SmartFusion2 SoC FPGA PCIe Control Plane Demo - Libero SoC v11.6

DG0456 Demo Guide

November 2015

Microsemi

Revision History

Date	Revision	Change
11 November 2015	5	Fifth release
February 2015	4	Fourth release
August 2014	3	Third release
April 2014	2	Second release
December 2013	1	First release
June 2013	0	Initial release

Confidentiality Status

This is a non-confidential document.

Table of Contents

SmartFusion2 SoC FPGA - PCIe Control Plane Demo 4
Introduction 4
Demo Requirements 5
Hardware and Software Requirements 5
Design Files 5
Demo Design Description 6
Building the Demo 7
Step 1: Creating a Libero SoC Project 7
Instantiating SERDESIF Component in PCIe_Demo SmartDesign 19
Instantiating Debounce Logic in PCIe_Demo SmartDesign 24
Connecting Components in PCIe_Demo SmartDesign 25
Configuring and Generating Firmware 31
Step 2: Creating an eNVM Client 33
Step 3: Developing the Simulation Stimulus 45
Step 4: Simulating the Design 48
Step 5: Generating the Program File 52
Running the Demo 55
Demo Setup 55
Running the Demo Design 56
Running the Demo Design on Windows 57
Running the Demo Design on Linux 70
Conclusion 80

List of Changes 81

Product Support 82
Customer Service 82
Customer Technical Support Center 82
Technical Support 82
Website 82
Contacting the Customer Technical Support Center 82
Email 82
My Cases 83
Outside the U.S. 83
ITAR Technical Support 83

SmartFusion2 SoC FPGA - PCIe Control Plane Demo

Introduction

SmartFusion®2 system-on-chip (SoC) field programmable gate array (FPGA) devices integrate a fourth generation flash-based FPGA fabric and an ARM® Cortex®-M3 processor, along with high performance communication interfaces on a single chip. The SmartFusion2 high speed serial interface (SERDESIF) provides a fully hardened PCIe endpoint (EP) implementation and is compliant with PCIe Base Specification Revision 2.0 and 1.1. For more information, refer to the

UG0447: IGLOO2 and SmartFusion2 High Speed Serial Interfaces User Guide.

The demo explains the SmartFusion2 embedded PCI Express feature and how this can be used as a low bandwidth control plane interface using the SmartFusion2 Security Evaluation Kit. The demo provides a simple design to access the SmartFusion2 PCIe EP from a Host PC. A GUI is provided for read and write access to the SmartFusion2 PCIe configuration space and memory space of BAR0 and BAR1. The demo also provides Host PC device drivers for the SmartFusion2 PCIe EP. This demo can run on both windows and Red Hat Linux operating system.

Figure 1 shows the top-level block diagram for the PCIe control plane demo. The demo design uses a SmartFusion2 PCIe interface with a maximum link width of x4 to interface with a Host PC PCIe Gen2 slot. The SmartFusion2 microcontroller subsystem (MSS) GPIOs control the LEDs and switches on the SmartFusion2 Security Evaluation Kit through the PCIe interface. The Host PC can also read memory and writes to the SmartFusion2 eSRAM through the GUI. The Host PC can also be interrupted by using the push button on the SmartFusion2 Security Evaluation Kit.

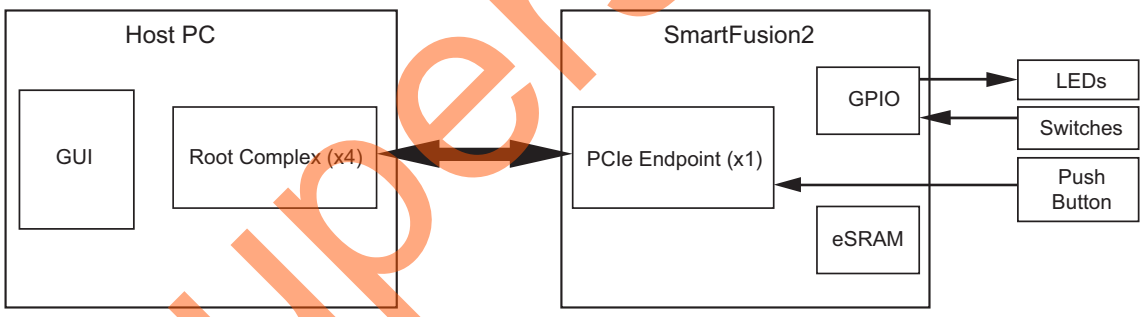


Figure 1 • **PCIe Control Plane Demo Top-Level Block Diagram**

The demo design performs the following tasks:

- Displays the PCIe link enable or disable, negotiated link width, and the link speed.
- Controls the status of LEDs on the SmartFusion2 Security Evaluation Kit according to the command from the GUI.
- Displays the position of DIP Switches on SmartFusion2 Security Evaluation Kit.
- Enables read and write to eSRAM.
- Interrupts the Host PC, when the push button is pressed. The GUI displays the count value of the number of interrupts sent from the SmartFusion2 Security Evaluation Kit.
- Displays the SmartFusion2 PCIe Configuration Space.

Demo Requirements

Hardware and Software Requirements

Table 1 shows the hardware and software required to run the demo.

Table 1 • **Hardware and Software Requirements**

Hardware	**Version**
SmartFusion2 Security Evaluation Kit	Rev C or later
12 V adapter (provided along with the kit)	-
FlashPro4 programmer (provided along with the kit)	-
Host PC with an available PCIe 2.0 Gen1 or Gen2 compliant slot	Operating system: Windows XP SP2: 64-bit Windows 7: 64-bit or Red Hat Linux Kernel Version: 2.6.18-308
Software	
Libero® System-on-Chip (SoC)	v11.6
SoftConsole	v3.4SP1
Host PC Drivers (provided along with the design files)	-
GUI executable (provided along with the design files)	-

Design Files

The design files for this demo can be downloaded from the Microsemi website:
http://soc.microsemi.com/download/rsc/?f=m2s_dg0456_liberov11p6_df

Design files include:

- Libero project
- Linux_64bit
- ProgrammingFile
- Windows_64bit
- Source files
- Readme file

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

Demo Design Description

This demo design implements the SmartFusion2 embedded PCI Express interface as a low bandwidth control plane interface. This design provides Host PC drivers and a Host PC interface over PCIe to control the SmartFusion2 device. Figure 2 shows a detailed block diagram of the design implementation. The PCIe EP device receives commands from the Host PC through the GUI and does corresponding memory writes to the SmartFusion2 MSS address space. The MSS address space provides a GPIO block and eSRAM memory block, which is accessed through a Fabric Interface Controller (FIC_0).

The SERDES_IF2_0 is configured for a PCIe 2.0, x1 link width with GEN2 speed. The PCIe interface to the fabric uses an AMBA High-speed Bus (AHB). The AHB master interface of SERDESIF is enabled and connected to the AHB slave interface of FIC_0 to access the MSS peripherals. The SmartFusion2 PCIe BAR0 and BAR1 are configured in 32-bit memory mapped memory mode.

The AXI master windows of the SERDESIF PCIe provide address translation for accessing one address space from another address space as the PCIe address is different from SmartFusion2 AHB bus matrix address space. The AXI master window 0 is enabled and configured to translate the BAR0 memory address space to the MSS GPIO address space to control the MSS GPIOs. The AXI master window 1 is enabled and configured to translate the BAR1 memory address space to the eSRAM address space to perform read and writes from PCIe.

MSS GPIO block is enabled and configured as below:

- GPIO_0 to GPIO_7 as outputs and connected to LEDs
- GPIO_8 to GPIO_11 as inputs and connected to DIP switches

The PCIe interrupt line is connected to the SW4 push button on the SmartFusion2 Security Evaluation Kit. The FPGA clocks are configured to run the FPGA fabric and MSS at 90 MHz.

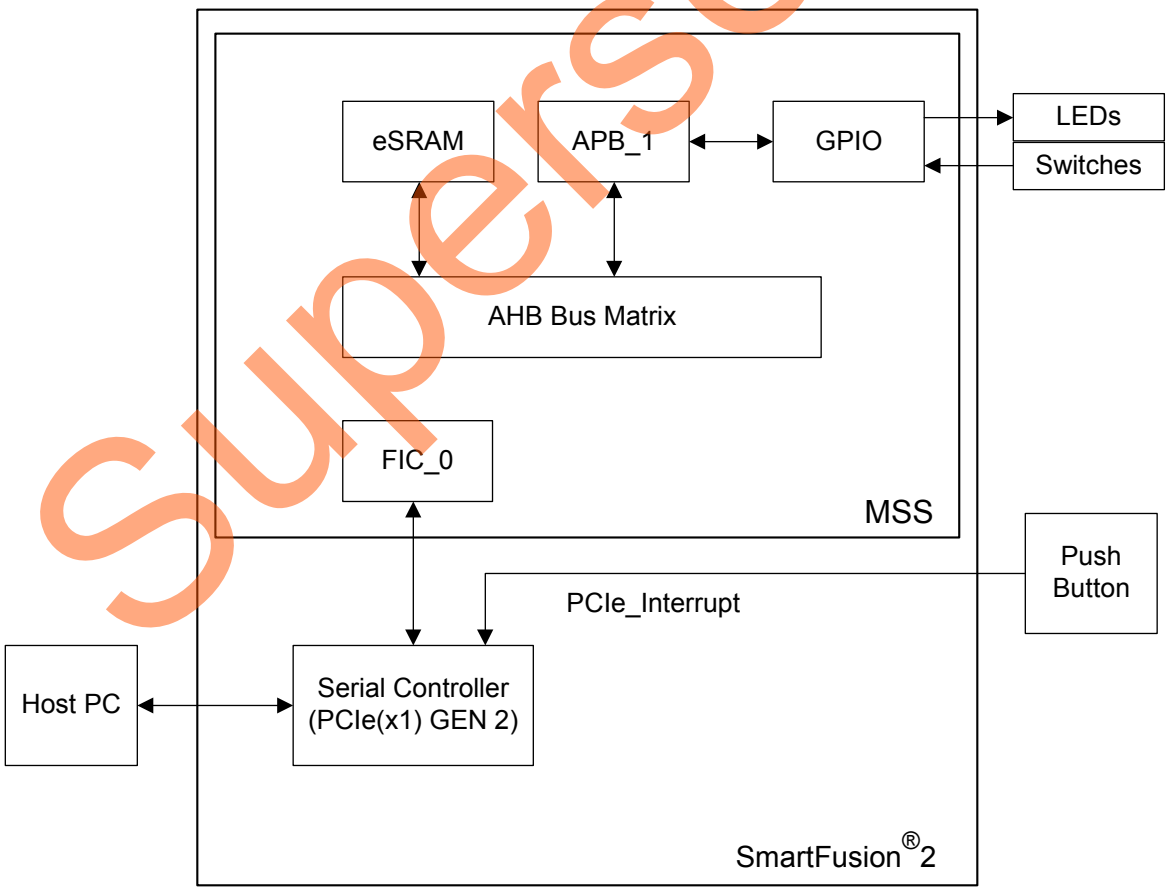


***Figure 2* • PCIe Control Plane Demo Block Diagram**

Building the Demo

This demo design provides a complete design flow starting from a new project to a working design on the SmartFusion2 Security Evaluation Kit. This process includes usage of the tools in the Libero SoC design suite to program the SmartFusion2 device.

Building the demo involves the following steps:

- Step 1: Creating a Libero SoC Project
- Step 2: Creating an eNVM Client
- Step 3: Developing the Simulation Stimulus
- Step 4: Simulating the Design
- Step 5: Generating the Program File

Step 1: Creating a Libero SoC Project

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

1. Click ***Start > Programs > Microsemi Libero SoC v11.6 > Libero SoC v11.6***, or click desktop shortcut. The Libero SoC v11.6 Project Manager is displayed, as shown in Figure 3.

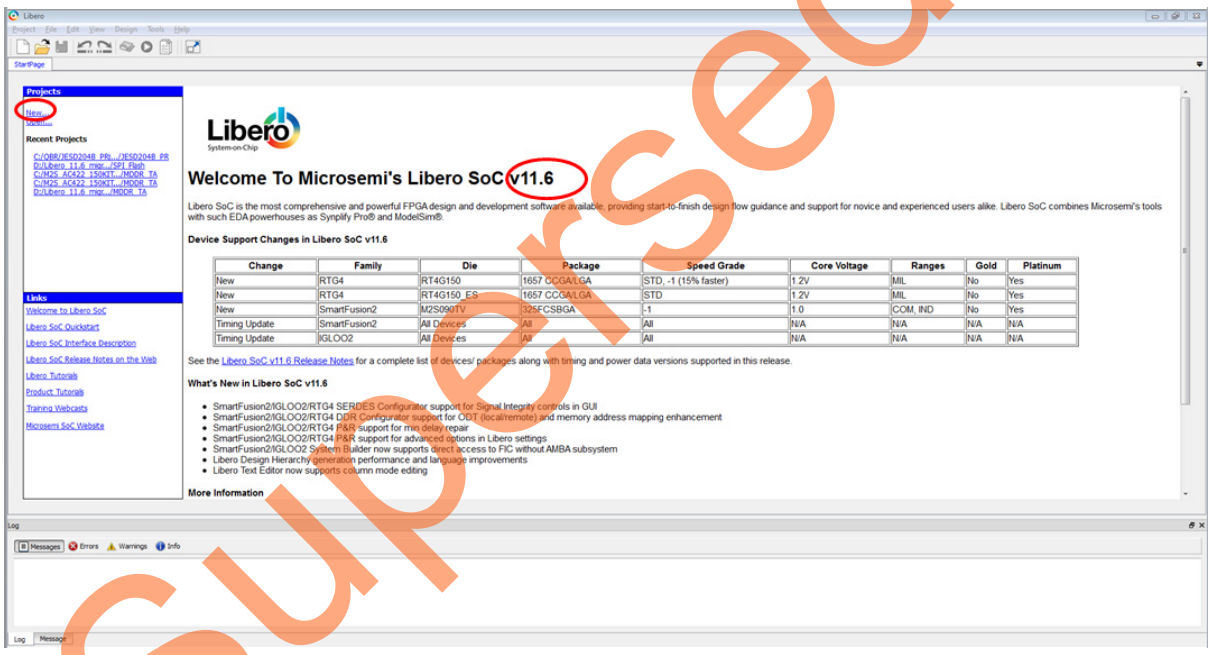


Figure 3 • **Libero SoC v 11.6 Project Manager**

2. Create a new project using one of the following options:
   – Select **New** on the Start Page tab as highlighted in Figure 3 on page 7.
   – Click **Project > New Project** from the Libero SoC menu.
3. Enter the following information in the **New Project-Project Details** tab as shown in Figure 4.
   – Project Name: PCIE_Demo
   – Project Location: Select an appropriate location (for example, D:/Microsemi_proj)
   – Preferred HDL type: Verilog or VHDL

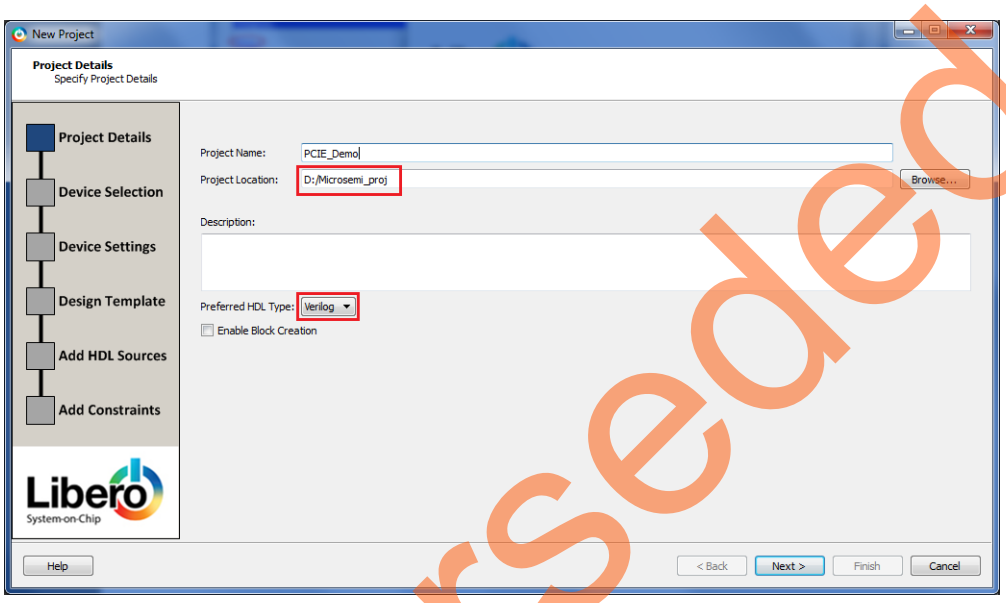


Figure 4 • **Project Details Tab**

4. Select the information for **Device Selection** as shown in Figure 5 on page 9 and click **Next**.
   – Family: SmartFusion2
   – Die: M2S090T
   – Package: 484 FBGA
   – Speed: -1
   – Core Voltage: 1.2
   – Operating conditions: COM

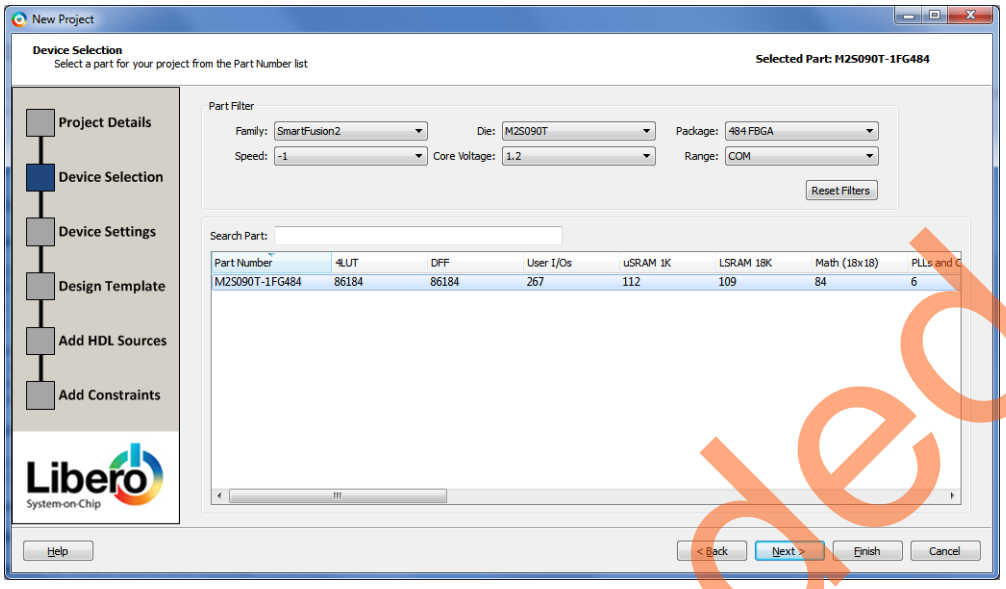


Figure 5 • **Device Selection Tab**

5. Select the information for **Device Settings**, as shown in Figure 6 and click **Next**.

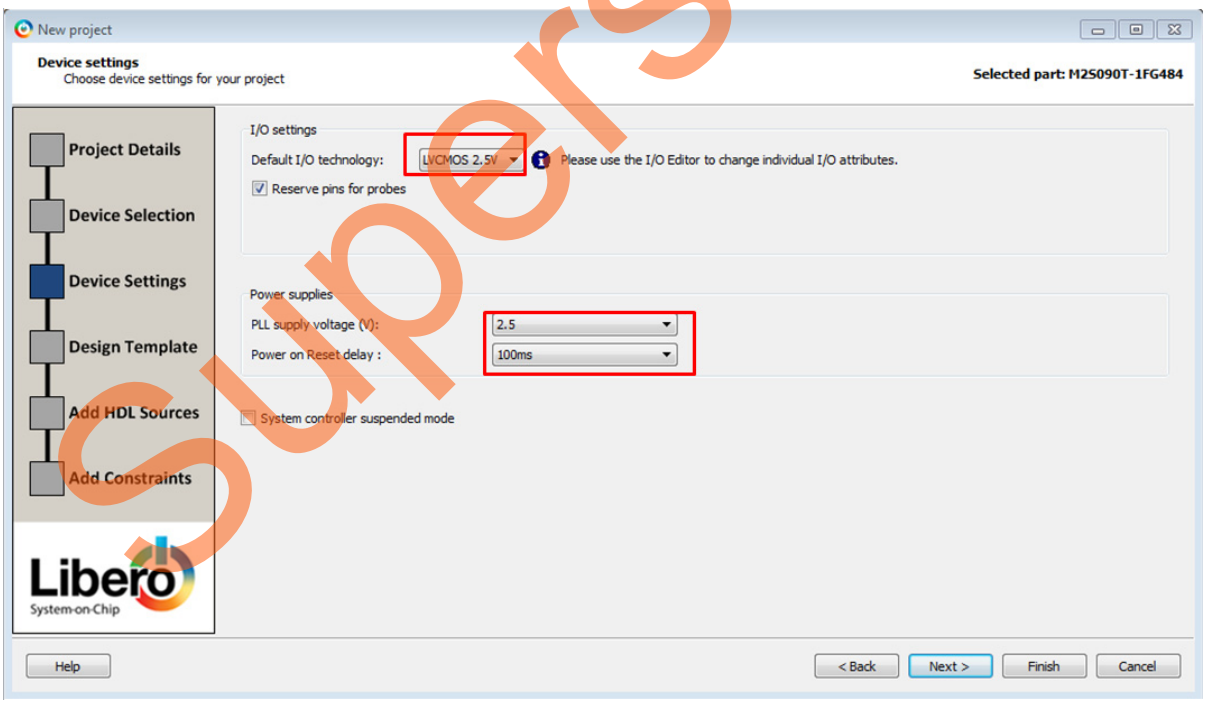


Figure 6 • **Device Settings Tab**

6. **Design Template** tab is displayed, as shown in Figure 7. Select **Create a System Builder based design** under **Design Templates and Creators** and click **Next**.

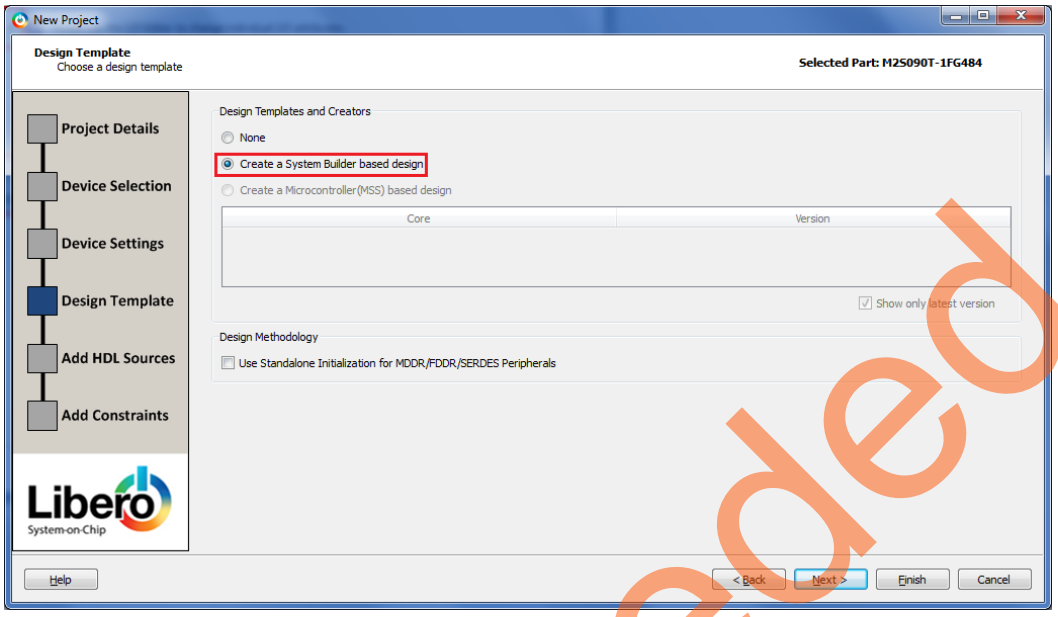


***Figure 7* • Design Template Tab**

7. **Add HDL Sources** tab is displayed, as shown in Figure 8. **Verilog/VHDL Source Files** can be added here.

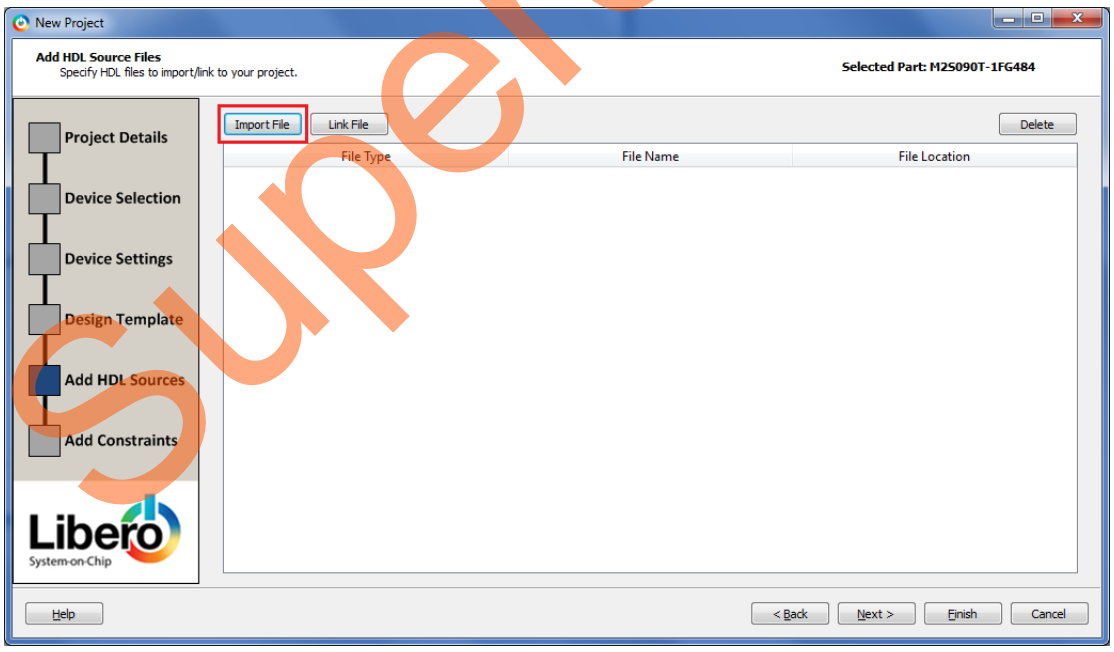


***Figure 8* • Add HDL Source Files Tab**

8. **Add Constraints** tab is displayed, as shown in Figure 9. **Constraints** file can be added using Import option.

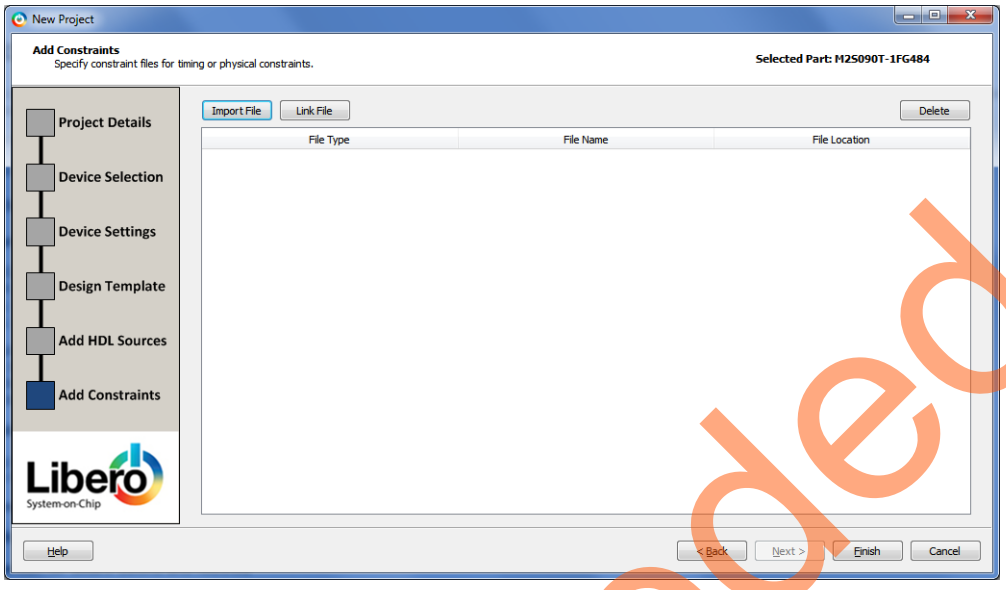


***Figure 9* • Add Constraints Tab**

9. Click **Finish**. This displays the **System Builder** dialog box.
10. Enter a name for your system in the column provided, as shown in Figure 10.

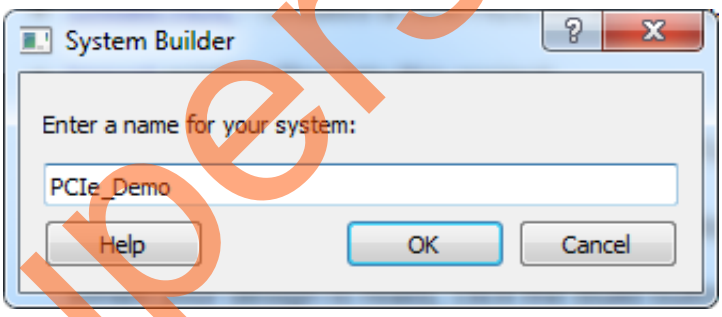


***Figure 10* • System Builder Dialog Box**

11. Enter **PCIe_Demo** as the name of the system and click **OK**. The System Builder dialog box is displayed with the Device Features page open by default.
12. Enter the following information in the **System Builder – Device Features page**, as shown in Figure 11 on page 12:
    - Memory: Clear all except MSS On-chip Flash Memory (eNVM)
    - High-speed serial interfaces: Check SERDESIF_0
    - Microcontroller Options: Clear All

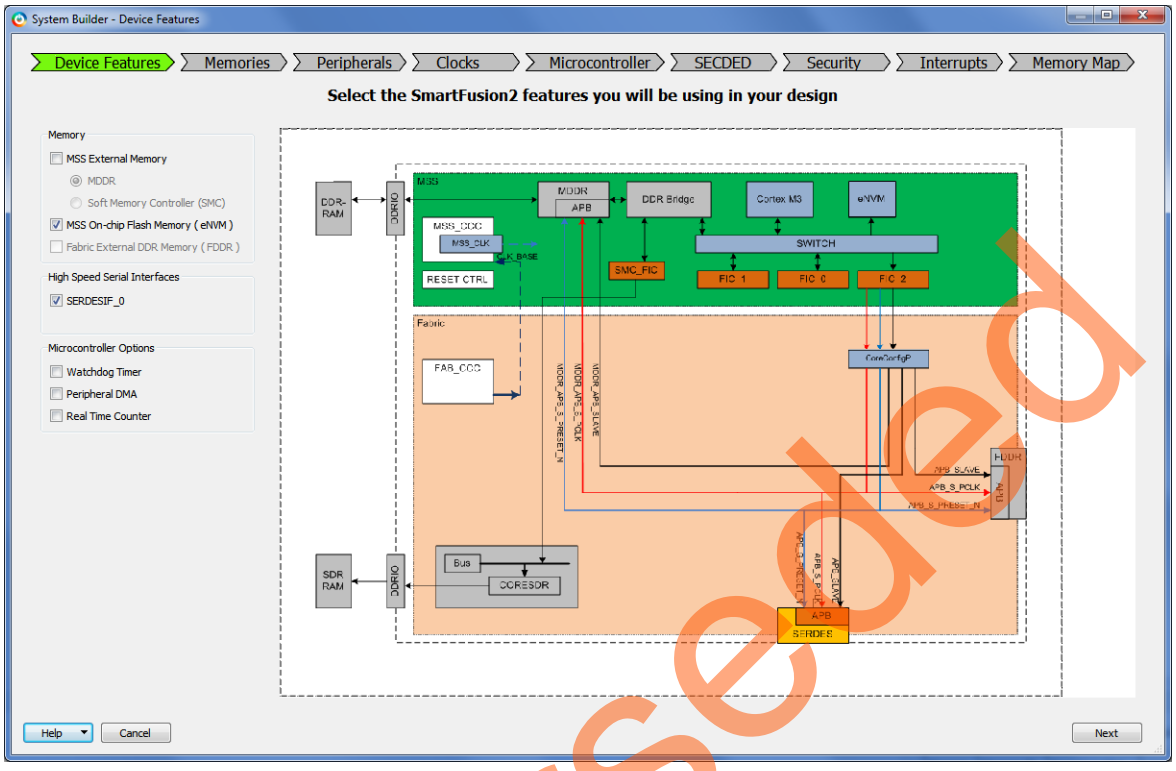


Figure 11 • **SmartFusion2 System Builder Configurator**

13. Click **Next**. The **System Builder – Memories** page is displayed.

14. Click **Next**. The **System Builder – Peripherals** page is displayed. Drag the **Fabric AMBA Master** to **MSS_FIC_0 – Fabric Master Subsystem**, as shown in Figure 12. It enables the MSS FIC_0 slave interface.

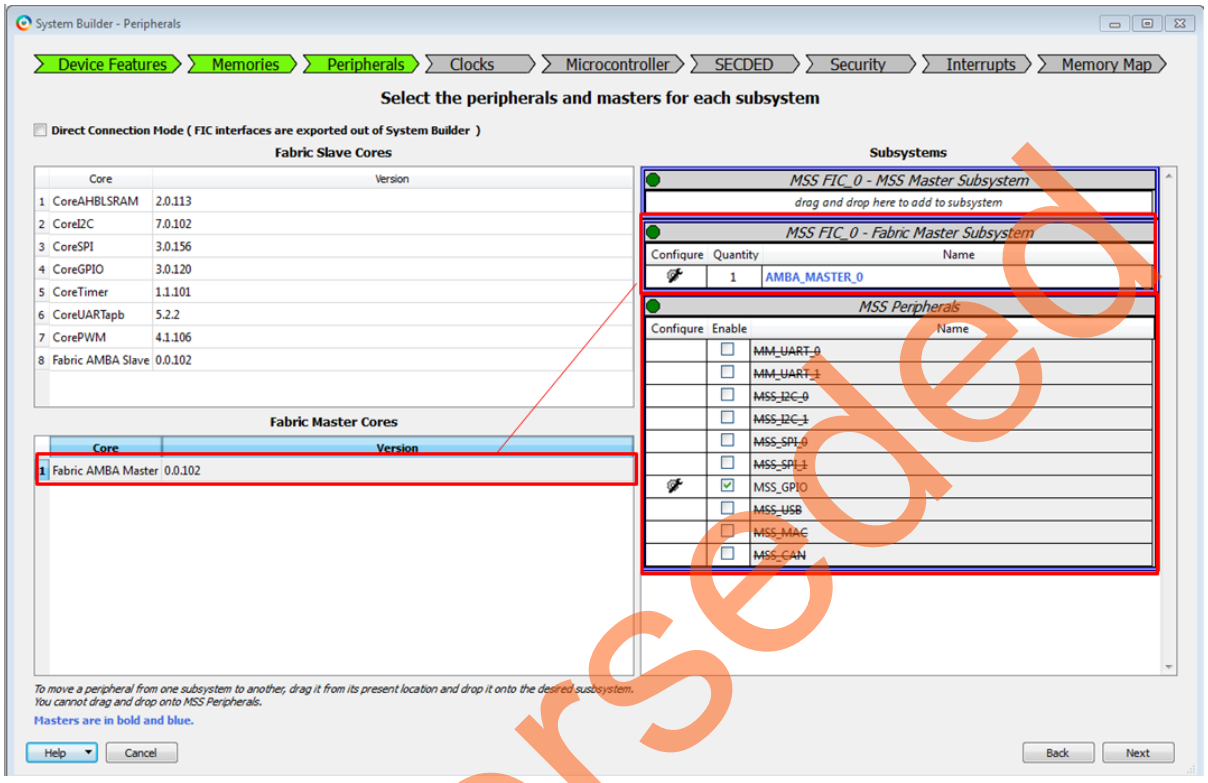


***Figure 12* • System Builder – Peripherals Page**

15. Disable the MSS Peripherals except MSS_GPIO. The **System Builder – Peripherals** page is displayed, as shown in Figure 13 on page 14. Configure **MSS_FIC_0 – Fabric Master Subsystem** for AHB-Lite by clicking on the AMBA_MASTER_0 configurator button highlighted in Figure 13 on page 14. This displays a drop-down list, as shown in Figure 14 on page 14.

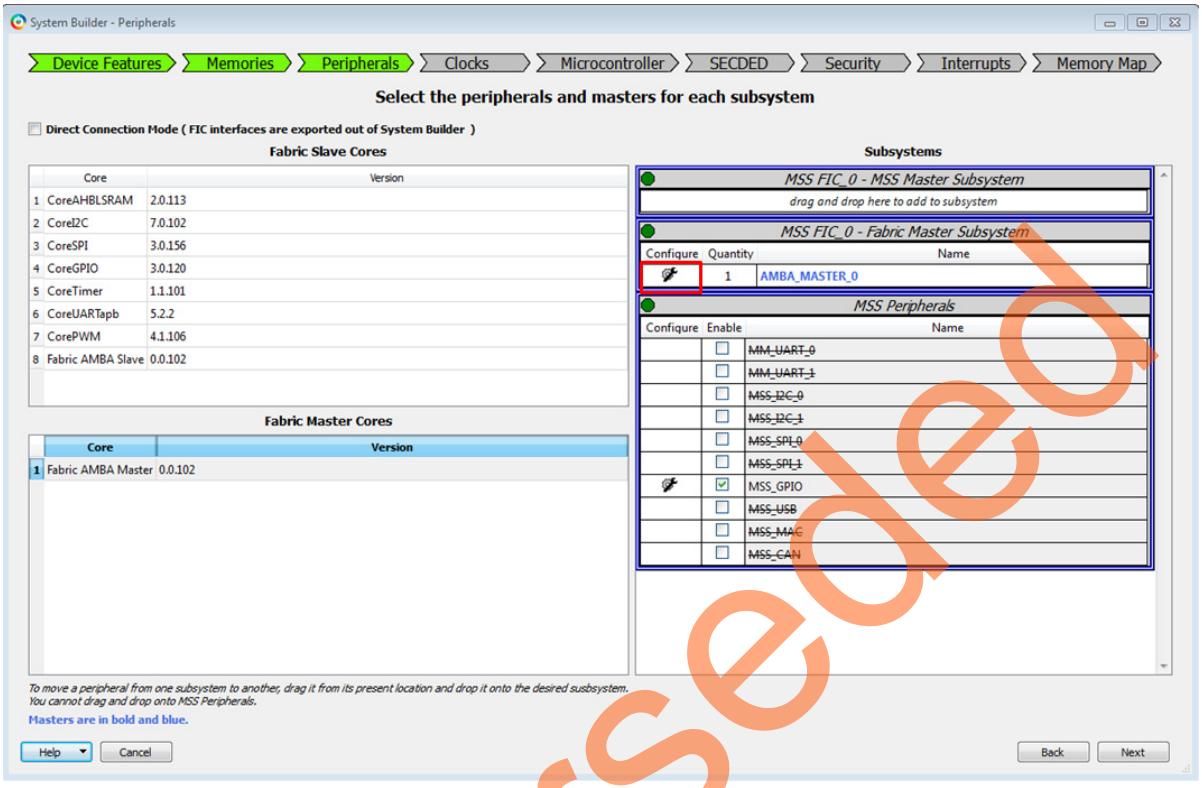


***Figure 13* • System Builder – Peripherals Page**

16. Select **AHBLite** from the drop-down list, as shown in Figure 14.

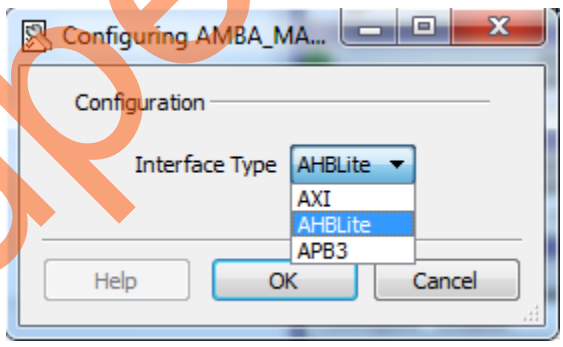


***Figure 14* • Configuring AMBA Master**

17. Configure MSS_GPIO by clicking **MSS_GPIO Configure**, as shown in Figure 15.

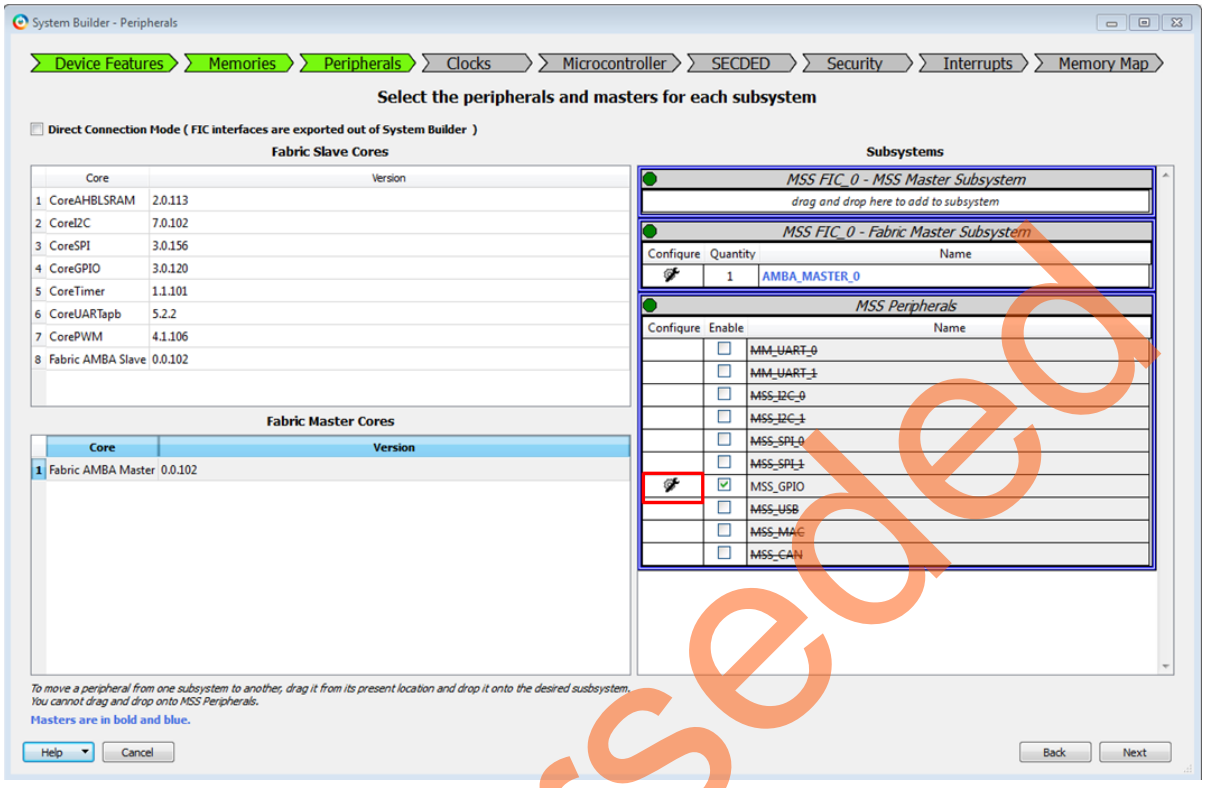


***Figure 15* • System Builder – Peripherals Page**

18. Double-click **MSS_GPIO** configuration button, as shown in Figure 15 and configure:
    - GPIO_0 to GPIO_7 as outputs and their connectivity to FABRIC_A to connect with LEDs
    - GPIO_8 to GPIO_11 as inputs and their connectivity to FABRIC_A, to connect with DIP switches

    This design requires configuring GPIO_0 to GPIO_7 to drive LED_1 to LED_8 on the SmartFusion2 Security Evaluation Kit, and GPIO_8 to GPIO_11 to connect DIP1 to DIP4. These signals are routed through the fabric to the I/O pins.

Figure 16 shows the MSS GPIO Configurator.

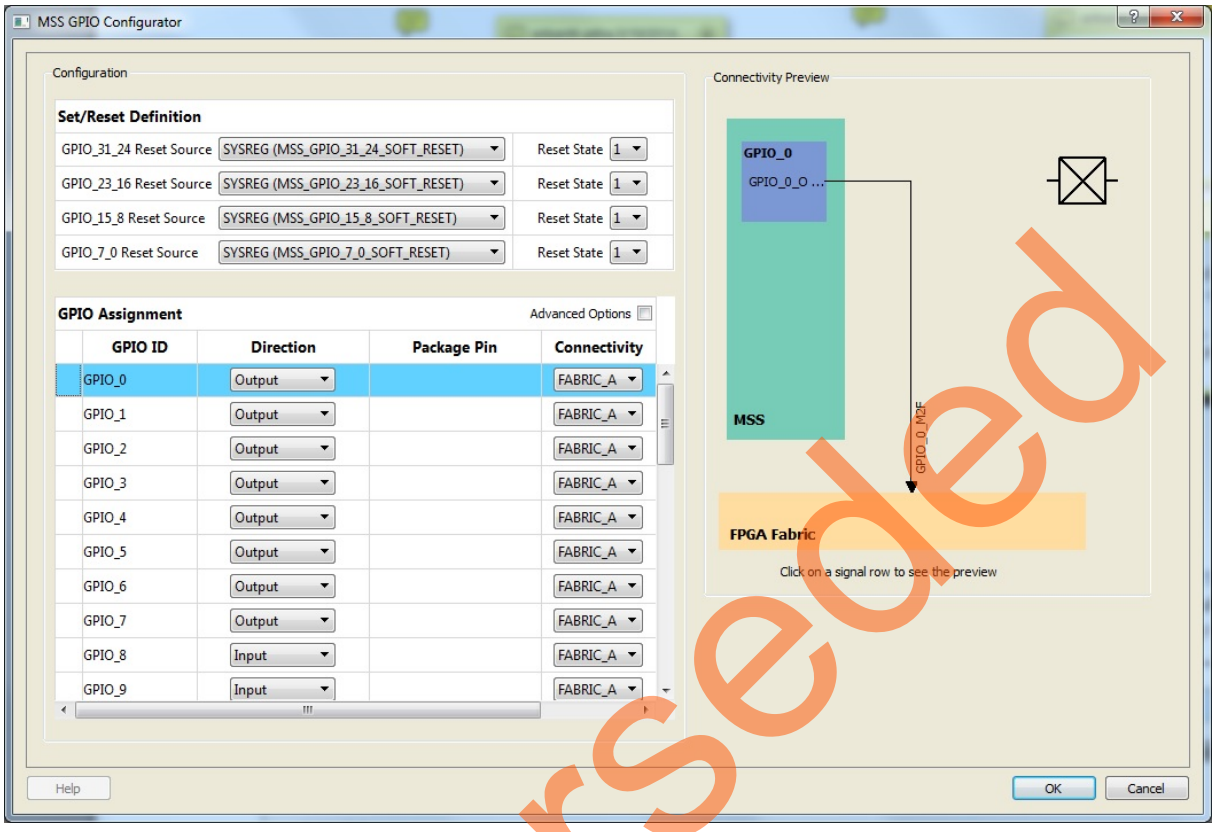


Figure 16 • **GPIO Configuration**

19. Click **OK** on MSS GPIO Configurator.

20. Click **Next**. The **System Builder – Clock** page is displayed, as shown in Figure 17. Change the configuration of **System Clock** from 100 MHz to 50 MHz. The dedicated input pad is connected to on board 50 MHz oscillator. The M3_CLK is configured to 90 MHz by default.

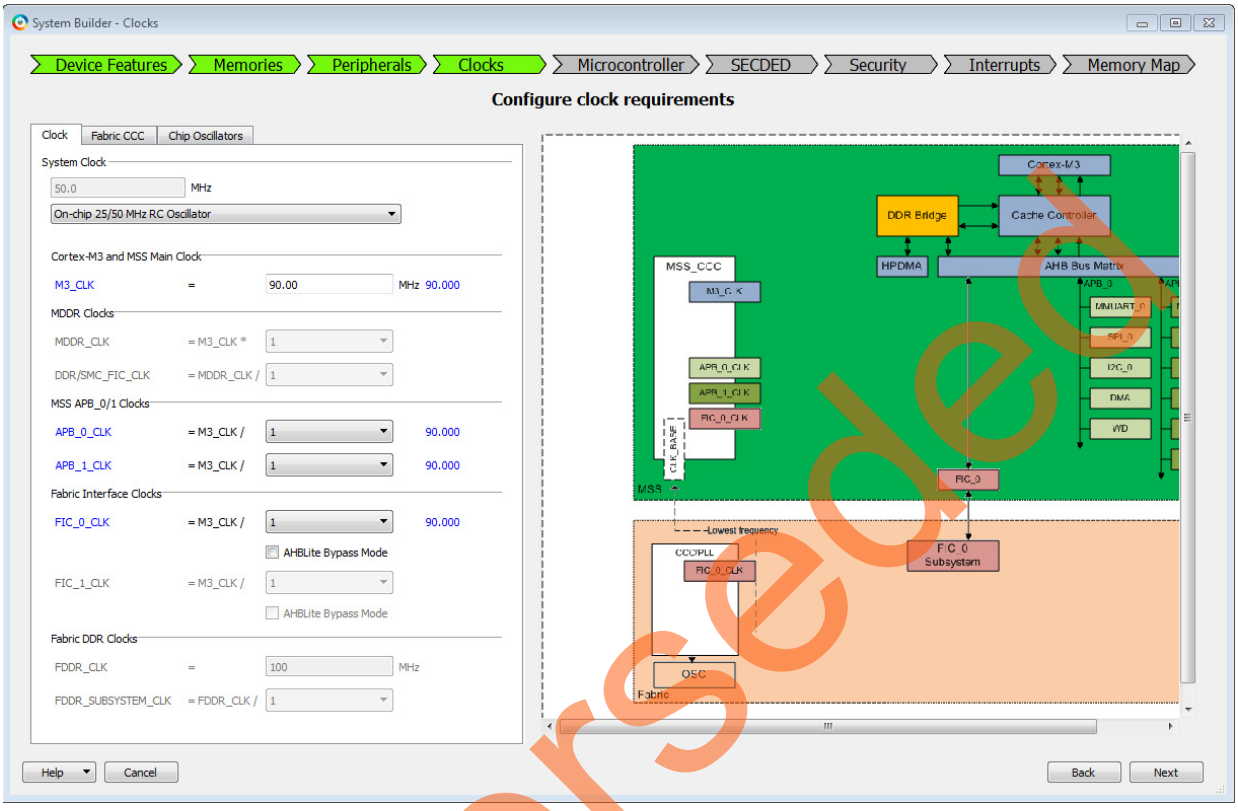


***Figure 17* • System Builder – Clock Page**

21. Click **Next**. The **System Builder - Microcontroller** page is displayed. Leave all the default selections.
22. Click **Next**. The **System Builder - SECDED** page is displayed. Do not change the default selections.
23. Click **Next**. The **System Builder - Security** page is displayed. Do not change the default selections.
24. Click **Next**. The **System Builder - Interrupts** page is displayed. Do not change the default selections.
25. Click **Next**. The **System Builder - Memory Map** page is displayed. Do not change the default selections.
26. Click **Finish**. The **System Builder** generates the system based on the selected options.

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

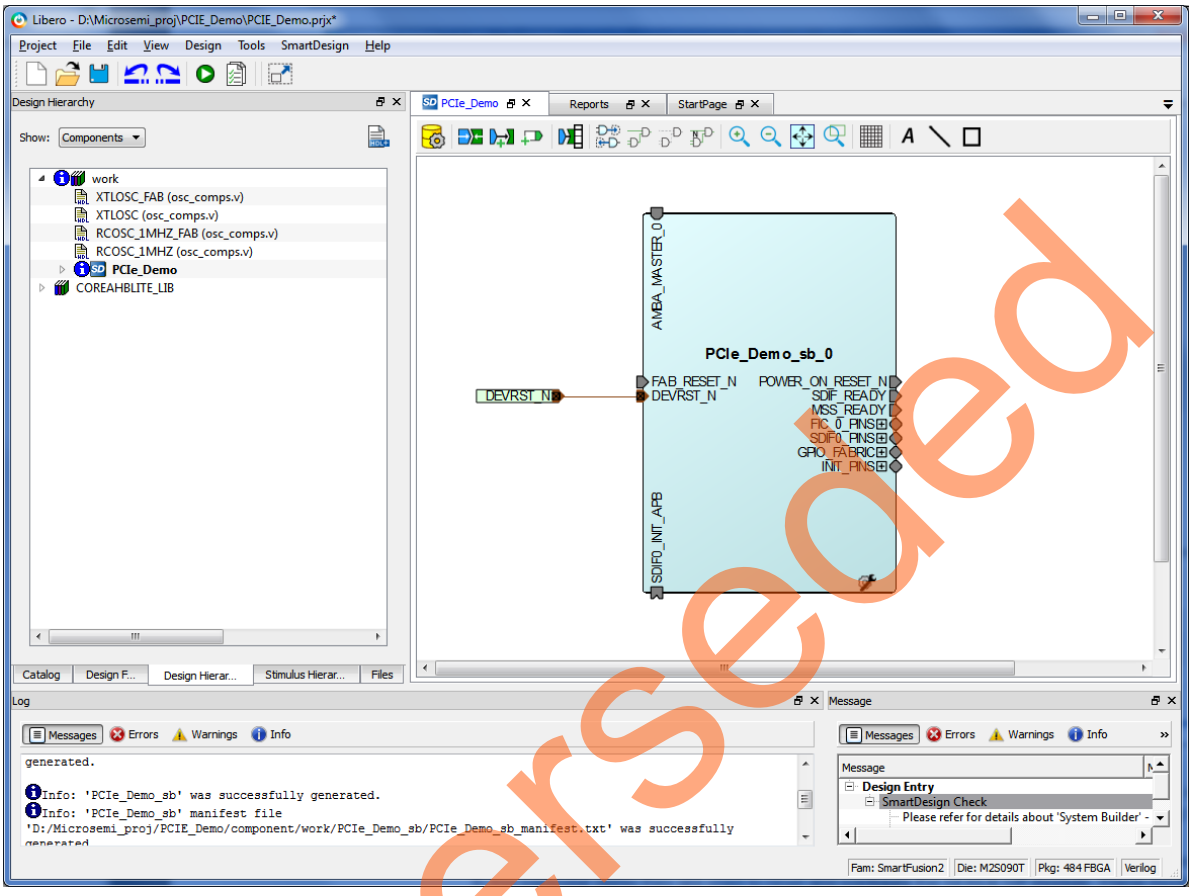


Figure 18 • **SmartFusion2 System Builder Generated System**

The two soft cores (CoreResetP and CoreConfigP) are automatically instantiated and connected by the System Builder. The block connections can be seen by opening the System Builder component in the SmartDesign canvas.

Note: CoreResetP and CoreConfigP are responsible for the reset and configuration of ASIC peripherals. In this particular demo they are used to reset and configure the SERDESIF module. These modules are included in the System Builder generated component when an ASIC peripheral is selected.

Instantiating SERDESIF Component in PCIe_Demo SmartDesign

The Libero SoC Catalog provides IP cores that can be easily dragged-and-dropped into the SmartDesign Canvas workspace. Many of these IPs are free to use while several require a license agreement. The SERDESIF module that supports the PCIe embedded interface is included in the catalog. To instantiate the SERDESIF component in the **PCIe_Demo** SmartDesign, expand the **Peripherals** category in the Libero SoC Catalog.

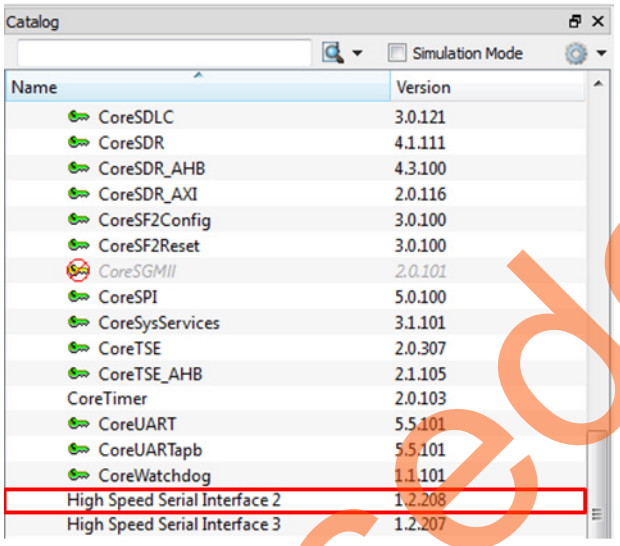


Figure 19 **• IP Catalog**

1. Drag the **High Speed Serial Interface2** onto the **PCIe_Demo SmartDesign** canvas. If the component appears shadowed in the Vault, right-click the name and select **Download**.
2. Double-click the **SERDES_IF2_0** component in the SmartDesign canvas to open the **SERDES** configurator. Configure the SERDES with the following settings, as shown in Figure 20:
   - Select SERDESIF_0
   - Simulation Level: BFM PCIe
   - Protocol1: Number of Lanes: x1
   - Protocol1: Type: PCIe
   - CLK_BASE Frequency (MHz): 90
   - Lane Configuration: Speed: 5.0 Gbps(Gen2)
   - Lane Configuration:
   - Reference Clock Source: REFCLK0 (Differential)

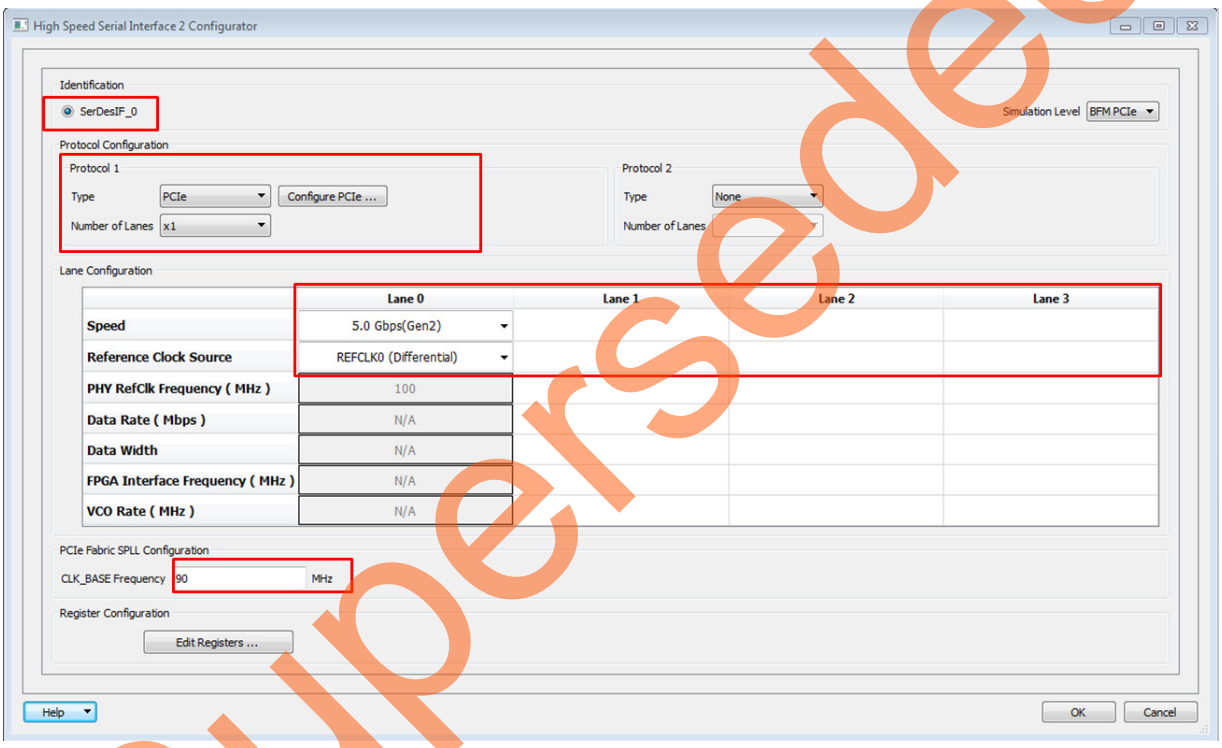


***Figure 20* • SERDES Configurator**

3. Click **Configure PCIe** in Protocol1, as shown in Figure 20 on page 20. The following settings are made in the Configuration tab, as shown in Figure 21 on page 21.
   – Fabric Interface (AXI/AHBLite)
     - Bus: select as AHBLite from the drop-down list
   – Base Address Registers
     - BAR 0 Width: 32-bit, Size: 1 MB (to access MSS Peripheral address space)
     - BAR 1 Width: 32-bit, Size: 64 KB (to access eSRAM memory)
   – Identification Registers
     - Device ID: 0x11AA (MicroSemi ID)
     - Subsystem Vendor ID: 0x11AA (MicroSemi ID)

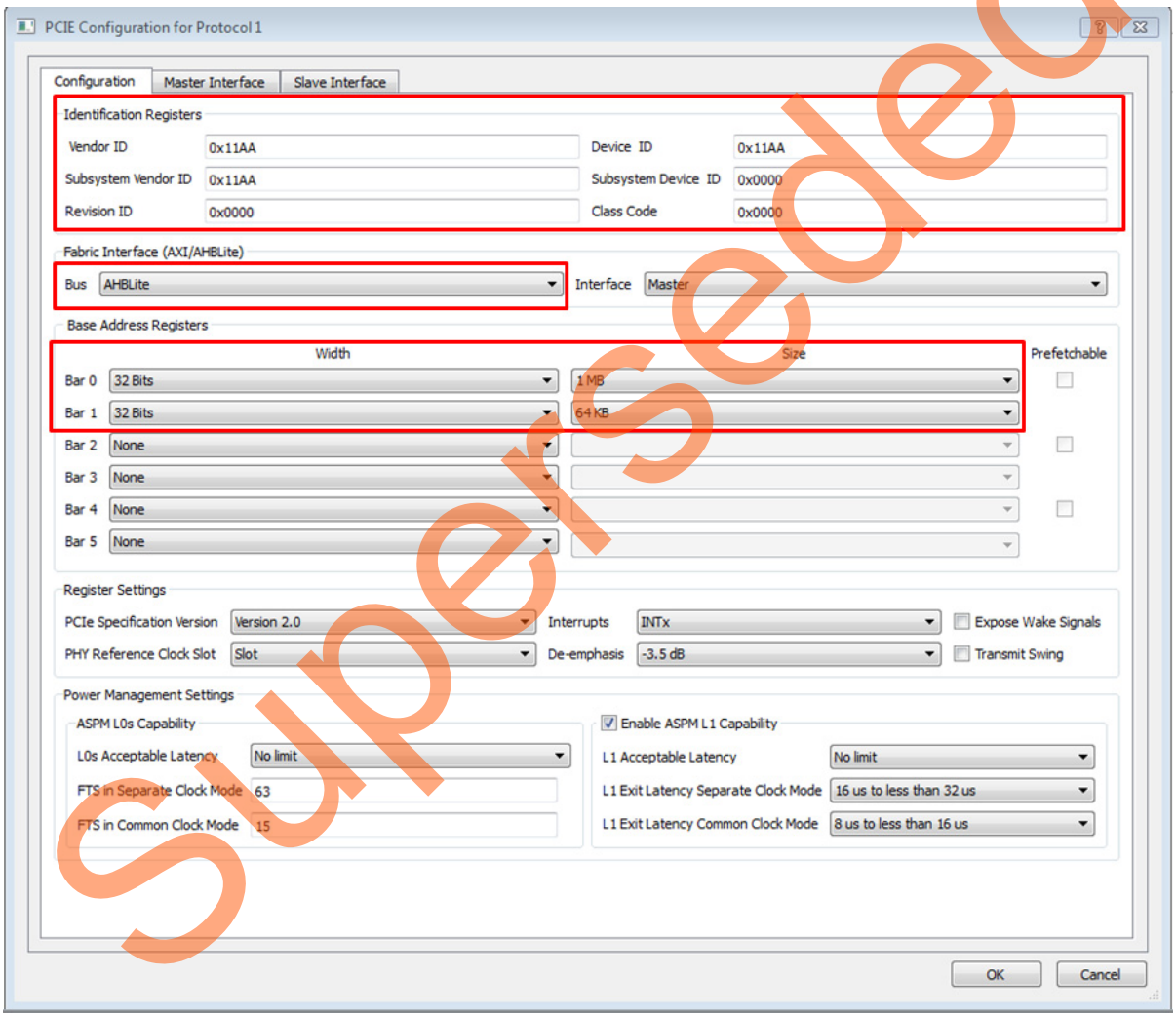


***Figure 21* • PCIe Configuration for Protocol 1**

4. Click the **Master Interface** tab to configure the PCIe master windows. The PCIe AXI master windows are used to translate the PCIe address domain to the local device address domain. In this demo the PCIe AXI master windows are used to translate the address of BAR0 and BAR1 to CoreGPIO address and COREAHBLSRAM address. Make settings as shown in Figure 22.
   – Select Window 0 and configure following settings:
     - Size: Select as 1MB from the drop-down list
     - PCIe BAR: Select as Bar0 from the drop-down list
     - Local Address: Enter values as 0x40000 to translate the BAR0 address space to CoreGPIO address (0x4000_0000)
   – Select Window 1 and configure following settings
     - Size: Select as 64KB from the drop-down list
     - PCIe BAR: Select as Bar1 from the drop-down list
     - Local Address: Enter values as 0x20000 to translate the BAR1 address space to COREAHBLSRAM address (0x2000_0000)

   For more information on PCIe address translation, refer to the **Address Translation on the AXI Master Interface** section of the *UG0447: IGLOO2 and SmartFusion2 High Speed Serial Interfaces User Guide.*

.

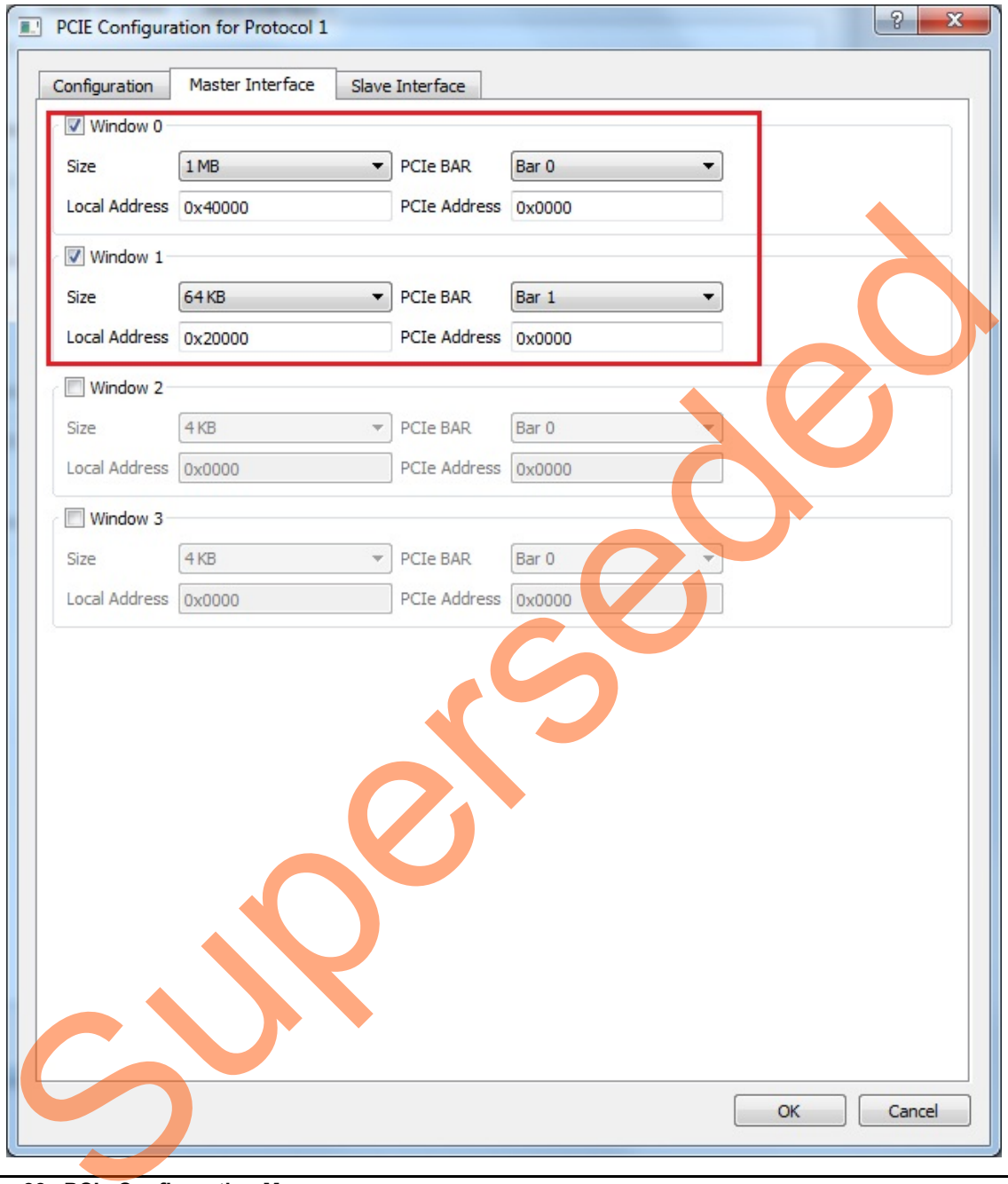


Figure 22 • **PCIe Configuration Memory**

5. Click **OK** to close PCIe Configuration window.
6. Click **OK** to save and close the High Speed Serial Interface Configurator.

Instantiating Debounce Logic in PCIe_Demo SmartDesign

1. The demo provides a push button on the SmartFusion2 Security Evaluation Kit to send an interrupt to the Host PC. This push button generates switch bounce that causes multiple interrupts to PCIe. Debounce logic is required to avoid the switch bounce.
2. To add the debounce logic to the PCIe demo design, click **File > Import > HDL Source files**.
3. Browse to the Debounce.v or Debounce.vhd file location in the design files folder: *M2S90_PCIE_Control_DEMO_DF/Source Files*. Figure 23 shows the DEBOUNCE component in the Design Hierarchy window.

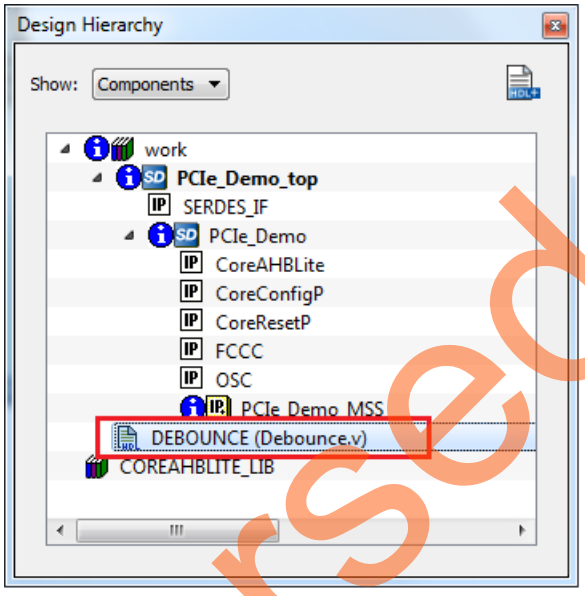


***Figure 23* • DEBOUNCE Component in Design Hierarchy Window**

4. Click the **PCIe_Demo** tab and drag the **DEBOUNCE** component from the **Design Hierarchy** into the **PCIe_Demo SmartDesign** canvas, as shown in Figure 24. A SmartDesign symbol for the Verilog HDL file is automatically generated.

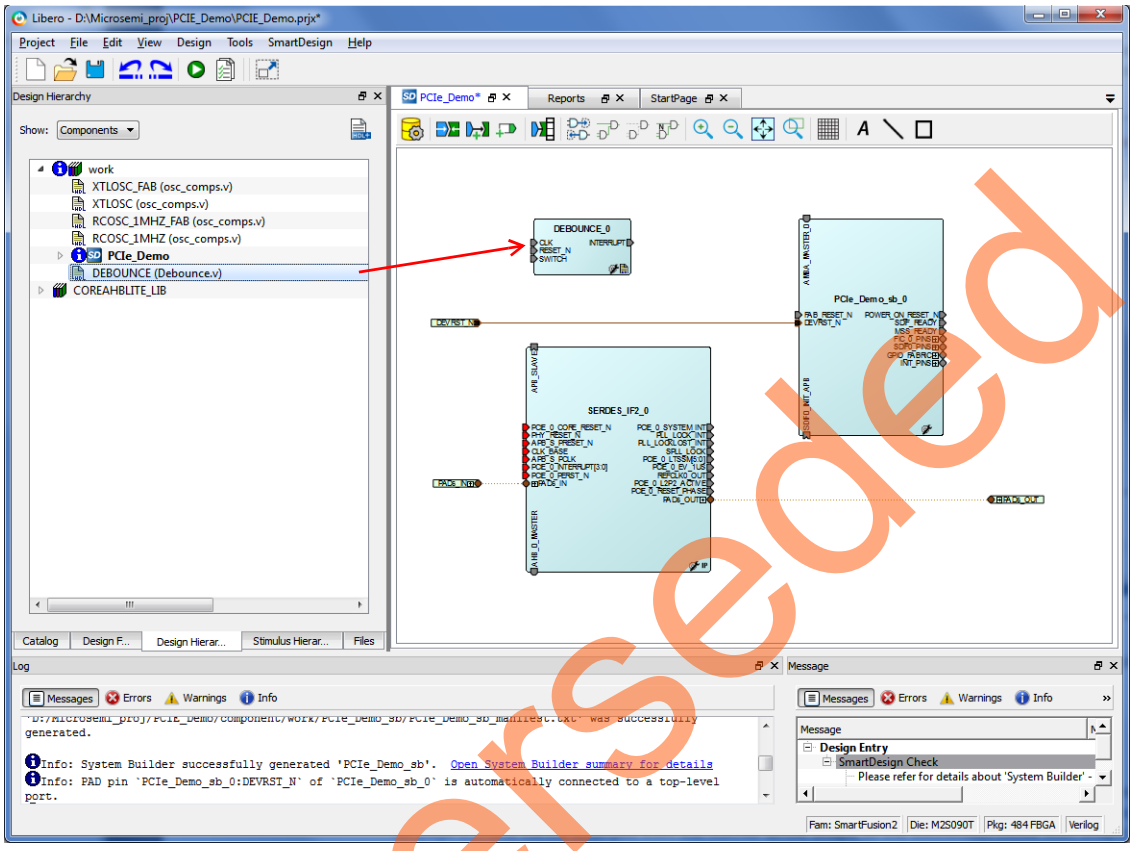


Figure 24 • **DEBOUNCE Component in Design Hierarchy**

The PCIe_Demo is displayed, as shown in Figure 26 on page 27. Connect the pins of all the blocks as described in the "Connecting Components in PCIe_Demo SmartDesign" section.

Connecting Components in PCIe_Demo SmartDesign

There are three methods for connecting components in PCIe_Demo SmartDesign.

The first method is by using the **Connection Mode** option. To use this method, change the SmartDesign to connection mode by clicking **Connection Mode** on the SmartDesign window, as shown in Figure 26 on page 27. The cursor changes from the normal arrow shape to the connection mode icon shape. To make a connection in this mode, click on the first pin and drag-drop to the second pin that you want to connect.

The second method is by selecting the pins to be connected together and selecting **Connect** from the context menu. To select multiple pins to be connected together, press down the **CTRL** key while selecting the pins. Right-click the input source signal and select **Connect** to connect all the signals together. Similarly, select the input source signal, right-click it, and select **Disconnect** to disconnect the signals already connected.

The third method is by using the **Quick Connect** option. To use this method, change the SmartDesign to quick connect mode by clicking **Quick Connect** mode on the SmartDesign window, as shown in Figure 25. **Quick Connect** window opens. Find the Instance Pin that needs to be connected and click to select it. In Pins to Connect, find the pin that needs to be connected, right-click and choose **Connect**, as shown in Figure 25.

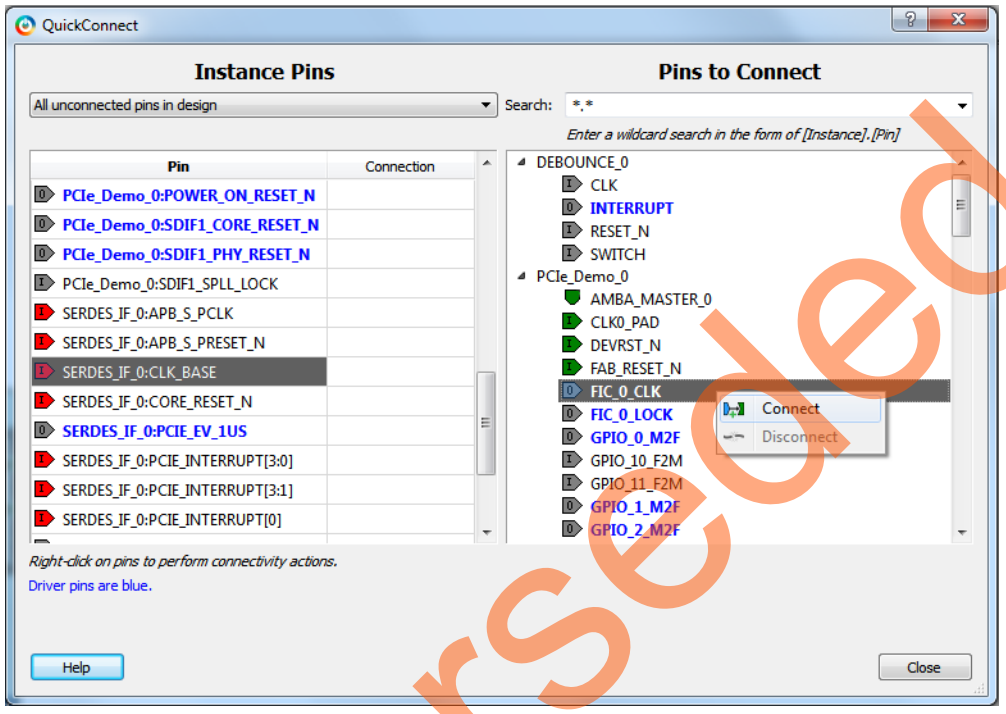


***Figure 25* • Quick Connect Window**

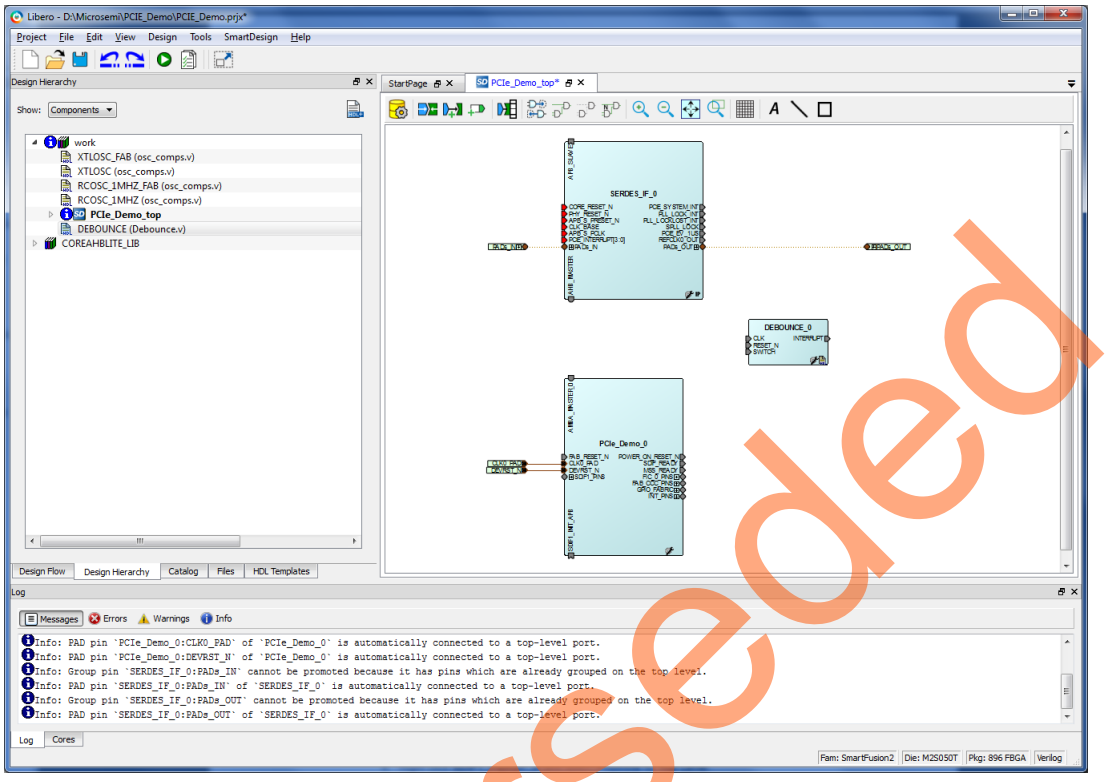


Figure 26 • **PCIe Demo Top in SmartDesign**

Use one of the three options and make the following connections:

1. Expand FIC_0_PINS of PCIe_Demo_sb_0 and make connections, as shown in Table 2.
2. Right-click **FIC_0_LOCK** and select **Mark Unused**

Table 2 • **FIC_0_PINS**

From PCIe_Demo_sb_0	To
FIC_0_CLK	CLK_BASE of SERDES_IF2_0
	CLK of DEBOUNCE_0

3. Expand SDIF0_PINS of PCIe_Demo_sb_0 and make connections, as shown in Table 3.

Table 3 • **SDIF0_PINS**

From PCIe_Demo_sb_0	To SERDES_IF2_0
SDIF0_PHY_RESET_N	PHY_RESET_N
SDIF0_0_CORE_RESET_N	PCIE_0_CORE_RESET_N
SDIF0_SPLL_LOCK	SPLL_LOCK

4. Right-click **SDIF0_1_CORE_RESET_N** and select **Mark Unused**.
5. Right-click **PCIE_0_PERST_N** and select **Promote to Top Level**.

6. Expand INIT_PINS of PCIe_Demo_sb_0 and make connections, as shown in Table 4.

Table 4 • **INIT_PINS**

From PCIe_Demo_sb_0	To SERDES_IF2_0
INIT_APB_S_PCLK	APB_S_PCLK
INIT_APB_S_PRESET_N	APB_S_PRESET_N

7. Right-click **INIT_DONE** and select **Mark Unused**.
8. Connect **MSS_READY** of **PCIe_Demo_sb_0** and **RESET_N** of **DEBOUNCE_0**.
9. Right-click **FAB_RESET_N** of **PCIe_Demo_sb_0** and select **Tie High**.
10. Right-click **GPIO_FABRIC** of **PCIe_Demo_sb_0** and select **Promote to Top Level**.
11. Right-click **POWER_ON_RESET_N** of **PCIe_Demo_sb_0** and select **Mark Unused**.
12. Right-click **SDIF_READY** of **PCIe_Demo_sb_0** and select **Mark Unused**.
13. Connect **AMBA_MASTER_0** of **PCIe_Demo_sb_0** and **AHB_MASTER** of **SERDES_IF2_0**.
14. Expand **FAB_CCC_PINS**, right-click **FAB_CCC_GL3** and select **Mark Unused**.
15. Connect **SDIF1_INIT_APB** of **PCIe_Demo_sb_0** and **APB_SLAVE** of **SERDES_IF2_0**.
16. Right-click the **SWITCH** of **DEBOUNCE_0** and select **Promote to Top Level**.
17. Select the following ports of **SERDES_IF2_0** by pressing down the **CTRL** key, right-click, and select **Mark Unused**.
    - PCIE_SYSTEM_INT
    - PLL_LOCK_INT
    - PLL_LOCKLOST_INT
    - PCIE_EV_1US
    - REFCLK0_OUT
    - PCIE_0_LTSSM[5:0]
    - PCIE_0_L2P2_ACTIVE
    - PCIE_0_RESET_PHASE

The PCIe supports four interrupts. This design uses only one interrupt out of four by connecting the unused interrupts to logic '0'. To connect unused interrupt pins to logic '0' split the interrupt pins to two groups. To do that right-click the **PCIE_INTERRUPT[3:0]** of **SERDES_IF2_0** and select **Edit Slices**. The Edit Slices window is displayed, as shown in Figure 27.

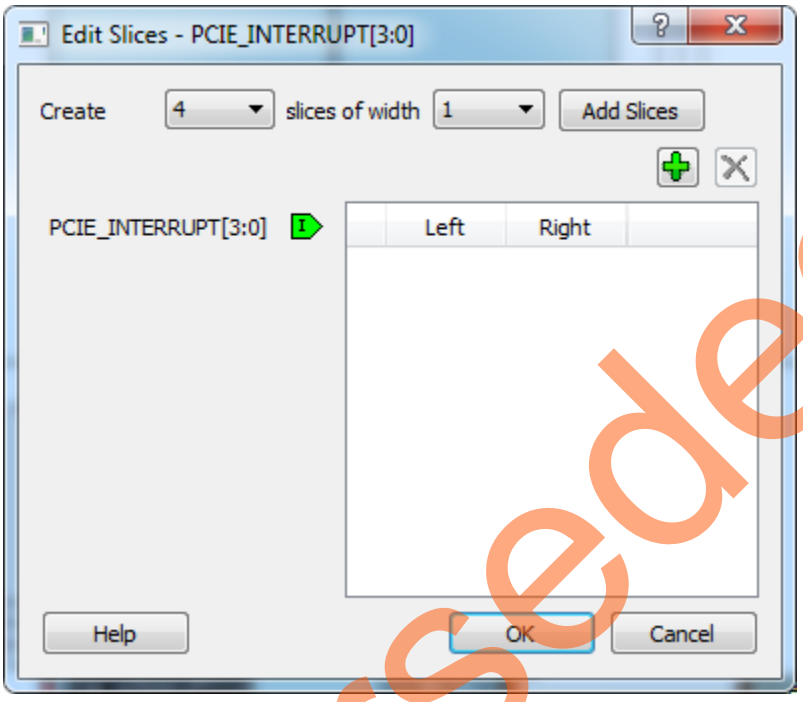


Figure 27 • **Edit Slices**

18. Click the + sign and create a slice with the Left index 0 and the Right index 0. Click + again to create a second slice with Left index 3 and Right index 1, as shown in Figure 28.

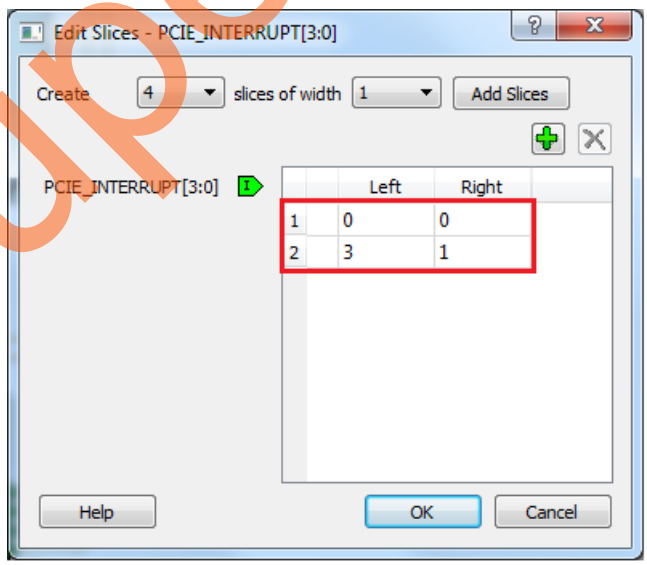


Figure 28 • **Edit Slices**

19. Expand **PCIE_INTERRUPT[3:0]**, right-click the **PCIE_INTERRUPT[3:1]**, and select **Tie low**.

20. Connect **INTERRUPT** of **DEBOUNCE_0** to the **PCIE_INTERRUPT[0]** of **SERDES_IF2_0**.
21. Click **Auto arrange instances** to arrange the instances and click **File > Save**. The PCIe_Demo is displayed, as shown in Figure 29.

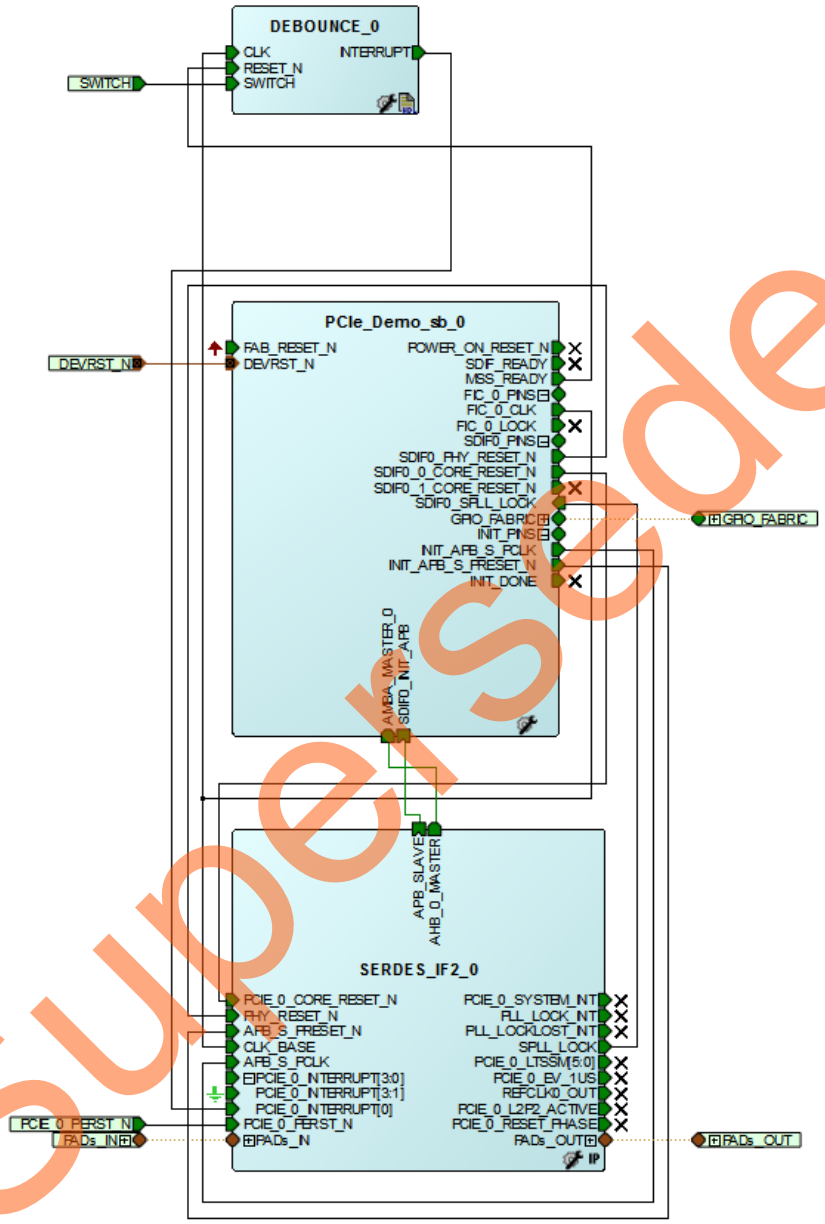


***Figure 29* • PCIe Demo Top Design**

22. Click the **PCIe_Demo** tab and click **Generate Component** icon, as shown in Figure 30.

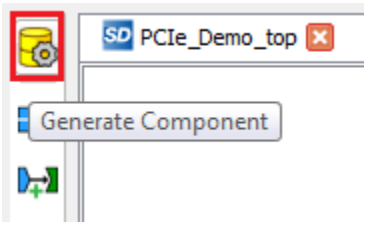


Figure 30 • **Generate Component**

The message "PCIe_Demo" was successfully generated is displayed in the Libero SoC log window, if the design is generated without any error. The log window is displayed on a successful component generation, as shown in Figure 31

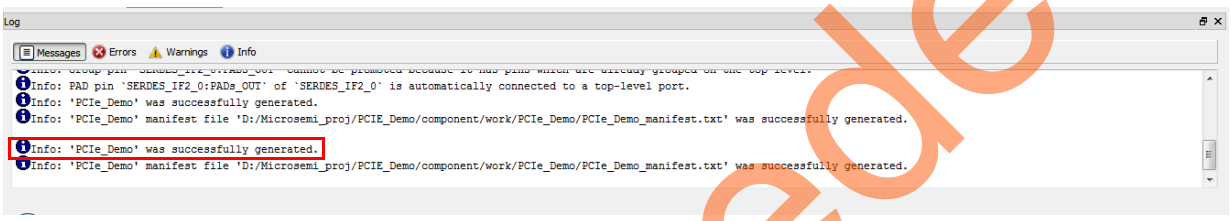


Figure 31 • **Log Window**

Configuring and Generating Firmware

The following steps describe how to configure and generate firmware.

1. Double click **Configure Firmware Cores** under **Handoff Design for Firmware Development** in Design Flow and clear all drivers except CMSIS, as shown in Figure 32.

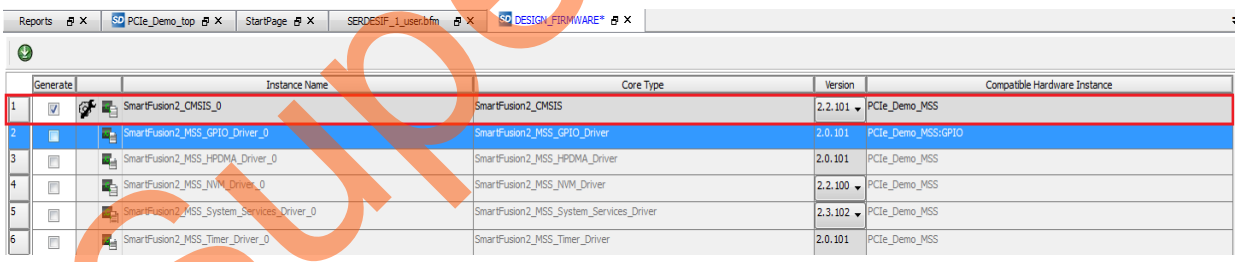


Figure 32 • **Configuring Firmware**

Click **Export Firmware**. The **Export Firmware** dialog box is displayed, as shown in Figure 33.

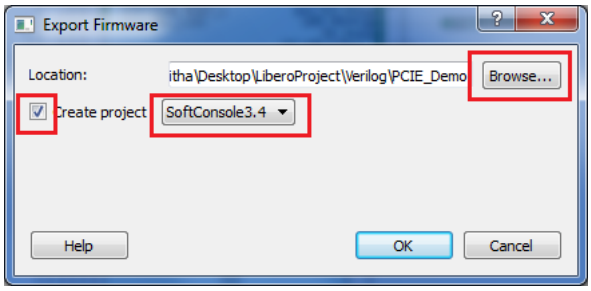


Figure 33 • **Export Firmware Dialog Box**

2. Browse the **Location** to export the firmware project.
3. Select the **Create project** check box.
4. Select **SoftConsole3.4** from the drop down list.
5. Click **OK**. The successful firmware generation window is displayed.
6. Click **OK**. The log window is displayed, as shown in Figure 34.

Figure 34 • **Log Window**

Step 2: Creating an eNVM Client

The HDL and logical design portion of the demo is now complete. The following sections describe the creation of the Cortex-M3 firmware used to initialize the MSS and SERDESIF.

The eNVM client has to be uploaded with the firmware application to initialize the SERDESIF through **CoreConfigP**. The Cortex-M3 processor executes the code in the eNVM after the SmartFusion2 device has been reset. In this design the eNVM client is created with the firmware application code to initialize the SERDESIF.

The following steps describe how to create an eNVM Client:

1. To build the firmware eNVM client, invoke the standalone SoftConsole IDE. The **SoftConsoleIDE Project Workspace** window is displayed, as shown in Figure 35.

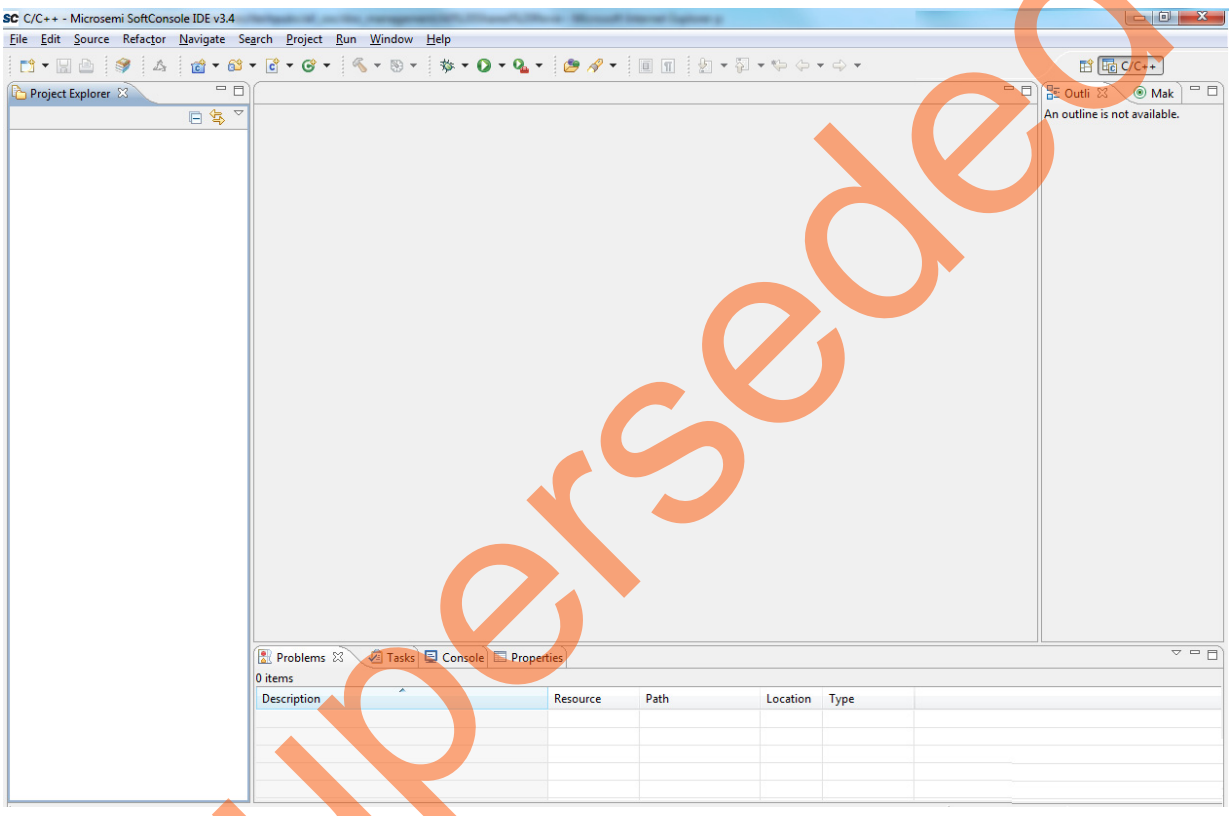


Figure 35 • SoftConsole IDE Project Workspace

2. Import the existing project into workspace, as shown in Figure 36.

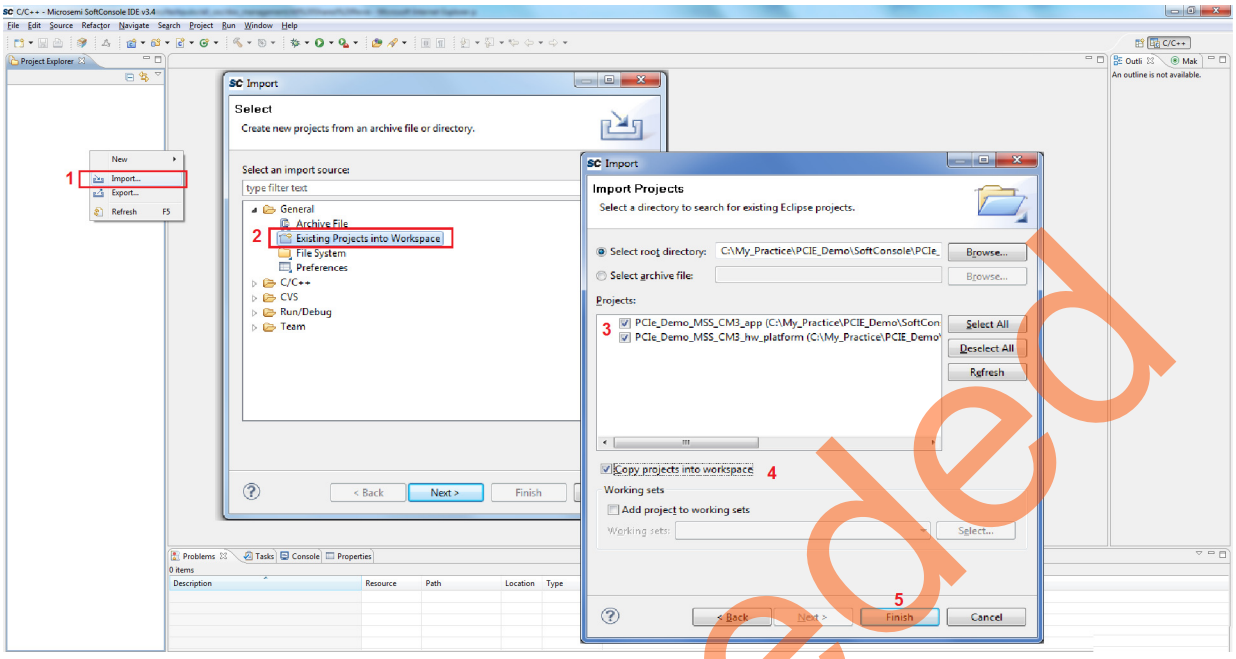


***Figure 36* • Importing Existing Project into Workspace**

3. Right-click **Project Explorer** tab on the left pane and select **Import…**. The **Import** dialog box is displayed.
4. Select **Existing Project into Workspace** under **General** folder and click **Next**. The **Import Projects** dialog box is displayed.
5. Click **Browse** to navigate to the SoftConsole project folder.
6. Select **PCIe_Demo_sb_MSS_CM3_app** and **PCIe_Demo_sb_MSS_CM3_hw_platform** check boxes under **Projects**.
7. Select **Copy projects into workspace** check box.

8. Click **Finish**. The **SoftConsole Workspace** window is displayed, as shown in Figure 37.

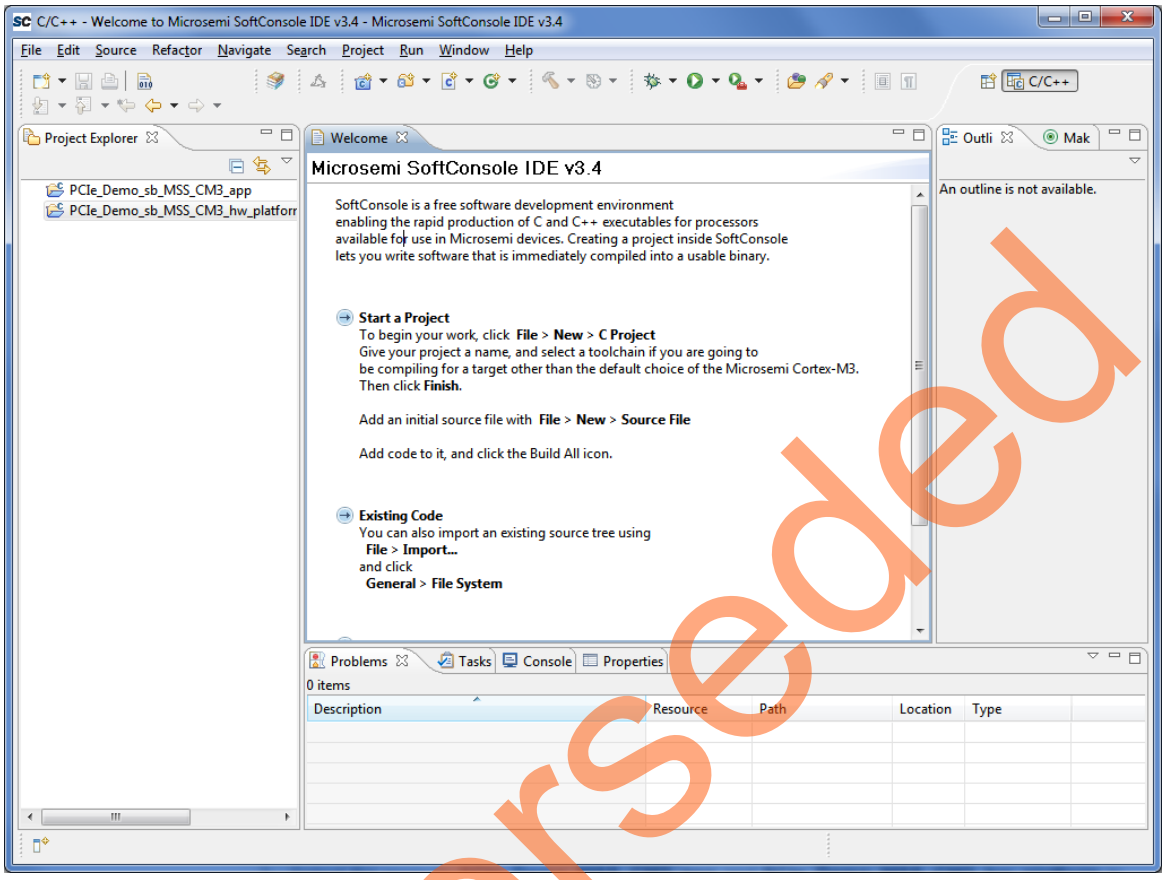


Figure 37 • **SoftConsole Workspace**

9. Select the projects **PCIe_Demo_sb_MSS_CM3_app** and **PCIe_Demo_sb_MSS_CM3_hw_platform** in the Project Explorer by using **CTRL** key.

10. Right-click and select **Build Configurations > Set Active > Release**, as shown in Figure 38.

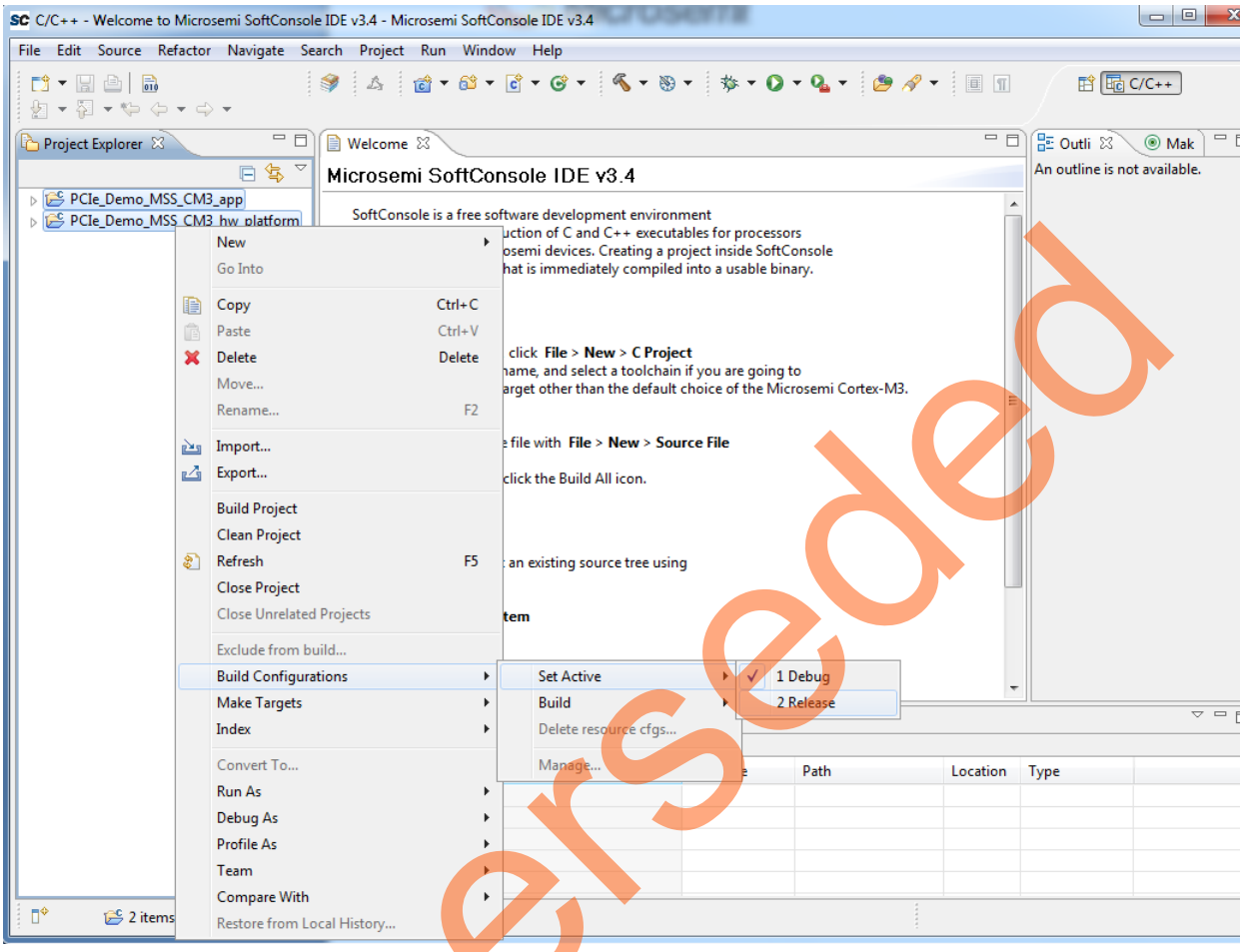


***Figure 38* • Release Mode Option**

11. Select **PCIe_Demo_sb_MSS_CM3_app**. Right-click and select **Properties**, as shown in Figure 39.

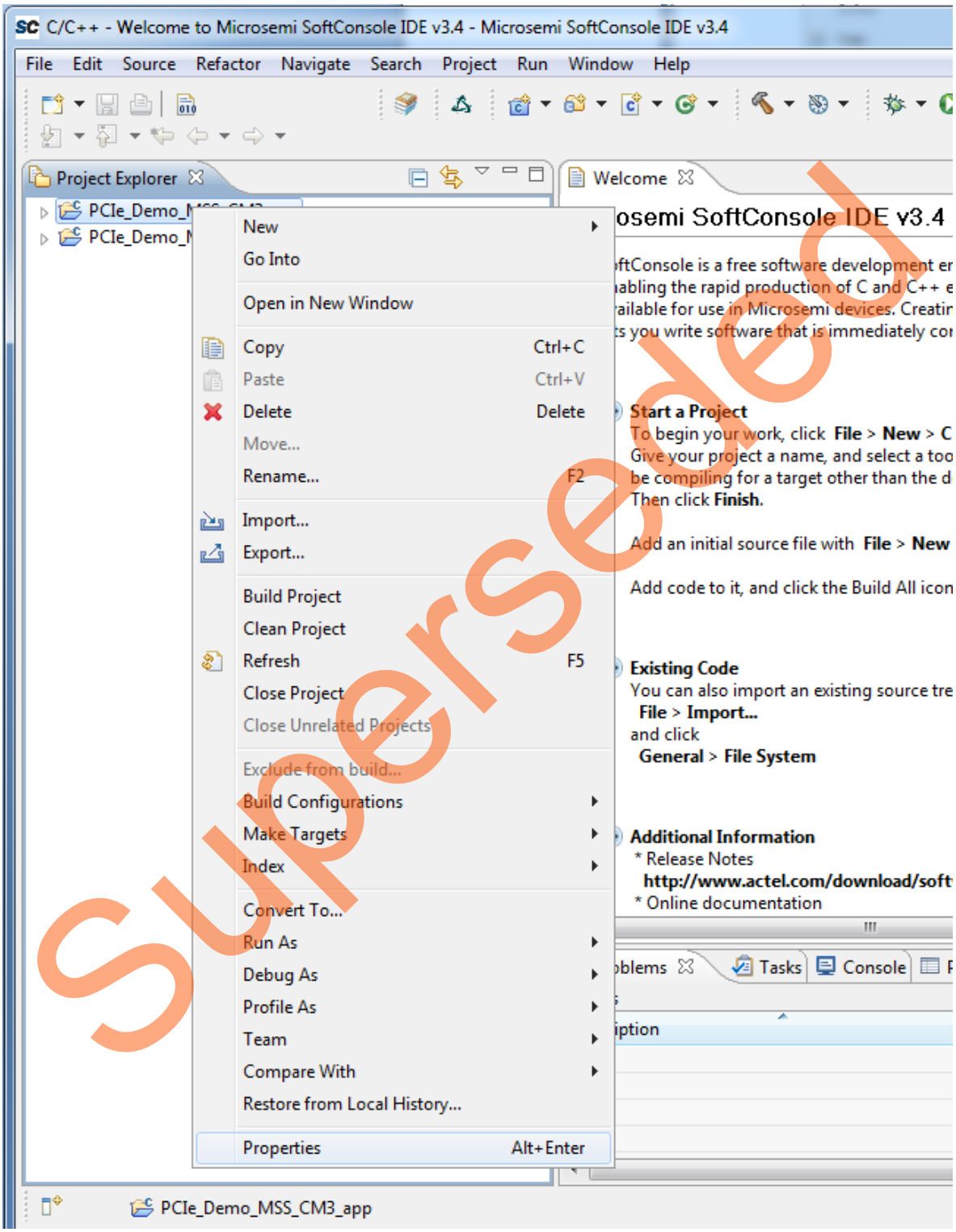


***Figure 39* • Properties Option**

The **Properties for PCIe_Demo_sb_MSS_CM3_app** window is displayed, as shown in Figure 40.

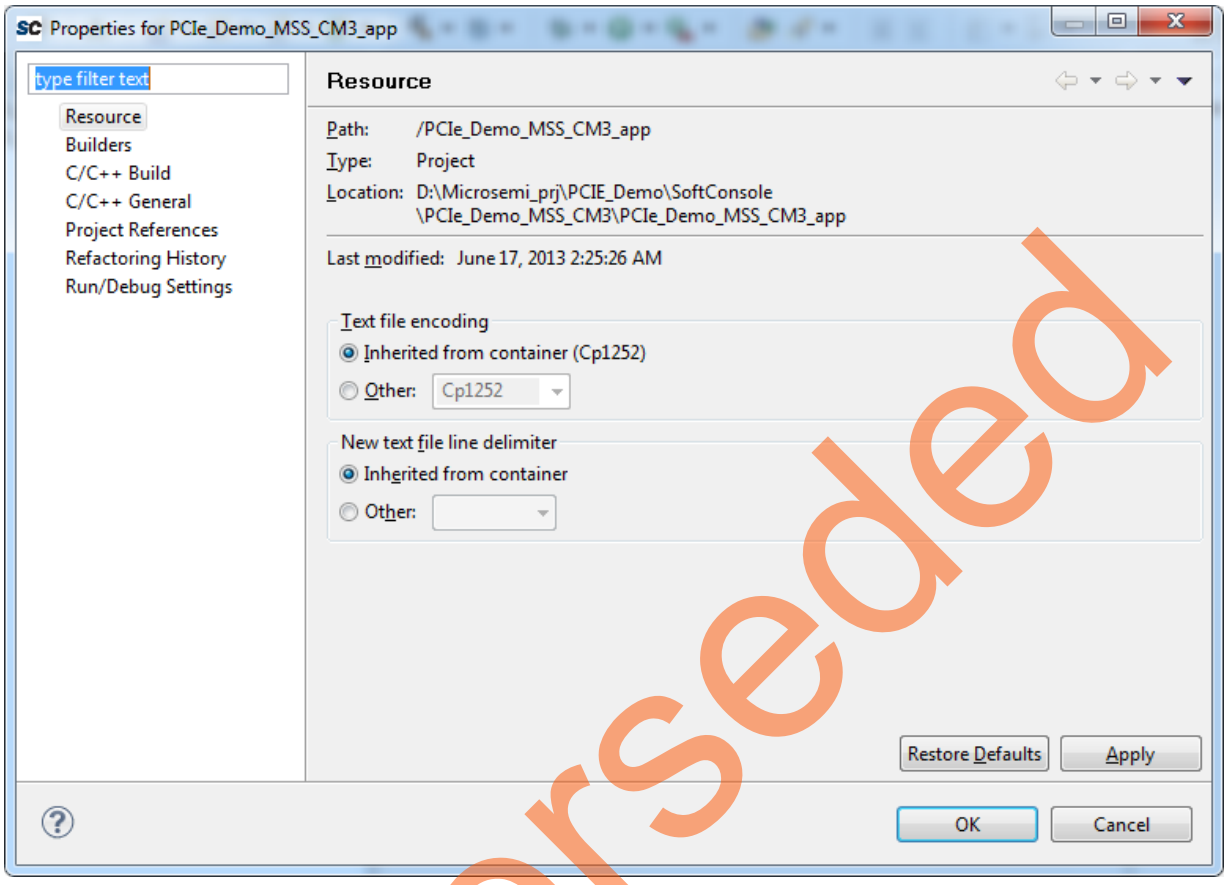


***Figure 40* • Properties Window**

12. In the **Properties for PCIe_Demo_sb_MSS_CM3_app** window, expand the **C/C++ Build** option and select **Settings**.

13. Select **Miscellaneous** and provide the release mode linker script file to the linker by changing the 'Linker flags' field to "*-T../../PCIe_Demo_sb_MSS_CM3_hw_platform/CMSIS/startup_gcc/production-smartfusion2-execute-in-place.ld*", as shown in Figure 41.

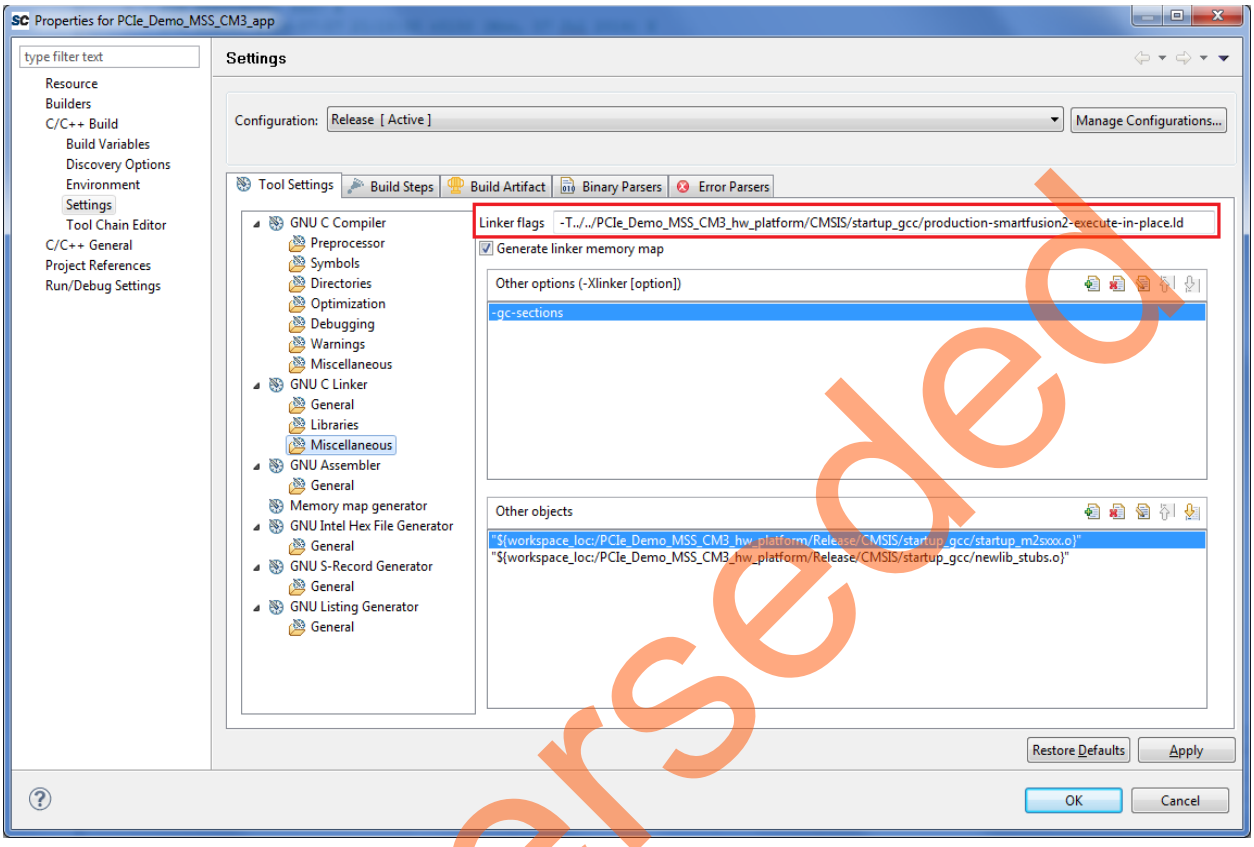


***Figure 41* • LD File Option**

14. Click **OK** to close the **Properties for PCIe_Demo_sb_MSS_CM3_app** window.
15. To clean and build the project, select **Project > Clean**, as shown in Figure 42.

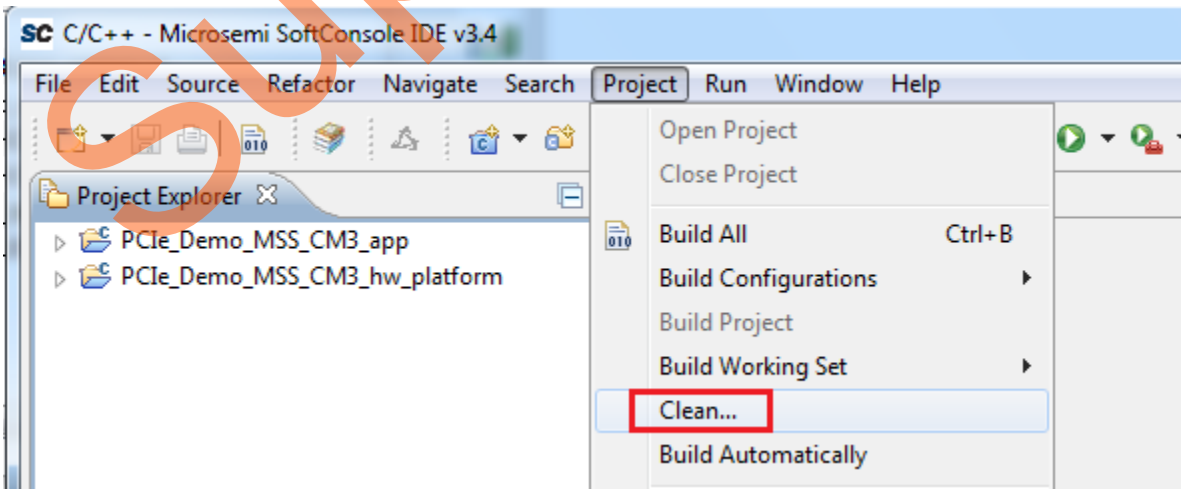


***Figure 42* • Building SoftConsole Project**

16. The **Clean** window is displayed. Click **OK** to build the SoftConsole projects, as shown in Figure 43.

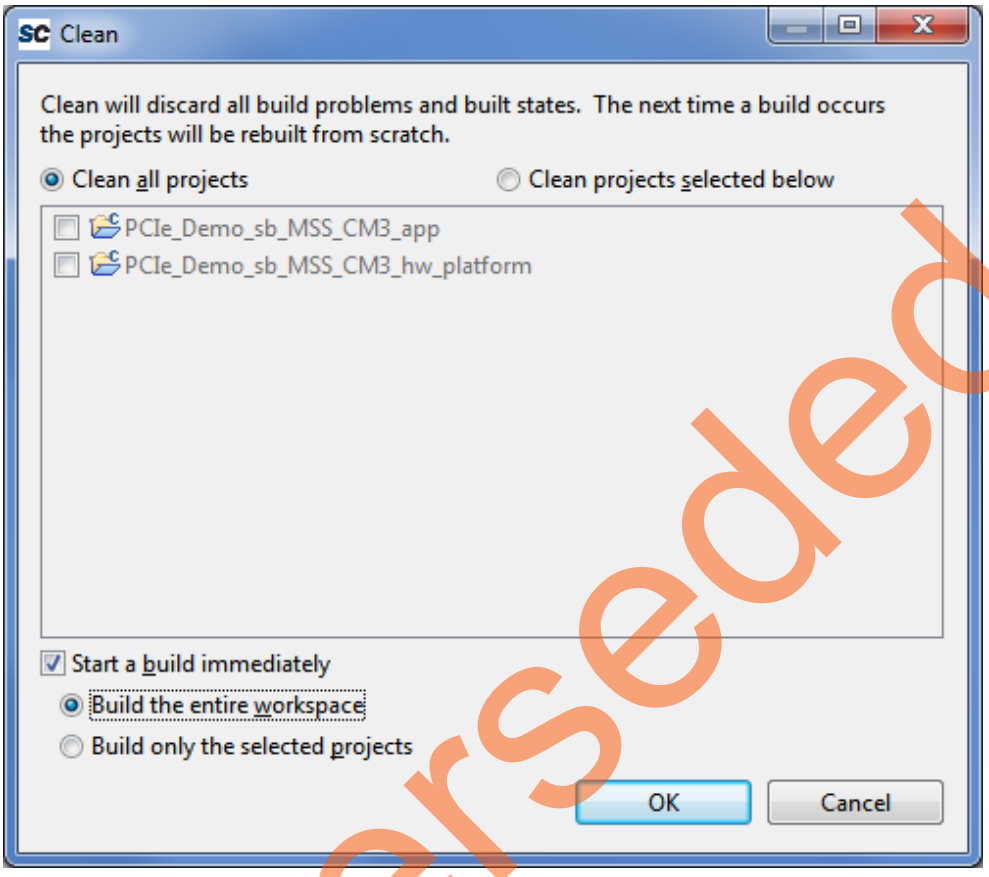


***Figure 43* • Clean and Build SoftConsole Projects**

17. The SoftConsole creates a hex file in the **Release** folder under the **PCIe_Demo_sb_MSS_CM3_app** project, as shown in Figure 44.

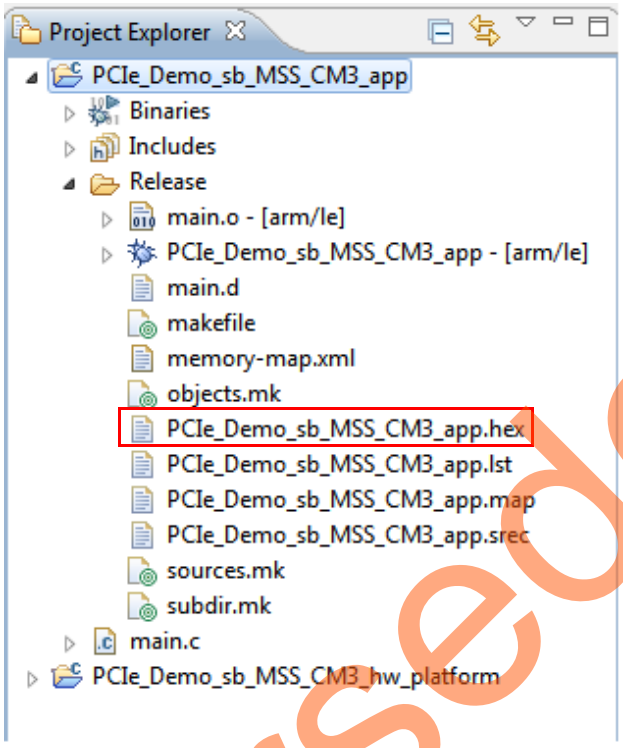


Figure 44 • **Generated Hex File**

18. Close the **SoftConsole** project window.
19. Open the Libero project and **PCIe_Demo** tab.
    Double-click **PCIe_Demo_sb_0** and go to **System Builder - Memories** tab to add the eNVM data storage client.

The eNVM configurator window is displayed, as shown in Figure 45.

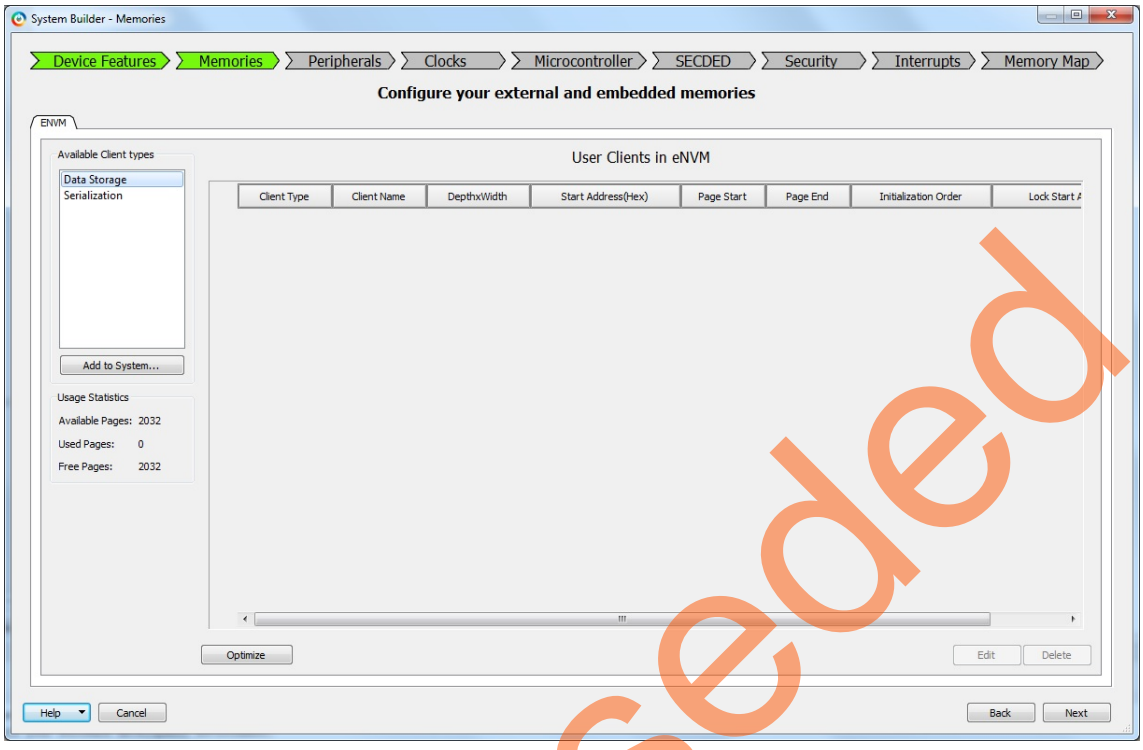


Figure 45 • **System Builder - Memory eNVM**

20. Select **Data Storage** under the **Available Client types** tab and click **Add to System**. The **Add Data Storage Client** window is displayed, as shown in Figure 46.

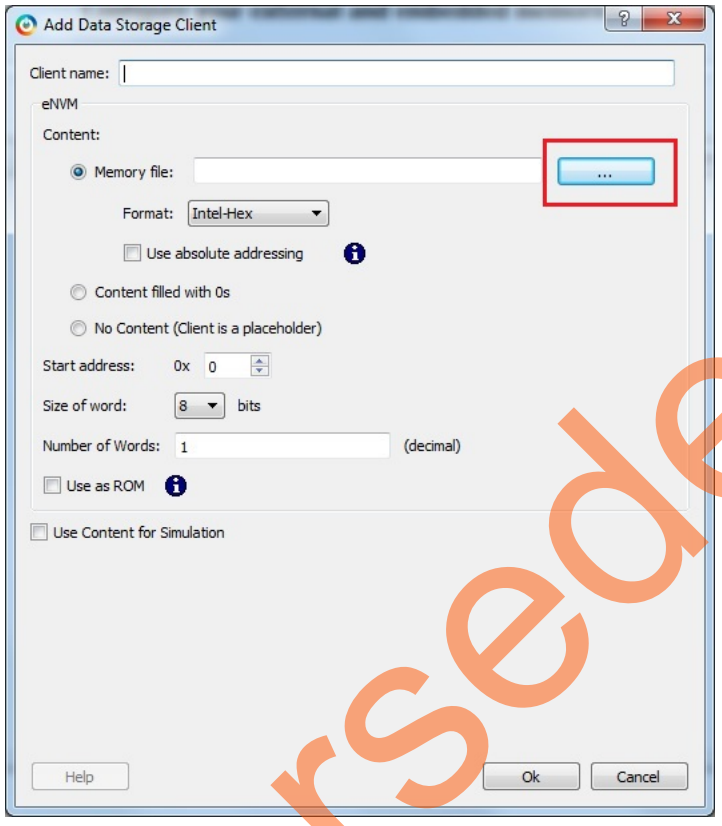


Figure 46 • **Add Data Storage Client**

21. Enter **Client Name** as eNVM in the **Add Data Storage Client** window.

22. Browse for the.hex file generated (as shown in Figure 44 on page 41). The generated executable image can be found in the **Release** folder under the SoftConsole project workspace, as shown in Figure 47.

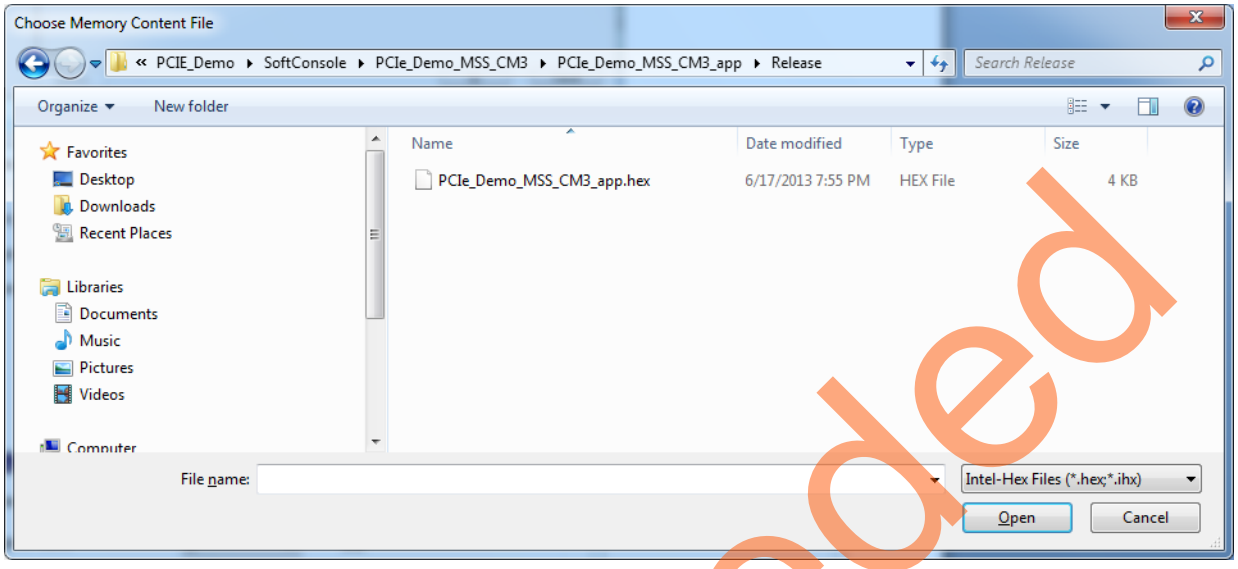


***Figure 47* • Browsing for .hex File**

23. Click **OK** in the **Add Data Storage Client** window, as shown in Figure 48.

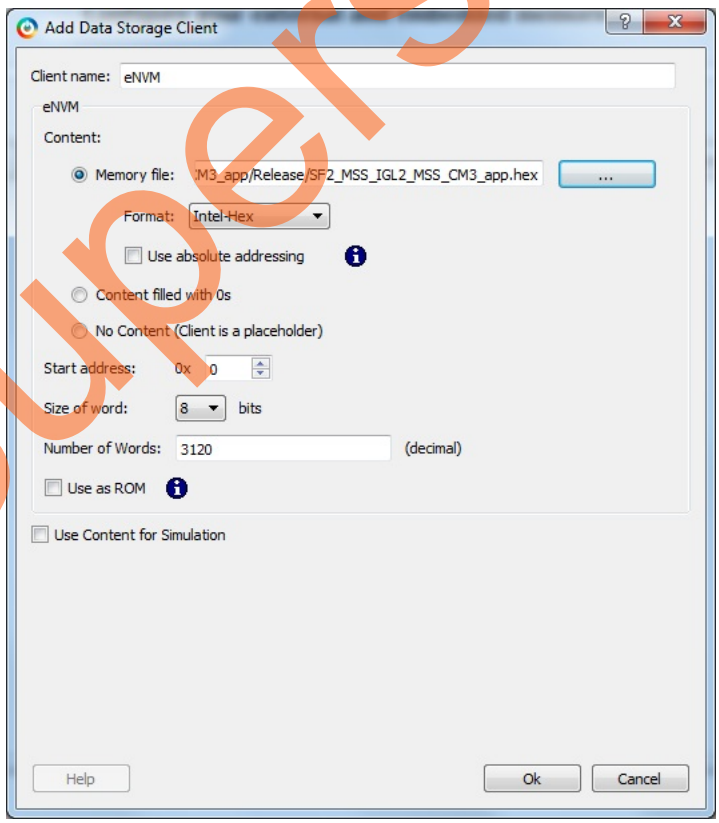


***Figure 48* • Add Data Storage Client**

24. Click **Next** and keep the rest of the System Builder tabs as default.

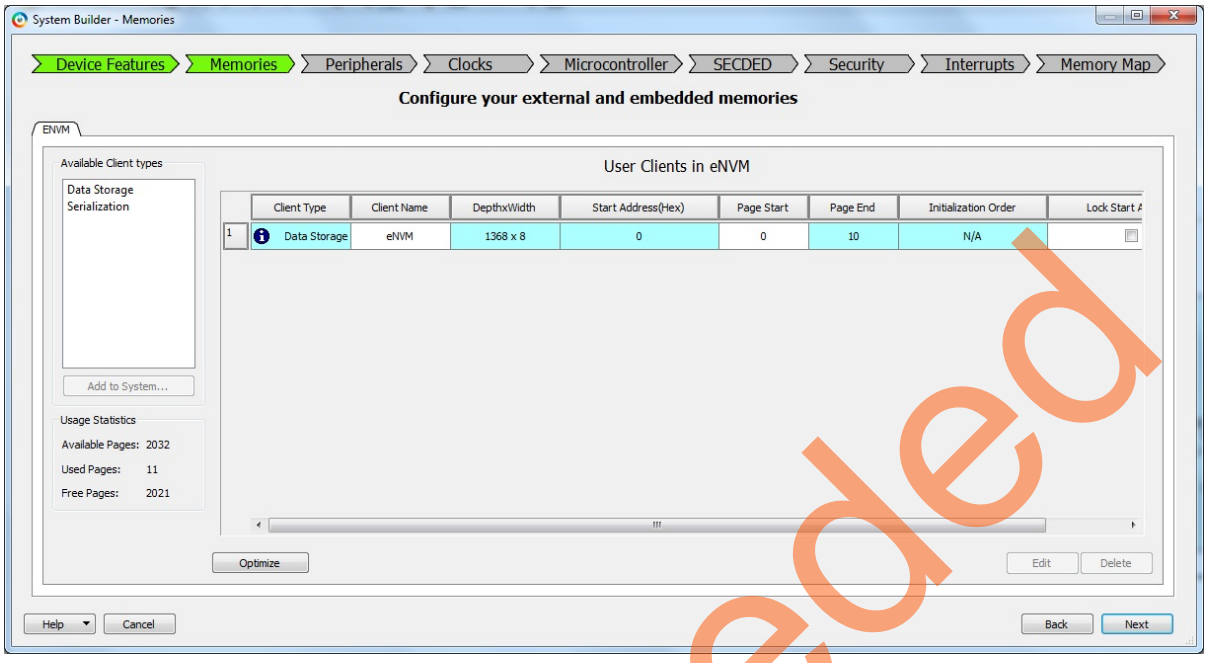


***Figure 49* • Modify Core - ENVM**

25. Save **PCIe_Demo** and regenerate the **PCIe_Demo** component by clicking **Generate Component** in SmartDesign.

Step 3: Developing the Simulation Stimulus

During the design process, SERDESIF is configured for the BFM simulation model. The BFM simulation model replaces the entire PCIe interface with a simple BFM that can send write transactions and read transactions over the AHB-Lite interface. These transactions are driven by a file and allow easy simulation of the FPGA design connected to a PCIe interface. This simulation methodology has the benefit of focusing on the FPGA design since the SmartFusion2 PCIe interface is a fully hardened and verified interface.

This section describes how to modify the BFM script (user.bfm) file that is generated by SmartDesign. The BFM script file simulates PCIe writing or reading to or from the MSS through the FIC_0.

1. Open the SERDESIF_0_PCIE_0_user.bfm file. To open the SERDESIF_0_PCIE_0_user.bfm, go to the **Files** tab > **Simulation** folder, and double-click the SERDESIF_0_PCIE_0_user.bfm. The SERDESIF_0_PCIE_0_user.bfm file is displayed, as shown in Figure 50.

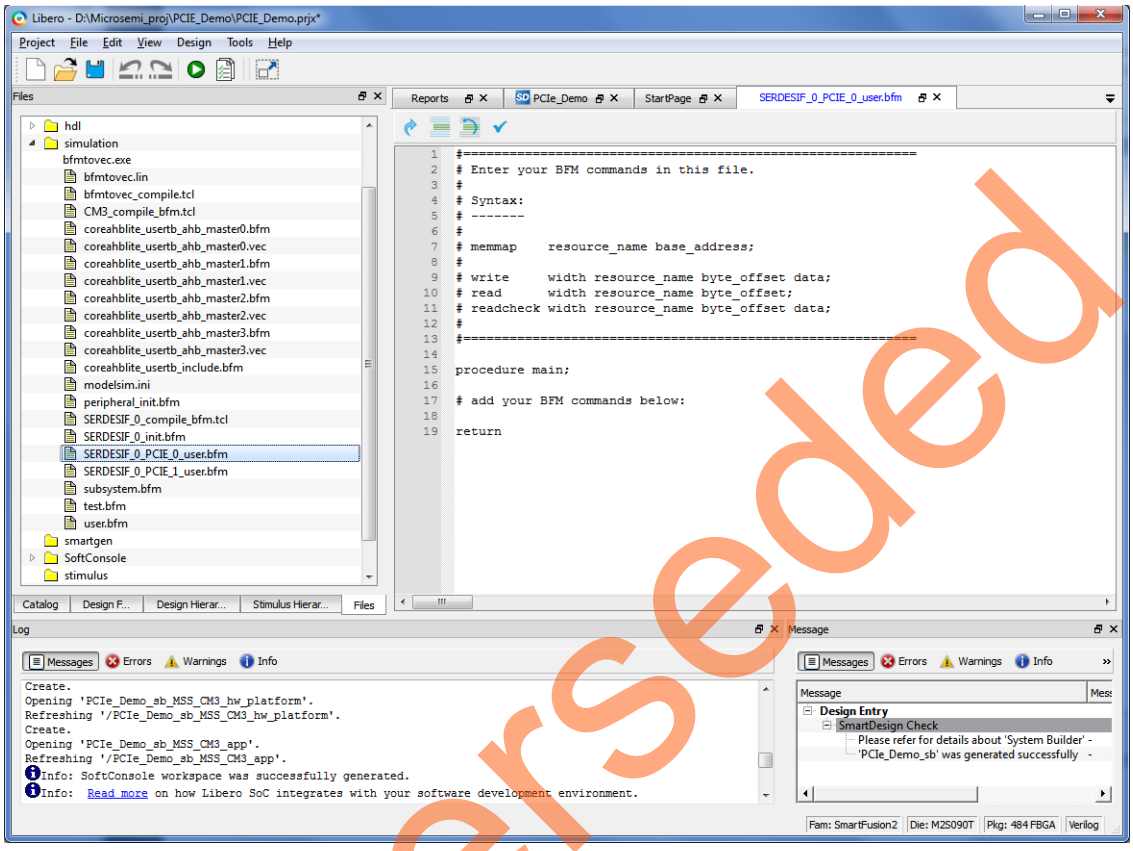


***Figure 50* • SmartDesign Generated SERDESIF_1_user.bfm File**

2. Modify the SERDESIF_0_PCIE_0_user.bfm to add the following bfm commands of writing and reading:

```
memmap GPIO 0x40013000;
memmap eSRAM 0x20000000;
procedure main;

# add your BFM commands below:
wait 500us;
wait 500us;
write w GPIO 0x00  0x5;
write w GPIO 0x04  0x5;
write w GPIO 0x08  0x5;
write w GPIO 0x0C  0x5;
write w GPIO 0x10  0x5;
write w GPIO 0x14  0x5;
write w GPIO 0x18  0x5;
write w GPIO 0x1C  0x5;

write w GPIO 0x88 0x00;
write w GPIO 0x88 0x01;
write w GPIO 0x88 0x02;
```

```
write w GPIO 0x88 0x04;
write w GPIO 0x88 0x08;
write w GPIO 0x88 0x10;
write w GPIO 0x88 0x20;
write w GPIO 0x88 0x40;
write w GPIO 0x88 0x80;

write w eSRAM 0x00  0x12345678;
write w eSRAM 0x04  0x87654321;
write w eSRAM 0x08  0x9ABCDEF0;
write w eSRAM 0x0C  0x0FEDCBA9;

readcheck w eSRAM 0x00  0x12345678;
readcheck w eSRAM 0x04  0x87654321;
readcheck w eSRAM 0x08  0x9ABCDEF0;
readcheck w eSRAM 0x0C  0x0FEDCBA9;

return
```

3. The modified BFM file appears similar to the file,as shown in Figure 51.
   BFM commands are added in the `SERDESIF_0_PCIE_0_user.bfm`. Perform the following:
   – Write to MSS GPIO
   – Write to eSRAM
   – Read-check from eSRAM

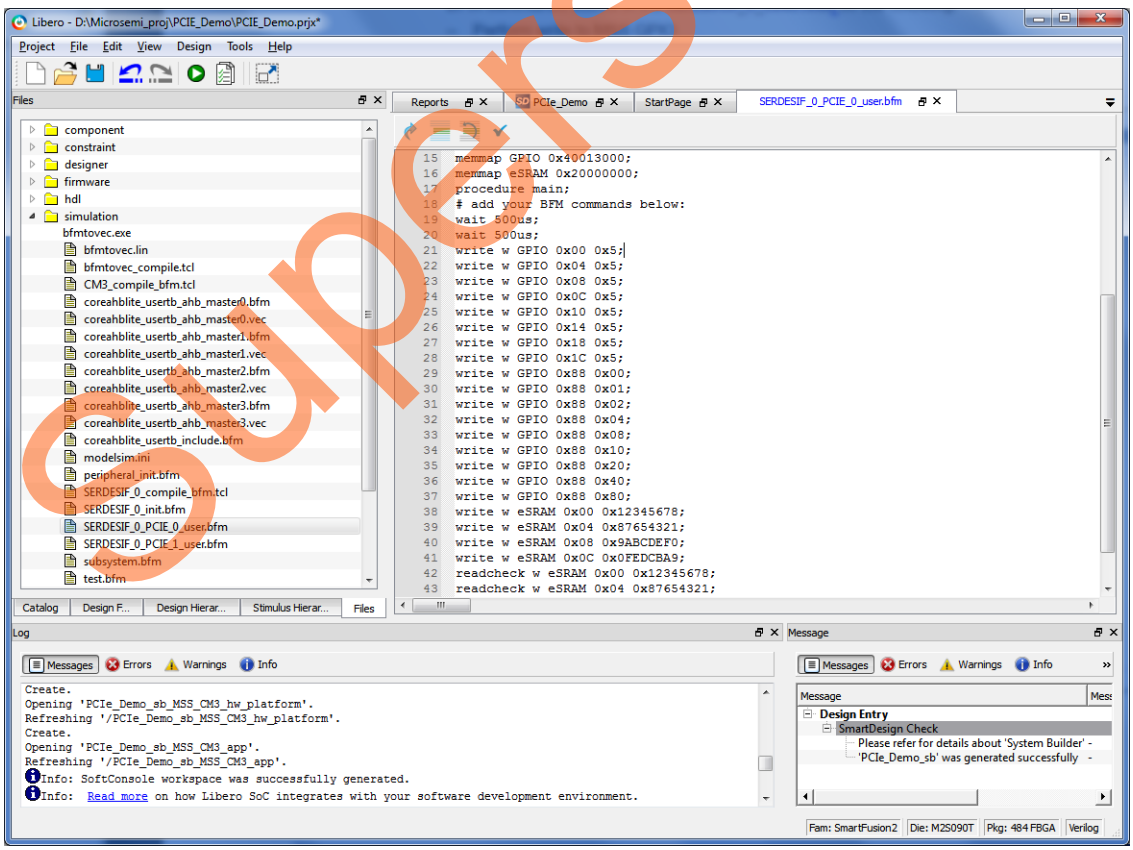


Figure 51 • **Modified SERDES User BFM**

Step 4: Simulating the Design

The design supports the BFM_PCIe simulation level to communicate with the High Speed Serial Interface block through the master AXI bus interface. Although, no serial communication actually goes through the High Speed Serial Interface block, this scenario allows validating the fabric interface connections. The SERDESIF_1_user.bfm file under the <Libero project>/simulation folder contains the BFM commands to verify the read or write access to MSS GPIOs and eSRAM.

The following steps describe how to use the SmartDesign testbench and BFM script file to simulate the design.

1. To generate the HDL testbench file follow the below instructions,

   a. From the **File** menu, choose **New** > **HDL Testbench**, as shown in Figure 52.

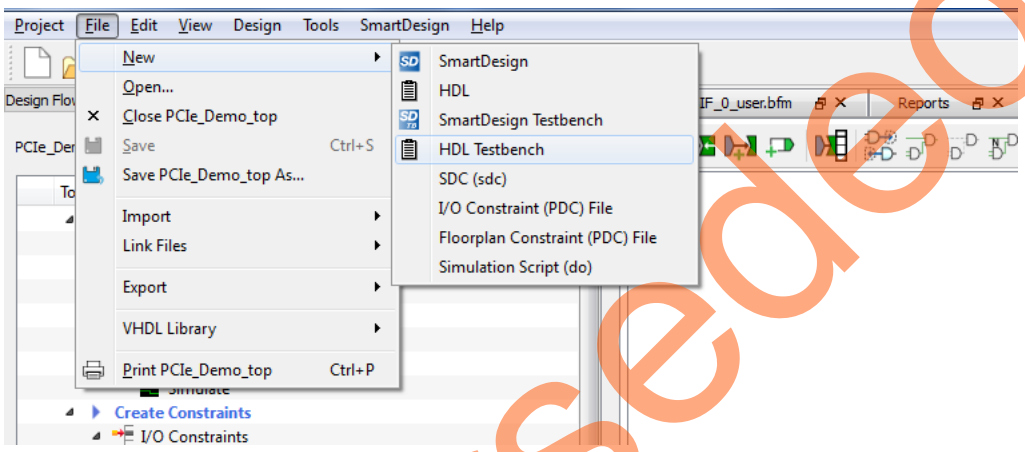


Figure 52 • **HDL Testbench**

Create New HDL Testbench File dialog box is displayed, as shown in Figure 53.

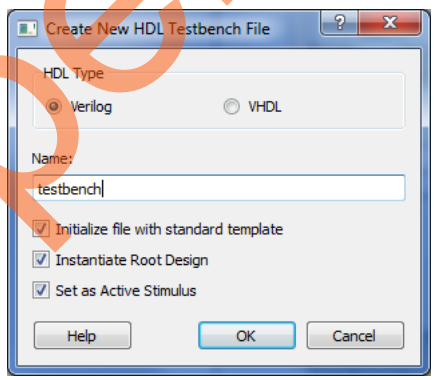


Figure 53 • **Create New HDL Testbench File**

   b. Select **Verilog** or **VHDL** under **HDL Type**.

   c. Enter testbench as a name of the new hdl testbench file and click **OK**.

2. Add the wave do file to the PCIe demo design simulation folder by clicking **File > Import > Others**.

3. Browse to the wave.do file location in the design files folder: *M2S90_PCIE_Control_Demo_DF/Source Files*. Figure 54 shows the wave.do file under simulation folder in the **Files** window.

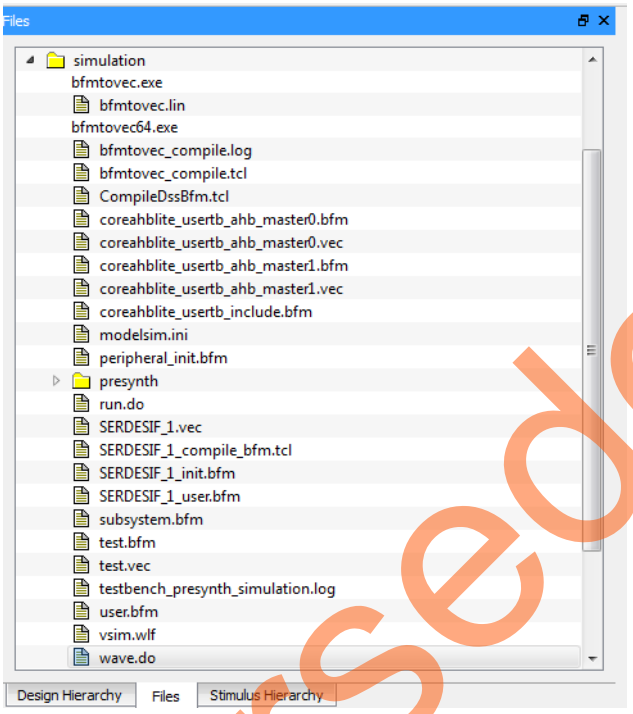


***Figure 54* • Wave.do File under Simulation Folder**

4. Open the Libero SoC project settings (**Project > Project Settings**).
5. Select **Do File** under **Simulation Options** in the Project Settings window. Change the **Simulation runtime** to **150us**, as shown in Figure 55.
6. Click **Save**.

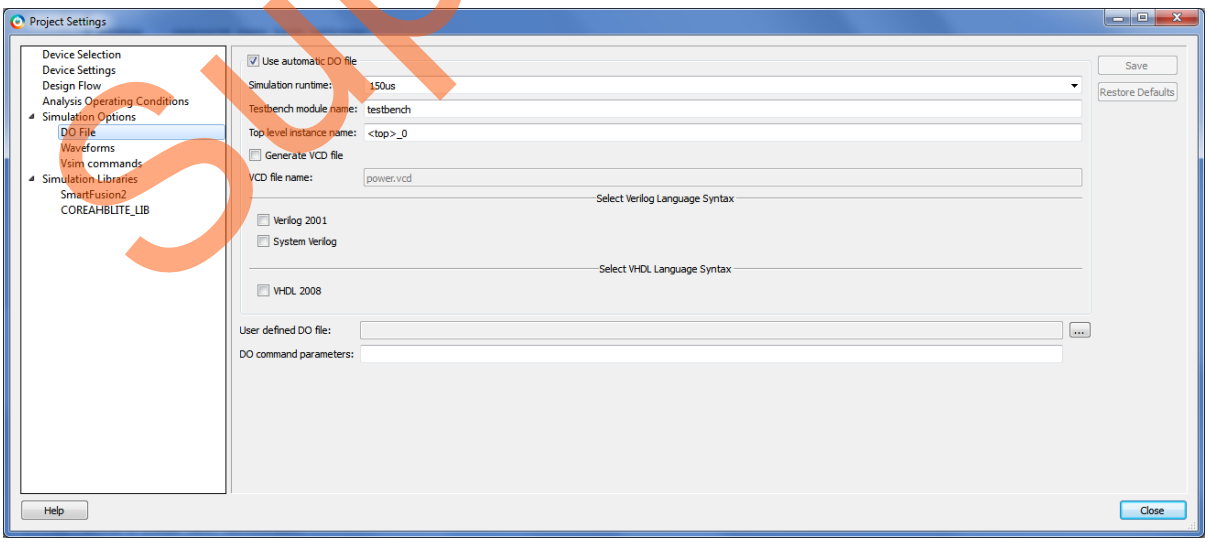


***Figure 55* • Project Setting – Do File Simulation Runtime Setting**

7. Select **Waveforms** under **Simulation Options**, as shown in Figure 56:
   – Select **Include Do file**.
   – Select **Log all signals in the design**.
   – Click **Close** to close the Project settings dialog box.
   – Select **Save** when prompted to save the changes.

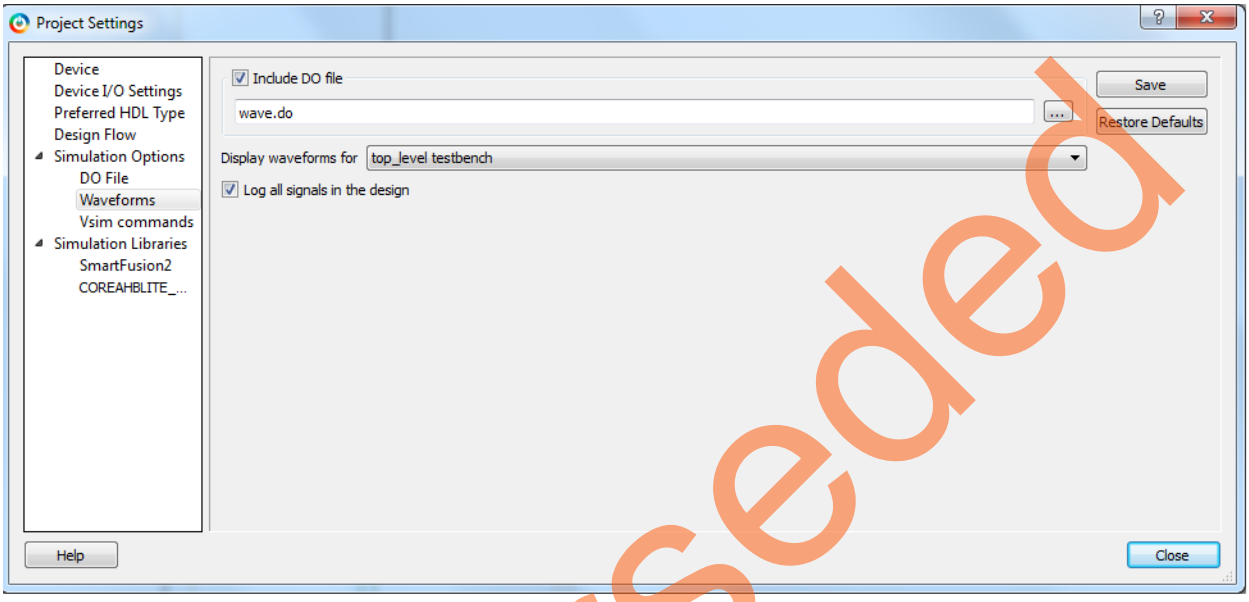


***Figure 56* • Project Setting – Waveform**

To run the simulation, double-click **Simulate** under **Verify Pre-Synthesized Design** in the **Design Flow** window.

ModelSim runs the design for approximately 150us. The ModelSim transcript window displays the BFM commands and the BFM simulation completed with no errors, as shown in Figure 57.

Figure 57 • **SERDES BFM Simulation**

Figure 58 shows the waveform window with MSS GPIO output signals.

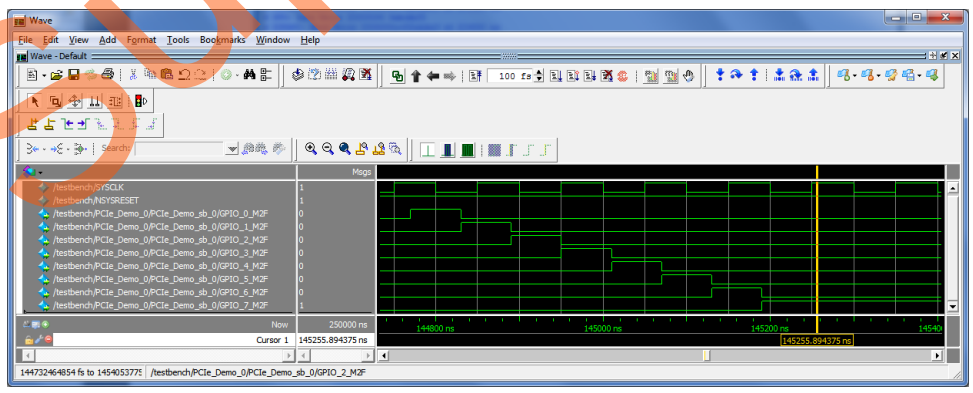


Figure 58 • **Simulation Result with MSS GPIO Signals**

Step 5: Generating the Program File

The following steps describe how to generate the program file.

1. Double-click **I/O Constraints** in the **Design Flow** window, as shown in Figure 59. The **I/O Editor** window is displayed after completing Synthesize and Compile.

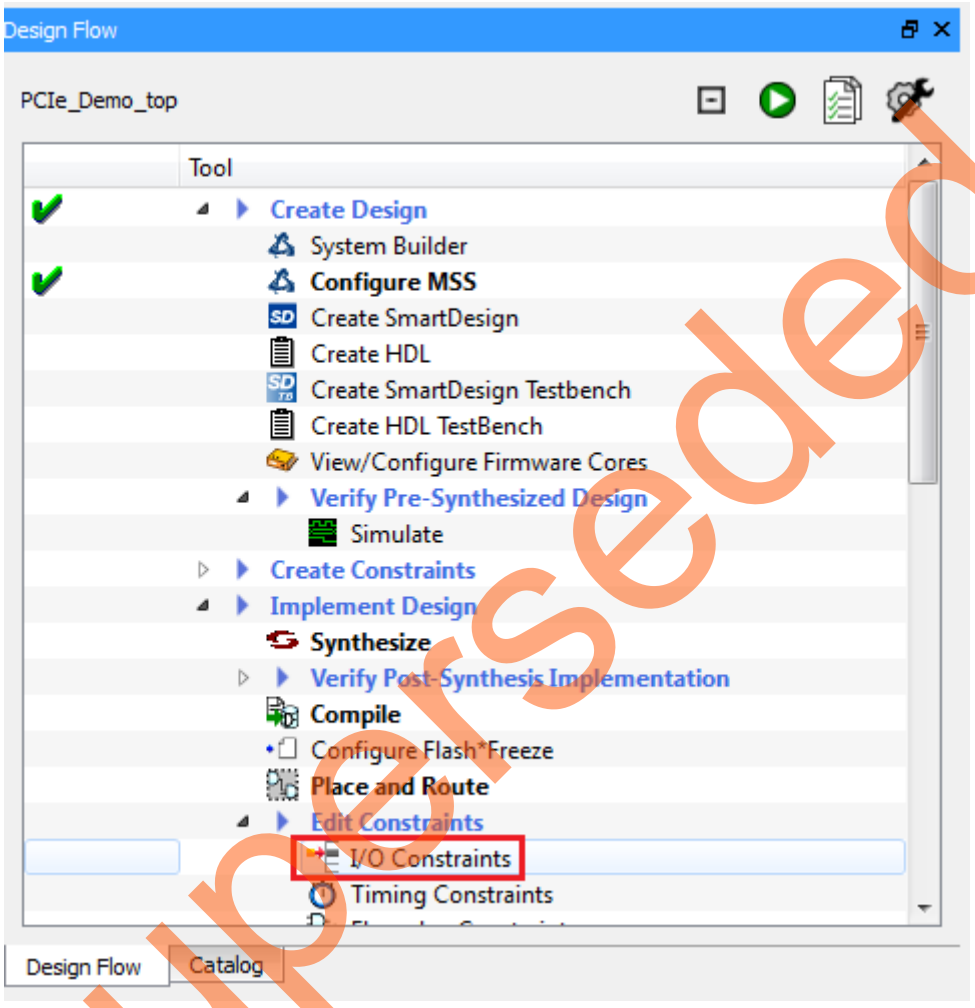


Figure 59 • **I/O Constraints**

2. The **I/O Editor** is displayed. Make the pin assignments,as shown in Table 5. After the pins have been assigned, the I/O Editor is displayed, as shown in Figure 60 on page 54.

Table 5 • **Port to Pin Mapping**

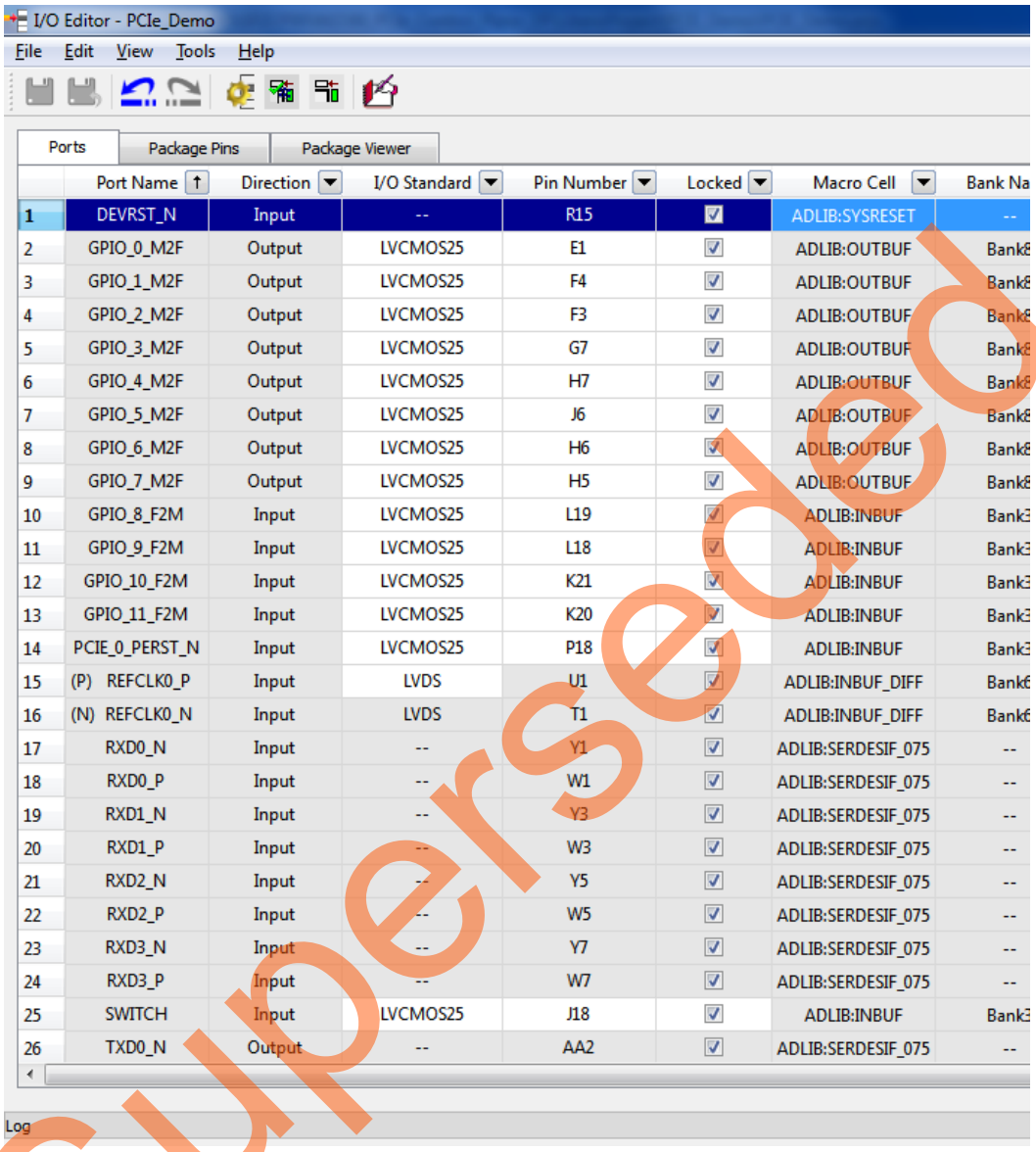
Port Name	Pin Number
GPIO_0_M2F	E1
GPIO_1_M2F	F4
GPIO_2_M2F	F3
GPIO_3_M2F	G7
GPIO_4_M2F	H7
GPIO_5_M2F	J6

Table 5 • **Port to Pin Mapping (continued)**

Port Name	Pin Number
GPIO_6_M2F	H6
GPIO_7_M2F	H5
GPIO_8_M2F	L19
GPIO_9_M2F	L18
GPIO_10_M2F	K21
GPIO_11_M2F	K20
SWITCH	J18
PCIE0_PERST_N	P18

These pin assignments are for connecting below on the SmartFusion2 Security Evaluation Kit.

- CLK0_PAD to 50 MHz Clock Oscillator
- GPIO_0 to GPIO_8 for LEDs
- GPIO_8 to GPIO_11 for DIP switches
- SWITCH for SW4
- PCIE_0_PERST_N to PERST of PCIe Edge connector



	Port Name	Direction	I/O Standard	Pin Number	Locked	Macro Cell	Bank Na
1	DEVRST_N	Input	--	R15	☑	ADLIB:SYSRESET	--
2	GPIO_0_M2F	Output	LVCMOS25	E1	☑	ADLIB:OUTBUF	Bank
3	GPIO_1_M2F	Output	LVCMOS25	F4	☑	ADLIB:OUTBUF	Bank
4	GPIO_2_M2F	Output	LVCMOS25	F3	☑	ADLIB:OUTBUF	Bank
5	GPIO_3_M2F	Output	LVCMOS25	G7	☑	ADLIB:OUTBUF	Bank
6	GPIO_4_M2F	Output	LVCMOS25	H7	☑	ADLIB:OUTBUF	Bank
7	GPIO_5_M2F	Output	LVCMOS25	J6	☑	ADLIB:OUTBUF	Bank
8	GPIO_6_M2F	Output	LVCMOS25	H6	☑	ADLIB:OUTBUF	Bank
9	GPIO_7_M2F	Output	LVCMOS25	H5	☑	ADLIB:OUTBUF	Bank
10	GPIO_8_F2M	Input	LVCMOS25	L19	☑	ADLIB:INBUF	Bank
11	GPIO_9_F2M	Input	LVCMOS25	L18	☑	ADLIB:INBUF	Bank
12	GPIO_10_F2M	Input	LVCMOS25	K21	☑	ADLIB:INBUF	Bank
13	GPIO_11_F2M	Input	LVCMOS25	K20	☑	ADLIB:INBUF	Bank
14	PCIE_0_PERST_N	Input	LVCMOS25	P18	☑	ADLIB:INBUF	Bank
15	(P) REFCLK0_P	Input	LVDS	U1	☑	ADLIB:INBUF_DIFF	Bank
16	(N) REFCLK0_N	Input	LVDS	T1	☑	ADLIB:INBUF_DIFF	Bank
17	RXD0_N	Input	--	Y1	☑	ADLIB:SERDESIF_075	--
18	RXD0_P	Input	--	W1	☑	ADLIB:SERDESIF_075	--
19	RXD1_N	Input	--	Y3	☑	ADLIB:SERDESIF_075	--
20	RXD1_P	Input	--	W3	☑	ADLIB:SERDESIF_075	--
21	RXD2_N	Input	--	Y5	☑	ADLIB:SERDESIF_075	--
22	RXD2_P	Input	--	W5	☑	ADLIB:SERDESIF_075	--
23	RXD3_N	Input	--	Y7	☑	ADLIB:SERDESIF_075	--
24	RXD3_P	Input	--	W7	☑	ADLIB:SERDESIF_075	--
25	SWITCH	Input	LVCMOS25	J18	☑	ADLIB:INBUF	Bank
26	TXD0_N	Output	--	AA2	☑	ADLIB:SERDESIF_075	--

Figure 60 • **I/O Editor**

3. After updating I/O editor, click **Commit and Check**.
4. Close the I/O editor.
5. Click **Generate Bitstream**, as shown in Figure 61 to complete place and route, verify timing, and generate the programming file.

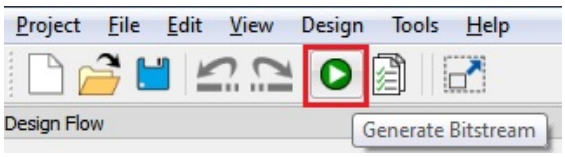


Figure 61 • **Generate Bitstream**

Running the Demo

Demo Setup

Following are the steps to setup the demo for SmartFusion2 Security Evaluation Kit board:

1. Connect the FlashPro4 programmer to the J5 connector of the SmartFusion2 Security Evaluation Kit board.
2. Connect the jumpers on the SmartFusion2 Security Evaluation Kit board, as shown in Table 4.

CAUTION: The power supply switch SW7 on the board should be in OFF position, while making the jumper connections.

Table 6 • **SmartFusion2 FPGA Security Evaluation Kit Jumper Settings**

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

3. Connect the power supply to the **J6** connector.
4. Switch the power supply switch **SW7** to **ON** position.
5. To program the SmarFusion2 device double-click **Run PROGRAM Action** in the **Design Flow** tab, as shown in Figure 62.

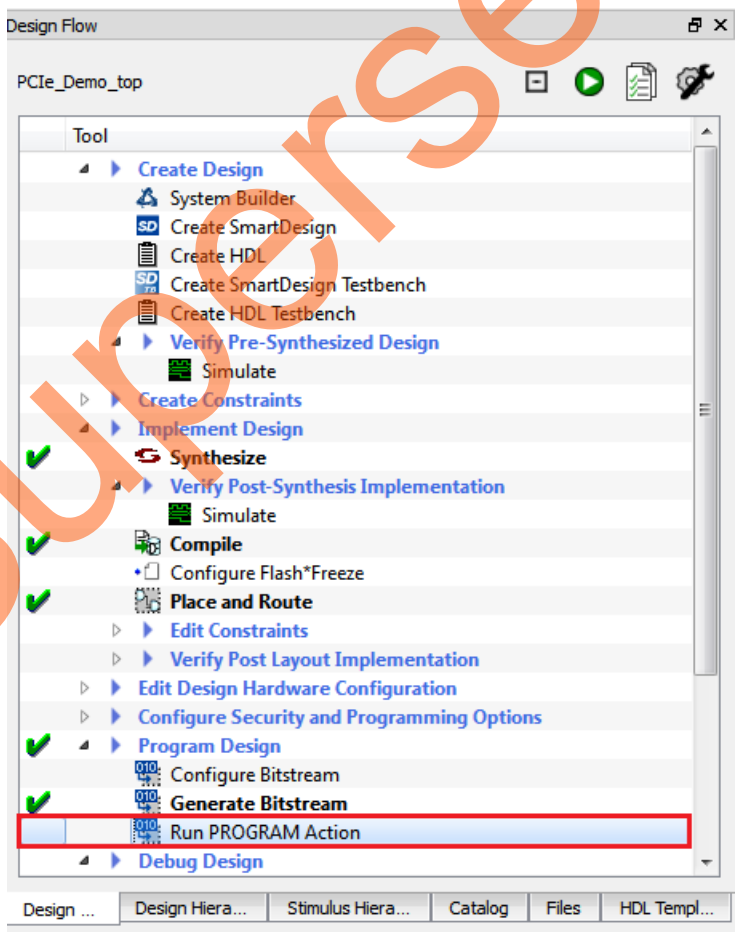


Figure 62 • **Run PROGRAM Action**

6. After Successful programming, power OFF the SmartFusion2 Security Evaluation Kit and shut down the Host PC.
7. Following are the steps to connect the **CON1-PCIe Edge Connector** either to Host PC or laptop:

   a. Connect the CON1-PCIe Edge Connector to Host PC PCIe Gen 2 slot or Gen 1 slot, as applicable. If the Host PC does not support the Gen 2 compliant slot, the design switches to the Gen 1 slot.

   b. Connect the CON1-PCIe Edge Connector to the laptop PCIe slot using the express card adapter. If you are using a laptop, the express card adapters typically support only Gen 1 and the design works on Gen 1 slot.

CAUTION: Host PC or laptop should be powered OFF while inserting the PCIe Edge Connector. If the system is not powered OFF, the PCIe device detection and selection of Gen 1 or Gen 2 does not occur properly. Microsemi recommends that the Host PC or laptop should be powered OFF during the PCIe card insertion.

The board setup is as shown in Figure 63.

Figure 63 • **SmartFusion2 Security Evaluation Kit Setup**

8. Switch ON the power supply switch, **SW7**.

Running the Demo Design

This demo can run on both windows and Red Hat Linux operating system.

To run the demo on Windows operating system GUI, Jungo drivers are provided. Refer to "Running the Demo Design on Windows" on page 57.

To run the demo on Linux operating system native Red Hat Linux drivers and command line scripts are provided. Refer to "Running the Demo Design on Linux" on page 70.

Running the Demo Design on Windows

The following steps describe how to run the Demo Design on Windows:

1. Power on the Host PC and check the Host PC Device Manager for PCIe Device. It is similar to Figure 64. If the device is not detected, power cycle the SmartFusion2 Security Evaluation Kit and click **Scan for hardware changes** in **Device Manager** window.

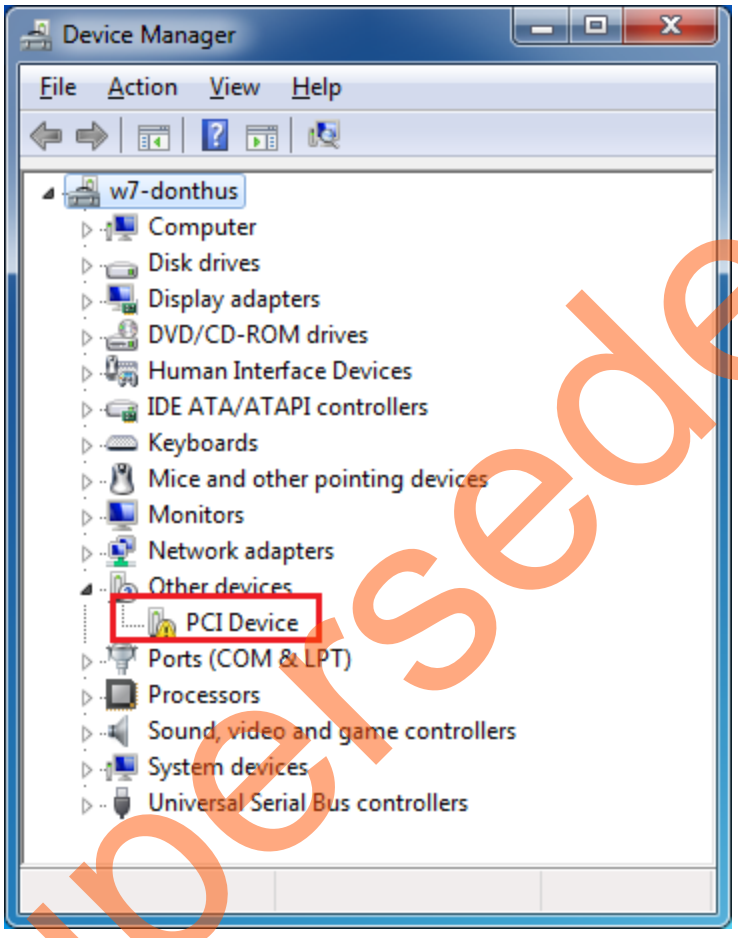


Figure 64 • **Device Manager - PCIe Device Detection**

2. If the Host PC has any other installed drivers (previous versions of Jungo drivers) for the SmartFusion2 PCIe device, uninstall them. To uninstall previous versions of Jungo drivers follow steps 3 and 4.
3. To uninstall previous Jungo drivers go to **Device Manager,** right-click on **DEVICE,** and click **Uninstall**, as shown in Figure 65 on page 58.

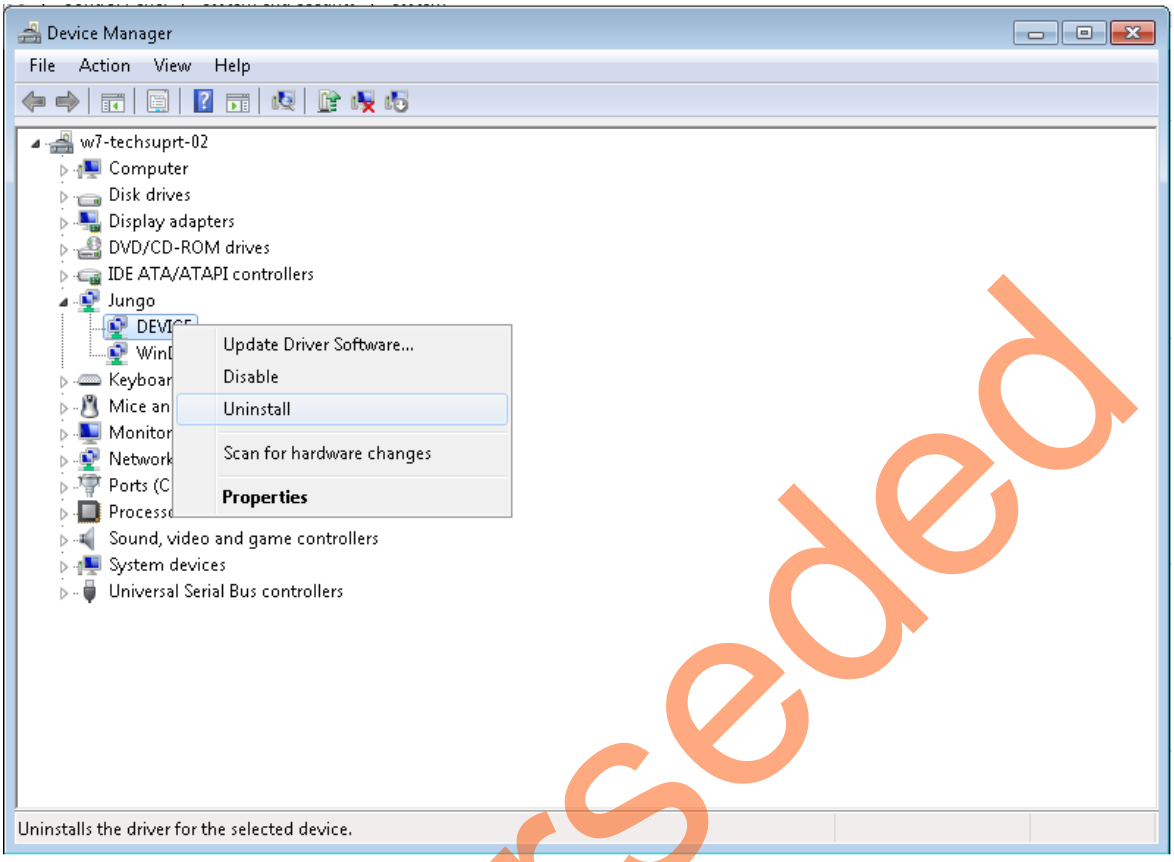


Figure 65 • **Device Manager Window**

4. **Confirm Device Uninstall** window is displayed, as shown in Figure 66. Confirm Device Uninstall.Select **Delete the driver software for this device**. After uninstalling previous Jungo drivers, ensure that the PCI Device is detected in the **Device Manager** window, as shown in Figure 66.

Figure 66 • **Confirm Device Uninstall Dialogue Box**

Note: If the device is still not detected, check if the BIOS version in Host PC is latest, and if PCI is enabled in the Host PC BIOS.

Drivers Installation

The PCIe Demo uses a driver framework provided by Jungo WinDriverPro. To install the PCIe drivers on Host PC for SmartFusion2 Security Evaluation Kit, use the following steps:

1. Extract the **PCIe_Demo.rar** to C:\ drive. The PCIe_Demo.rar is located in the provided design files:
   – *M2S90_PCIE_Control_DEMO_DF\Windows_64bit\Drivers\PCIe_Demo.rar*
2. Run the batch file **C:\PCIe_Demo\DriverInstall\Jungo_KP_install.bat**

   Note:Installing these drivers require Host PC Administration rights.

3. In the Windows Security dialog box, click **Install**, as shown in Figure 67.

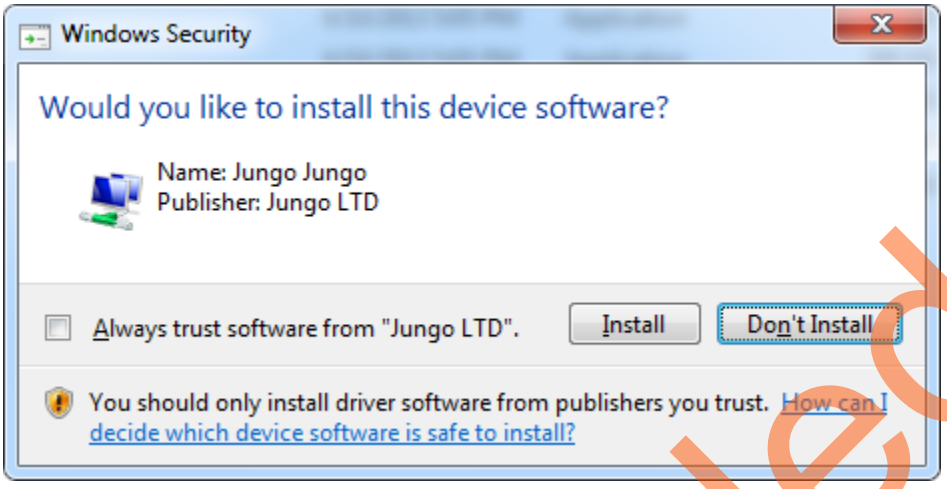


Figure 67 • **Jungo Driver Installation**

Note: If the installation is not in progress, right-click on the command prompt and select Run as administrator. Run the batch file C:**PCIe_Demo\DriverInstall\Jungo_KP_install.bat** from command prompt.

4. Click **Install this driver software anyway**, as shown in Figure 68.

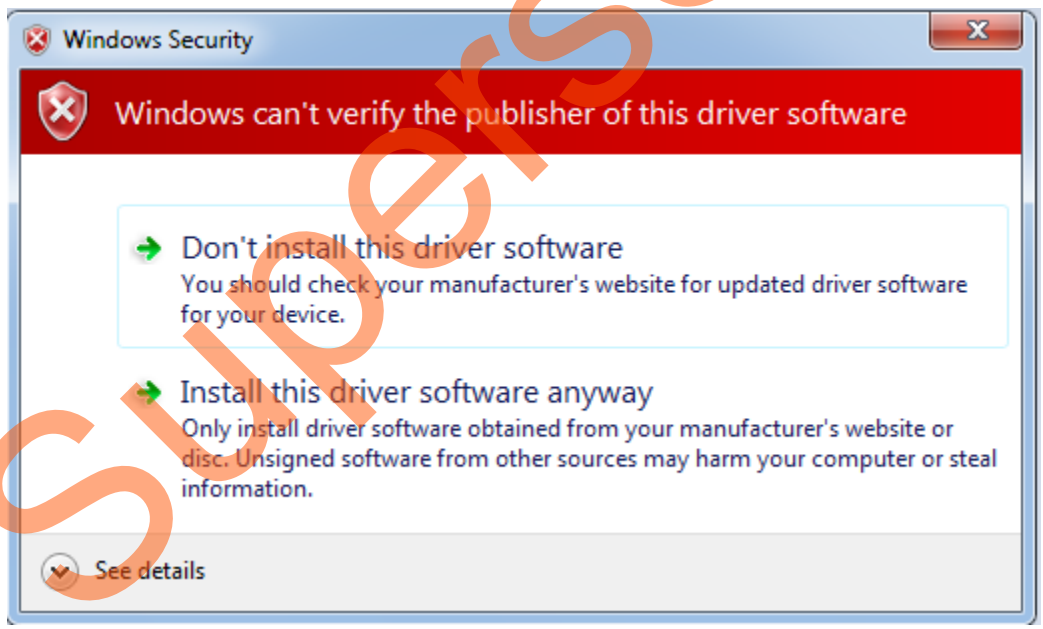


Figure 68 • **Windows Security**

PCIe Demo GUI

SmartFusion2 PCIe Demo GUI is a simple graphic user interface that runs on the Host PC to communicate with the SmartFusion2 PCIe EP device. The GUI provides the PCIe link status, driver information, and demo controls. The GUI invokes the PCIe driver installed on the Host PC and provides commands to the driver according to the user selection.

The following steps are used to install the GUI:

1. Download the PCIe_Demo_GUI Installer from the following link.
   http://soc.microsemi.com/download/rsc/?f=PCIe_Demo_GUI_Installer
2. Double-click **setup.exe** in the provided GUI installation (PCIe_Demo_GUI_Installer\setup.exe). Apply default options, as shown in Figure 69.

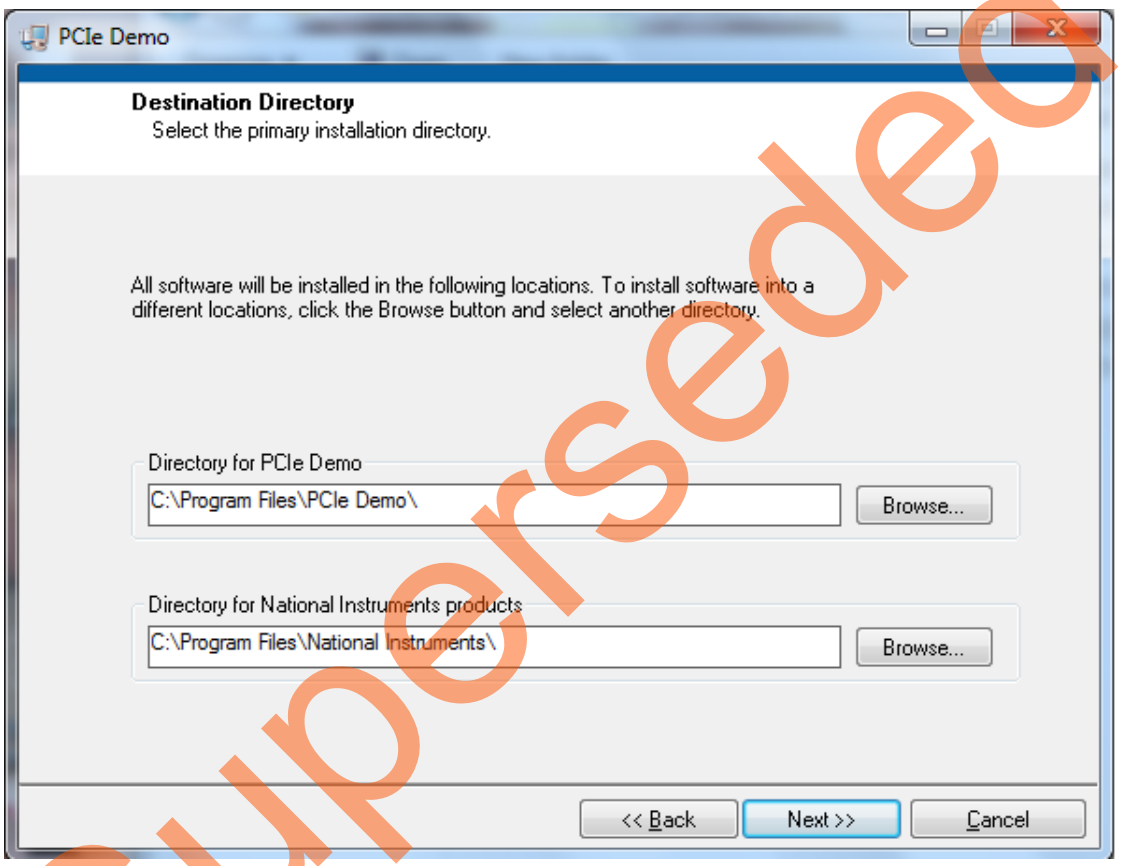


***Figure 69* • GUI Installation**

3. Click **Next** to complete the installation. After successful installation, the following window is displayed, as shown in Figure 70.

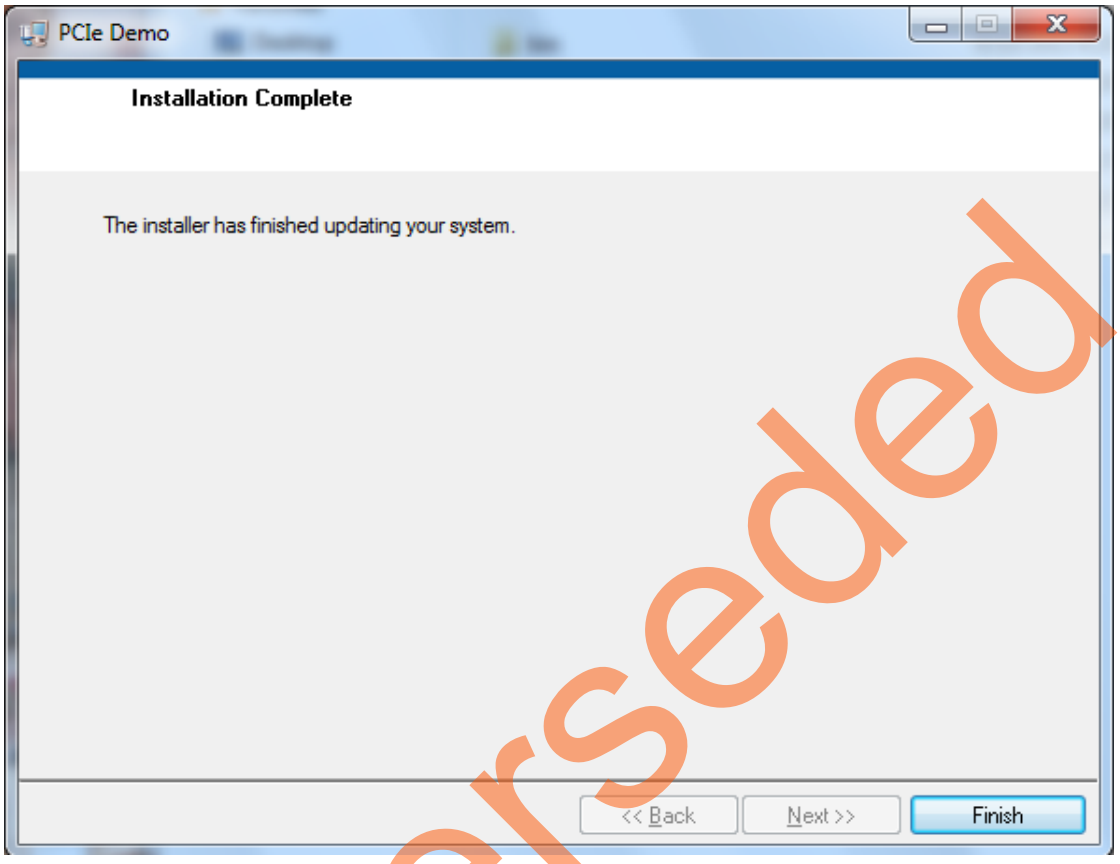


***Figure 70* • Successful GUI Installation**

4. Restart the host PC.

Running the Design

The following steps describe how to run the design.

1. Check the Host PC **Device Manager** for the drivers. If the device is not detected, power cycle the SmartFusion2 Security Evaluation Kit and click **Scan for hardware changes** in Device Manager. Ensure that the board is switched on.

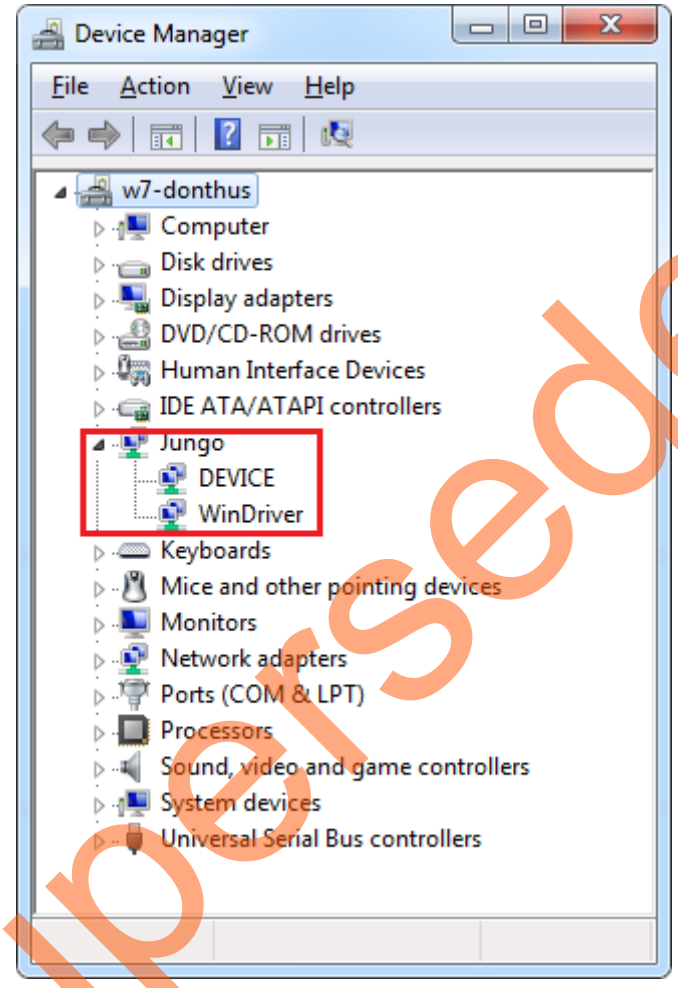


Figure 71 • **Device Manager - PCIe Device Detection**

Note: If a warning symbol is displayed on the **DEVICE** or **WinDriver** icons in the **Device Manager**, uninstall them and start from step1 of "Drivers Installation" on page 59.

2. Invoke the GUI from **ALL Programs > PCIeDemo > PCIe Demo**. The GUI is displayed, as shown in Figure 72.

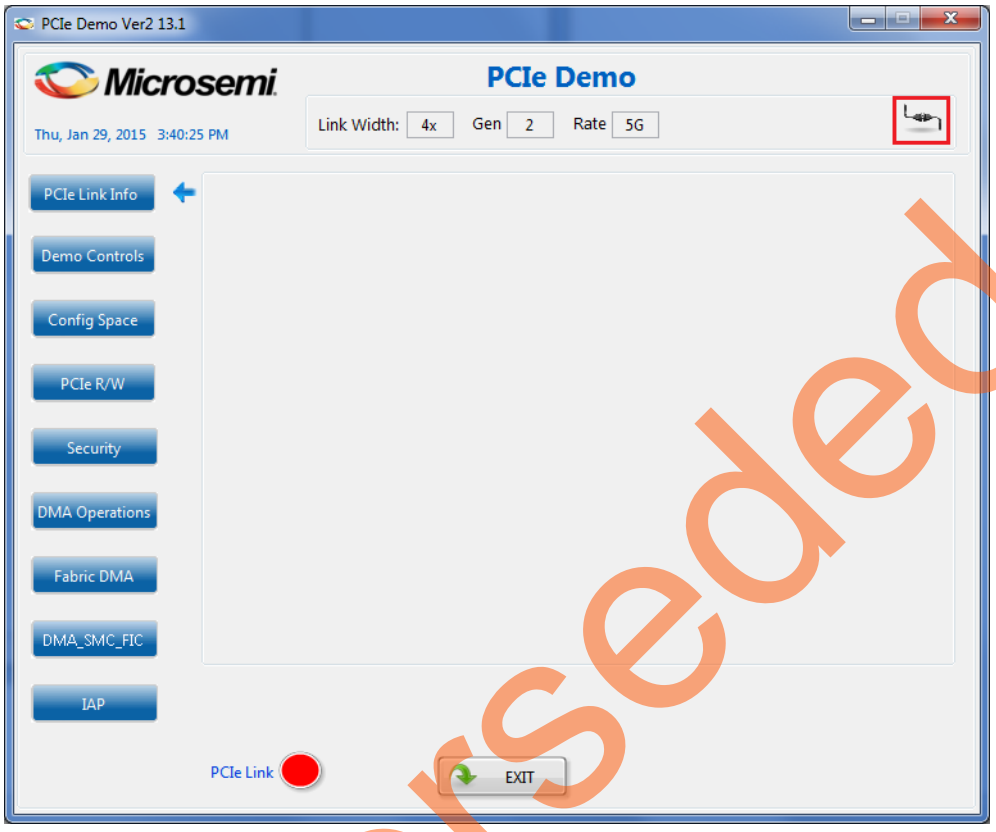


Figure 72 • **PCIe Demo GUI**

3. Click **Connect** button at the top-right corner of the GUI. The messages are displayed on the GUI, as shown in Figure 73.

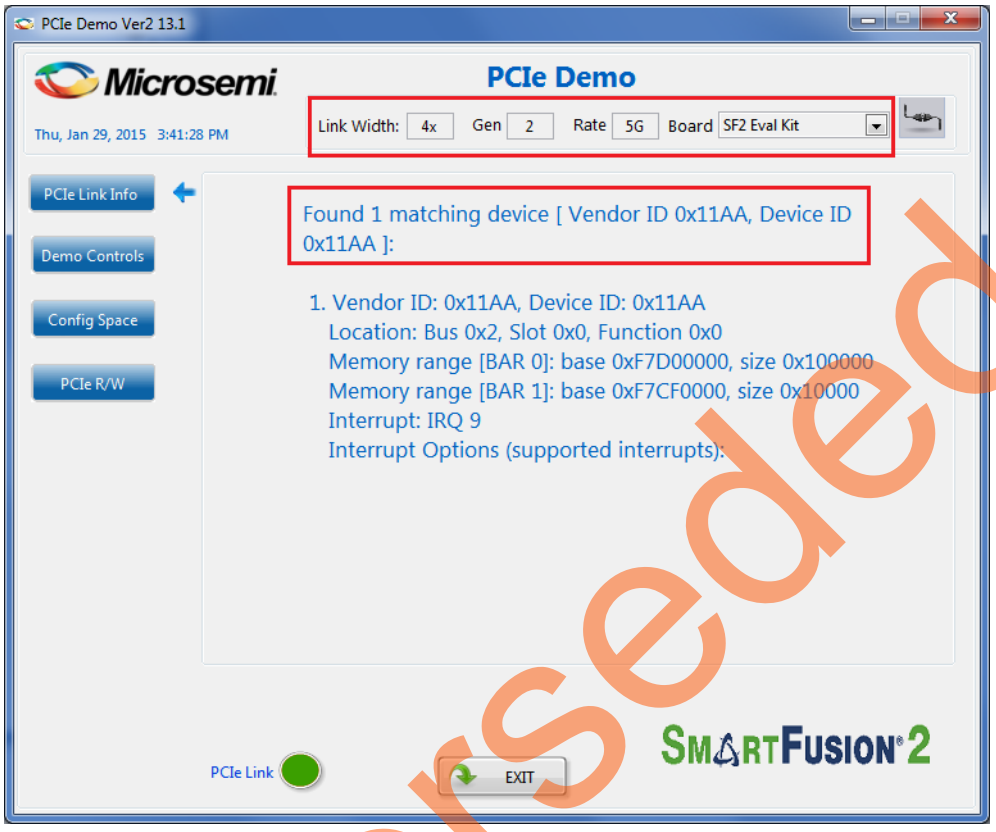


***Figure 73* • Version Information**

4. Click **Demo Controls** in the GUI displays the LEDs options and DIP switch positions, as shown in Figure 74.

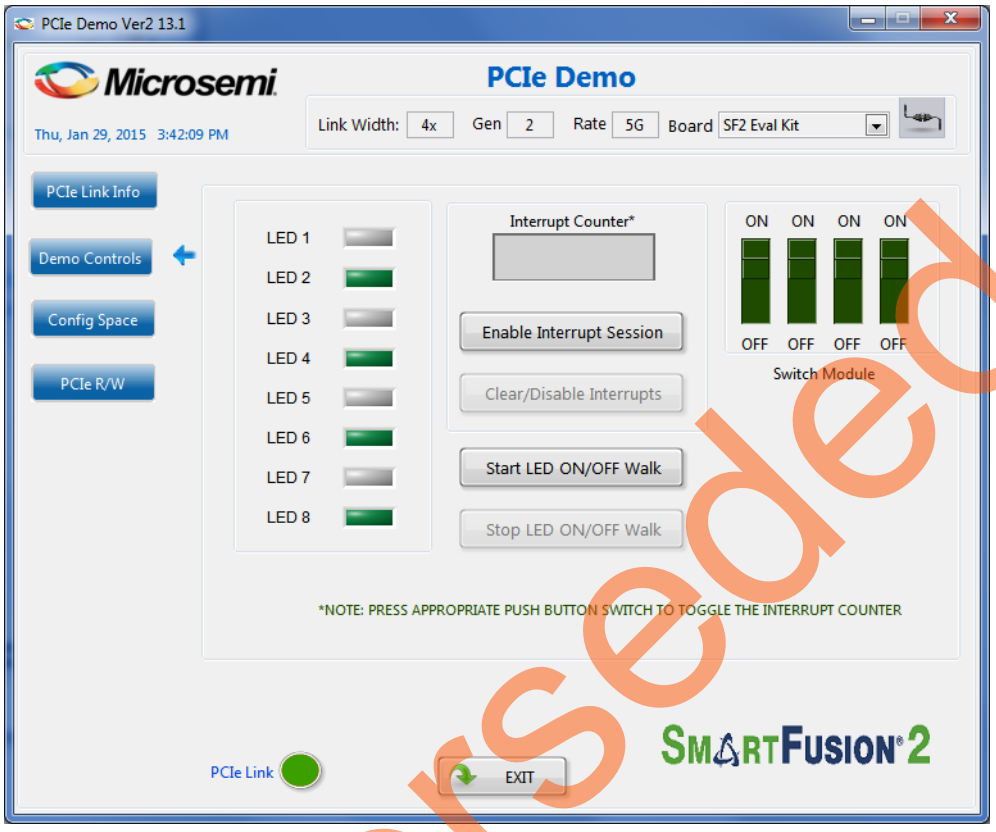


***Figure 74* • Demo Controls**

5. Click LEDs in GUI to ON/OFF the LEDs on the SmartFusion2 Security Evaluation Kit.
6. Click **Start LED ON/OFF Walk** to blink the LEDs on the SmartFusion2 Security Evaluation Kit.
7. Click **Stop LED ON/OFF Walk** to stop the LEDs blinking.
8. Change the DIP switch positions on the SmartFusion2 Security Evaluation Kit (SW10) and observe the similar position of switches in GUI SWITCH MODULE.
9. Click **Enable Interrupt Session** to enable the PCIe interrupt.

10. Press the push button SW3 on the SmartFusion2 Security Evaluation Kit and observe the interrupt count on the **Interrupt Counter** field in GUI, as shown in Figure 75.

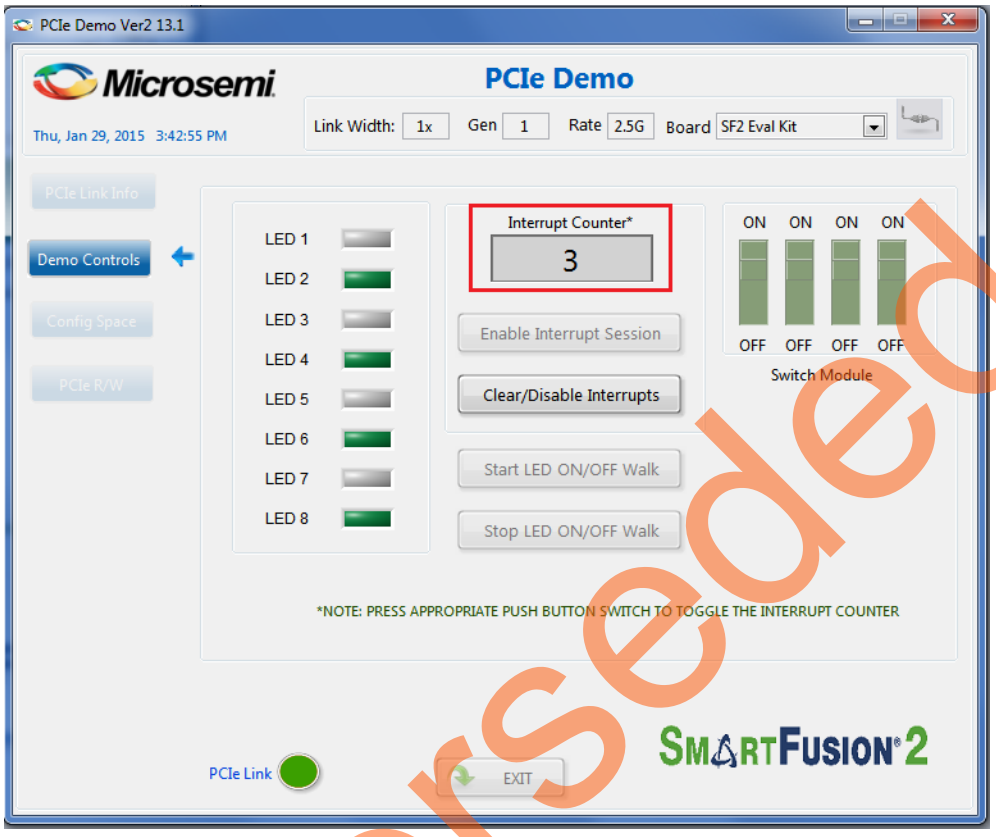


Figure 75 • **Interrupt Counter**

11. Click **Clear/Disable Interrupts** to clear and disable the PCIe interrupts.

12. Click **Config Space** to read details about the PCIe configuration space. Figure 76 shows the PCIe configuration space.

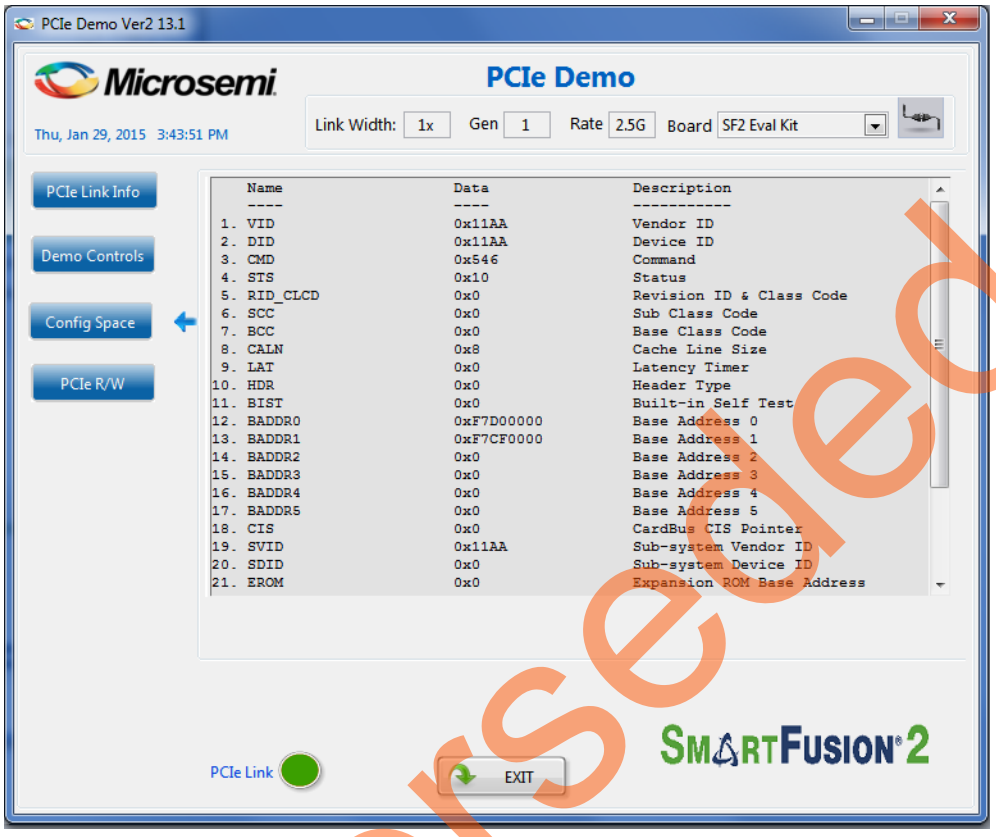


Figure 76 • **Configuration Space**

13. Click PCIe R/W to perform read and writes to eSRAM memory through BAR1 space. Figure 77 shows the PCIe R/W window.

14. Enter the address in the Address field between 0x0000 to 0xFFFC. The Data field accepts a 32-bit hexadecimal value.

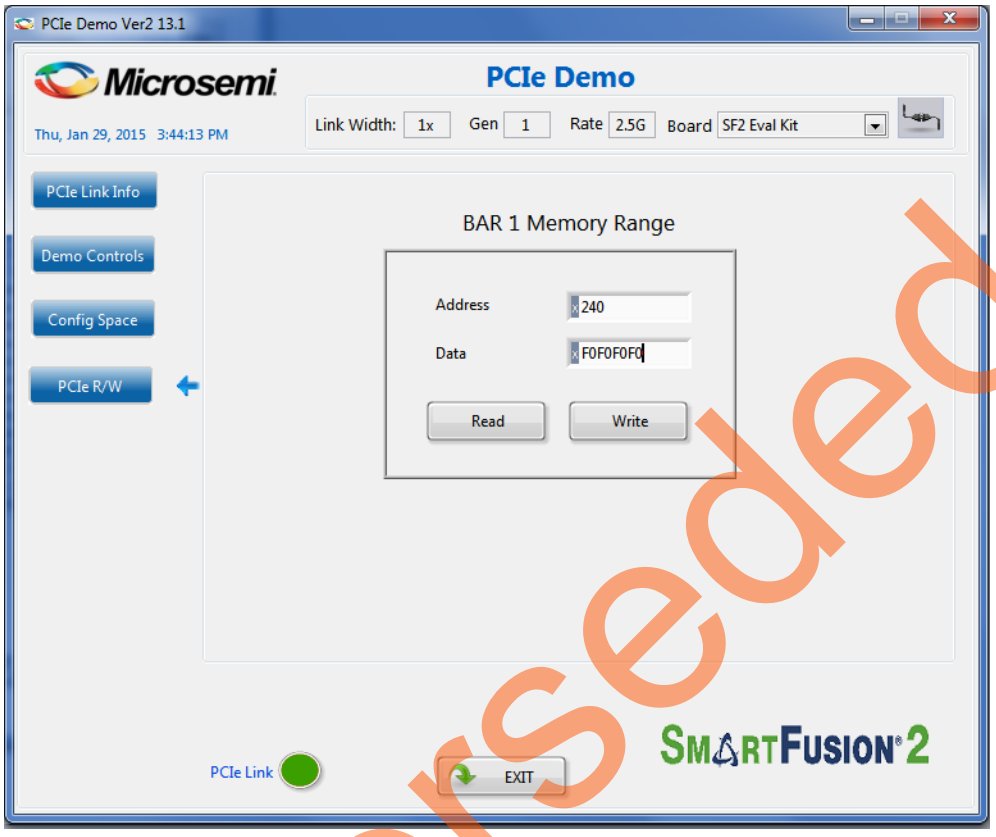


***Figure 77* • Perform Read and Write to eSRAM Using PCIe**

15. Click **Exit** to quit the demo.

Running the Demo Design on Linux

The following steps describe how to run the Demo Design on Linux.

1. Switch **ON** the Red Hat Linux Host PC.
2. Red Hat Linux Kernel detects the SmartFusion2 PCIe end point as Actel Device.
3. On Linux Command Prompt Use `lspci` command to display the PCIe info.

```
# lspci
```

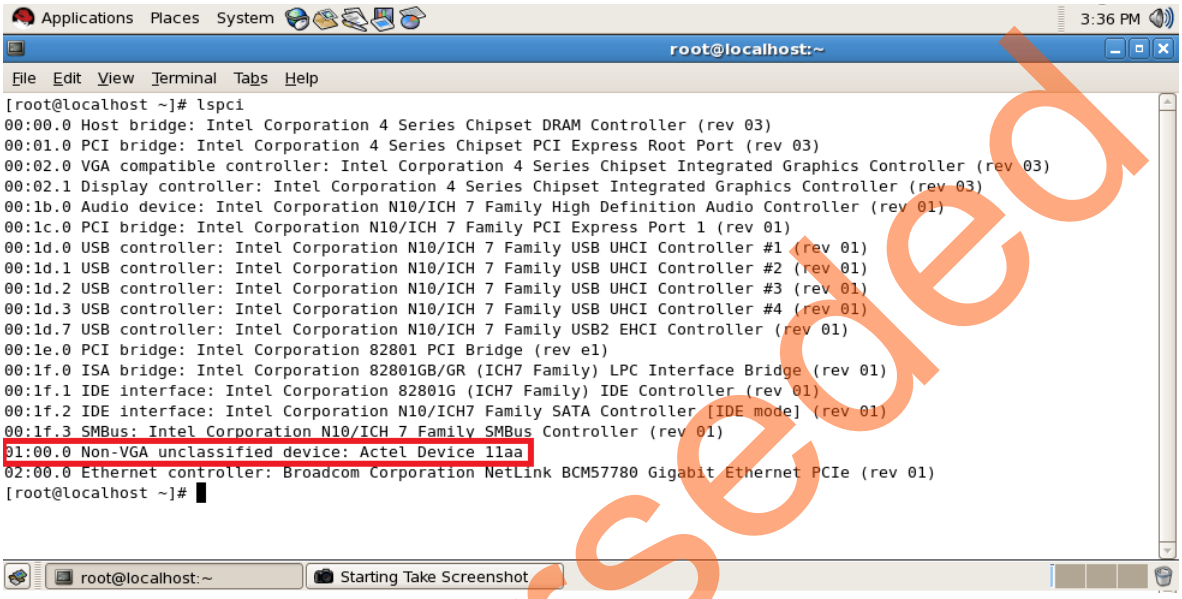


Figure 78 • **PCIe Device Detection**

Drivers Installation

Enter the following commands in the Linux command prompt to install the PCIe drivers:

1. Create the **sf2** directory under the **home/** directory using the following command:

   ```
   # mkdir /home/sf2
   ```

2. Copy the M2S90_PCIe_Control_Plane_DF design files folder under **/home/sf2** directory, which contains the Linux PCIe device driver files and Linux PCIe application utility files.
3. Copy the Linux PCIe Device Driver file (PCIe_Driver.rar) from M2S90_PCIe_Control_Plane_DF/design files folder.

   ```
   cp -rf/home/sf2/M2S90_PCIe_Control_Plane_DF/Linux_64bit/Drivers/
   PCIe_Driver.rar/home/sf2# unzip PCIe_Driver.rar
   ```

4. Execute `ls` command to display the contents of /home/sf2 directory.

   ```
   # ls
   ```

5. Change to inc/ directory by using the following command:

   ```
   #cd /home/sf2/inc
   ```

6. Edit the `board.h` file for SmartFusion2 Security Evaluation Kit, as shown in Figure 79.

```
#vi board.h
#undef SF2_ADV_KIT
#undef IGL2
#undef SF2_DEV_KIT
#define SF2_EVAL_KIT
```

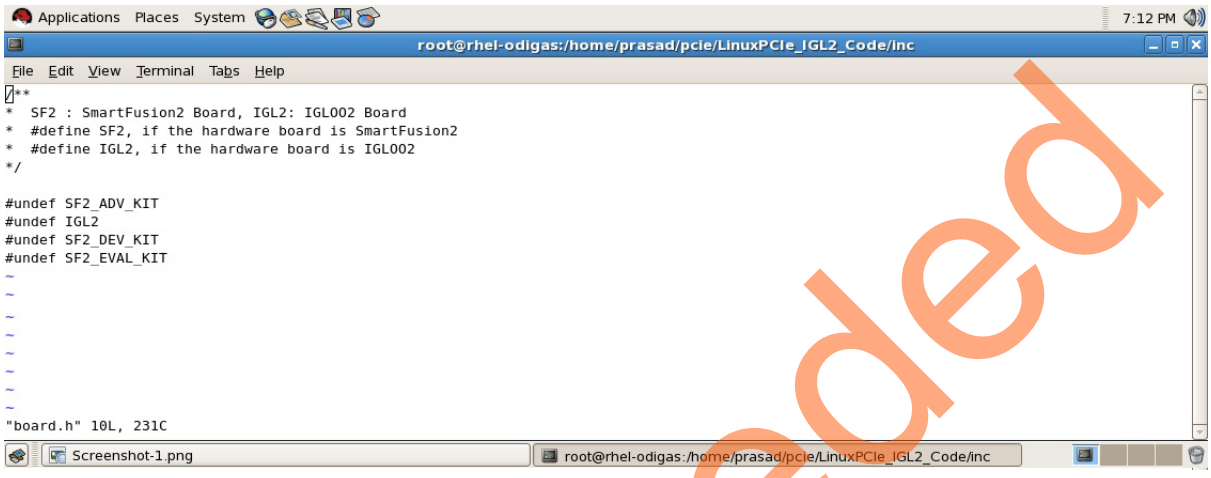


Figure 79 • **Edit board.h File**

7. To save the selected file, execute the `:wq` command
8. Change to PCIe_Driver/ directory using the `cd` command:

```
#cd /home/sf2/PCIe_Driver
```

9. To compile the Linux PCIe device driver code, execute `make` command.

```
#make clean [To clean any *.o, *.ko files]
#make
```

10. The kernel module, `pci_chr_drv_ctrlpln.ko` creates in the same directory.
11. To insert the Linux PCIe device driver as a module, execute `insmod` command.

```
#insmod pci_chr_drv_ctrlpln.ko
```

Note: Root privileges are required to execute this command.

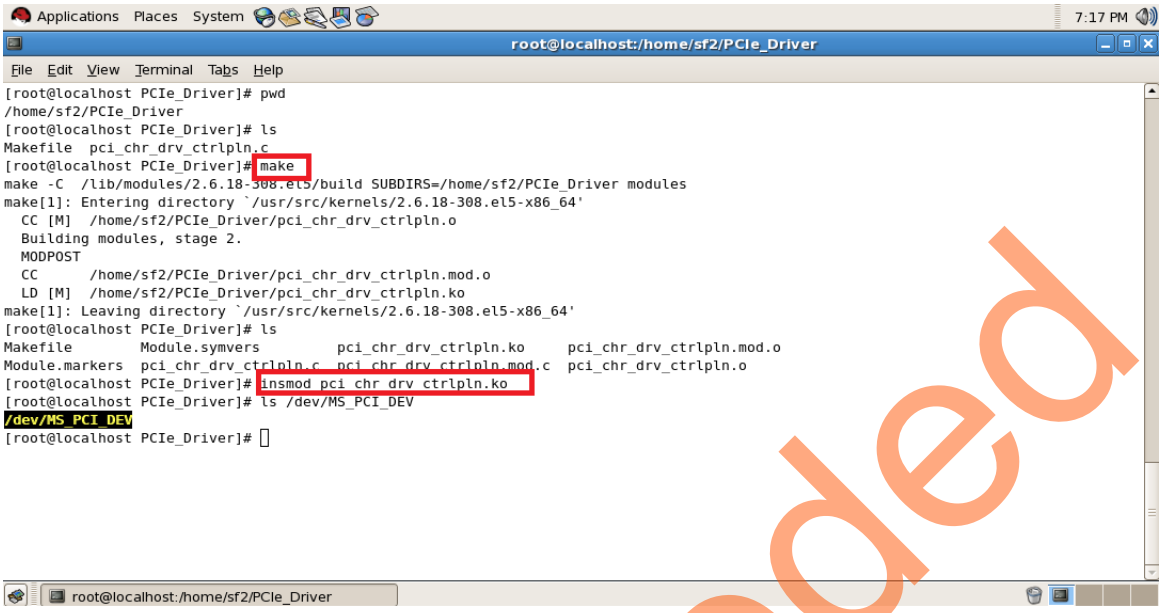


Figure 80 • **PCIe Device Driver Installation**

12. After successful Linux PCIe device driver installation, check `/dev/MS_PCI_DEV` got created by using the following Linux command:

    ```
    #ls /dev/MS_PCI_DEV
    ```

Note: `/dev/MS_PCI_DEV` interface is used to access the SmartFusion2 PCIe end point from Linux user space.

Linux PCIe Application Compilation and PCIe Control Plane Utility Creation

1. Change to the */home/sf2/* directory using the following command:

   ```
   #cd /home/sf2
   ```

2. Copy the *M2S90_PCIE_Control_DEMO_DF\Linux_64bit\Util\PCIe_App* folder from the Windows host PC and place it into the /home/sf2 directory of RedHat Linux host PC.
3. Change to the **/home/sf2/PCIe_App** directory using the following command:

   ```
   #cd /home/sf2/PCIe_App
   ```

4. Compile the Linux user space application `pcie_appln_ctrlpln.c` by using `gcc` command.

   ```
   #gcc -o pcie_ctrlplane pcie_appln_ctrlpln.c
   ```

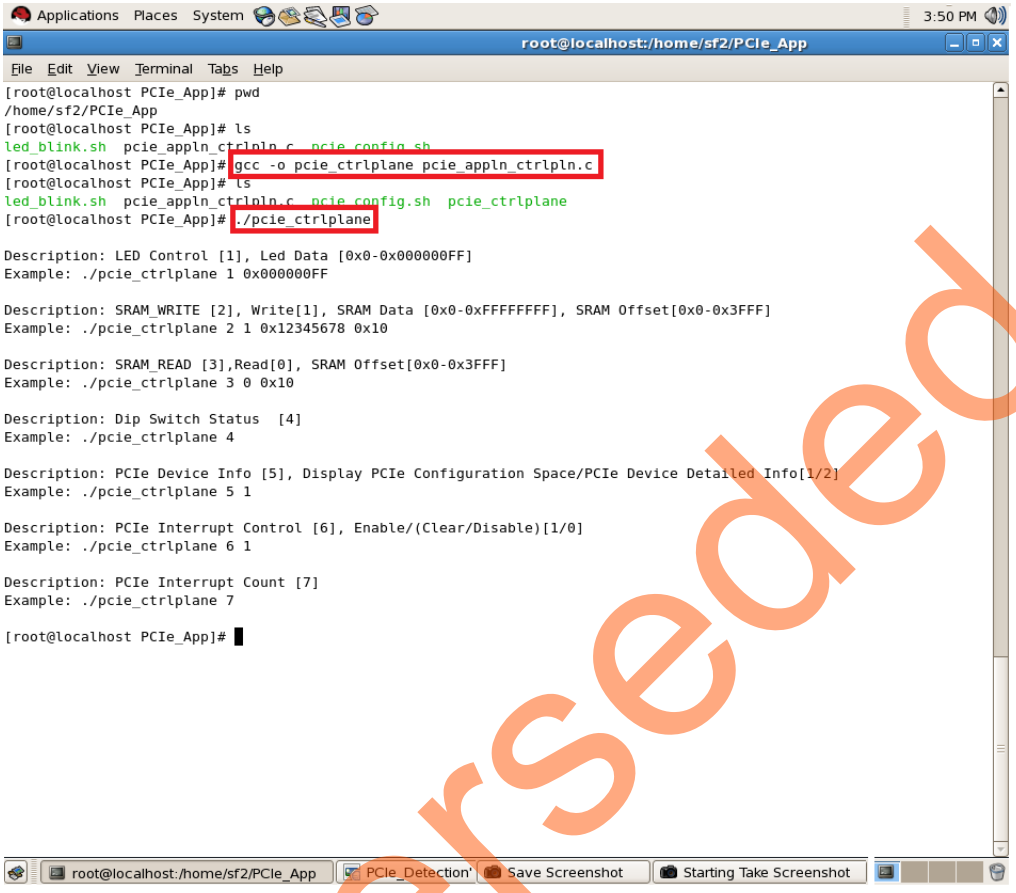


Figure 81 • **Linux PCIe Application Utility**

5. After successful compilation, Linux PCIe application utility `pcie_ctrlplane` creates in the same directory.
6. On Linux Command Prompt run the `pcie_ctrlplane` utility as:

   `#./pcie_ctrlplane`

Help menu displays as shown in Figure 81.

Execution of Linux PCIe Control Plane Features

LED Control

LED1 to LED8 is controlled by writing data to SmartFusion2 LED Control Registers.

```
#./pcie_ctrlplane 1 0x000000FF [LED OFF]
#./pcie_ctrlplane 1 0x00000000 [LED ON]
```

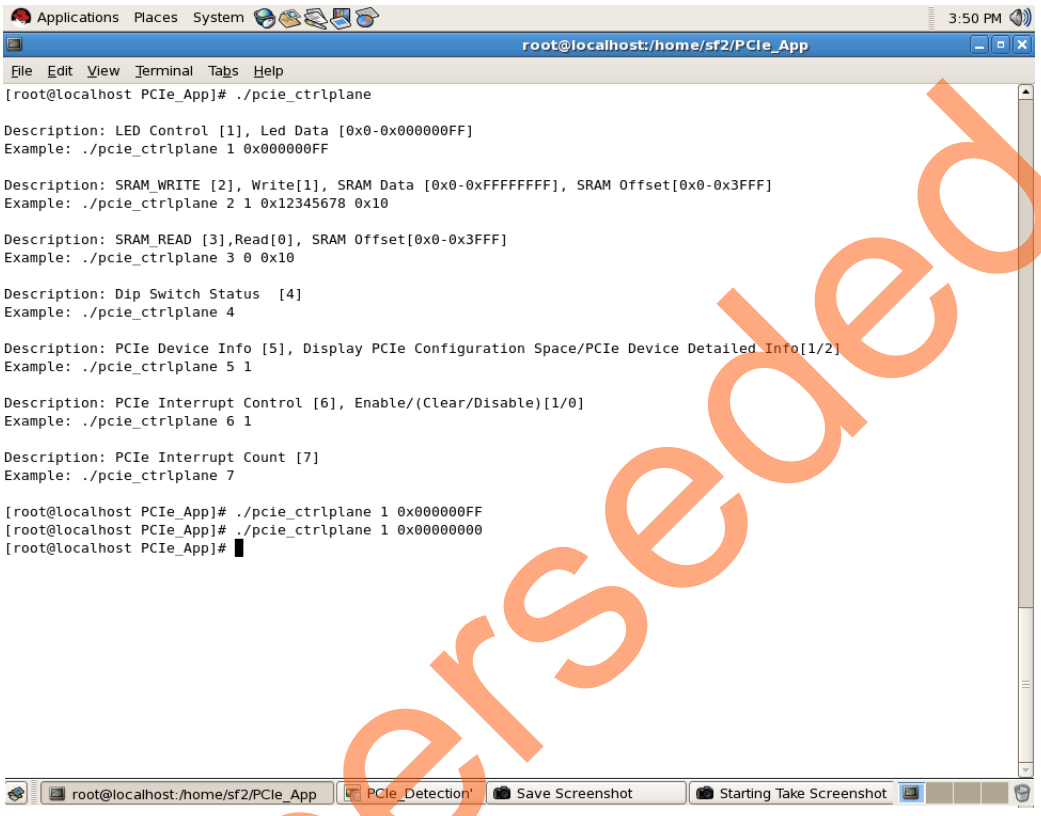


Figure 82 • **Linux Command - LED Control**

`led_blink.sh`, contains the shell script code to perform LED Walk ON where as `Ctrl C` exits the shell script and LED Walk turns OFF.

Run the led_blink.sh shell script using `sh` command.

```
#sh led_blink.sh
```

SRAM Read/Write

64 KB SRAM is accessible for SmartFusion2 Security Evaluation Kit.

```
#./pcie_ctrlplane 2 1 0xFF00FF00 0x1000 [SRAM WRITE]
#./pcie_ctrlplane 3 0 0x1000 [SRAM READ]
```

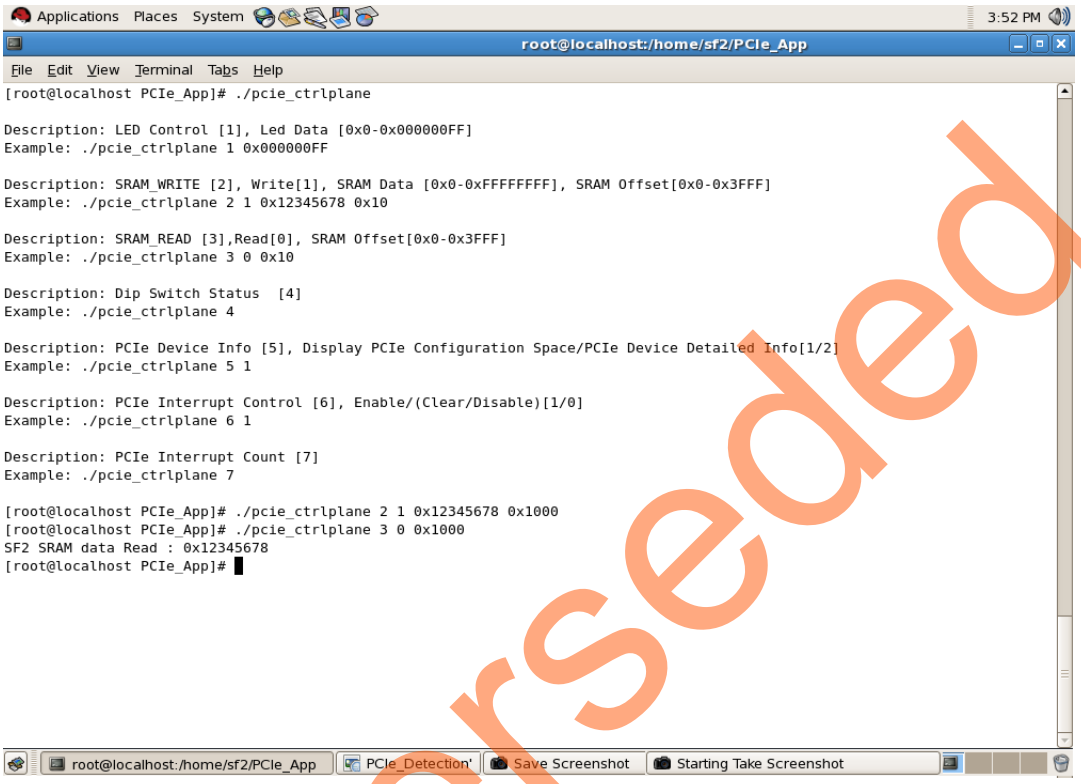


Figure 83 • **Linux Command - SRAM Read/Write**

DIP Switch Status

Dip Switch on SmartFusion2 Security Evaluation Kit consists of four electric switches to hold the device configurations. Linux PCIe utility reads the corresponding switches (ON/OFF) state.

```
#./pcie_ctrlplane 4 [DIP Switch Status]
```

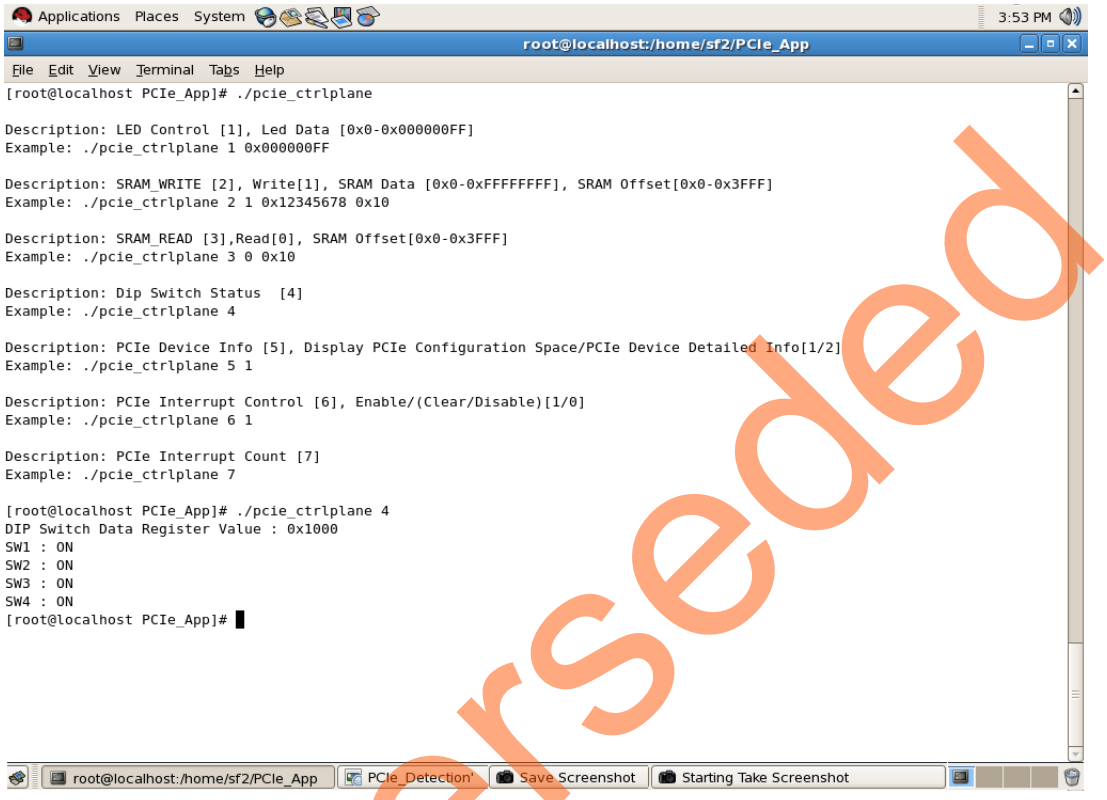


Figure 84 • **Linux Command - DIP Switch**

PCIe Configuration Space Display

PCIe Configuration Space contains the PCIe device data, such as Vendor ID, Device ID, and Base Address 0.

Note: Root Privileges are required to execute this command.

```
#./pcie_ctrlplane 5 1 [Read PCIe Configuration Space]
```

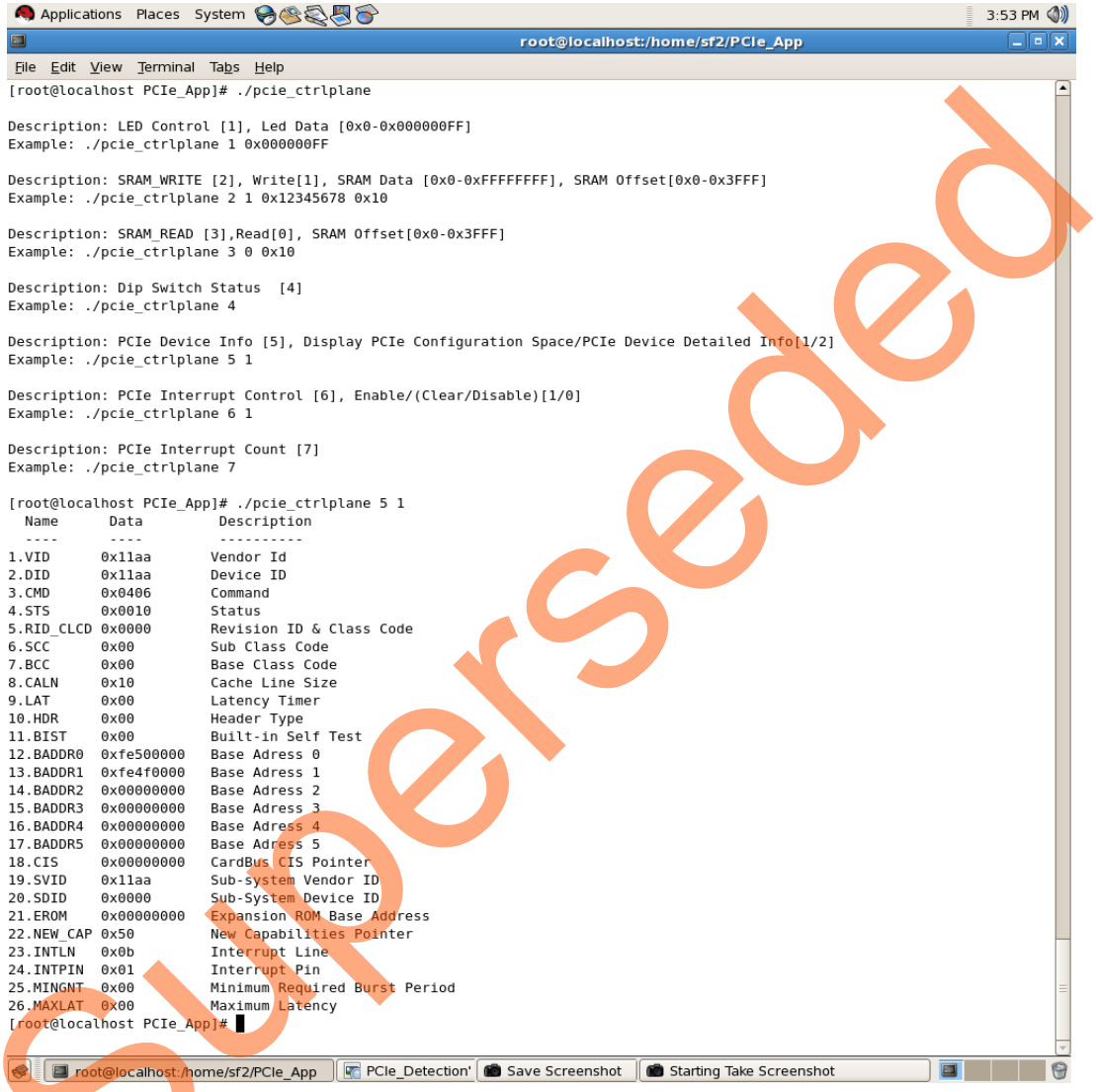


Figure 85 • **Linux Command - PCIe Configuration Space Display**

PCIe Link Speed and Width

Note: Root Privileges are required to execute this command.

```
#./pcie_ctrlplane 5 2 [Read PCIe Link Speed and Link Width]
```

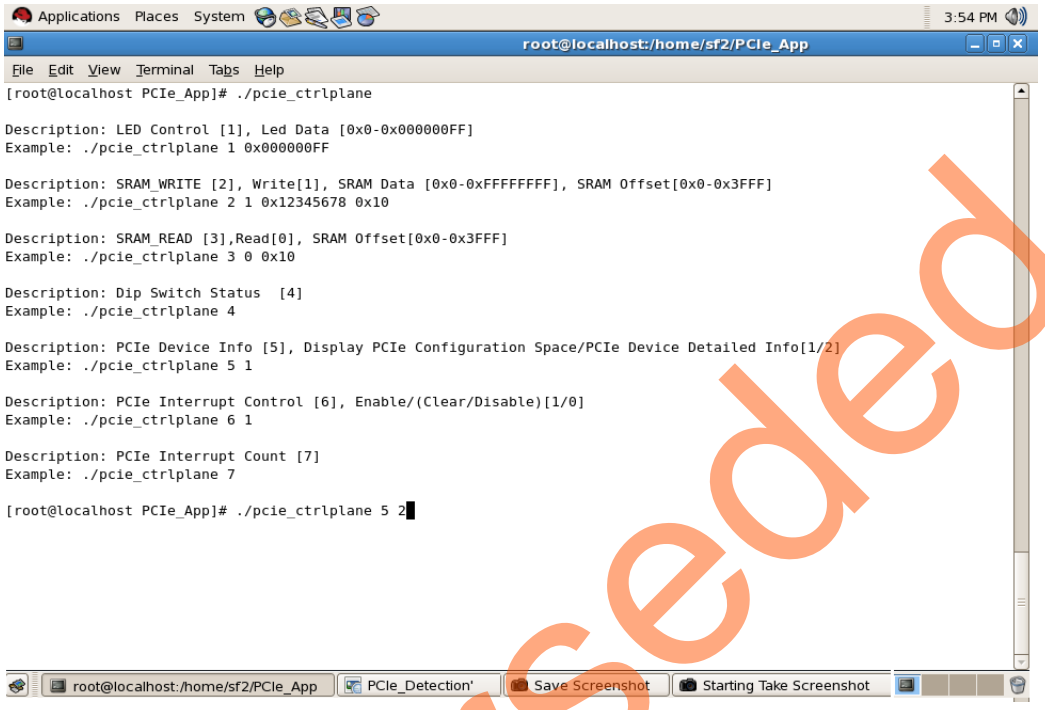


Figure 86 • **Linux Command - PCIe Link Speed and Width**

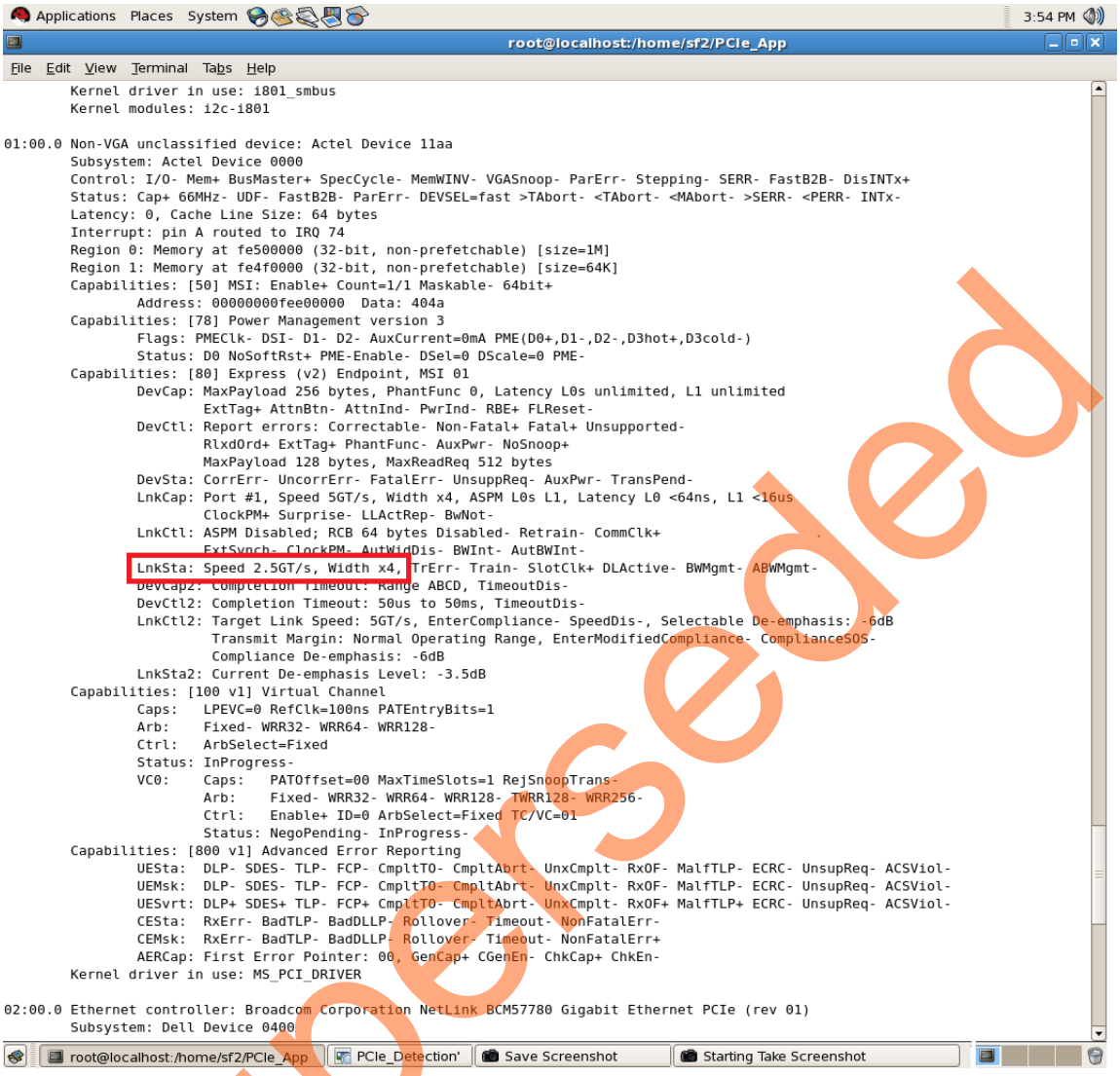


***Figure 87* • Linux Command - PCIe Link Speed and Width**

PCIe Interrupt Control (Enable/Disable) and Interrupt Counter

SmartFusion2 Security Evaluation Kit enable or disable the MSI interrupts by writing data to its PCIe configuration space.

Interrupt Counter holds the number of MSI interrupts got triggered by pressing the SW3 Push button.

```
#. /pcie_ctrlplane 6 0 [Disable Interrupts]
#. /pcie_ctrlplane 6 1 [Enable Interrupts]
#. /pcie_ctrlplane 7 [Interrupt Counter Value]
```

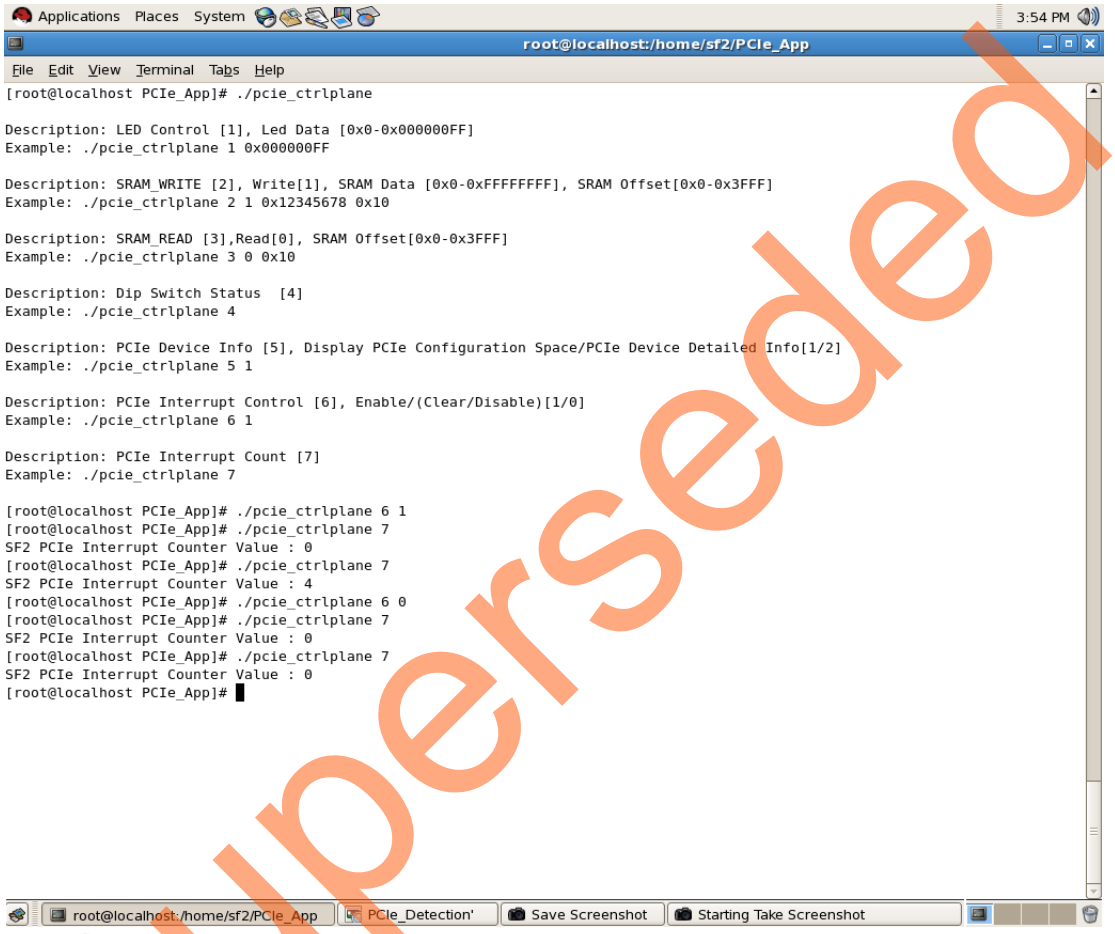


Figure 88 • **Linux Command - PCIe Interrupt Control**

Conclusion

This demo describes how to access the PCIe endpoint features of SmartFusion2, create a simple design, and verify the design using BFM simulation. This demo demonstrates that the Host PC can easily communicate with the SmartFusion2 Security Evaluation Kit through the provided GUI and Drivers. This demo also provides a Linux PCIe application for accessing PCIe EP device through Linux PCIe Device Driver.

List of Changes

The following table lists the important changes that were made in each revision of the chapter in the demo guide.

Date	Changes	Page
Revision 5 (November 2015)	Updated the document for Libero v11.6 software release (SAR 73139).	NA
Revision 4 (February 2015)	Updated the document for Libero v11.5 software release (SAR 64184).	NA
Revision 3 (August 2014)	Updated the document for Libero v11.4 software release (SAR 59644).	NA
Revision 2 (April 2014)	Updated the document for Libero v11.3 software release (SAR 56081).	NA
Revision 1 (December 2013)	Updated the document for Libero v11.2 software release (SAR 52109) (SAR 52909) and (SAR 50779).	NA
Revision 0 (June 2013)	Initial Release	NA

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, **650.318.8044**

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

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

Website

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

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

ITAR Technical Support

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

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

Within the USA: +1 (800) 713-4113
Outside the USA: +1 (949) 380-6100
Sales: +1 (949) 380-6136
Fax: +1 (949) 215-4996

E-mail: sales.support@microsemi.com

Microsemi Corporation (Nasdaq: MSCC) offers a comprehensive portfolio of semiconductor and system solutions for communications, defense & security, aerospace and industrial markets. Products include high-performance and radiation-hardened analog mixed-signal integrated circuits, FPGAs, SoCs and ASICs; power management products; timing and synchronization devices and precise time solutions, setting the world's standard for time; voice processing devices; RF solutions; discrete components; security technologies and scalable anti-tamper products; Ethernet solutions; Power-over-Ethernet ICs and midspans; as well as custom design capabilities and services. Microsemi is headquartered in Aliso Viejo, Calif., and has approximately 3,600 employees globally. Learn more at **www.microsemi.com**.

50200456-5/11.15