFusion2 MSS HPDMA Driver 0 Version 2.0.101
    - SmartFusion2_MSS_MMUART_Driver_1 Version 2.0.101
    - SmartFusion2_MSS_NVM_Driver_0 Version 2.1.102
    - SmartFusion2_MSS_System_Services_Driver_0 Version 2.0.103
    - SmartFusion2_MSS_Timer_Driver_0 Version 2.0.101

Note: If you select the wrong core version, the Application may not build successfully.

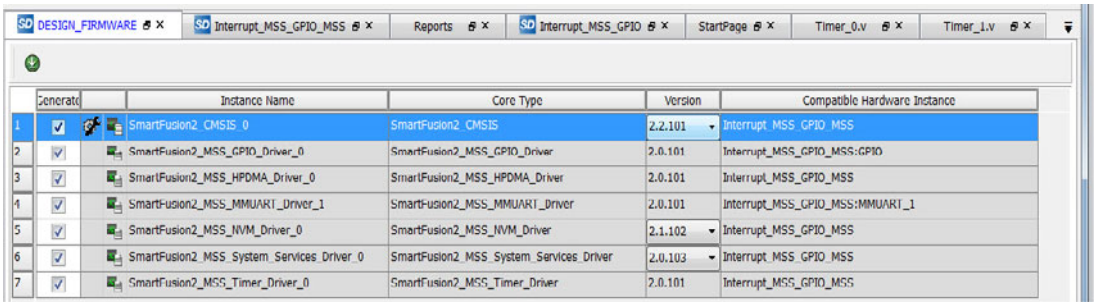


Figure 3-8 **• Configure Firmware**

19. From the **SmartDesign** menu, choose **Generate Component**.

    If you do not see **Generate Component** in the SmartDesign menu, make sure that you are viewing the Interrupt_MSS_GPIO tab (Figure 3-9).

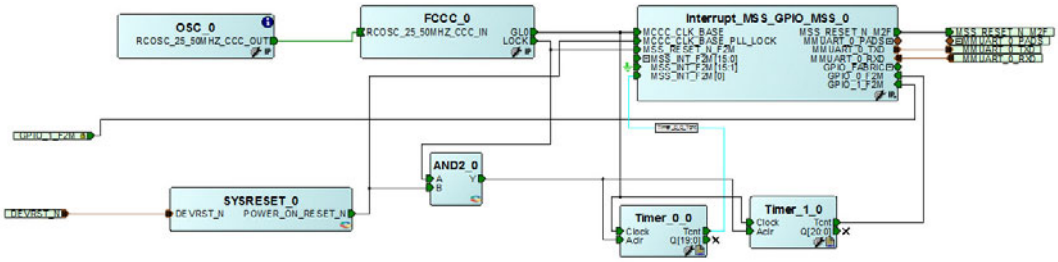


Figure 3-9 **• Generate Component**

A green check mark under Create Design in the Libero SoC Design Flow window indicates the design was created without any errors (Figure 3-10).

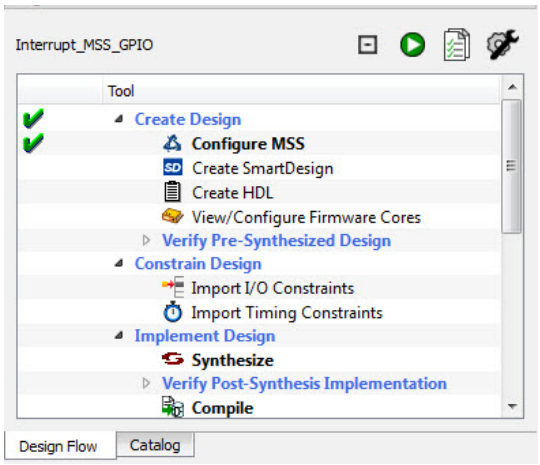


Figure 3-10 **• Successful Generation of SmartDesign Component**

After successful MSS Component generation, the Log window displays the message **Info: 'Interrupt_MSS_GPIO' was successfully generated** (Figure 3-11).

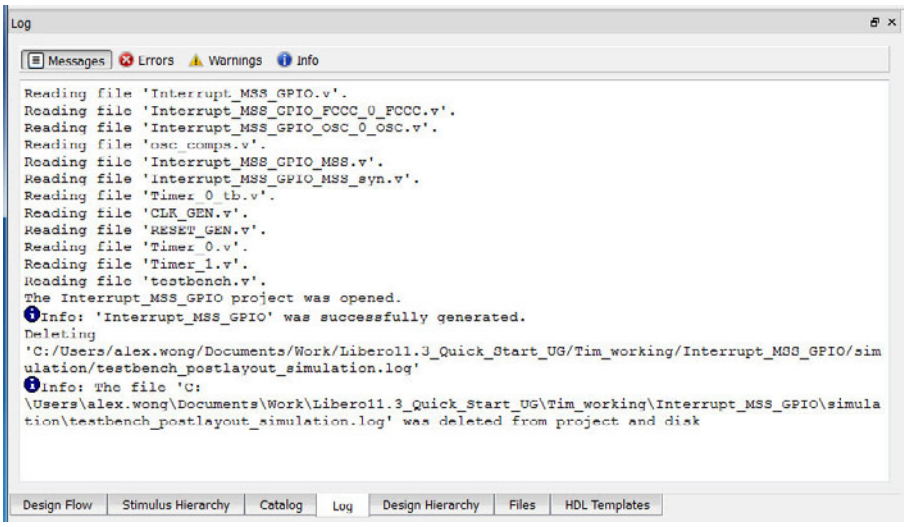


Figure 3-11 **• Log Window**

20. From the **Design** menu choose **Reports > Interrupt_MSS_GPIO_DataSheet.xml**. Examine the DataSheet.xml file to familiarize yourself with the Generated Files, Firmware and Memory Map

sections (click the hyperlink at the top of the datasheet to move to the section of interest). See Figure 3-12.

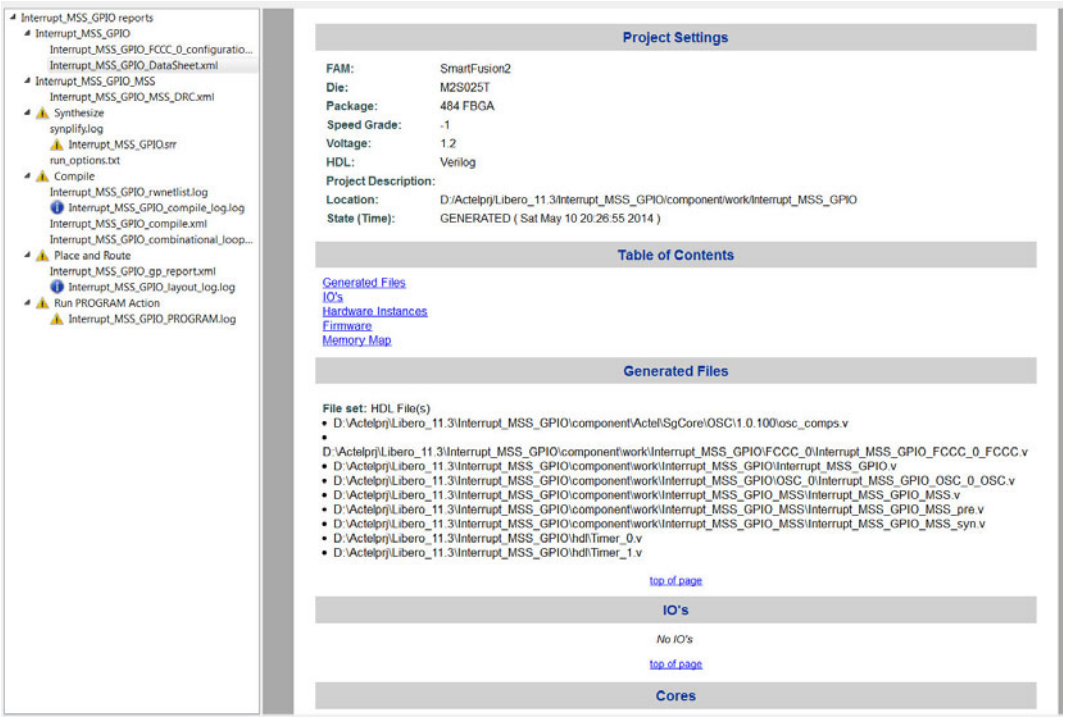


Figure 3-12 **• Datasheet File**

4 – Step 4 - Performing Pre-Synthesis Simulation

Libero can create an HDL testbench for you or you can create a testbench using SmartDesign. For this tutorial, you use an HDL testbench from Libero.

1. In the Design Hierarchy window, right-click **Interrupt_MSS_GPIO** and choose **Create Testbench > HDL**. Enter **Interrupt_MSS_GPIO_tb** for the testbench name (Figure 4-1).

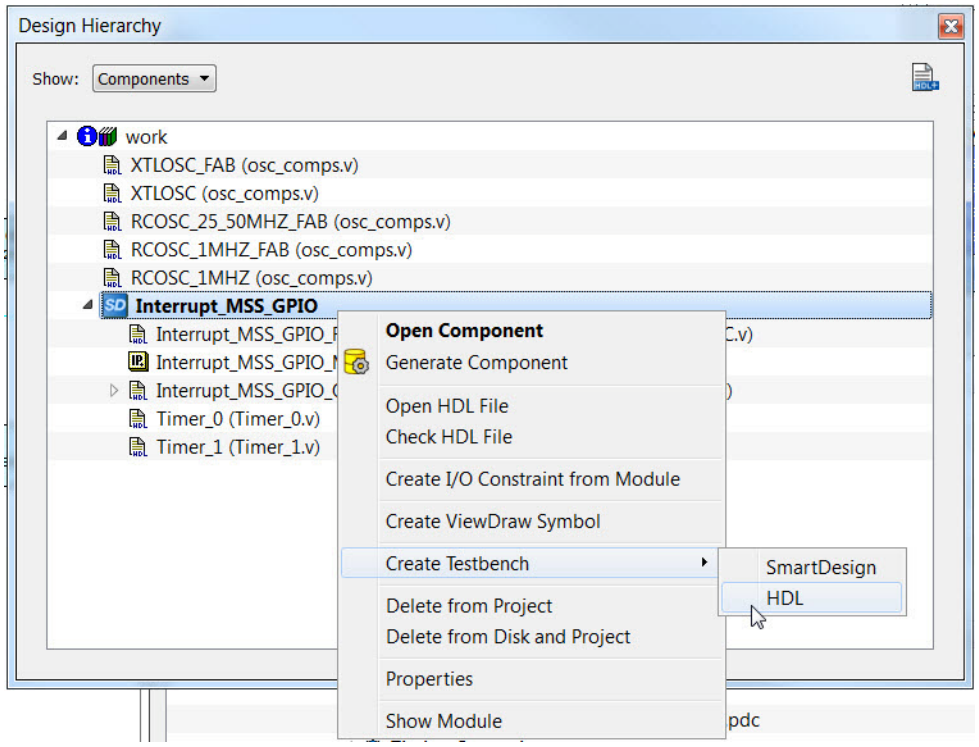


Figure 4-1 **• Testbench Creation**

2. From the **Project** menu, choose **Project Settings** and choose **DO File** under **Simulation Options**. Under **Simulation Runtime**, enter a value of **1ms** (Figure 4-2).

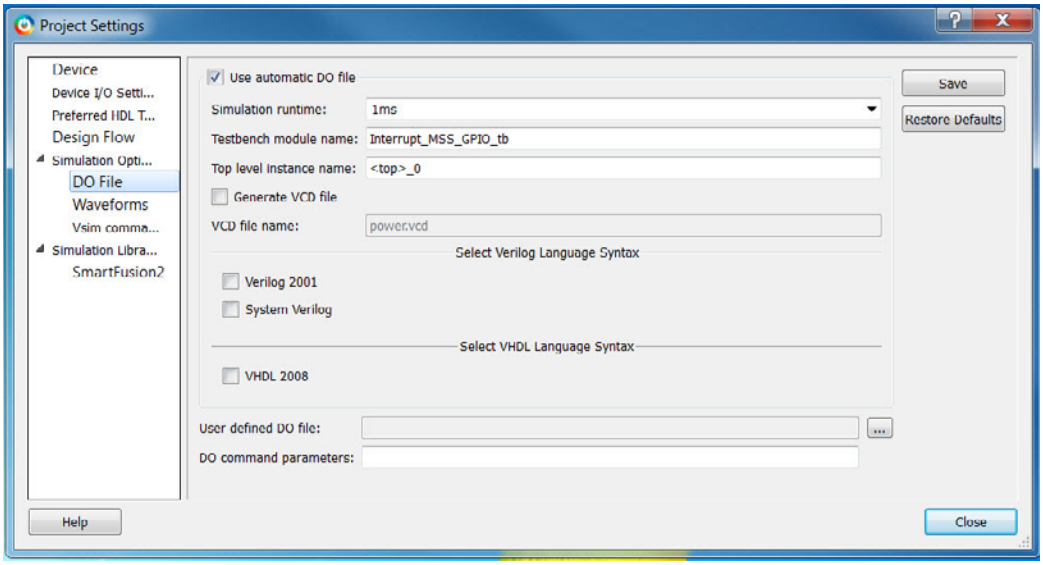


Figure 4-2 **• Project Settings Dialog Box**

3. Select **Vsim commands** in the Project Settings dialog box to set Vsim command options.
4. Change the default simulation resolution from 1fs to 1ps in the Resolution field. Ignore the warning message. This will speed up the simulation. See Figure 4-3.

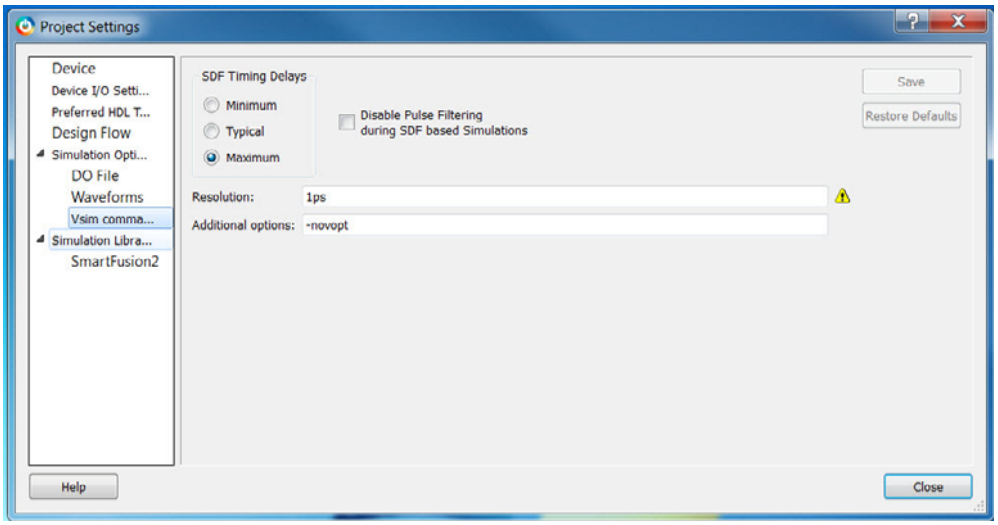


Figure 4-3 **• Vsim command Settings**

5. Under **Simulation Options**, click **Waveforms** and click the checkbox to enable the option **Log all signals in the design** (Figure 4-4).

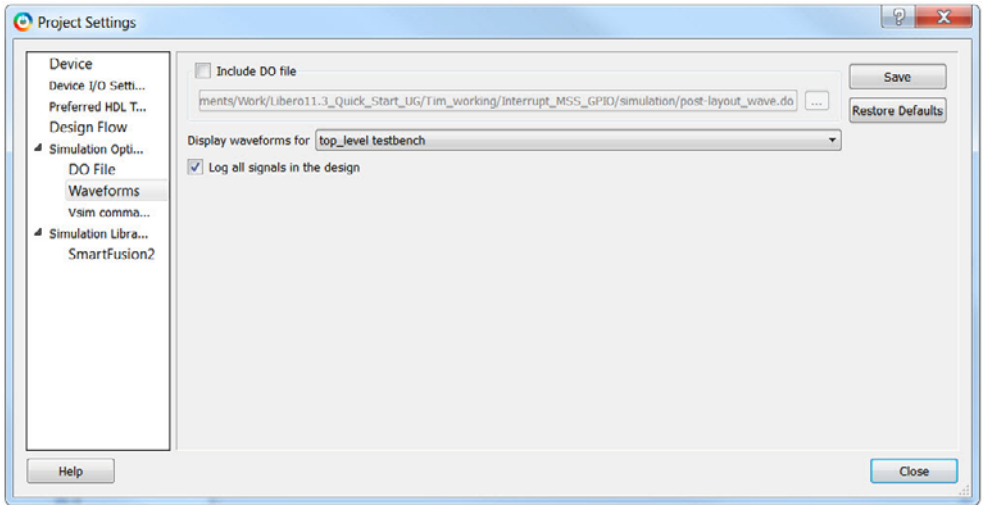


Figure 4-4 **• Changing Waveform Options**

6. Click **Save** to save the changes to the project settings. Click **Close** to close the Project Settings dialog box.
7. In the Design Flow window, expand Verify Pre-Synthesized Design and right-click **Simulate** and choose **Open Interactively** to launch ModelSim in GUI mode (Figure 4-5).

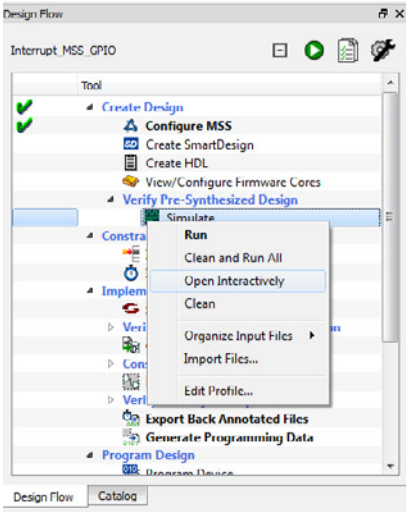


Figure 4-5 **• Invoking ModelSim**

ModelSim opens and automatically imports a **run.do** macro file that contains the links to the design files and gives simulation commands. The simulator compiles the source files and loads the design. The Wave window appears, as shown in Figure 4-6.

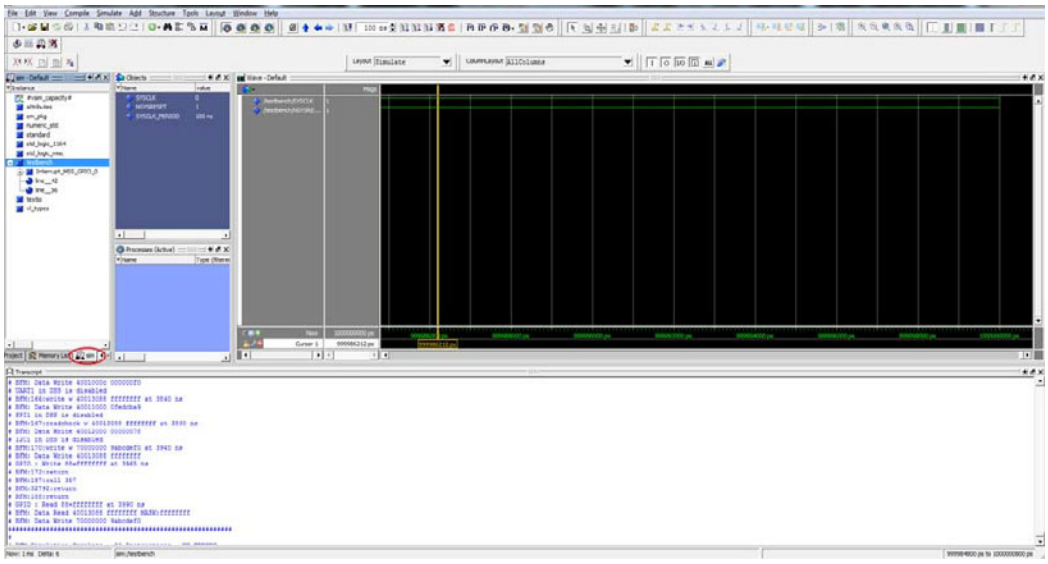


Figure 4-6 • **ModelSim Wave Window**

You must add additional signals to the Wave window to confirm the design is functioning properly.

8. Click the ModelSim **sim** tab (circled in Figure 4-6). Expand the Design Hierarchy and select **Timer_1_0** (Figure 4-7).

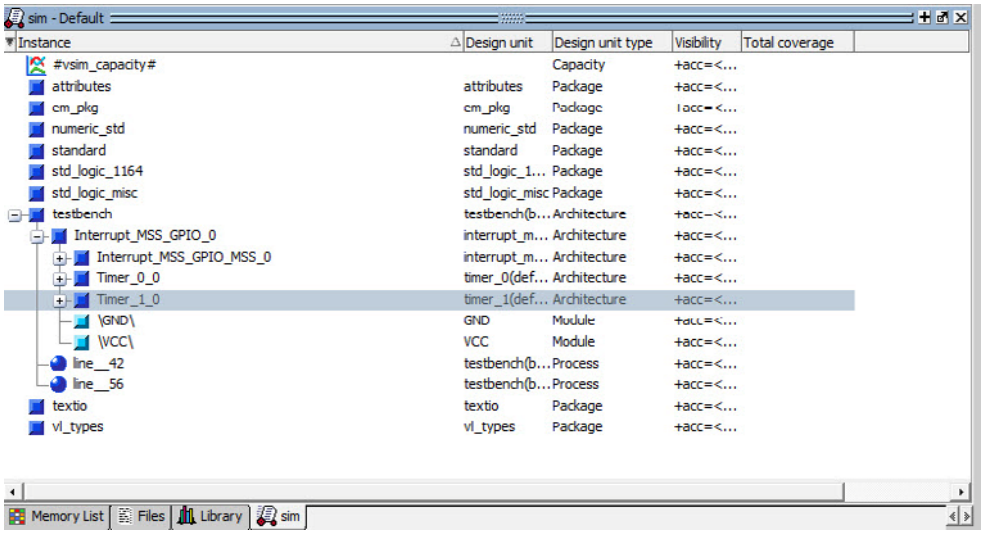


Figure 4-7 • **Timer_1_0 Selected in the ModelSim sim Tab**

9. Click the **Objects** tab. Ctrl + click to select **Aclr**, **Clock**, **Q** and **Tcnt**.
10. Add the signals to the Wave window: from the **ModelSim** menu, choose **Add > Wave > Selected Signals** (Figure 4-8).

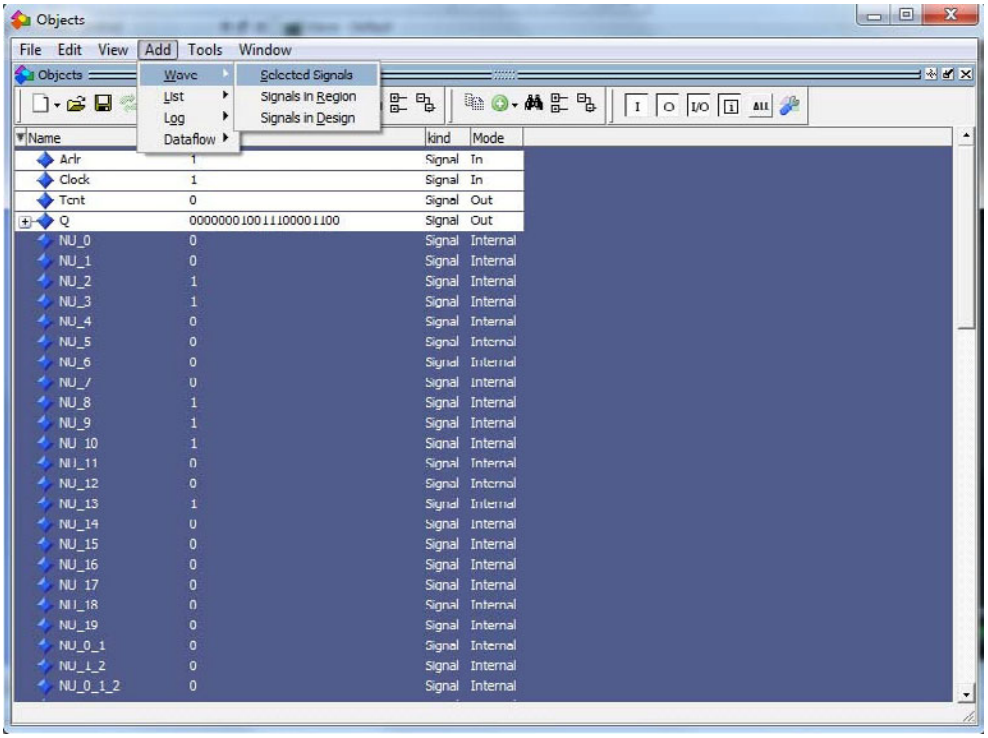


Figure 4-8 • **Adding Signals to the Wave Window**

11. Click the ModelSim **sim** tab (circled in Figure 4-6). Expand the Design Hierarchy and select **Timer_0_0**.
12. Click the **Objects** tab. Ctrl + click to select **Aclr**, **Clock**, **Q** and **Tcnt**.
13. Add the signals to the Wave window: from the **ModelSim** menu, choose **Add > Wave > Selected Signals**.

    After the signals are added the Wave window should look like Figure 4-9.

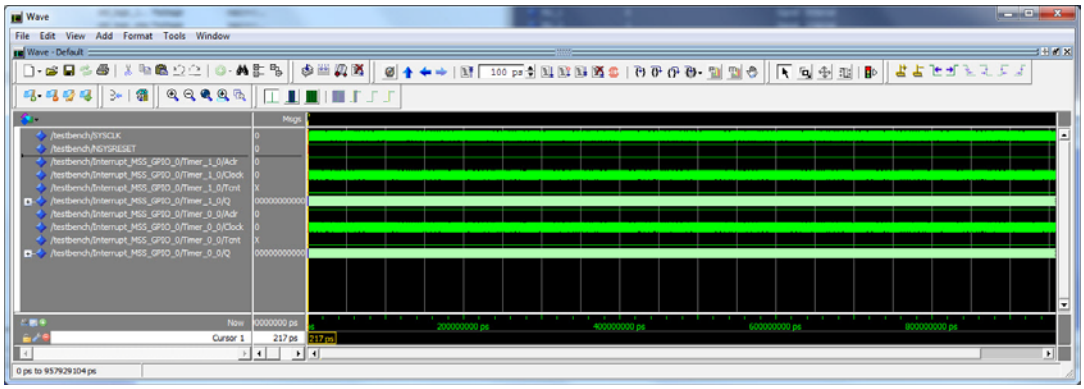


Figure 4-9 • **ModelSim Wave Window After Adding Timer_1_0 and Timer_0_0 Signals**

14. Observe the operation of the counters to confirm the design is working. Use the zoom buttons to zoom in and out as necessary (Figure 4-8).

    Undock the Wave window to make it easier to observe the signals.

    Change the radix of the Timer Q output to hex to make it easier to view the values (Figure 4-10).

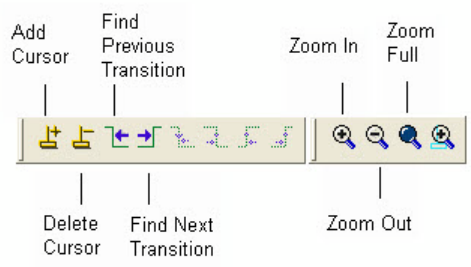


Figure 4-10 **• Wave Window Zoom Controls**

ModelSim displays waveforms of the Timers, as shown in Figure 4-11.

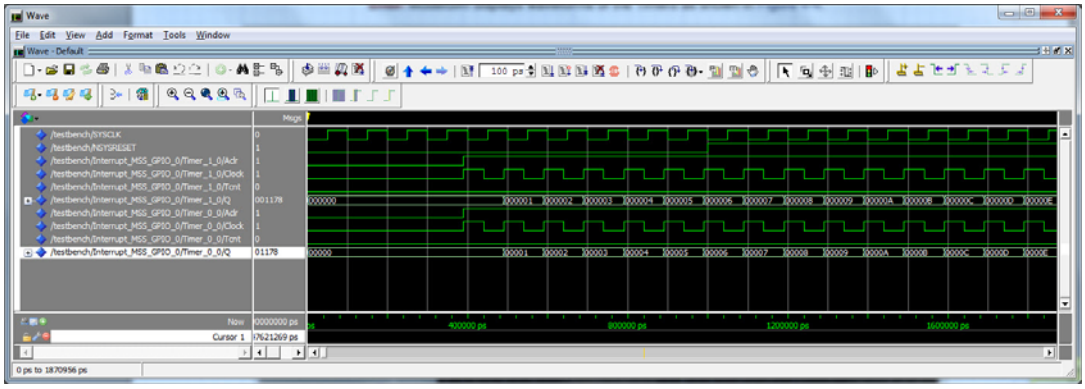


Figure 4-11 **• ModelSim Wave Window**

15. You can quit the Simulator after you confirm the Timers are counting. From the ModelSim toolbar **File** menu, choose **Quit** to exit the simulator. Click **Yes** when asked if you want to quit.

Synthesizing the Design using Synplify® Pro

Synplify Pro compiles and synthesizes the design into an EDIF (*.edn) file. Your EDIF Netlist is then automatically translated by Libero SoC into an HDL Netlist. The resulting EDIF (*.edn), Verilog (*.v) or VHDL (*.vhd) files are visible in the **Files** tab under **Synthesis**.

1. In the Design Flow window, right-click **Synthesis** and choose **Open Interactively** to launch Synplify Pro (Figure 4-12).

.

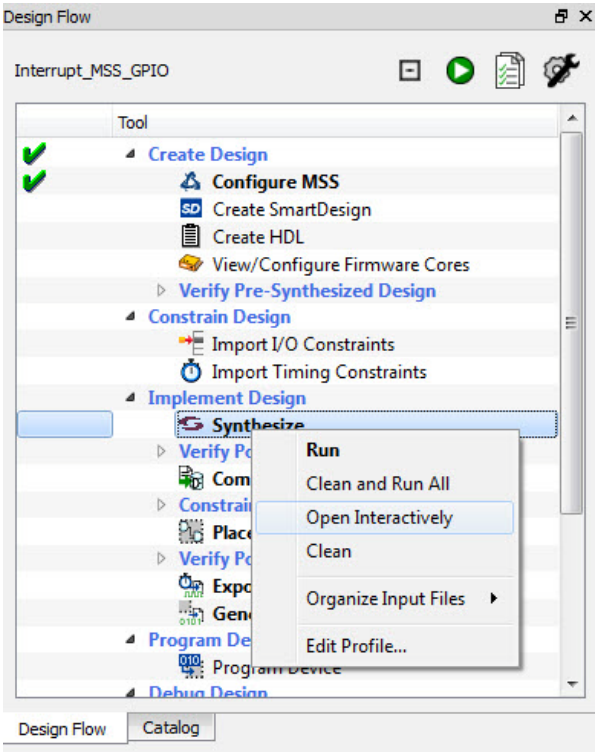


Figure 4-12 **• Invoking the Synthesis Tool**

You can change the frequency in the Synplify Pro GUI to meet your design requirements. In this design, we are not changing the frequency (Figure 4-13).

.

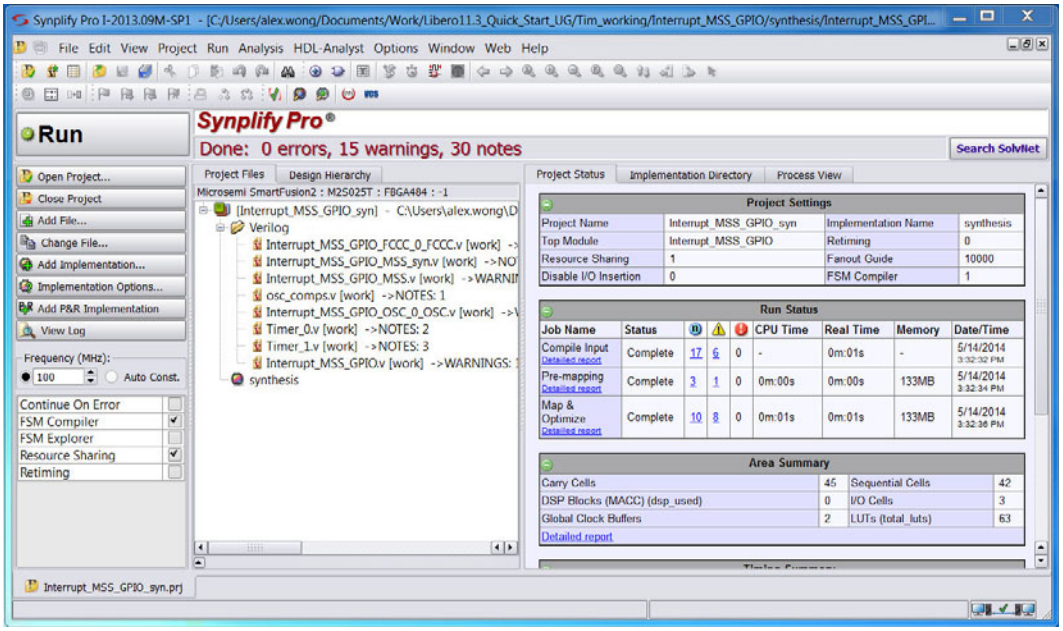


Figure 4-13 **• Synplify Pro GUI**

2. Click **Run** to map the design (Figure 4-13). When **Ready** in Synplify Pro changes to **Done:0 errors**, the design has been mapped successfully.
3. From the **File** menu, choose **Exit** to close Synplify Pro. Click **Yes** if prompted to save changes to the project.

In the Libero SoC Design Flow window, a green check mark adjacent to Synthesis indicates that the design has been synthesized without any errors (Figure 4-14).

.

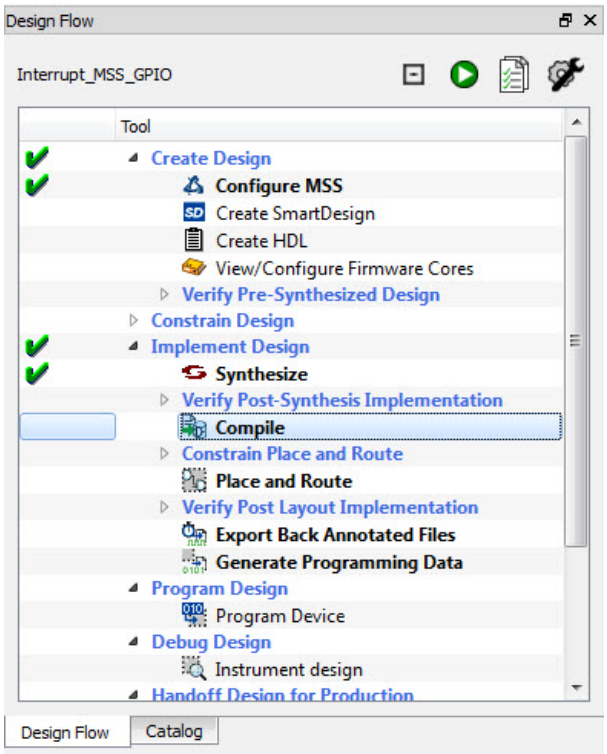


Figure 4-14 **• Successful Synthesis**

5 – Step 5 - Implementing the Design

The next step is to implement the design. You must assign I/O and timing constraints, run Compile, then Place and Route. Once place and route is complete, you can perform the static timing and power analysis.

In this design, all ports are hard-wired except for GPIO_1_F2M, which must be assigned to an I/O pin.

1. Expand Place and Route. Right-click **I/O Constraints** and choose **Open Interactively** to open the I/O Constraints Editor (Figure 5-1).

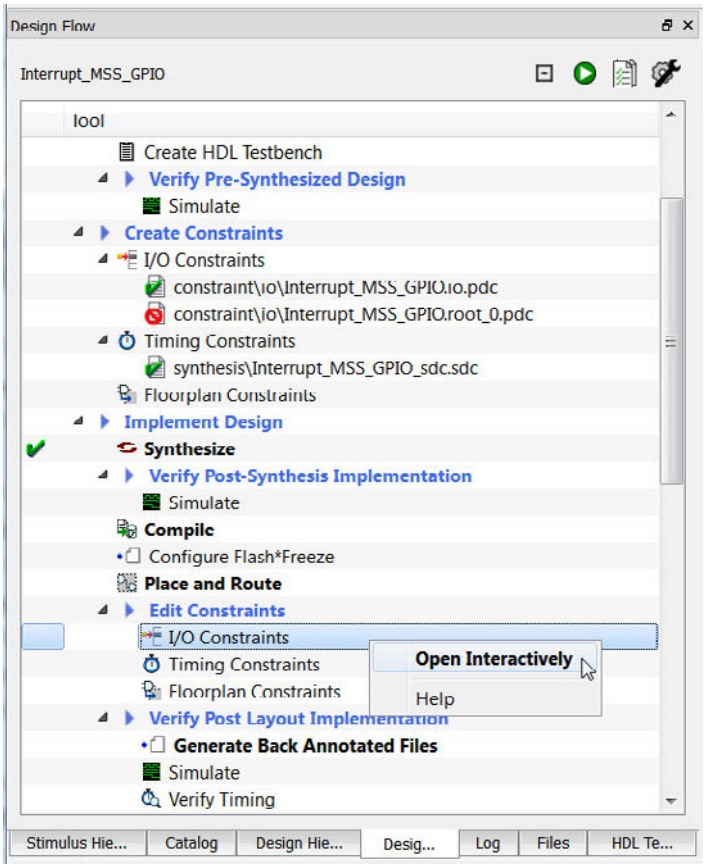


***Figure 5-1* • I/O Constraints Editor Invocation**

Libero SoC runs Compile. After successful completion, a green check mark appears next to Compile in the Design Flow window. The I/O Editor then appears for you to assign the I/O pins (Figure 5-2).

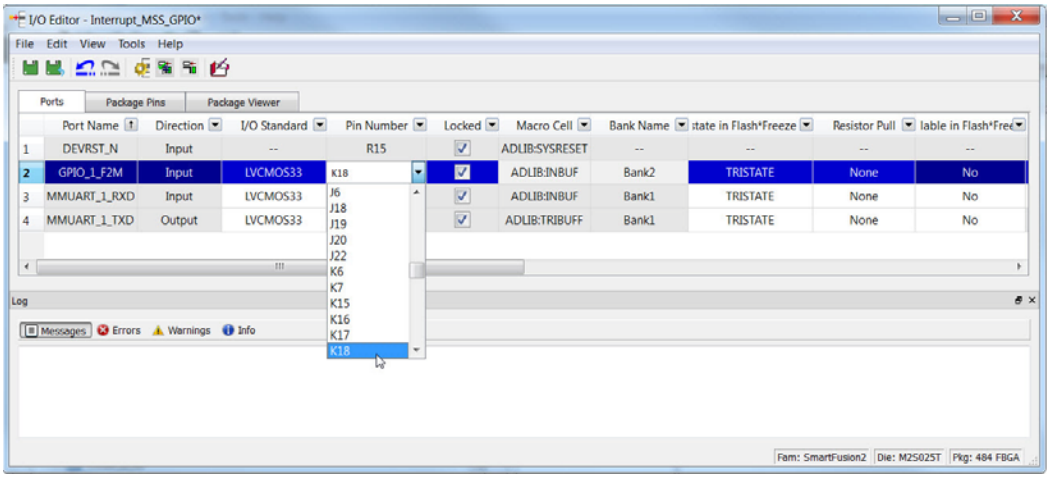


Figure 5-2 **• Pin Assignment of Port GPIO_1_F2M**

2. In the I/O Editor, assign Pin K18 to the Port GPIO_1_F2M and check the Locked checkbox. Choose **Commit and Check** to save the pin assignment. Correct any errors that are reported in the Log window**.**
3. From the **File** menu, choose **Exit** to close the I/O Pin Editor.
4. Right-click **Place and Route Layout** in the Design Flow window and choose **Configure Options** to set the Place and Route Options (Figure 5-3).

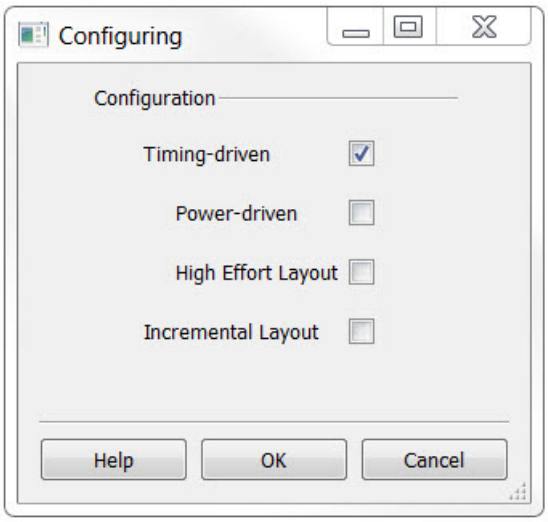


Figure 5-3 **• Layout Options**

5. Click the checkbox to enable Timing-Driven place and route and leave all other options unchecked. Click **OK** to close the Configuring Options dialog box.
6. Double-click **Place and Route**.

Libero generates the *io.pdc file (Pin Assignment file) and passes the file to Place and Route. A green check mark appears next to Place and Route to indicate successful completion (Figure 5-4).

.

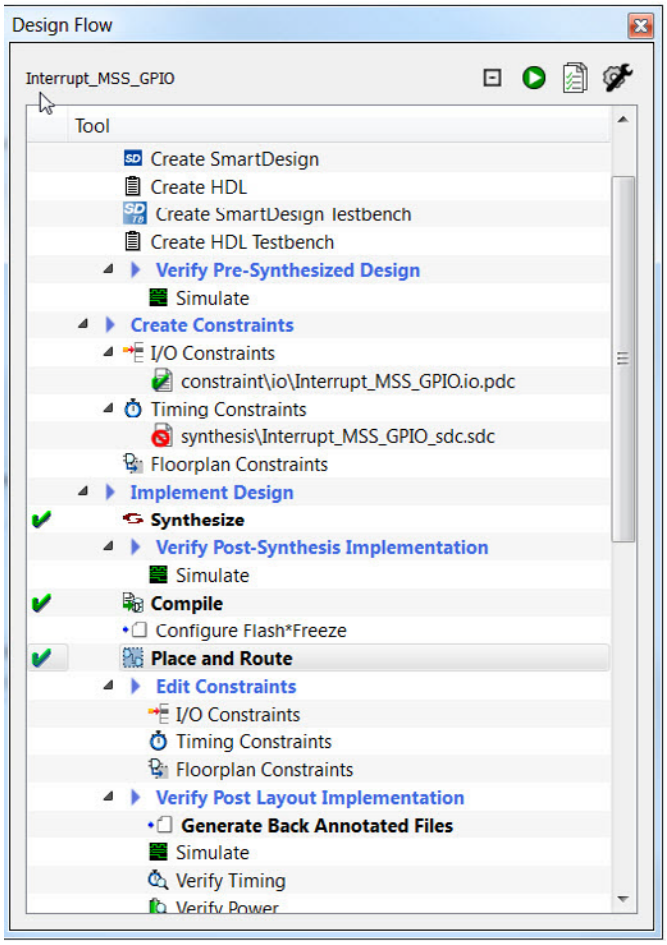


Figure 5-4 **• Successful Compile and Place and Route**

Timing Analysis

The next step is to perform static timing analysis using SmartTime. SmartTime reads your design and displays post-layout timing information (pre-layout, if invoked before place and route). For more information, see the Help included with the software.

SmartTime includes a Constraints Editor and a Timing Analyzer.

1. In the Design Flow window, right-click **Verify Timing** and choose **Open Interactively** to open SmartTime. Confirm that the fabric clock (mss_ccc_glb) meets the timing requirement (10 MHz). (Figure 5-5).

.

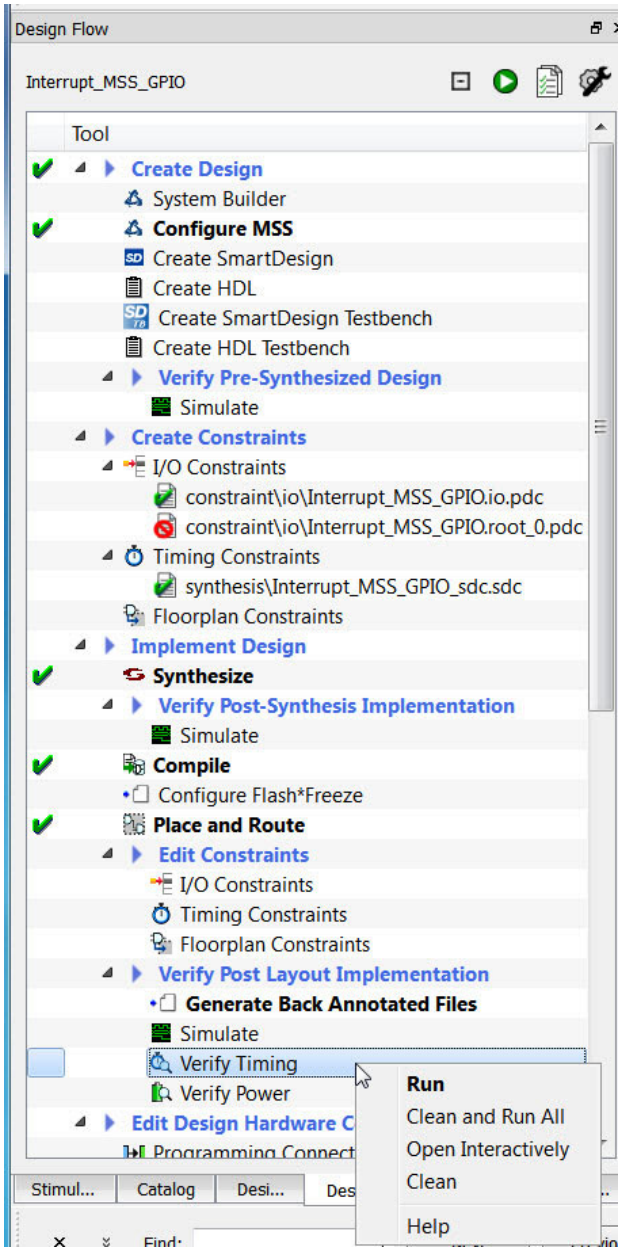


Figure 5-5 **• Verify Timing**

2. The Maximum Delay Analysis View window opens. Maximum Delay Analysis checks for any setup violations. A green check mark next to **Register to Register delay** indicates that there are no setup violations. Select the **Register to Register** path set for the GL0_net_inferred_clock domain (Figure 5-6).

.

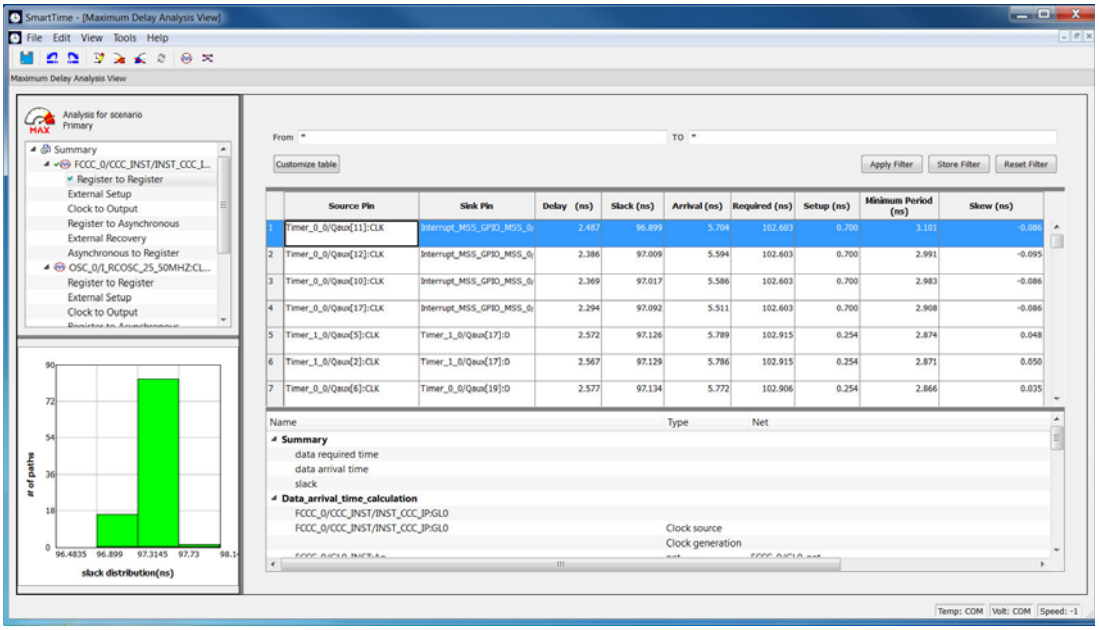


Figure 5-6 **• SmartTime Max Delay Timing Analysis**

3. Double-click one of the source pins to view detailed timing analysis for the selected path in the path details (as shown in Figure 5-7).

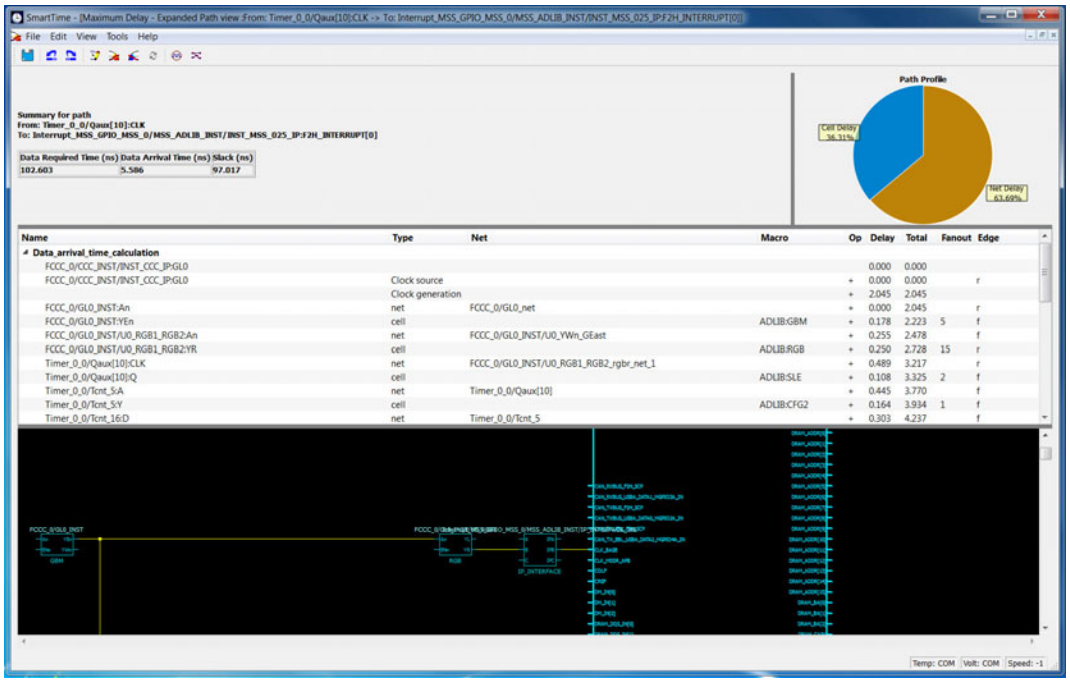


Figure 5-7 **• Detailed View of the Register Path in SmartTime**

4. From the **SmartTime Tools** menu, choose **Minimum Delay Analysis** to open the Minimum Delay Analysis View window. Minimum Delay Analysis checks for hold time violations. A green check mark next to **Register to Register delay** indicates no hold time violations.
5. From the **File** menu, choose **Exit** to close SmartTime.

Power Analysis

SmartPower enables you to estimate the power consumption in your design. This enables you to make adjustments to reduce power consumption.

1. From the Design Flow window, right-click **Verify Power** and choose **Open Interactively** to start power analysis (Figure 5-8).

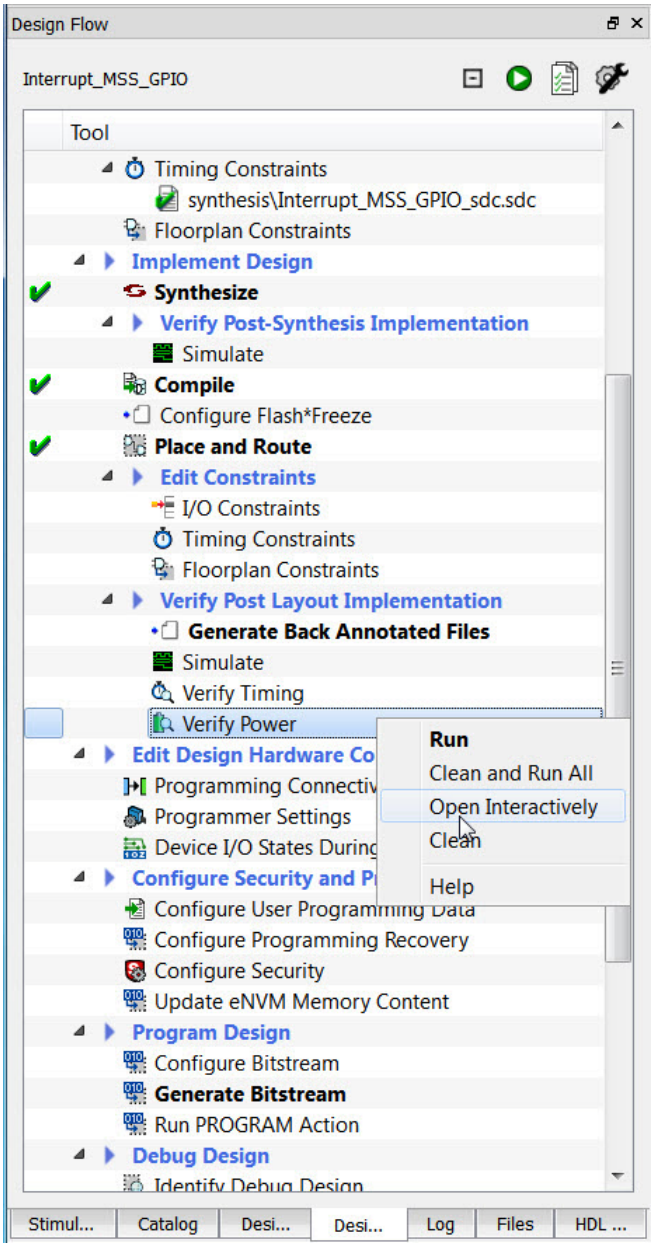


Figure 5-8 **• Power Analysis**

Figure 5-9 shows the SmartPower Analysis window.

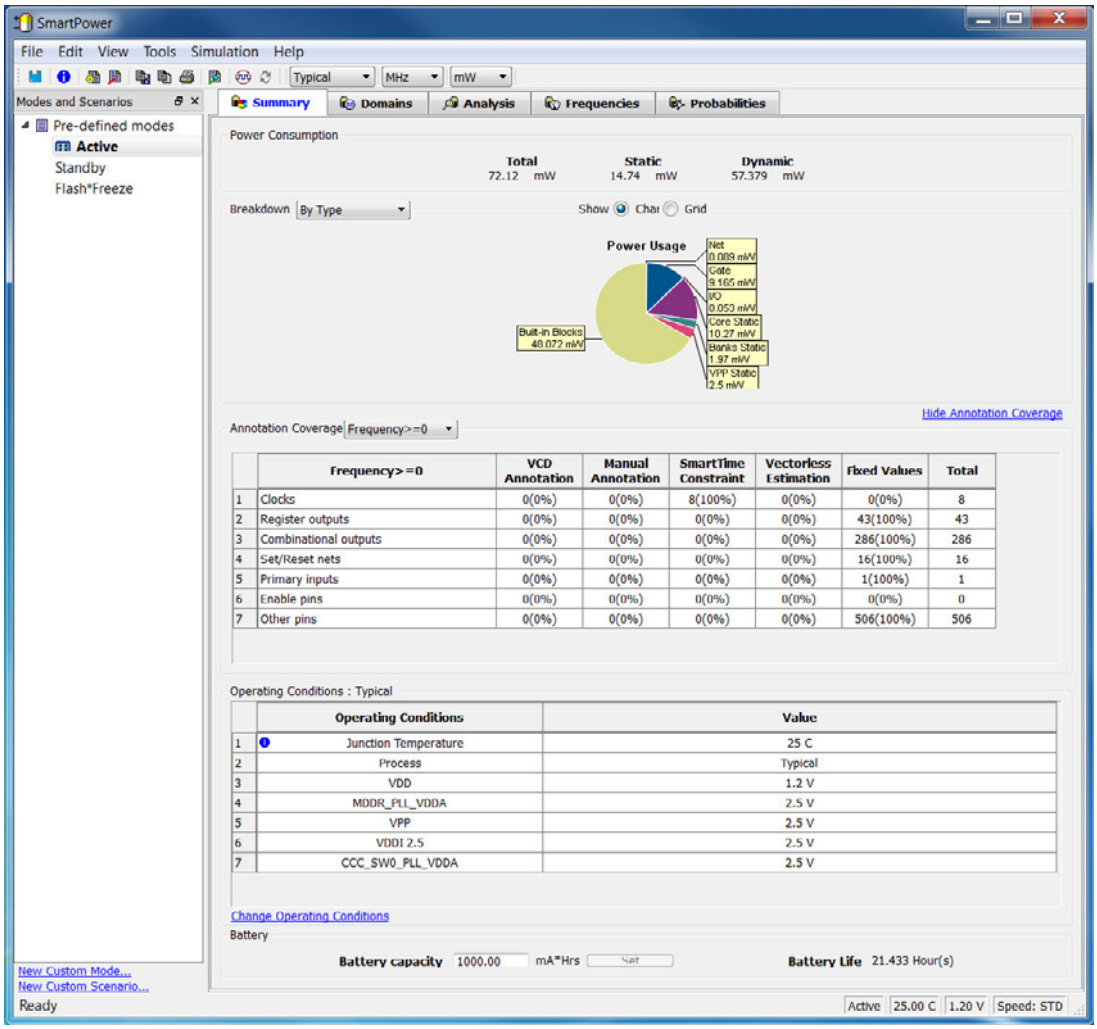


***Figure 5-9* • SmartPower Analysis Window**

SmartPower includes the following tabs (as shown in Figure 5-9):

- **Summary** - Displays the total power consumption, temperature and voltage operating conditions, battery capacity in mA/hr, and the projected battery life.
- **Domains** - Displays a list of existing domains with their corresponding clock and data frequencies. Use the **Domains** tab to set different clock frequencies and observe the effect on power consumption.
- **Analysis** - Displays detailed hierarchical reports of the power consumption.
- **Frequencies** - Used to attach switching frequency attributes to the interconnects of the design.
- **Probabilities** - Used to control the probabilities for all pins.

SmartPower enables you to globally visualize power consumption and potential power consumption problems within your design. You can then make adjustments, when possible, to reduce power consumption to meet your design's power requirements.

For more information, see the Help included with the software.

2. Select **File > Exit** to close SmartPower.

Back-Annotation

Back-Annotation generates a *.sdf file that contains timing information for your design. It is used for post-layout and timing simulation.

1. From the Design Flow window, right-click **Generate Back Annotate Files** and choose **Run** to extract post-layout timing delay from your design for simulation. (Figure 5-10)

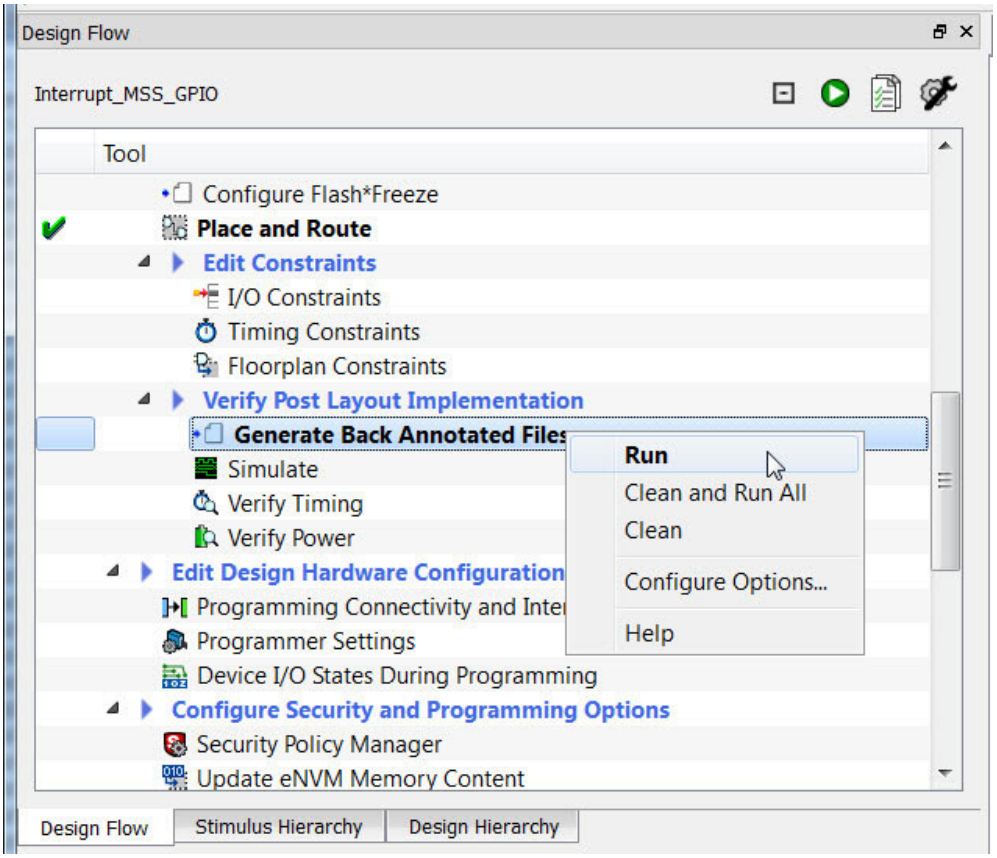


Figure 5-10 **• Generated Back Annotated Files**

Post-Layout Simulation

A green check mark appears next to the Generate Back Annotated Files in the Design Flow window when a back annotated file is generated successfully.

2. Click **Project > Project Settings > Simulation Options > DO File** to open the dialog box.
3. Check **Use automatic DO File**.
4. Enter **1ms** for Simulation runtime.
5. Click **Save** and then click **Close**.
6. Click **Project > Project Settings > Simulation Options > Vsim Commands** to open the dialog box.
7. In Simulation Resolution, enter **1ps**. This will speed up the simulation significantly. If you see warnings when you make the change, they can be ignored. The default value is 1 fs.
8. Click **Project > Project Settings > Simulation Options > Waveforms** and in the open window, uncheck **Log all signals in the design**. This will speed up the simulation run. After the simulation run, you may add to the waveform window the signals you want to examine.

   For Verilog projects, a wave.do file is available for you to include. Check **Include Do File**. Browse to the location of the post-layout_wave.do file you have extracted from the source files included in

this tutorial. Select the file and click **Open**. This file will be loaded into the simulator's waveform window and the signals will be displayed during simulation (Figure 5-11).

9. In the Libero Design Flow window, expand **Verify Post Layout Implementation**, right-click **Simulate** and choose **Open Interactively** to open ModelSim.

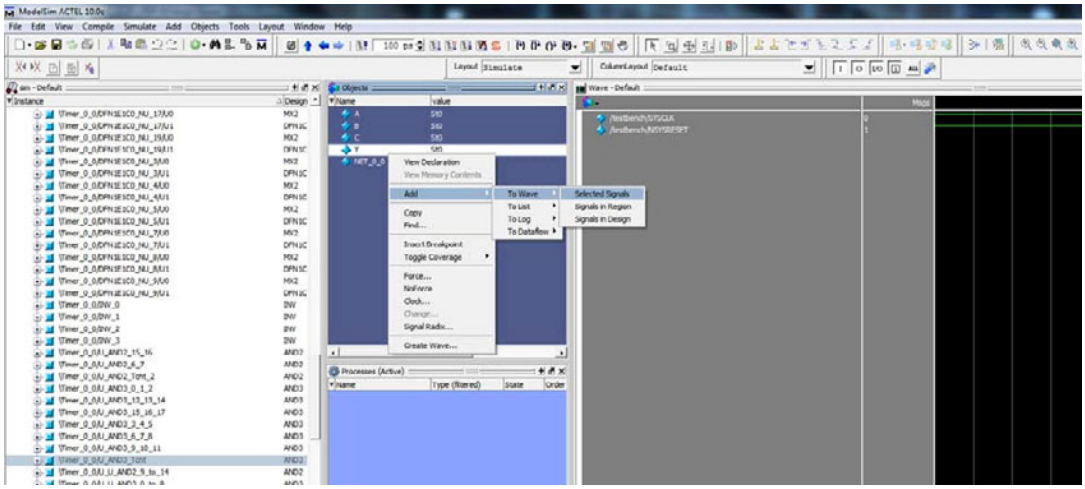


Figure 5-11 **• Adding Signals to the Wave Window in ModelSim**

10. In ModelSim, let the simulation run for 100 - 300 ms of run time. This may take a a few hours to complete. You may want to stop the simulation (**Simulate > Break**) after you have verified that the lower bits of the two counters are counting/toggling correctly in the waveform window. Use the zoom buttons to change the scale and view details in the waveforms (Figure 5-12).

    Note that the Tnct signal of Counter_0 asserts at about 100 ms and Tnct of Counter1_0 asserts at about 200 ms.

    Figure 5-12 shows the waveform window after 300 ms of simulation runtime.

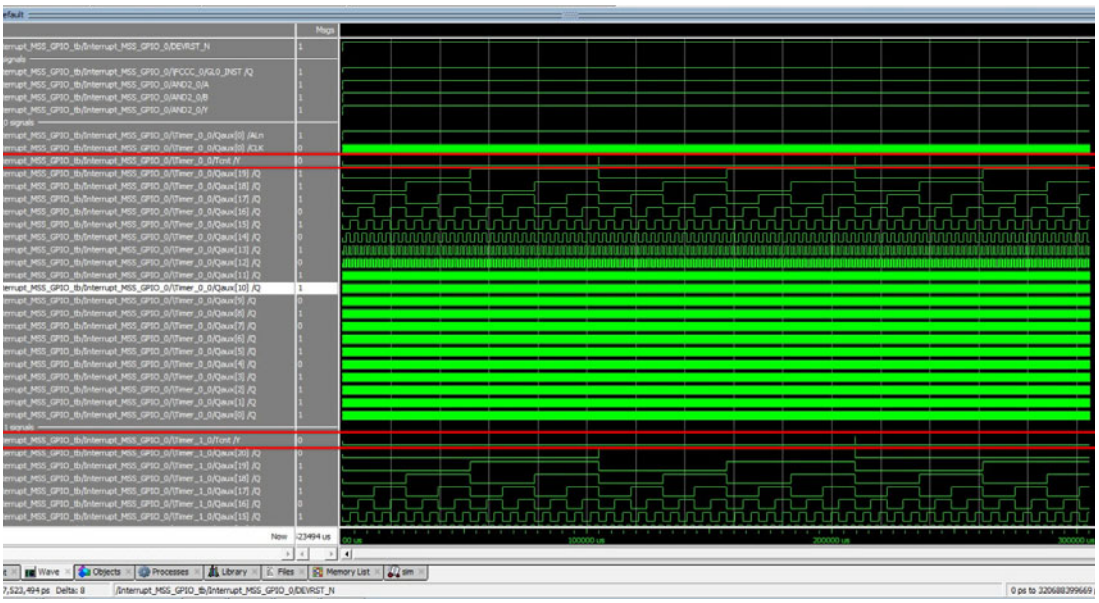


Figure 5-12 **• Post-Layout Simulation Window after 300 ms of simulation time**

Note: If ModelSim reports signals not found, you must edit the post-layout_wave.do file. Open the file in the Libero Text Editor. Replace the testbench name Interrupt_MSS_GPIO_tb in the post-layout_wave.do file with your testbench name. Replace the Component name Interrupt_MSS_GPIO_0 with your top level component name. Save the file and restart ModelSim.

You can stop the simulation from the ModelSim menu (**Simulate > Break**).

11. Close ModelSim.

Programming

In this step you will launch FlashPro and program the SmartFusion2 device on the Evaluation Board. Refer to the SmartFusion2 Evaluation Kit User Guide for instructions on how to set up the hardware for programming.

1. Connect one of the mini USB cables between the JTAG Programming Header and a USB port on your PC. Install the FlashPro4 drivers if prompted. The drivers are located in the <drive>:\<Microsemi \<Libero SoC v11.4 installation>\Designer\Drivers folder.
2. Connect the AC adaptor to the 12 V Power Supply Input on the board and plug the Adaptor to the AC wall outlet.
3. Turn the on/off switch on the board to the ON position.
4. In the Design Flow window, expand **Program Design**.
   – To program the device directly, double-click **Run PROGRAM Action**.

Do not interrupt the programming sequence; it may damage the device or the programmer.

The following message is visible in the Libero SoC log window when the device is programmed successfully: **Chain Programming PROGRAM PASSED.**

A green check mark next to **Run PROGRAM Action** appears in the Design Flow window to indicate that the programming completed successfully (Figure 5-13).

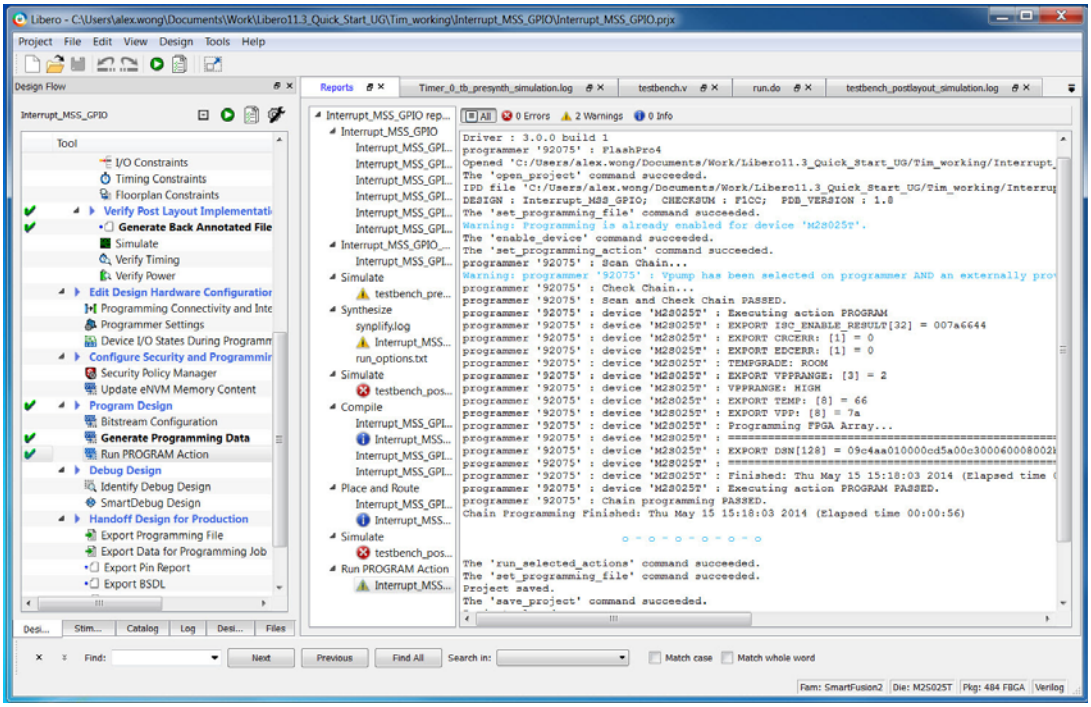


Figure 5-13 • **Design Flow Window After Programming**

Exporting Firmware

1. From the Libero SoC Design Flow window, right-click **Export Firmware** and choose **Export Firmware**.

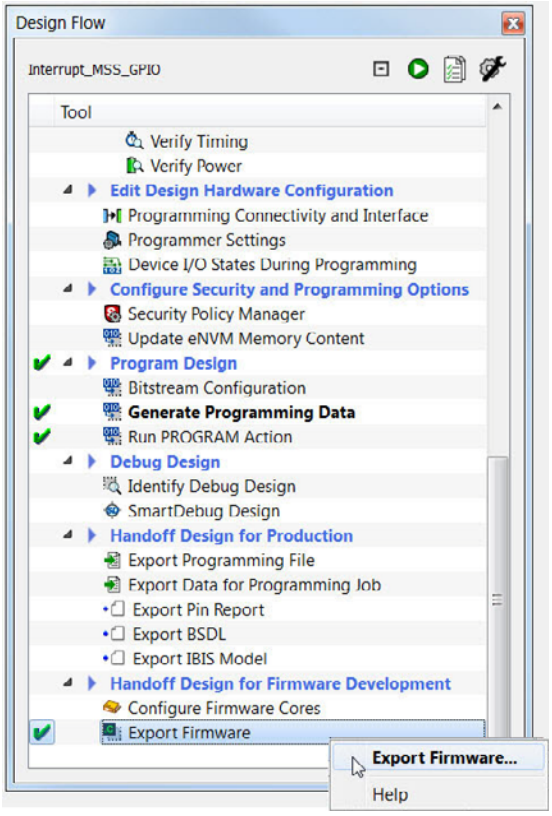


Figure 5-14 **• Export Firmware**

2. Accept the default location for the destination of the firmware exported.
3. Check **Create Project** and choose **SoftConsole** from the drop-down menu (Figure 5-15).

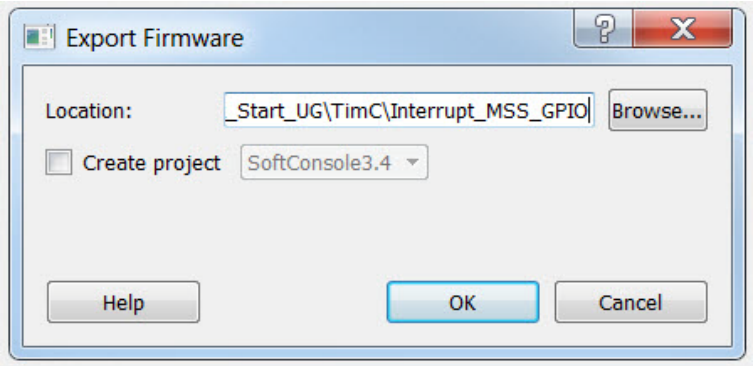


Figure 5-15 **• Export Firmware Dialog Box**

6 – Software Implementation

Step 1 – Invoking SoftConsole

Note: A software patch SP1 is available for SoftConsole 3.4. Before your proceed, install the SP1 patch first from

http://www.soc.microsemi.com/download/reg/download.aspx?p=f=SoftConsolev34_SP1.

1. Double-click the SoftConsole icon on your Desktop or choose **Start > All Programs > Microsemi SoftConsole v.3.4**.

   SoftConsole opens and prompts for a Workspace Location (Figure 6-1).

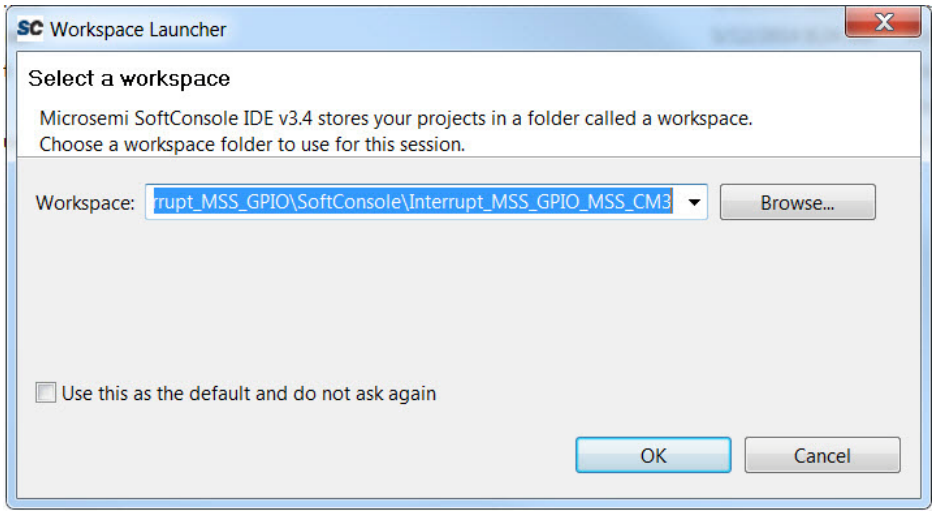


***Figure 6-1* • SoftConsole Workspace**

1. Make sure the Workspace location is set to:

   <Proj_location>\SoftConsole\Interrupt_MSS_GPIO_MSS_CM3.
2. Expand **Interrupt_MSS_GPIO_MSS_MSS_CM3_0_app** and double-click **main.c** to open it.
3. Delete the existing code in main.c and copy the code below and paste it in the main.c file.

   Note: The main.c file is included in the Source File you have extracted.

```
/*******************************************************************************
 * (c) Copyright 2010-2014 Microsemi SoC Products Group. All rights reserved.
 *
 *
 *
 * SmartFusion2 Quick Start User Guide main.c Source File.
 *
 *
 *
 * Revision: 1.0
 *
 * Date: 07/02/2014
 *
*******************************************************************************/
 *
/*******************************************************************************
 * Comment: main.c function in the Libero SoC Quick Start Guide for software v11.4.        *
```

```
 * This main.c file invokes three fabric to MSS Interrupts (two synchronous from timers and
 * one asynchronous from the switch of the SmartFusion2 Evaluation Kit).                    *
 * Interrupt messages are output from the MSS to the Terminal Emulator via MMUART of
 * the MSS.                                                                                  *
*******************************************************************************************/
/*Including Directories*/

#include<stdio.h>
#include "mss_uart.h"
#include "mss_gpio.h"
#include "cortex_nvic.h"

/*==============================================================================
  Messages displayed over the UART.
 ==============================================================================*/
const uint8_t g_greeting_msg[] ="\r\n\r\n\
*******************************************************************************\r\n\
**************** SmartFusion2 Libero SoC v11.4 Tutorial **************** *\r\n\
*******************************************************************************\r\n\
 This example project demonstrates the use of the SmartFusion2\r\n\
 Fabric and GPIO interrupts.\r\n";

/*Interrupt Handlers*/

/* GPIO 0 Interrupt Handler */
/* This Interrupt handler executes upon the occurrence of
   GPIO 0 interrupt, which is from the Timer 1 in the FPGA
   Fabric. This function prints source of the interrupt to the
   hyperterminal */
void GPIO0_IRQHandler( void )
  {


const uint8_t tim2[] = "\n\r Timer 1 Interrupt occurred - GPIO 0 \n\r";
MSS_UART_polled_tx (&g_mss_uart1, tim2, sizeof(tim2));
MSS_GPIO_clear_irq( MSS_GPIO_0 );
NVIC_ClearPendingIRQ( GPIO0_IRQn );
  }


/* GPIO 1 Interrupt Handler */
/* This Interrupt handler executes upon the occurrence of
   GPIO 1 interrupt, which is from the Switch.
   This function prints source of the interrupt to the
   hyperterminal */

void GPIO1_IRQHandler( void )
  {
const uint8_t sw1[] = "\n\r Switch Interrupt occurred - GPIO 1 \n\r";
MSS_UART_polled_tx (&g_mss_uart1, sw1, sizeof(sw1));
MSS_GPIO_clear_irq( MSS_GPIO_1 );
NVIC_ClearPendingIRQ( GPIO1_IRQn );

  }
/* Fabric Interrupt Handler */
/* This Interrupt handler executes upon the occurrence of
   fabric interrupt, which is from the Timer 0.
   This function prints the source of the interrupt to the
   hyperterminal */
void FabricIrq0_IRQHandler( void )
{
        const uint8_t tim1[] = "\n\r Timer 0 Interrupt occurred - FABINT \n\r";
        MSS_UART_polled_tx (&g_mss_uart1, tim1, sizeof(tim1));
        NVIC_ClearPendingIRQ (FabricIrq0_IRQn);

  }

/*==============================================================================
```

```
  Display greeting message when application is started.
 */
static void display_greeting(void)
{
    MSS_UART_polled_tx(&g_mss_uart1, g_greeting_msg, sizeof(g_greeting_msg));

}


/*******************************************************************************
 Function Main
*******************************************************************************/

int main()
{

/*UART initialization*/
MSS_UART_init(
&g_mss_uart1,
MSS_UART_57600_BAUD,
MSS_UART_DATA_8_BITS | MSS_UART_NO_PARITY | MSS_UART_ONE_STOP_BIT );

    /* Display greeting message */

    display_greeting();


/* Enabling of GPIO 0, GPIO 1 and Fabric Interrupts*/
    /* MSS GPIO_0 used for Timer 2 input */
    NVIC_EnableIRQ(GPIO0_IRQn);
MSS_GPIO_config( MSS_GPIO_0, MSS_GPIO_INPUT_MODE | MSS_GPIO_IRQ_EDGE_POSITIVE );
MSS_GPIO_enable_irq( MSS_GPIO_0 );

/* Configure MSS GPIO_1 - used for switch input */
NVIC_EnableIRQ(GPIO1_IRQn);
MSS_GPIO_config( MSS_GPIO_1, MSS_GPIO_INPUT_MODE | MSS_GPIO_IRQ_LEVEL_LOW );
MSS_GPIO_enable_irq( MSS_GPIO_1 );

/* Enable Fabric_0 IRQ - used for Timer 1 input */
NVIC_EnableIRQ (FabricIrq0_IRQn);

for(;;)
{

}
return 0;
}
/*******************************************************************************
 End of function Main
*******************************************************************************/
```

4. From the **Project** menu, choose **Clean** to perform a clean build. Accept the default settings in the Clean dialog box and click **OK** (Figure 6-2).

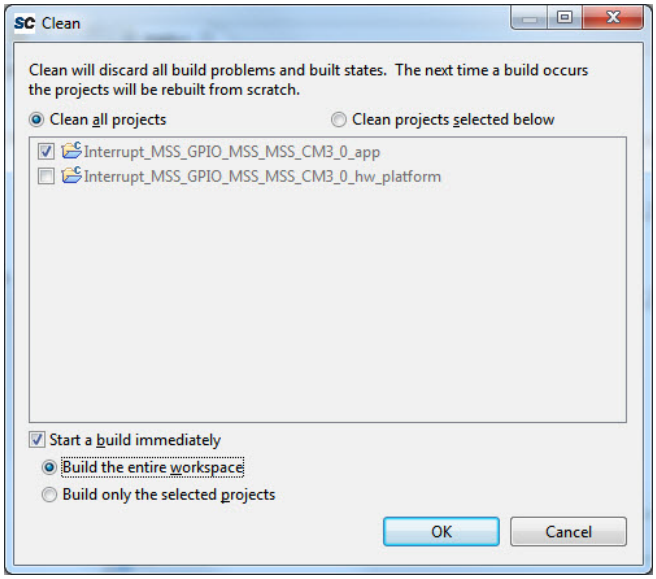


Figure 6-2 **• Settings for a Clean Build**

5. Click the **Problems** tab and make sure there are no errors and warnings. Correct errors before you continue (Figure 6-3).

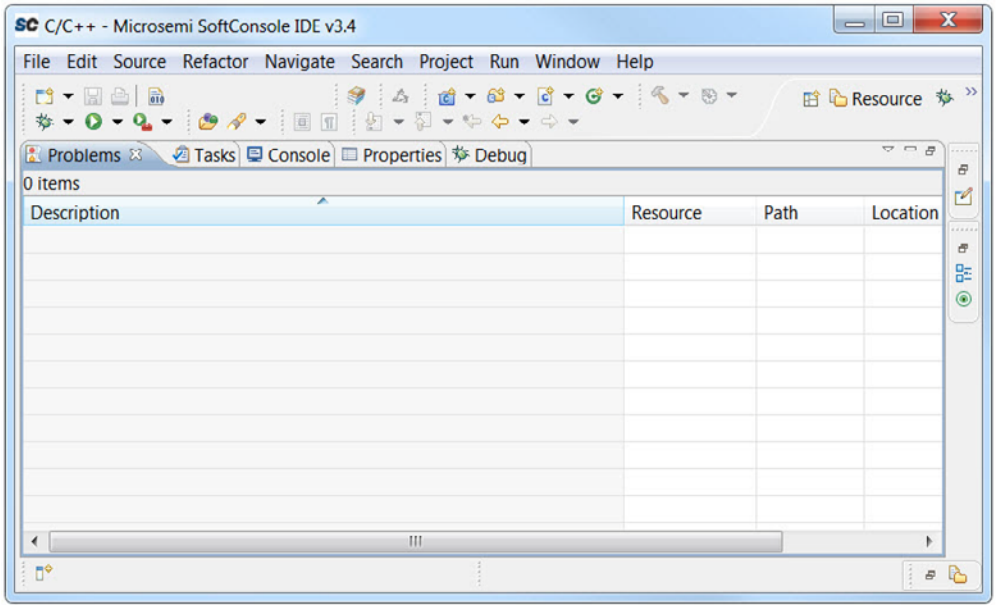


Figure 6-3 **• Softconsole Problems tab**

Step 2 - Configuring Hyper Terminal/Other Terminal Emulator Programs

Prior to running the application program, you need to configure a terminal emulator program (HyperTerminal, included with Windows®) on your PC. Perform the following steps to use the SmartFusion2 Evaluation Kit Board:

1. Connect a second mini USB cable between the USB connector (J14) on the SmartFusion2 Evaluation Kit Board and a USB port of your computer. If Windows prompts you to connect to Windows Update, select **No, not at this time** and click **Next**.
2. If the USB Serial Ports are automatically detected (displayed in Device Manager), as shown in Figure 6-4, it means that the USB-UART driver is already installed on your PC. Four consecutively numbered COM ports are listed (COM 4 through COM7, as in Figure 6-4). For your PC, the four COM port numbers may be different.

Note: If COM ports are not detected in the Device Manager, see "Appendix A Step 3 - Installing Drivers for the USB-UART" to install the USB-UART driver before proceeding.

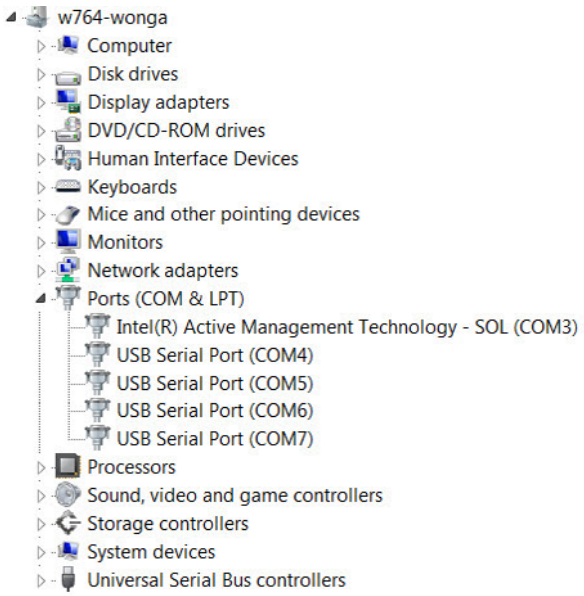


***Figure 6-4* • Device Manager Listing USB Serial Port**

3. From the Windows **Start** menu, choose **Programs > Accessories > Communications > HyperTerminal**. This opens HyperTerminal. If your PC does not have HyperTerminal, use any free serial terminal emulation program, such as PuTTY or Tera Term. Refer to the Configuring Serial Terminal Emulation Programs tutorial for configuring HyperTerminal, Tera Term, and PuTTY.

4. Enter **Hyperterminal** in the **Name** field in the Connection Description dialog box and click **OK** (Figure 6-5).

Figure 6-5 **• New Connection**

5. Choose the appropriate COM port (to which USB-RS232 drivers are pointed) from the **Connect using** drop-down list and click **OK** (Figure 6-6). Different systems may show different COM Port#. Choose the highest COM port displayed (COM7 in this case, as displayed in the Device Manager in Figure 6-4)..

Figure 6-6 **• Selecting the COM Port**

6. Set the following in the COM Properties window and click **OK** (Figure 6-7):
   - Bits per second: 57600
   - Data bits: 8
   - Parity: None
   - Stop Bits: 1

– Flow control: None

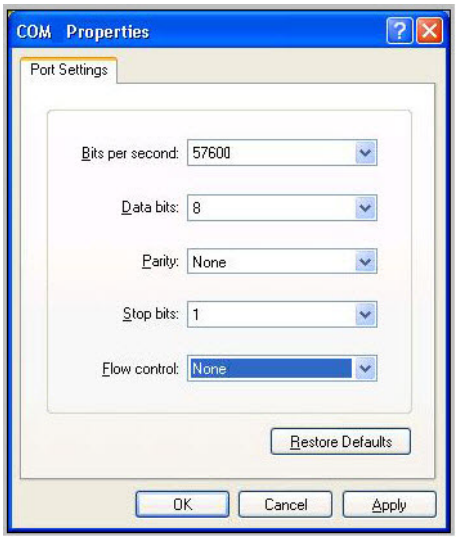


***Figure 6-7* • Setting the COM Properties**

7. Click **OK** to close the HyperTerminal Properties dialog box.

Alternatives to Hyper Terminal

Hyper Terminal is no longer a standard feature on newer Windows installations.

One alternative to the Hyper Terminal is the Tera Term, available for download from:

http://ttssh2.sourceforge.jp/

Another alternative is PuTTY, available for download from:

http://en.softonic.com/s/putty-download-windows-7

Configuring Tera Term

To configure Tera Term after installation:

1. Click the Tera Term icon to launch Tera Term.
2. When the New Connection dialog box appears (Figure 6-8), select **Serial**.
3. From the drop-down list of ports, choose the highest COM # from the group of four consecutive COM ports.

Note: Only one COM Port is activated on the SmartFusion2 kit. In this case, select **COM7:USB Serial PORT (COM7)**. Your PC may have different COM numbers.

.

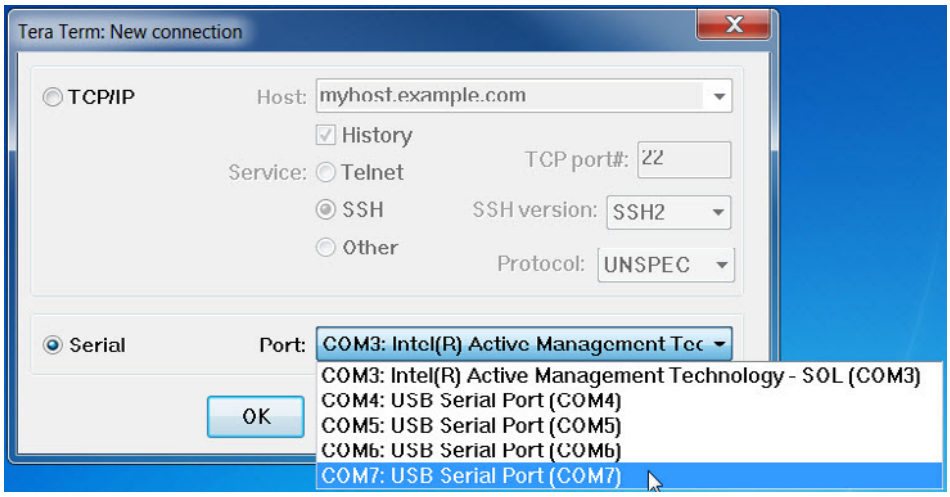


***Figure 6-8* • Configuring the Serial Port in Tera Terminal**

4. Click **OK** to close the Tera Term: New connection dialog box.
5. In the new pop-up Tera Terminal, click **Setup** and choose **Serial Port**. Enter the configuration as follows (Figure 6-9):
   – Port: COM7
   – Different systems may display different COM Port#. Only one COM port is activated on the Board for serial communication use. Always select the Port with the highest COM# (COM7 in this case).
   – Baud Rate: 57600
   – Data: 8 bit
   – Parity: None
   – Stop: 1 bit
   – Flow Control: None

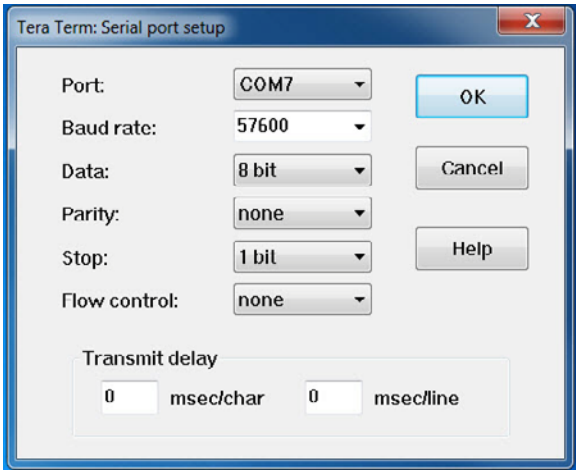


***Figure 6-9* • Tera Term Serial Port Setup**

6. Click **OK** to close the Tera Term: Serial port setup dialog box.

7. Click **Setup** and choose **Windows**. The Tera Term: Window setup dialog box appears (Figure 6-10).

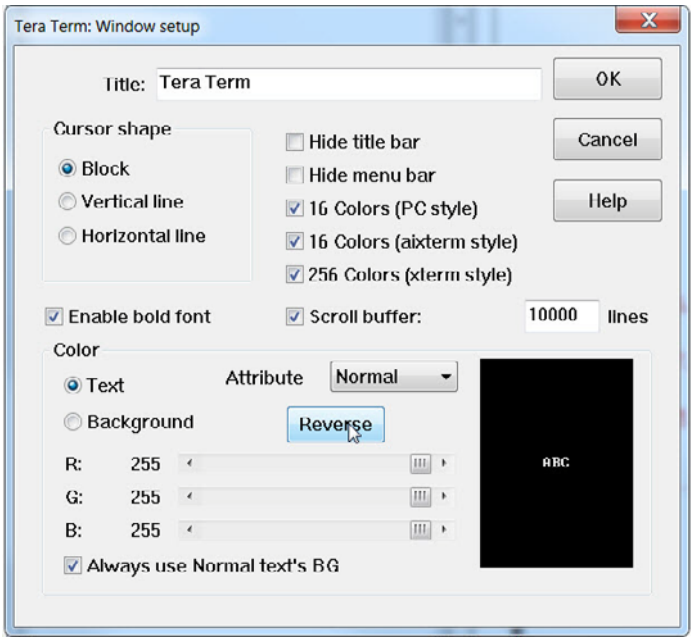


Figure 6-10 **• Tera Term Window Setup Dialog Box**

8. Click **Reverse** to change the background to white (black text on a white background) for easy viewing.

Step 3- Debugging the Application Project Using SoftConsole

To debug the application project using SoftConsole:

1. From the **Run** menu in SoftConsole, choose **Debug Configurations**. The Debug Configurations dialog box appears.

2. Double-click **Microsemi Cortex-M3 RAM Target** to display the Options for the RAM target (Figure 6-11).

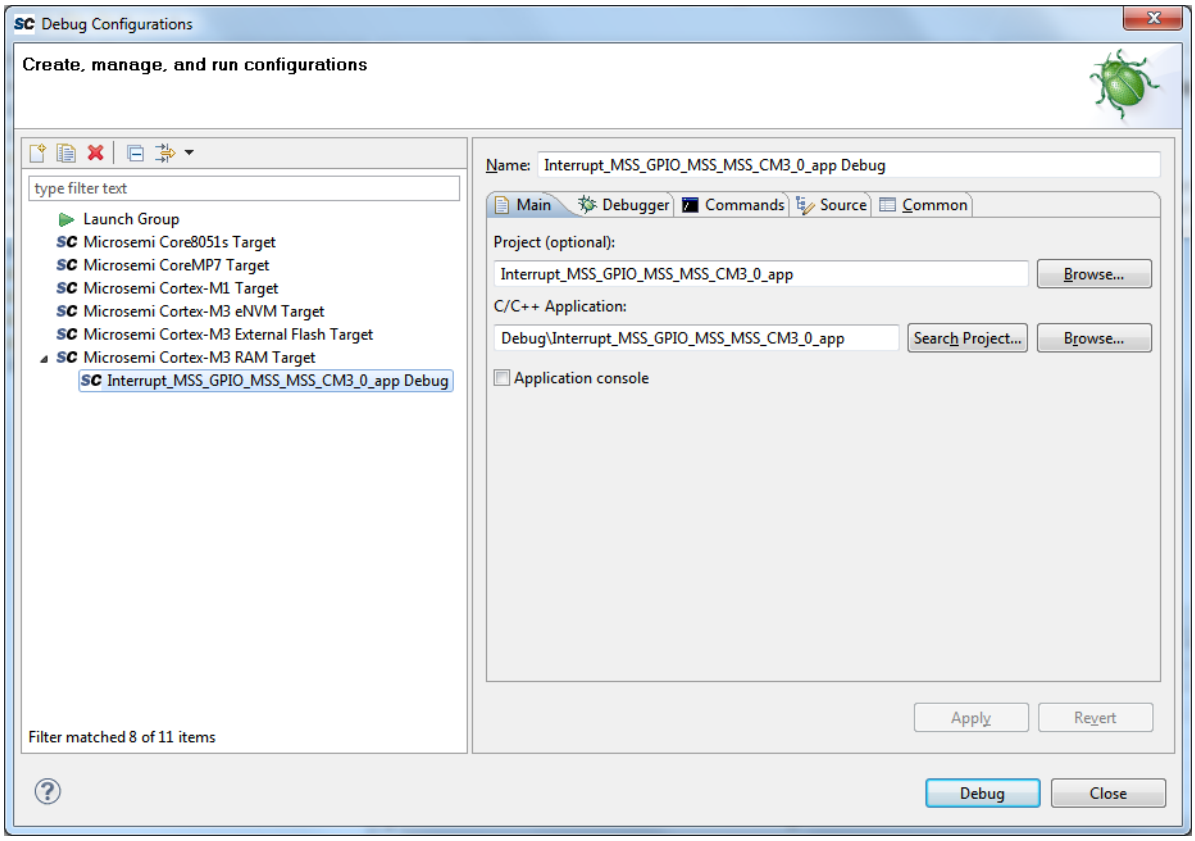


Figure 6-11 **• Debug Configurations Dialog Box**

3. Confirm that the following appear on the **Main** tab in the Debug Configuration dialog box:
   - **Name**: Interrupt_MSS_GPIO_MSS _MSS_CM3_app Debug
   - **Project**: Interrupt_MSS_GPIO_MSS _MSS_CM3_app
   - **C/C++ Application**: Debug\ Interrupt_MSS_GPIO_MSS _MSS_CM3_app
4. Click **Apply** and **Debug**.
5. Click **Yes** when prompted for **Confirm Perspective Switch** (Figure 6-12). This displays the Debug view mode.

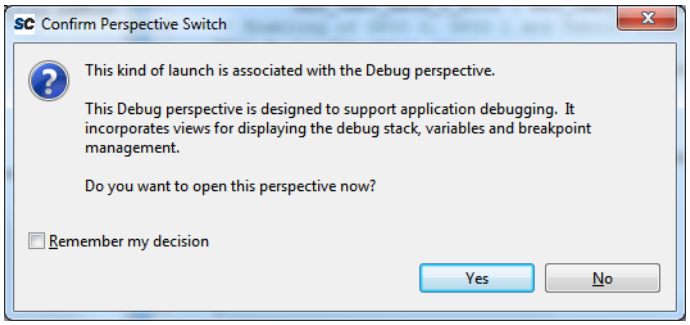


Figure 6-12 **• Confirm Perspective Switch**

Your Debug Perspective should resemble Figure 6-13:

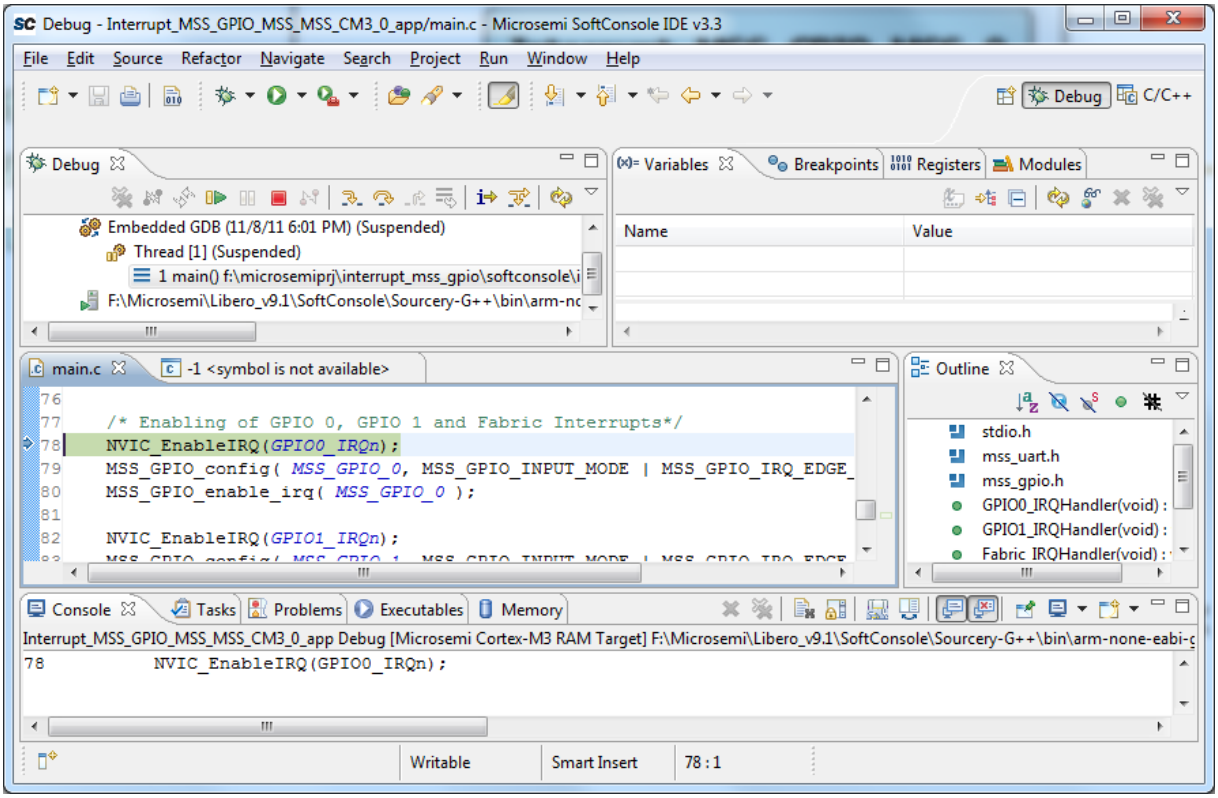
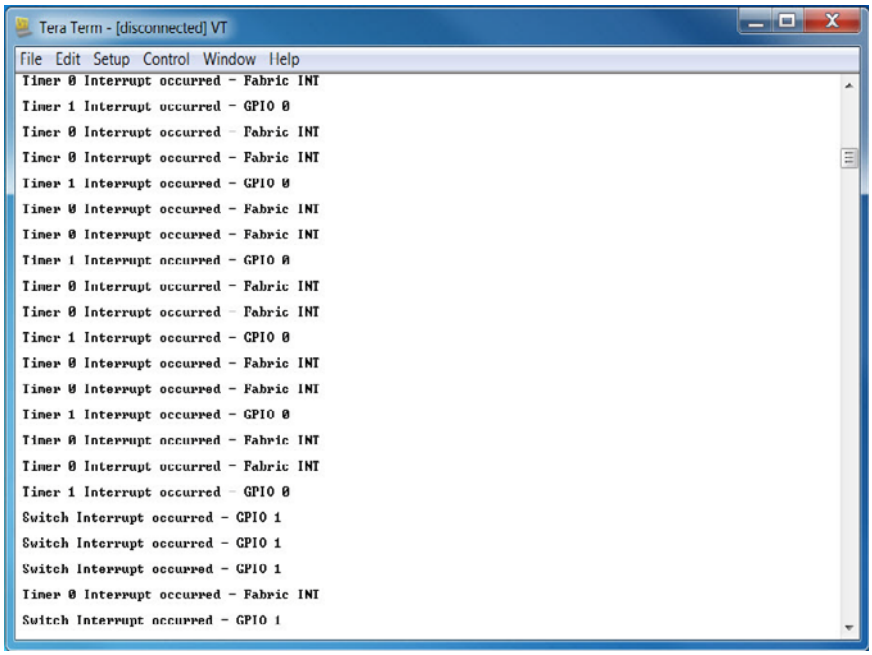


Figure 6-13 **• Debug Perspective**

6. Click **Run** and choose **Resume** to run the application.

The interrupt sequence messages are displayed in the terminal program window. Press the SW3 switch on the SmartFusion2 Evaluation Kit and observe the message in the serial terminal that indicates the switch interrupt occurred (Figure 6-14).



```
Timer 0 Interrupt occurred - Fabric INT
Timer 1 Interrupt occurred - GPIO 0
Timer 0 Interrupt occurred - Fabric INT
Timer 0 Interrupt occurred - Fabric INT
Timer 1 Interrupt occurred - GPIO 0
Timer 0 Interrupt occurred - Fabric INT
Timer 0 Interrupt occurred - Fabric INT
Timer 1 Interrupt occurred - GPIO 0
Timer 0 Interrupt occurred - Fabric INT
Timer 0 Interrupt occurred - Fabric INT
Timer 1 Interrupt occurred - GPIO 0
Timer 0 Interrupt occurred - Fabric INT
Timer 0 Interrupt occurred - Fabric INT
Timer 1 Interrupt occurred - GPIO 0
Timer 0 Interrupt occurred - Fabric INT
Timer 0 Interrupt occurred - Fabric INT
Timer 1 Interrupt occurred - GPIO 0
Switch Interrupt occurred - GPIO 1
Switch Interrupt occurred - GPIO 1
Switch Interrupt occurred - GPIO 1
Timer 0 Interrupt occurred - Fabric INT
Switch Interrupt occurred - GPIO 1
```

Figure 6-14 • **Serial Display of the Interrupts in Tera Term**

7. Terminate the Debugger (**Run > Terminate**) on SoftConsole.

Step 4 - Building an Executable Image in Release mode

You can build an application executable image with the Release Configuration and load it into eNVM. The Application code can then be executed from the eNVM after the SmartFusion2 device is programmed.

To load the application image into the eNVM of the SmartFusion2 device and use it for execution, you need to:

1. Configure the eNVM for a data storage client.
2. Program the eNVM data storage client.

In release mode, you cannot use the SoftConsole debugger to load the executable image into eNVM. Make sure you exit the Debugger.

Steps

1. Right-click on both project names in the Project Explorer view and choose **Build Configurations > Set Active > Release**, as shown in Figure 6-15.

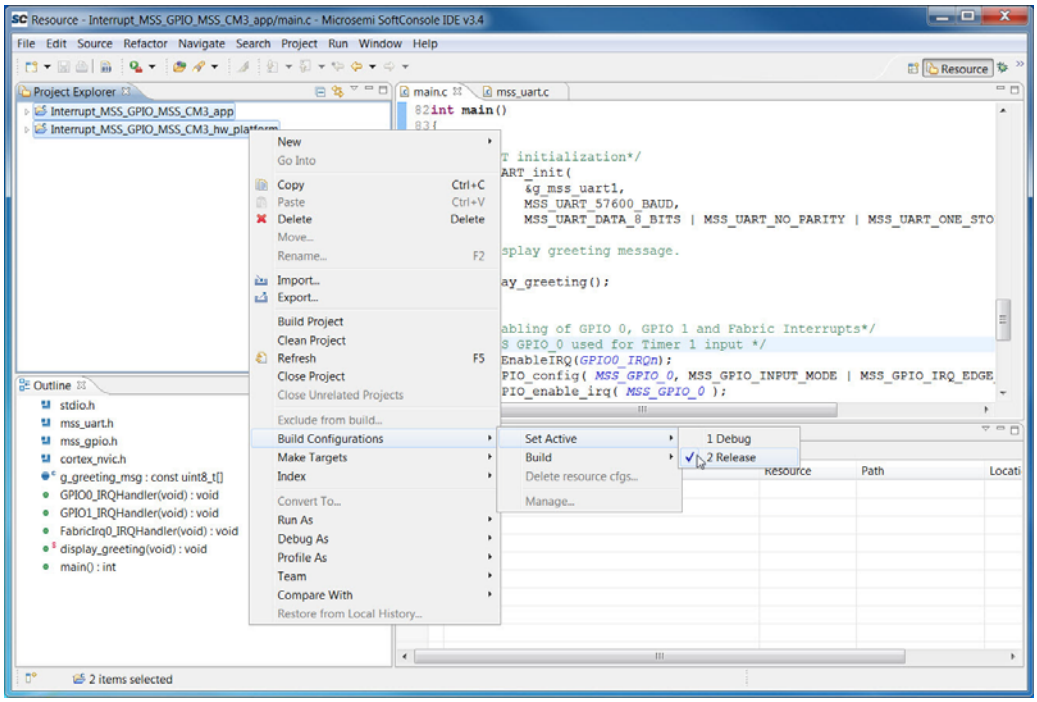


Figure 6-15 **• Configuring Release Mode**

2. Select **Project > Properties** from the SoftConsole menu. Expand C/C++ Build and select 'Settings'.
3. Provide the release mode linker script file to the linker by changing the 'Linker flags' field to "-T../../Interrupt_MSS_GPIO _MSS_CM3_0_hw_platform/CMSIS/startup_gcc/production-execute-in-place.ld", as shown in Figure 6-16. This step directs SoftConsole to use the production linker script, instead of the default linker script (that allows debugging), to build the application image for the Release Application.

.

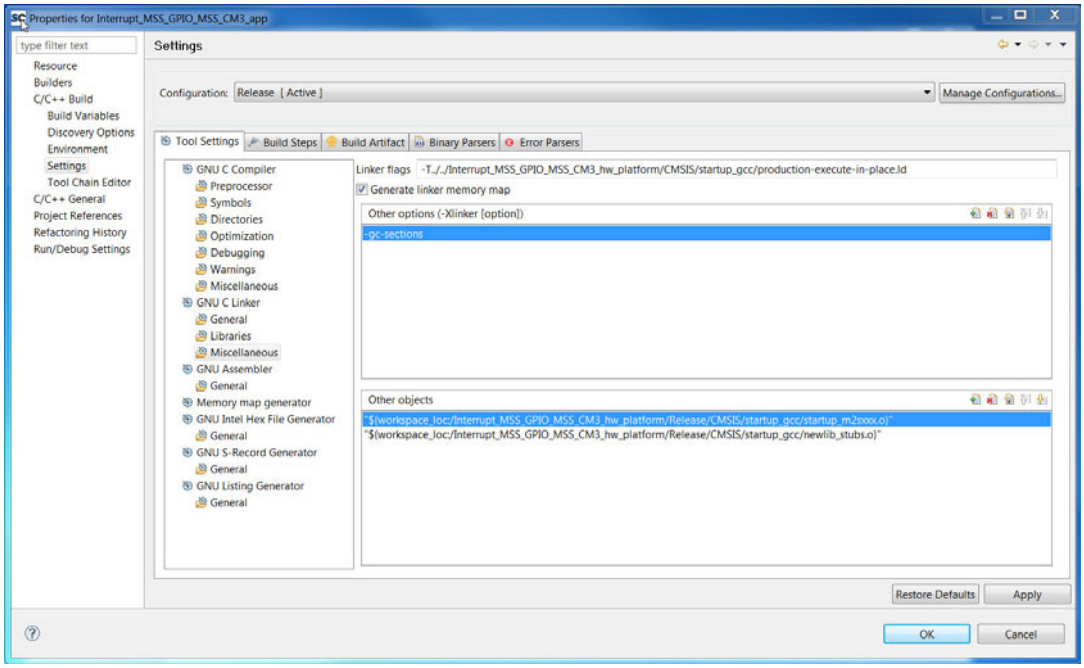


Figure 6-16 **• Linker Script Settings**

4. Click **Apply** and then click **OK**.
5. Build the project (**Project > Build All**) and observe that the '.hex' file is generated in the 'Release' folder created in the project folder, as shown in Figure 6-17 and Figure 6-18.

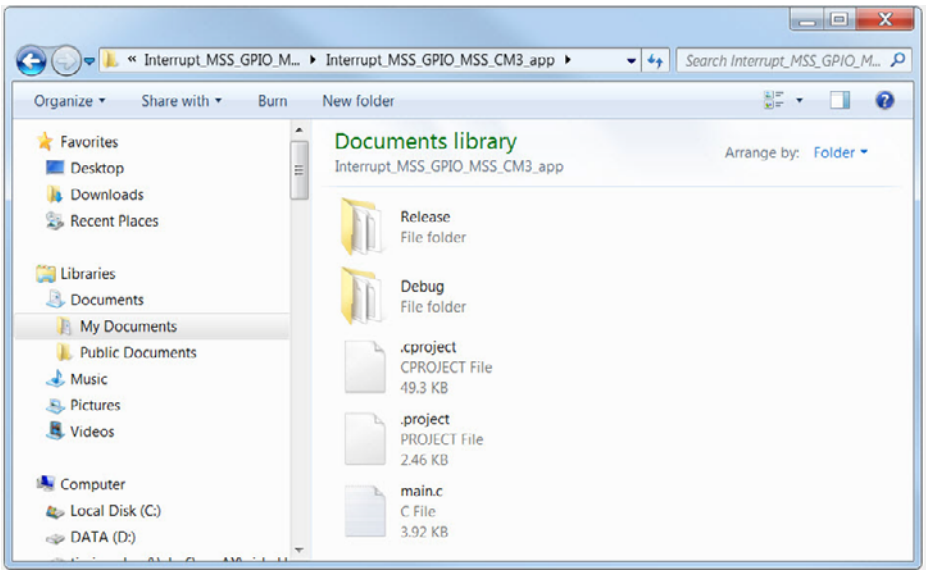


Figure 6-17 **• Release Folder**

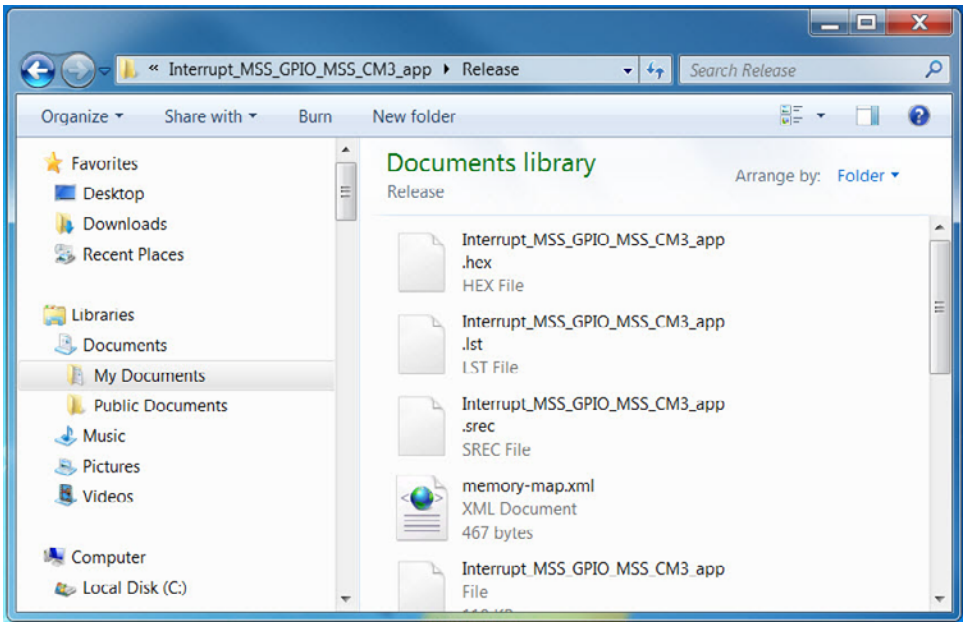


Figure 6-18 **• Hex File in Release Folder of SoftConsole Project Location**

6. Close SoftConsole.

Loading the Executable Image into eNVM

This section provides the steps required to load the generated executable image into eNVM of the SmartFusion2 cSoC device using the SmartDesign MSS Configurator.

1. From the Libero SoC Design Flow window, right-click **Update eNVM Memory Content** and choose **Open Interactively** (Figure 6-19).

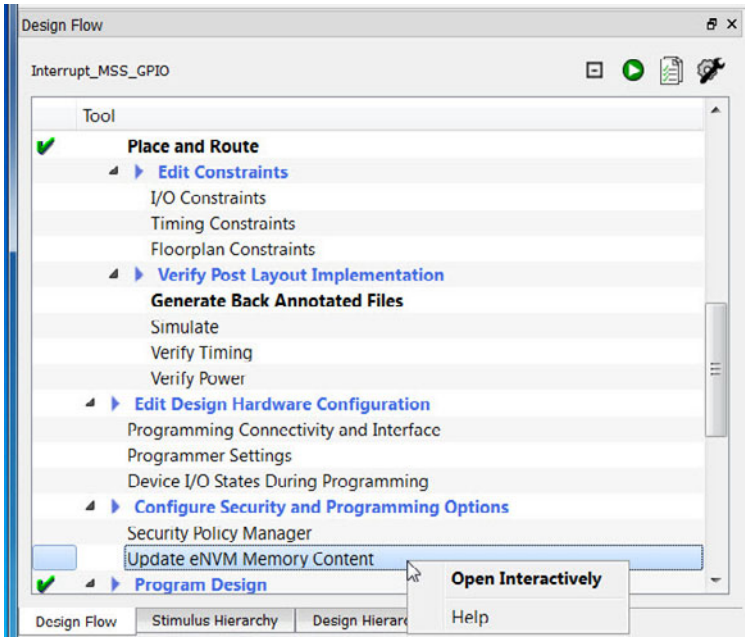


Figure 6-19 **• Update eNVM Memory Content**

2. The Update eNVM Memory Content dialog box appears (Figure 6-20).

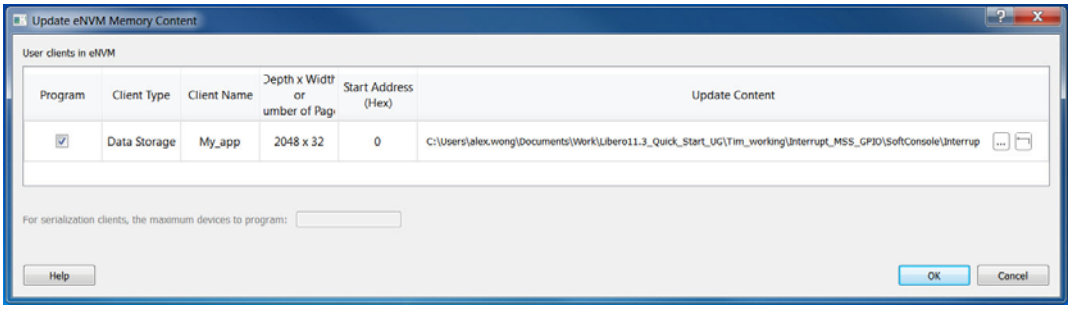


Figure 6-20 **• Update eNVM Memory Content Dialog Box**

3. Click the Program checkbox.
4. Click the Browse button (second from right) to navigate to the location of the *.hex file (<SoftConsole Project Folder>/../Interrupt_MSS_GPIO_MSS_CM3_app/Release folder) created by SoftConsole when you build the Application codes. Click the *.hex file and click **Open** (Figure 6-21).

.

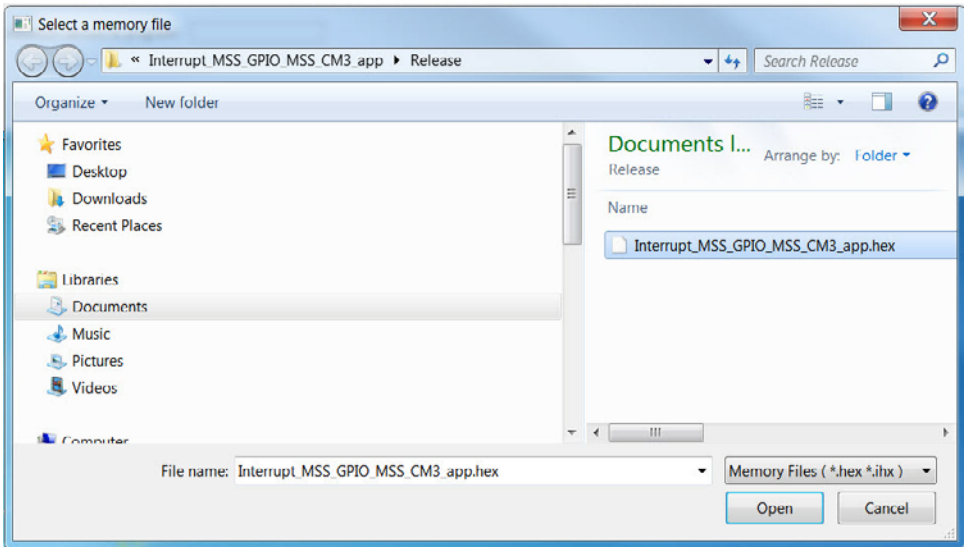


Figure 6-21 **• Loading the *hex file into eNVM Memory Content**

5. Click **OK** to exit the dialog box.
6. In the Design Flow window, right-click **Bitstream Configuration** and choose **Configure Options**.
7. Under Selected Features, deselect Fabric. Make sure eNVM is selected.

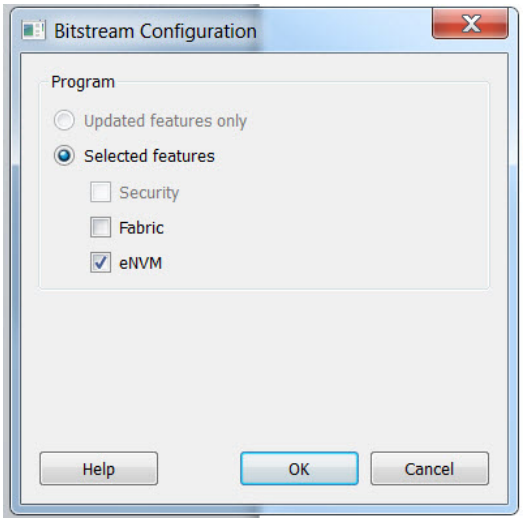


Figure 6-22 **• Bitstream Configuration for eNVM**

8. Click **OK** to exit.
9. Make sure the SoftConsole Debugger is terminated (**Run > Terminate**). Right-click **Run Program Action** and choose **Update and Run**.

   A green check mark appears next to **Run PROGRAM Action**. The application is programmed into the eNVM of the SmartFusion2 device. It is ready to run off the device's eNVM without launching SoftConsole.

10. Disconnect the FlashPro Programmer from the SF2-Eval-Board. Make sure that SoftConsole closes.
11. Open the Terminal Emulator. Verify that the Application Code is running off the eNVM flash memory of the device.

Congratulations! You have successfully completed this tutorial.

A – Installing Drivers for the USB-UART

Note: You must have Admin privileges on your PC to install the USB-RS232 drivers.

To install the USB to UART driver:

1. Connect a second mini USB cable between the USB connector (J14) on the SmartFusion2 Evaluation Kit Board and a USB port of your computer.
2. Download the USB to RS232 bridge USB to UART driver from http://www.microsemi.com/document-portal/doc_download/131593-usb-uart-driver-files
3. Click **Downloads**.
4. Extract the downloaded zip file into a folder on your PC.
5. Open the Device Manager on your PC.
6. Right-click Intel® Active Management Technology - SOL (COM3) under Ports (COM & LPT)
7. Choose **Update Driver Software** (Figure A-1).

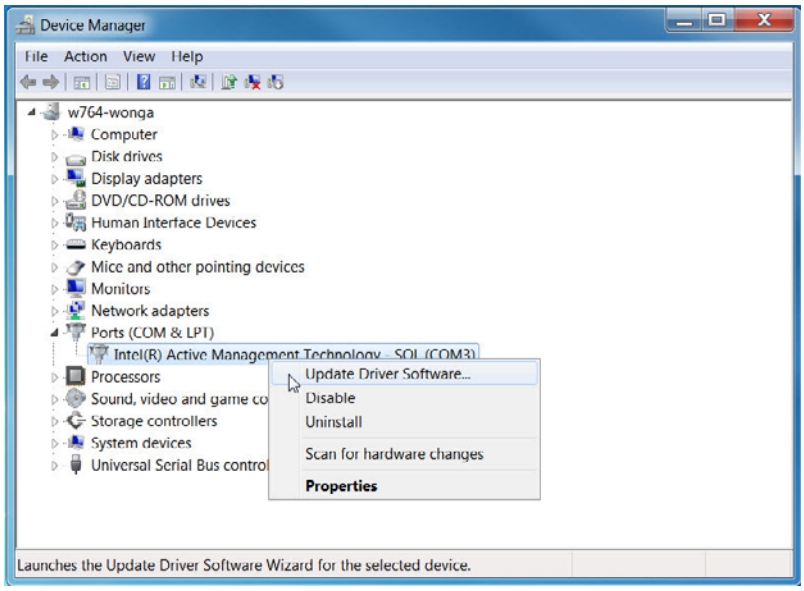


Figure A-1 **• Device Manager - Update Driver Software**

8. Select Browse my computer for driver software (Figure A-2).

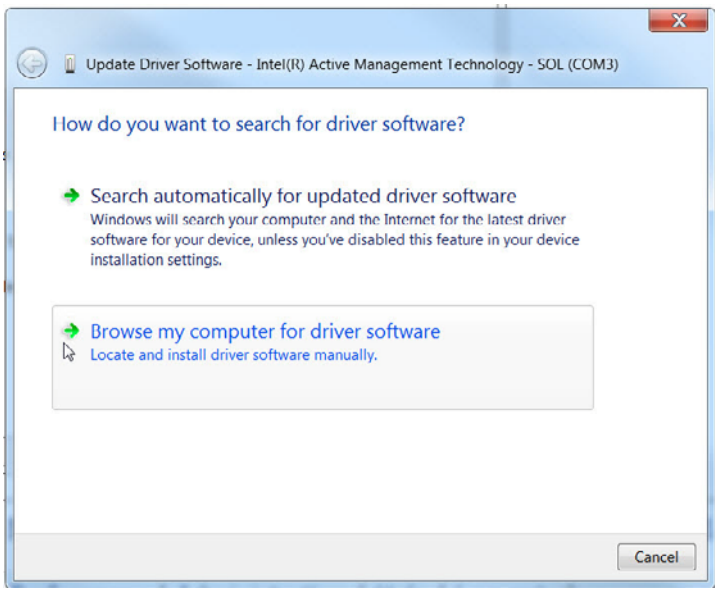


Figure A-2 **• Update Driver Software from driver software located locally on PC**

9. Browse to the folder location of your driver on your PC (Figure A-3).

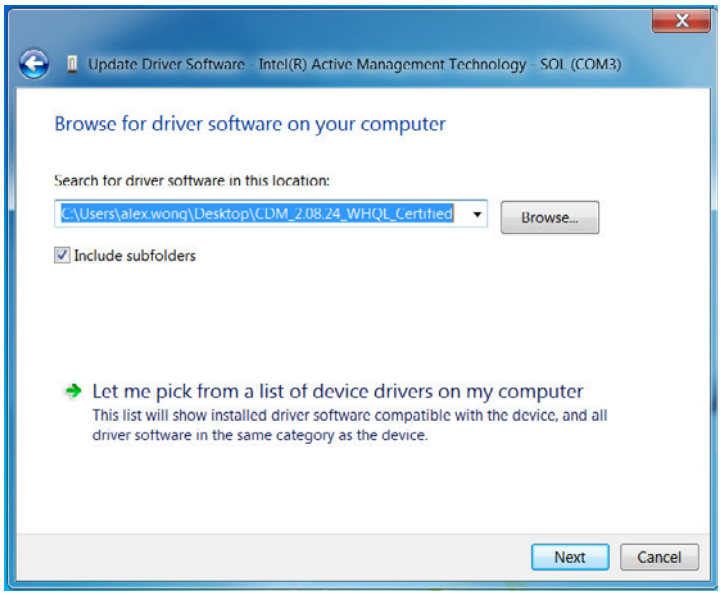


Figure A-3 **• Local Drive Software Location**

10. Follow the prompts to complete the driver installation.
11. When the installation is complete, click **OK**. The Ports (COM & LPT) section of the Device Manager (Figure A-4) lists four USB Serial Ports (COMn) numbered consecutively where n can be any number. The USB Serial Port numbers on your PC may be different from Figure A-4.

.

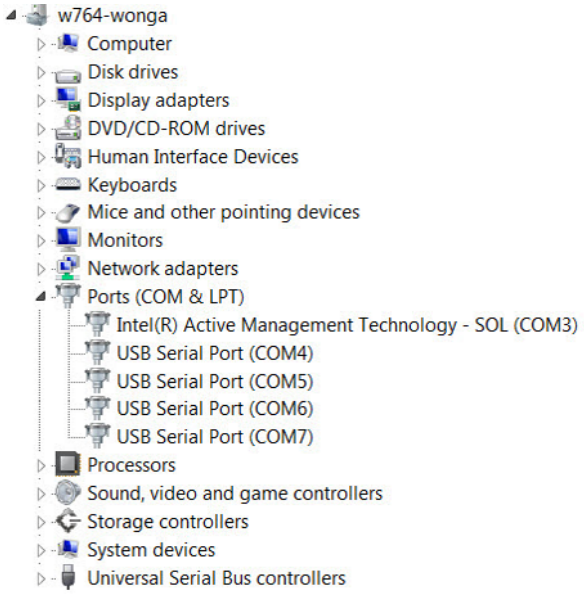


Figure A-4 **• Device Manager USB Serial Ports**

B – Product Support

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

Customer Service

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

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

Customer Technical Support Center

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

Technical Support

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

Website

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

Contacting the Customer Technical Support Center

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

Email

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

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

My Cases

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

Outside the U.S.

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

ITAR Technical Support

For technical support on RH and RT FPGAs that are regulated by International Traffic in Arms Regulations (ITAR), contact us via soc_tech_itar@microsemi.com. Alternatively, within My Cases, choose **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

5-02-9123-20/10.14