

DG0584
Demo Guide
SmartFusion2 SoC FPGA In-Application Programming
Using PCIe Interface - Libero SoC v11.8



Microsemi Corporate Headquarters
One Enterprise, Aliso Viejo,
CA 92656 USA
Within the USA: +1 (800) 713-4113
Outside the USA: +1 (949) 380-6100
Fax: +1 (949) 215-4996
Email: sales.support@microsemi.com
www.microsemi.com

© 2017 Microsemi Corporation. All rights reserved. Microsemi and the Microsemi logo are trademarks of Microsemi Corporation. All other trademarks and service marks are the property of their respective owners.

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

About Microsemi

Microsemi Corporation (Nasdaq: MSCC) offers a comprehensive portfolio of semiconductor and system solutions for aerospace & defense, communications, data center and industrial markets. Products include high-performance and radiation-hardened analog mixed-signal integrated circuits, FPGAs, SoCs and ASICs; power management products; timing and synchronization devices and precise time solutions, setting the world's standard for time; voice processing devices; RF solutions; discrete components; enterprise storage and communication solutions, security technologies and scalable anti-tamper products; Ethernet solutions; Power-over-Ethernet ICs and midspans; as well as custom design capabilities and services. Microsemi is headquartered in Aliso Viejo, California, and has approximately 4,800 employees globally. Learn more at www.microsemi.com.

Contents

1	Revision History	1
1.1	Revision 5.0	1
1.2	Revision 4.0	1
1.3	Revision 3.0	1
1.4	Revision 2.0	1
1.5	Revision 1.0	1
2	In-Application Programming Using PCIe Interface	2
2.1	Design Requirements	2
2.2	Demo Design	2
2.2.1	Features	4
2.2.2	Description	4
2.3	Running GUI Application on Host PC	5
2.3.1	Programming Files	5
2.3.2	IAP Execution Flow	6
2.4	Setting Up the Demo Design	7
2.4.1	Programming the SmartFusion2 Security Evaluation Kit Board	8
2.5	Connecting the SmartFusion2 Security Evaluation Kit Board to Host PC	10
2.6	Drivers Installation	12
2.6.1	Installing PCIe_Demo Application GUI	13
2.7	Running the Demo Design	14
2.8	IAP Step1: Loading SPI Flash with Programming Bitstream	15
2.9	IAP Step2: Initiating the IAP Services	17
2.9.1	Authenticate and Program Operation Mode	17
2.9.2	Verify Operation Mode	21
2.10	Known Issue	22
3	Appendix: SmartFusion2 Security Evaluation Kit Board	23
4	Appendix: Error Codes	24
5	Appendix: Generating .spi Programming File using Libero	25
6	Appendix: Hardware Implementation	27
6.1	Standby Clock Source Configuration	28
6.2	Configuring I/Os for Flash*Freeze Mode	29
6.3	SoftConsole Project Generation	29
7	Appendix: Implementing Workaround to Access Fabric LSRAM after IAP/ISP Program Operation	31
7.1	Changes Required in Libero Design	31
7.1.1	Option 1: Creating SmartDesign	31
7.1.2	Option 2: Importing the .cxf file in Libero Design	35

Figures

Figure 1	Demo Design Files Top-Level Structure	3
Figure 2	Top-Level Demo Block Diagram	3
Figure 3	IAP GUI Application	5
Figure 4	IAP Execution Flow	7
Figure 5	FlashPro New Project	8
Figure 6	FlashPro4 Programmer Type	9
Figure 7	FlashPro Project Configuration	9
Figure 8	FlashPro Programming Passed	10
Figure 9	Device Manager - PCIe Device Detection	11
Figure 10	Uninstall Jungo Driver	11
Figure 11	Confirm Device Uninstall Dialog	12
Figure 12	Jungo Driver Installation	12
Figure 13	Windows Security	12
Figure 14	GUI Installation	13
Figure 15	Successful Installation of GUI	13
Figure 16	Device Manager - PCIe Device Detection	14
Figure 17	PCIe Demo GUI	14
Figure 18	PCIe Device Information	15
Figure 19	Selecting Programming File in PCIe_Demo Application	16
Figure 20	SPI Flash Programming	16
Figure 21	IAP Authentication Status	17
Figure 22	Authentication Success Message in PCIe Demo GUI	18
Figure 23	Authentication Failed	18
Figure 24	IAP Programming Status	19
Figure 25	Microsemi PCIe Device Found Message in PCIe Demo GUI	20
Figure 26	IAP Verification Status	21
Figure 27	IAP Verification Success	21
Figure 28	IAP Verification Error Message	22
Figure 29	SmartFusion2 Security Evaluation Kit Board	23
Figure 30	Configuring Export Bitstream	25
Figure 31	Export Bitstream Window	25
Figure 32	SPI File Location	26
Figure 33	Demo Design Libero Top-Level Diagram	27
Figure 34	Flash*Freeze Hardware Settings	28
Figure 35	Configuring SPI_0 Ports Available During F*F	29
Figure 36	Export Firmware Options	29
Figure 37	SoftConsole Project Workspace	30
Figure 38	Tamper Macro - Before Configuration	31
Figure 39	Tamper Macro Configuration Window	32
Figure 40	Tamper Macro - After Configuration	32
Figure 41	Ram_interafce FSM Component	33
Figure 42	Two-Port SRAM Configurator Window	34
Figure 43	Dev_Restart_after_IAP_blk SmartDesign	34
Figure 44	PCIE_IAP_top SmartDesign	35

Tables

Table 1	Design Requirements	2
Table 2	SmartFusion2 Security Evaluation Kit Jumper Settings	8
Table 3	IAP Programming Results	20
Table 4	Error Codes	24

1 Revision History

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

1.1 Revision 5.0

Updated the document for Libero v11.8 software release.

1.2 Revision 4.0

Updated the document for Libero v11.6 software release (SAR 71564).

1.3 Revision 3.0

Updated the document for Libero v11.5 software release (SAR 68427).

1.4 Revision 2.0

The following changes are done in revision 2.0 of this document.

- Added [Appendix: Implementing Workaround to Access Fabric LSRAM after IAP/ISP Program Operation](#), page 31.
- Added [Known Issue](#), page 22.

1.5 Revision 1.0

Revision 1.0 was the first publication of this document.

2 In-Application Programming Using PCIe Interface

In-application programming (IAP) is a SmartFusion2 device programming feature used to reprogram the device to accommodate design iterations and field upgrades. Using the IAP feature the application can re-program the flash components of the SmartFusion2 devices. SmartFusion2 devices support IAP using a number of different interfaces. In this demo design, PCIe is used to transfer new programming data. This document describes how to program SmartFusion2 devices through the PCIe interface, and use SmartFusion2 system controller services.

For more information about the different programming modes supported by SmartFusion2 SoC FPGAs, see the [UG0451: IGLOO2 and SmartFusion2 Programming User Guide](#). For more information about system controller programming services, see the [UG0450: SmartFusion2 SoC and IGLOO2 FPGA System Controller User Guide](#).

2.1 Design Requirements

The following table lists the hardware and software design required to run the design.

Table 1 • Design Requirements

Design Requirements	Description
Hardware	
SmartFusion2 Security Evaluation Kit (M2S090TS):	Rev D or later
– 12 V adapter (provided along with the kit)	
– USB A to Mini-B cable (provided along with the kit)	
Host PC or Laptop	Any 64-bit Windows
Software	
Libero® System-on-Chip (SoC)	v11.8
FlashPro programming software	v11.8
Softconsole	4.0
Host PC Drivers (provided along with the design files)	–
PCIe Demo application	PCIe Demo GUI Installer

2.2 Demo Design

The demo design files are available for download at:

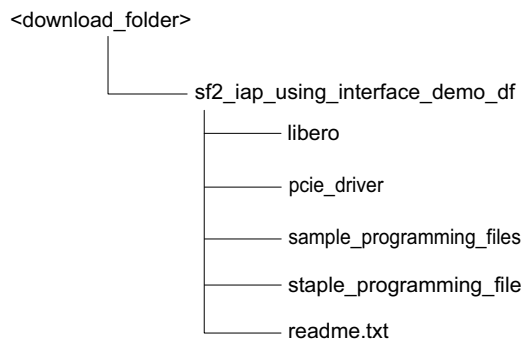
http://soc.microsemi.com/download/rsc/?f=m2s_dg0584_liberov11p8_df

The demo design files include:

- Libero SoC software project
- PCIe driver
- STAPL programming file
- Sample programming files
- Readme file

The following figure shows the top-level structure of the design files. For more information, see the `Readme.txt` file.

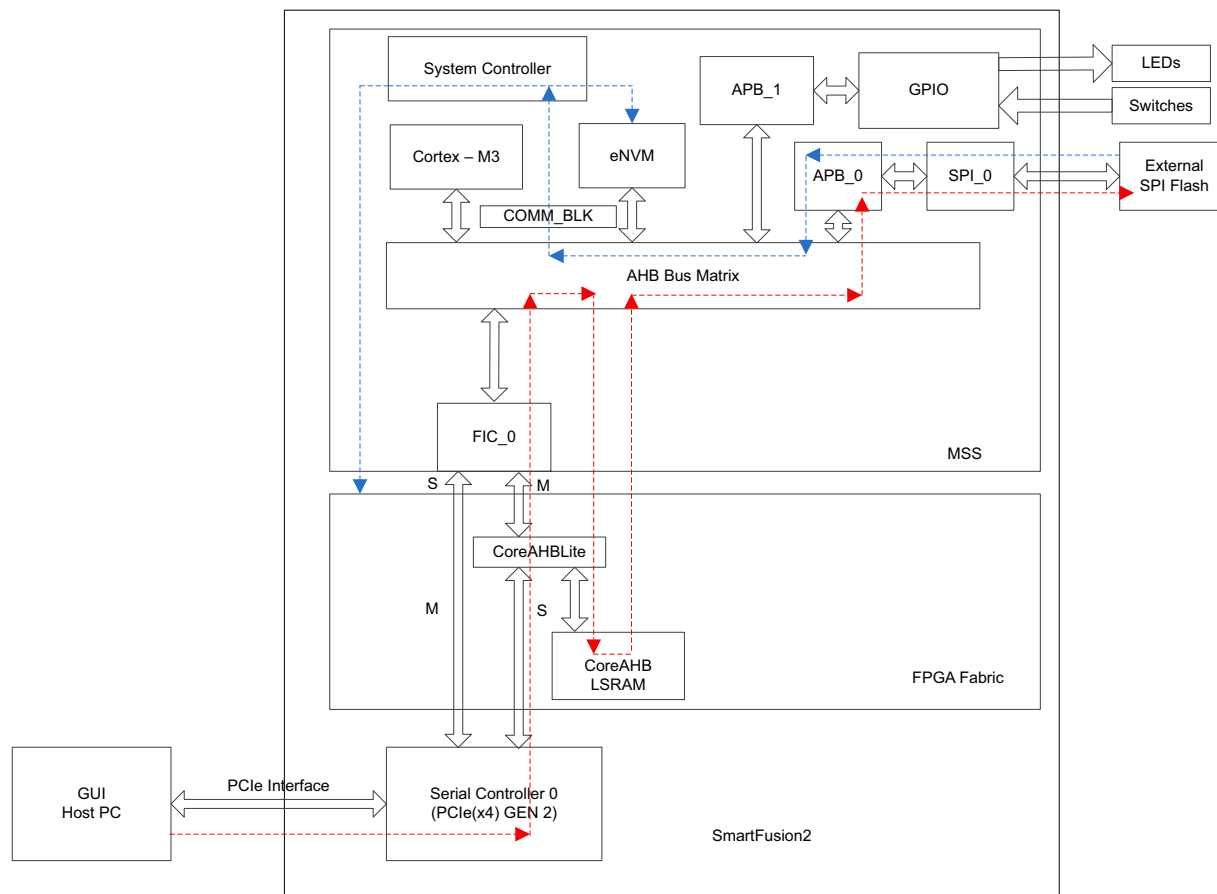
Figure 1 • Demo Design Files Top-Level Structure



The following figure describes the demo design. The arrows in red show the data flow between the host PC and the on-board external serial peripheral interface (SPI) flash memory using the PCIe interface. The ARM® Cortex®-M3 processor copies the programming data from the host PC to the SPI flash by using the large static random access memory (LSRAM) memory as a temporary buffer.

The lines in blue show the system controller reading data from the external SPI flash memory to program the SmartFusion2 device. For more information about IAP process, see [Description](#), page 4.

Figure 2 • Top-Level Demo Block Diagram



- Step1** Transferring data bitstream from host PC to external Flash through PCIe interface
- Step2** System controller reads data bitstream from the external flash to program the SmartFusion2 device

2.2.1 Features

The demo design performs the following types of programming based on the input provided by the programming file:

- **eNVM programming:** The IAP programming service programs only eNVM. In this case, the input programming file has only eNVM content.
- **FPGA Fabric programming:** The IAP programming service programs only the FPGA fabric. In this case, the input programming file has only the FPGA fabric content.
- **eNVM and FPGA Fabric programming:** The IAP programming service programs both the FPGA fabric and eNVM. In this case, the input programming file has both the FPGA fabric and eNVM content.

2.2.2 Description

IAP in SmartFusion2 devices is a two-step process as follows:

Step1: Loading SPI Flash with Programming Bitstream

The Cortex-M3 processor receives bitstream data in blocks of 2 KB from the host PC through the PCIe interface, and stores the bitstream data in the LSRAM 2 KB temporary buffer. The Cortex-M3 processor reads the bitstream data stored in the LSRAM temporary buffer and writes to the external SPI Flash attached to the microcontroller subsystem (MSS) SPI_0 controller. The Cortex-M3 processor continues to receive blocks of 2 KB bitstream data through the PCIe interface until the entire programming file gets transferred from the host PC to the external SPI Flash.

Step 2: Initiating the IAP Service

The bitstream data is verified by requesting the AUTHENTICATE IAP service from the system controller. The system controller reads the bitstream data from the external SPI Flash using the SPI interface to check the data integrity of the bitstream data. During authentication, the rest of the device functions normally. On successful authentication, the Cortex-M3 processor initiates a PROGRAM IAP system controller system service. The system controller fetches the bitstream data from the SPI Flash and programs the flash components of the SmartFusion2 device. IAP can be used to program the FPGA fabric, eNVM, or both FPGA fabric and eNVM.

The Cortex-M3 processor in the SmartFusion2 device can execute an application image from embedded SRAM (eSRAM), eNVM or DDR/SDR memories. For more information about remapping techniques, see [AC390: SmartFusion2 SoC FPGA Remapping eNVM, eSRAM, and DDR/SDR SDRAM Memories Application Note](#). In this demo design, the Cortex-M3 processor executes the IAP application image from eSRAM while the eNVM programming takes place, that is, during the Program operation mode. To execute the application image from eSRAM, the Cortex-M3 processor copies the IAP application image that resides in eNVM data client to eSRAM and remaps the eSRAM to the Cortex-M3 processor code region. For the Verify and Authenticate operation modes, the application image can be executed from either eNVM or eSRAM as the eNVM programming does not take place.

The system controller executes the IAP programming service in the following modes:

- **Authenticate:** The system controller IAP service validates the integrity of the programming bitstream.
 - For security and reliability reasons, Microsemi recommends that the bitstream must be authenticated before the program is executed, using the Authenticate Operation mode. The SmartFusion2 device application must commit the bitstream for programming, only after successful authentication and validation of the integrity of the bitstream.
- **Program:** Depending on the programming bitstream, system controller IAP service programs the following:
 - eNVM
 - FPGA fabric
 - Both eNVM and FPGA fabric
- **Verify:** The system controller IAP service verifies the contents of the SmartFusion2 device against the programming bitstream data stored in the SPI Flash.

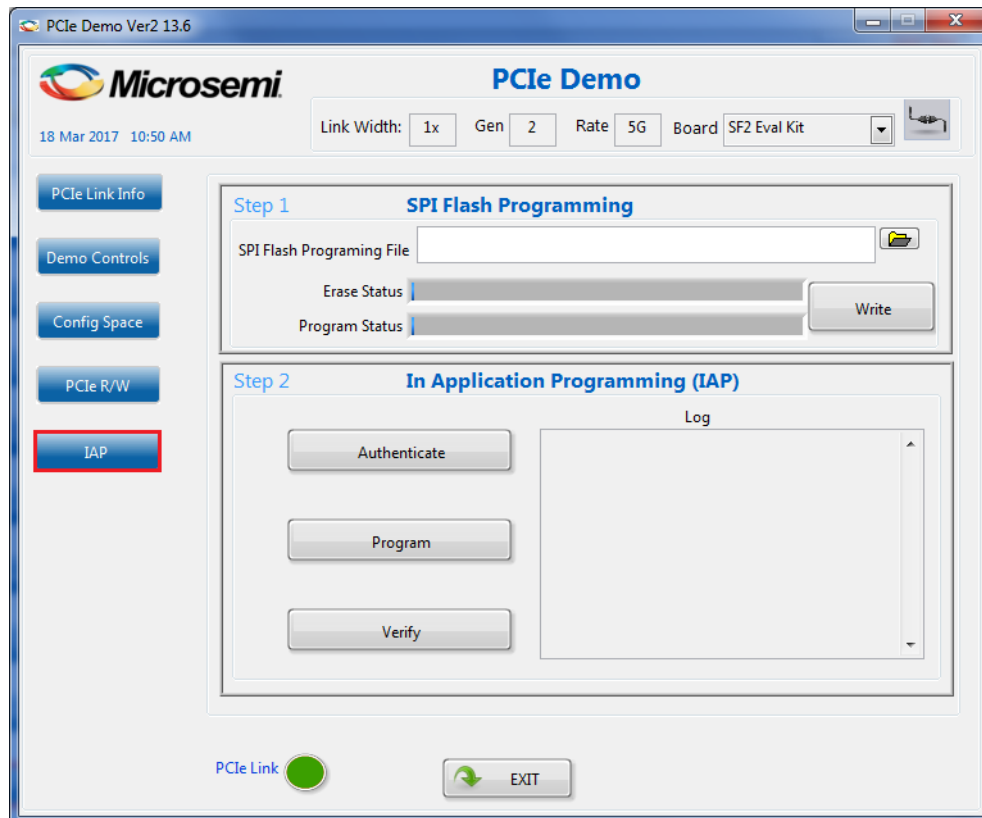
Note: The FPGA fabric is not operational during Program or Verify operations as the device enters into Flash*Freeze (F*F) mode. During Program or Verify operations, the PCIe communication link is in reset

state. On completion of the Verify operation, the PCIe communication link is up. On completion of the Program operation, the PCIe communication link is restored to the active state when the device is programmed with the PCIe-enabled programming file. For more information on hardware implementation, see [Appendix: Hardware Implementation](#), page 27.

2.3 Running GUI Application on Host PC

The GUI application is an executable program running on the host PC that transfers the programming file (*.spi) from the host PC to the SmartFusion2 Security Evaluation Kit on-board SPI Flash through the PCI interface. The GUI also allows the user to perform the IAP operations (Authenticate, Program, and Verify) by clicking the corresponding options, as shown in the following figure.

Figure 3 • IAP GUI Application



2.3.1 Programming Files

Sample programming files with the file extension .spi are provided to program the following:

- eNVM
- FPGA fabric
- Both eNVM and FPGA fabric

The folder `<download_folder>\sf2_iap_using_interface_demo_df\sample_programming_files` contains the following sample programming files:

- `iap_envm_only.spi`: Programs only eNVM. The eNVM client has a light-emitting diode (LED) counter logic.
- `iap_fabric_only.spi`: Programs only the FPGA fabric. The FPGA fabric has an LED blinking logic.
- `iap_fabric_and_envm.spi`: Programs both the FPGA fabric and eNVM. The eNVM client has an LED counter logic and the FPGA fabric has an LED blinking logic. The folder `<download_folder>\sf2_iap_using_interface_demo_df\sample_programming_files\fabric_and_envm` contains the Libero design to generate this sample programming file.

- `pcie_iap_top.spi`: This is the .spi file format version of the `pcie_iap_top.stp` file provided in `<download_folder>\sf2_iap_using_interface_demo_df1stapl_programming_file`.

Note: For more information on generating .spi programming files, see [Appendix: Generating .spi Programming File using Libero](#), page 25.

2.3.2 IAP Execution Flow

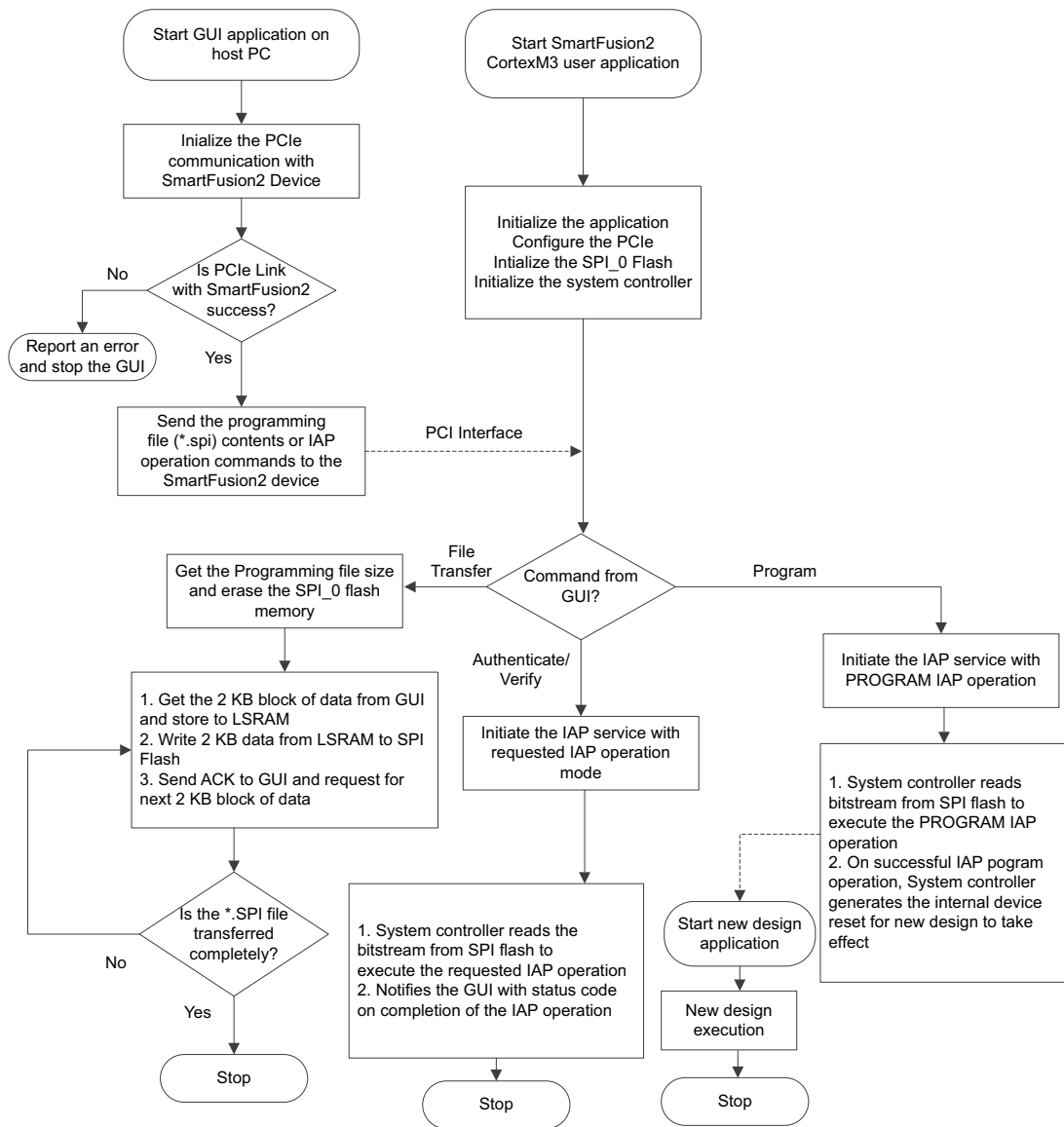
Browse to the .spi programming file and click WRITE in the GUI. The GUI application running on the host PC starts communicating with the SmartFusion2 device using the PCIe interface. On connecting with the SmartFusion2 device, the GUI sends the programming file size to the Cortex-M3 processor user application and requests to erase the external SPI Flash contents available on the SmartFusion2 Security Evaluation Kit for storing the bitstream data. The Cortex-M3 processor user application receives the bitstream data from the GUI through the PCIe interface and stores the data into the SPI Flash. The GUI application copies 2 KB bitstream data from the programming file to the host PC buffer. The Cortex-M3 processor user application reads the bitstream data from the host PC buffer through the PCIe interface and copies the data to LSRAM 2 KB temporary buffer. The Cortex-M3 processor writes the SPI Flash memory with the data bitstream stored in the LSRAM. The Cortex-M3 processor user application sends the acknowledgment to the GUI for every 2 KB block of bitstream data and requests the GUI for the next block of 2 KB data. The GUI sends the bitstream data in 2 KB blocks until the entire programming file is transferred from the host PC to the external SPI Flash.

The IAP services can be executed using the GUI. If the Authenticate or Verify option is selected from the GUI, the Cortex-M3 processor user application initiates the IAP service with the requested IAP Operation mode and notifies the GUI with a status code indicating the completion of the authentication or verification service. For Program mode, the Cortex-M3 user application does not notify the GUI with any status code as the Flash components of the device are programmed with new bitstream data. On successful IAP program operation, an internal device reset is generated for the new design to take effect.

Note: You can modify/edit the programming file (*.spi) contents and run the authenticate operation to get the authentication fail message with the corresponding error code. For information about error codes, see [Appendix: Error Codes](#), page 24.

The following figure shows the IAP execution flow.

Figure 4 • IAP Execution Flow



2.4 Setting Up the Demo Design

The following steps describe how to setup the demo design:

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 the following table.

CAUTION: Switch **OFF SW7** while connecting the jumpers.

Note: Snapshots of the SmartFusion2 Security Evaluation Kit board with the complete set up is given in the [Appendix: SmartFusion2 Security Evaluation Kit Board](#), page 23.

Table 2 • SmartFusion2 Security Evaluation Kit Jumper Settings

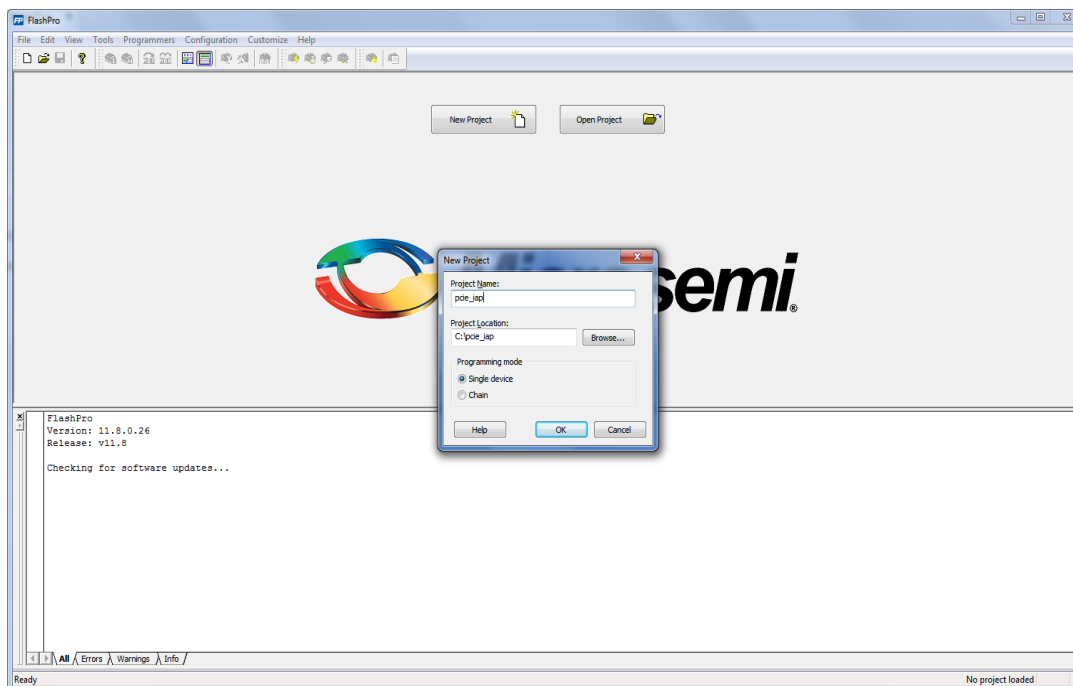
Jumper	Pin From	Pin To	Comments
J22	1	2	Default
J23	1	2	Default
J24	1	2	Default
J8	1	2	Default
J3	1	2	Default

2.4.1 Programming the SmartFusion2 Security Evaluation Kit Board

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

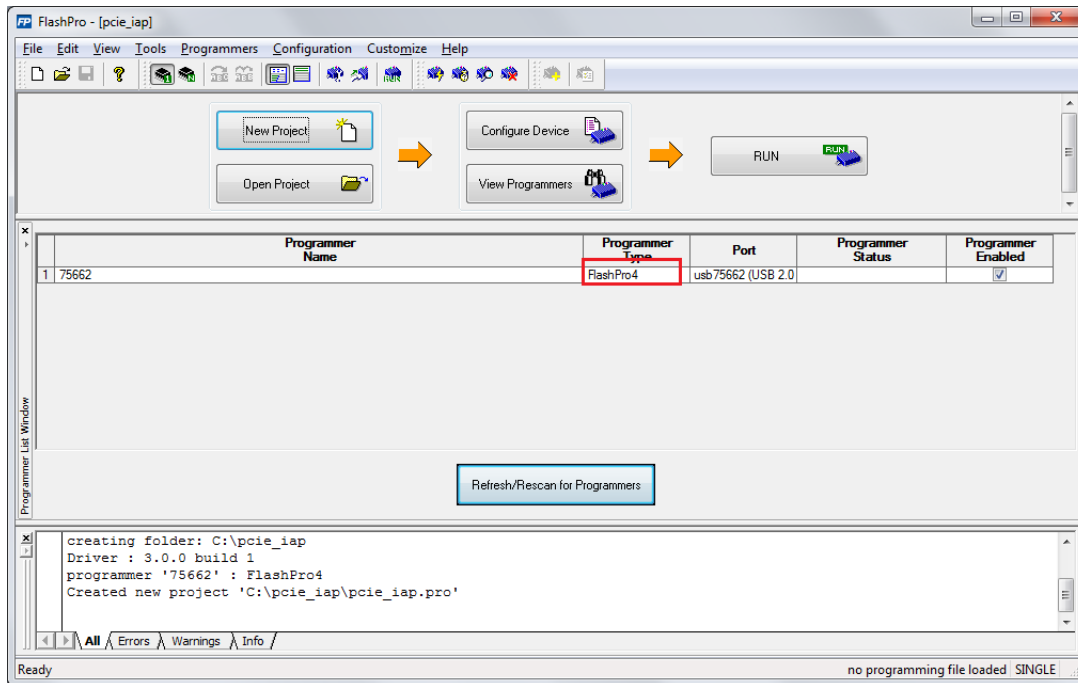
1. Download the demo design from:
http://soc.microsemi.com/download/rsc/?f=m2s_dg0584_liberov11p8_df
2. Switch **ON** the power supply switch, SW7.
3. Launch the FlashPro software.
4. Click New Project.
5. In the **New Project** window, enter the project name as **pcie_iap**.
6. Click Browse and navigate to the location where you want to save the project.
7. Select Single device as Programming mode.
8. Click **OK** to save the project.

Figure 5 • FlashPro New Project



The following figure shows the programmer type as FlashPro4.

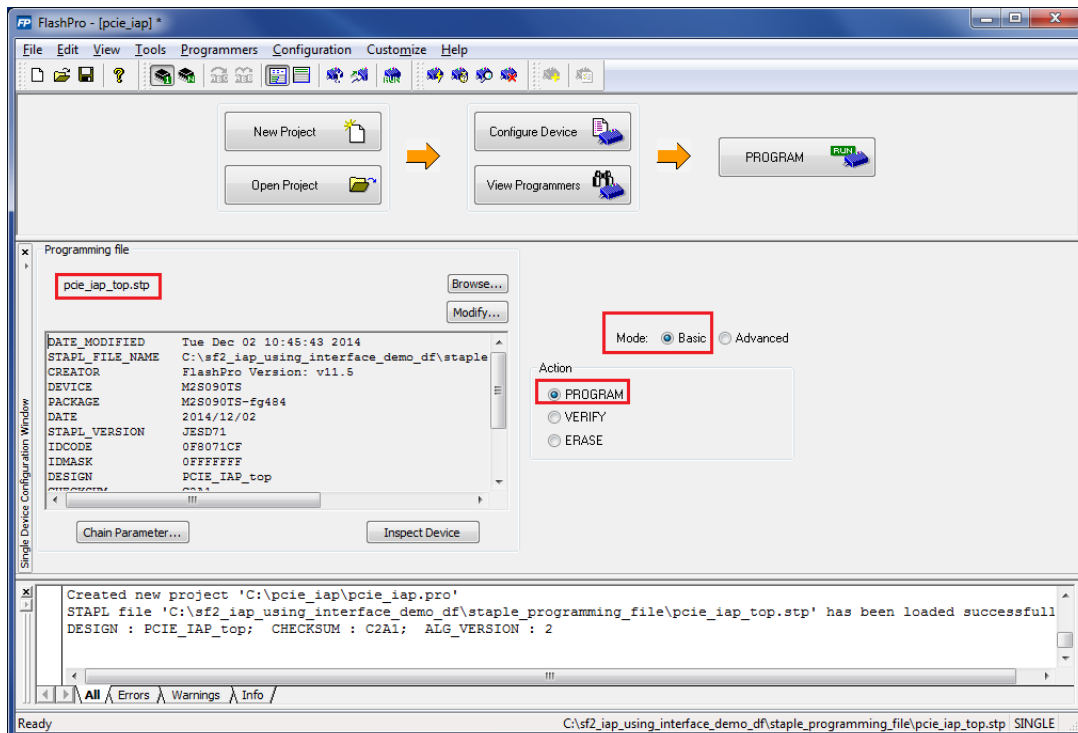
Figure 6 • FlashPro4 Programmer Type



The following steps describe how to configure the device:

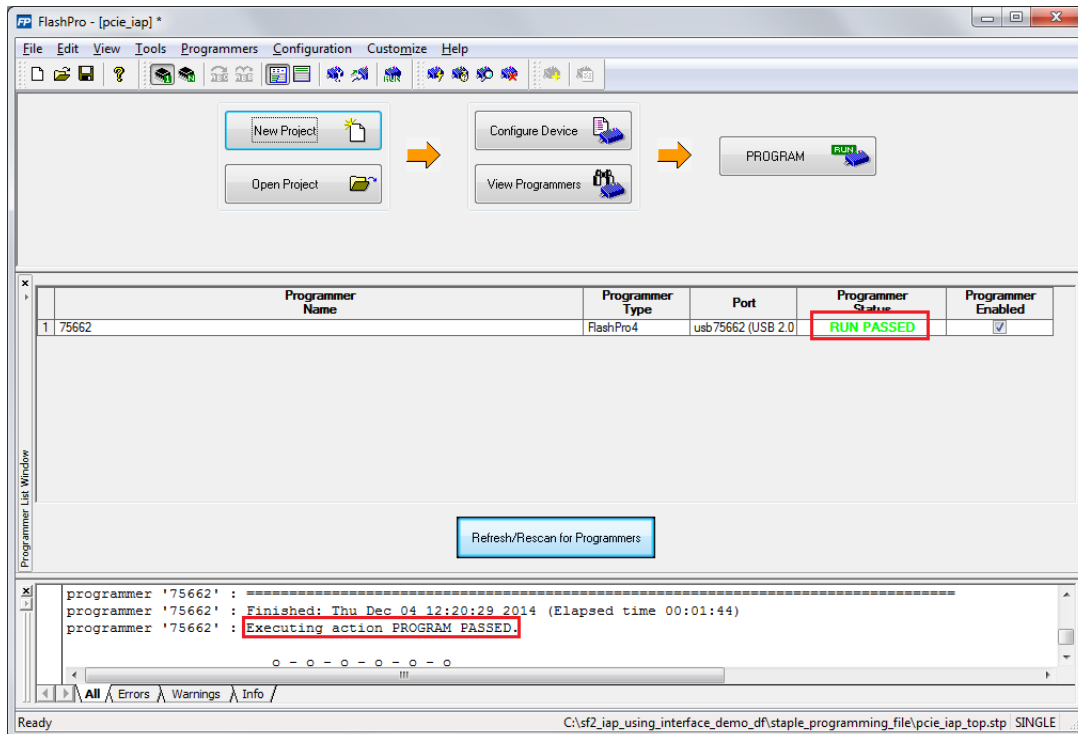
1. Click **Configure Device**.
2. Click **Browse** and navigate to the location where pcie_iap_top.stp is located and select the file. The default location is: <download_folder>\sf2_iap_using_interface_demo_df\stapl_programming_file
3. Click **Open**. The required programming file is selected and is ready to be programmed in the device.

Figure 7 • FlashPro Project Configuration



- Click PROGRAM to start programming the device. Wait until the Programmer Status is changed to RUN PASSED, as shown in the following figure.

Figure 8 • FlashPro Programming Passed



2.5 Connecting the SmartFusion2 Security Evaluation Kit Board to Host PC

The following steps describe how to connect the Security Evaluation Kit board to the host PC:

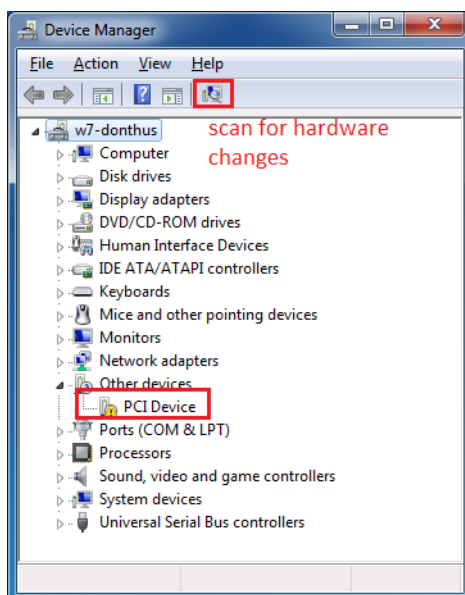
- After successful programming, switch OFF the SmartFusion2 Security Evaluation Kit board and shut down the host PC.
- The following steps describe how to connect the CON1–PCIe Edge Connector either to the host PC or laptop:
 - Connect the CON1–PCIe Edge Connector to the host PC PCIe Gen2 slot or Gen1 slot. This demo is designed to run in any PCIe Gen2 compliant slot. If the host PC does not support the Gen2 compliant slot, the design switches to the Gen1 mode.
 - 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 support only Gen1 and the design works on Gen1 mode.

Note: The host PC or laptop must be switched OFF while inserting the PCIe Edge Connector. If the system is not switched OFF, the PCIe device detection and selection of Gen1 or Gen2 do not occur properly.

- Switch ON the power supply switch, SW7.

4. Switch on the host PC and check the Device Manager for PCIe Device. The following figure shows the example Device Manager window. If the device is not detected, power cycle the SmartFusion2 Security Evaluation Kit and click option in the Device Manager.

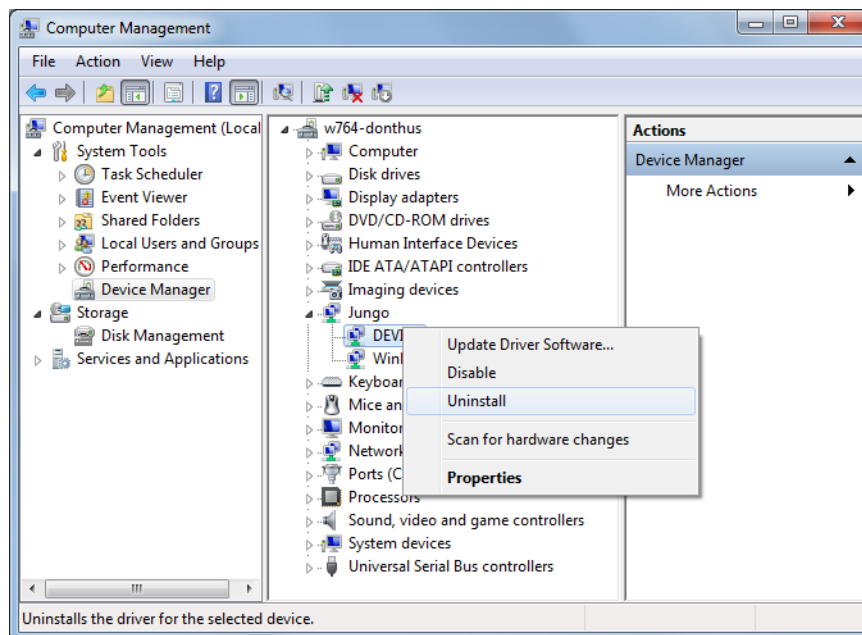
Figure 9 • Device Manager - PCIe Device Detection



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

5. If the host PC has any other installed drivers such as previous versions of Jungo drivers for the PCIe device, uninstall them. To uninstall previous versions of Jungo drivers, follow steps i and ii.
 - To uninstall previous Jungo drivers, go to **Device Manager** and right-click **DEVICE**, as shown in the following figure.

Figure 10 • Uninstall Jungo Driver



- In the **Confirm Device Uninstall** dialog box, select **Delete the driver software for this device** and click **OK**, as shown in the following figure. After uninstalling previous Jungo drivers, ensure that the PCI device is detected in the **Device Manager** window.

Figure 11 • Confirm Device Uninstall Dialog



2.6 Drivers Installation

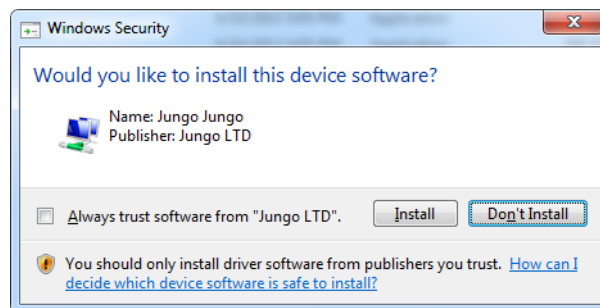
The PCIe demo uses a driver framework provided by Jungo WinDriverPro.

The following steps describe how to install the PCIe drivers on the host PC:

1. Extract the PCIe_Demo.rar to drive C. The PCIe_Demo.rar is at `<download_folder>\sf2_iap_using_interface_demo_df\pcie_driver\Drivers_64bitOS\PCIe_Demo.rar`
2. Run the batch file Jungo_KP_install.bat located at `C:\PCIe_Demo\DriverInstall\`
3. When the **Windows Security** dialog box appears asking to install or not, click **Install**.

Note: Installing these drivers require administration rights.

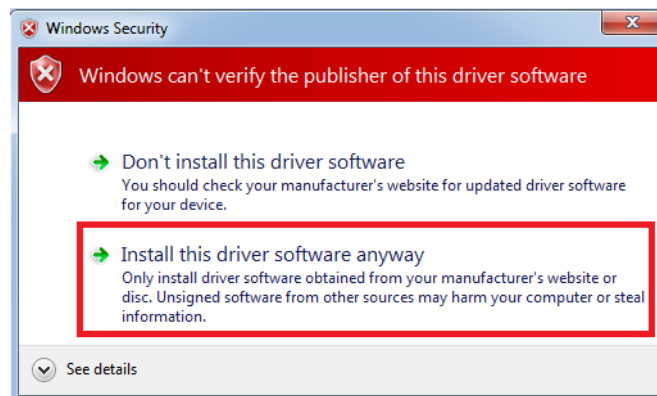
Figure 12 • Jungo Driver Installation



Note: If the installation fails, invoke the **Command Prompt** in administrator mode and run the batch file Jungo_KP_install.bat located at `C:\PCIe_Demo\DriverInstall\`

4. When the **Windows Security** dialog box appears, click **Install this driver software anyway**.

Figure 13 • Windows Security



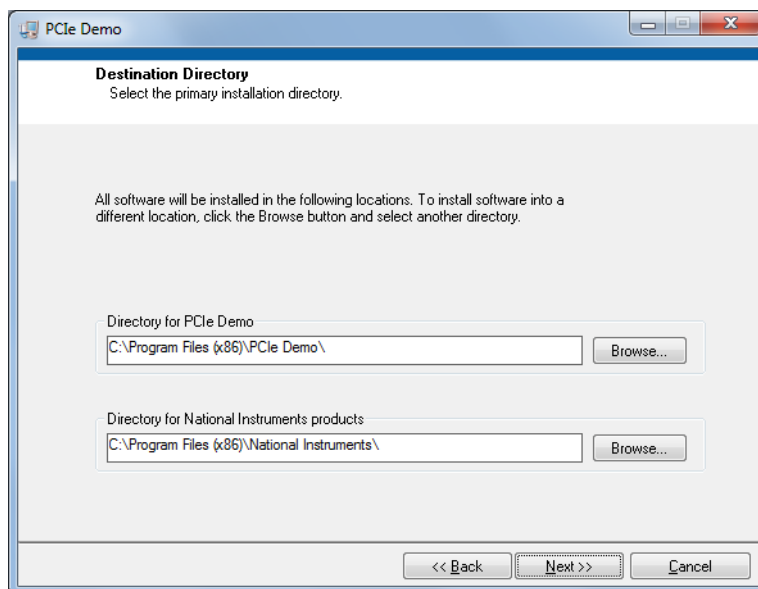
2.6.1 Installing PCIe_Demo Application GUI

The PCIe_Demo application is a GUI that runs on the host PC to communicate with the SmartFusion2 PCIe endpoint device. It provides PCIe link status, driver information, and demo controls. The PCIe_Demo application invokes the PCIe driver installed on the host PC and provides commands to the driver according to the selection made.

The following steps describe how to install the PCIe_Demo application:

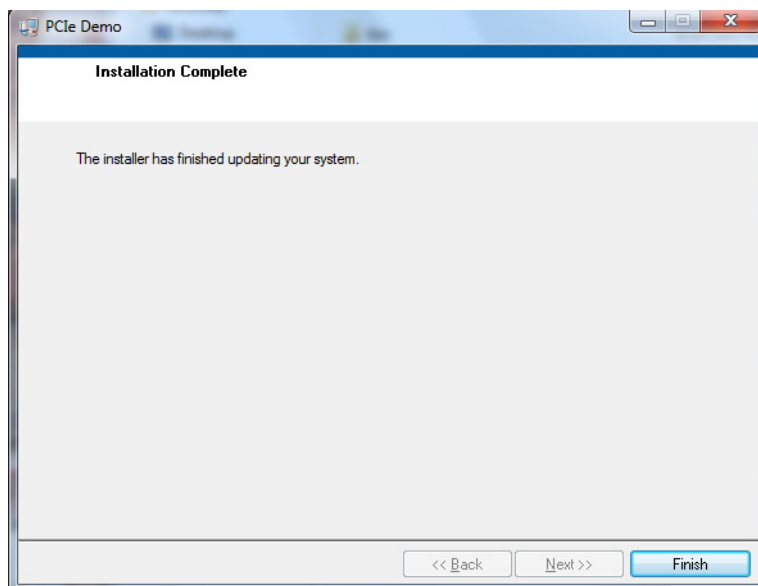
1. Download the PCIe Demo application from [PCIe_Demo_GUI_Installer](#). Go to <download_folder> *PCIe_Demo_GUI_Installer* and double-click **setup**. Do not change the default options.

Figure 14 • GUI Installation



2. Click **Next** to complete the installation. The following figure is displayed after successful installation.

Figure 15 • Successful Installation of GUI



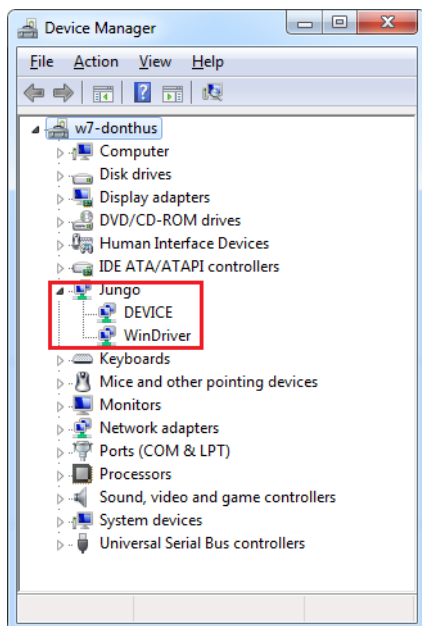
3. Shut down the host PC.
4. Power cycle the SmartFusion2 Security Evaluation Kit.
5. Restart the host PC.

2.7 Running the Demo Design

The following steps describe how to run the demo design:

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

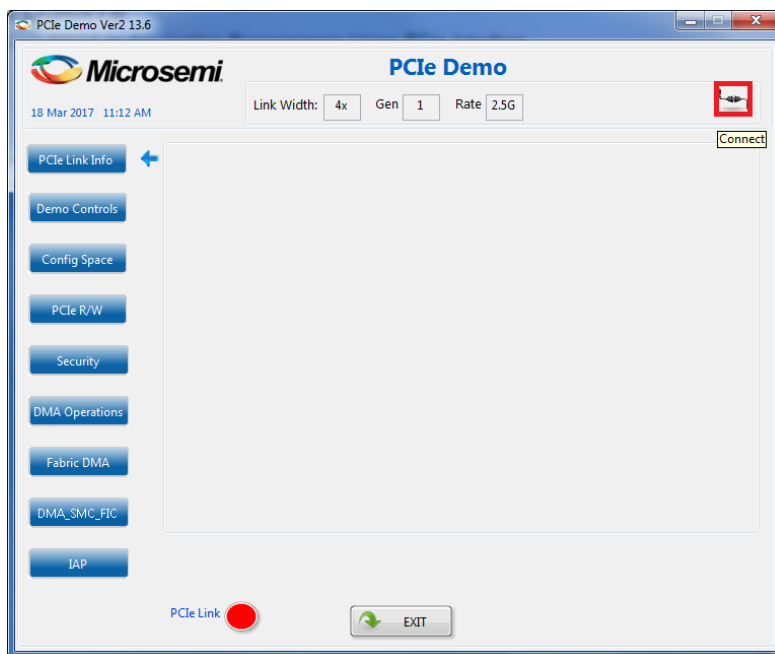
Figure 16 • Device Manager - PCIe Device Detection



Note: If a warning symbol appears on the **DEVICE** or **WinDriver** icons in **Device Manager**, uninstall them and start from step1 of [Drivers Installation](#), page 12.

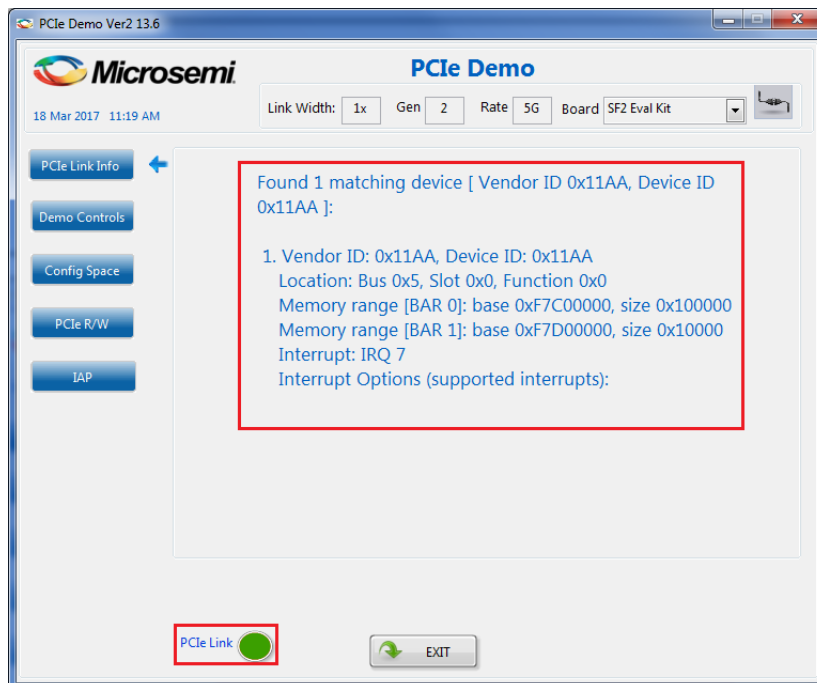
2. Invoke the GUI from **All Programs > PCIeDemo > PCIe Demo GUI**. The GUI is displayed as shown in the following figure.

Figure 17 • PCIe Demo GUI



- Click **Connect**. The application detects and displays the connected Kit, demo design, and PCIe link. The following figure shows the example messages after the connection is established.

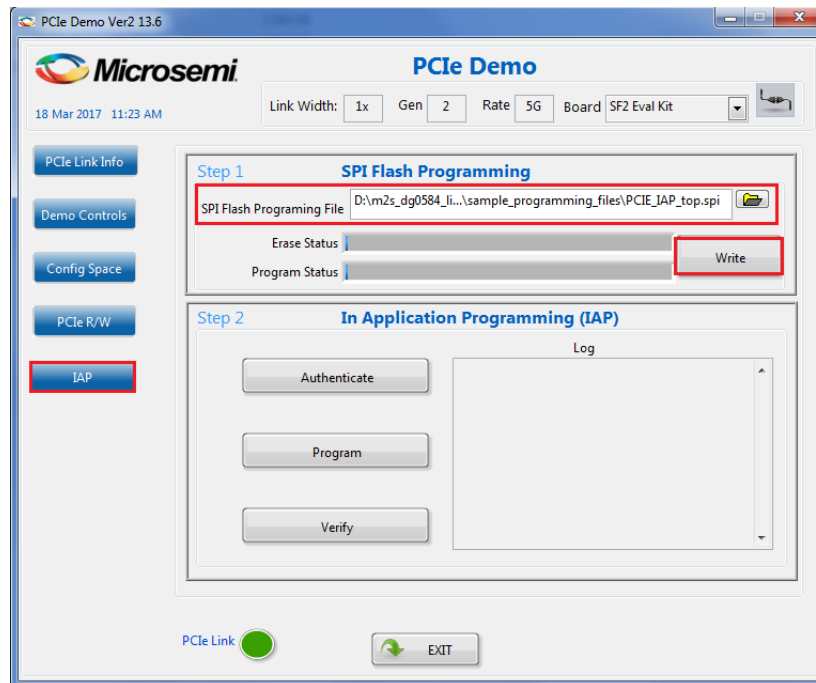
Figure 18 • PCIe Device Information



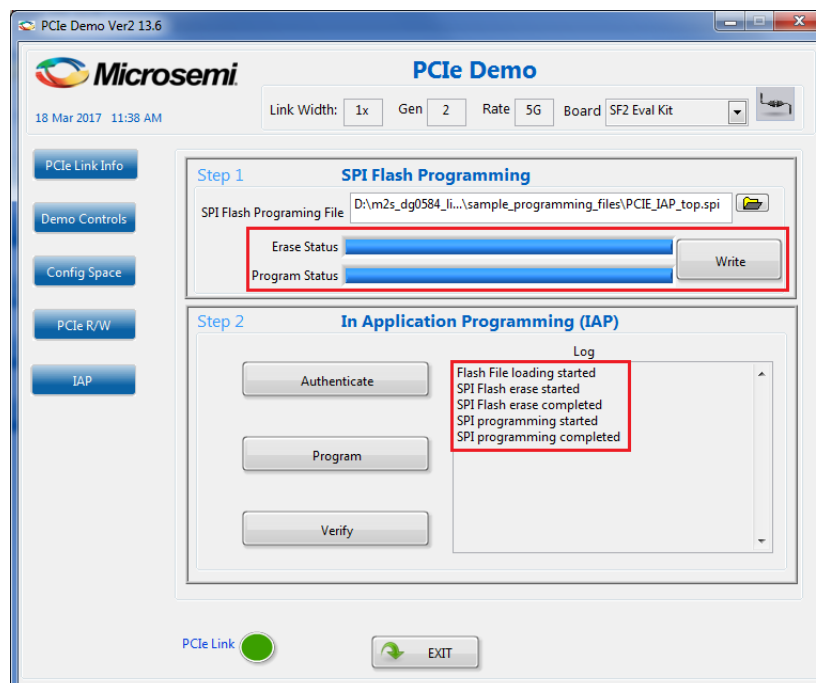
2.8 IAP Step1: Loading SPI Flash with Programming Bitstream

The following steps describe how to load the SPI Flash with programming bitstream:

- Click **IAP**, as shown in [Figure 19](#), page 16.
- Click **Browse** to select the *.spi programming file to write the bitstream data to the SPI Flash memory as shown in [Figure 19](#), page 16. The sample *.spi programming file can be selected from <download_folder>\sf2_iap_using_interface_demo_df\sample_programming_files.

Figure 19 • Selecting Programming File in PCIe_Demo Application


- Click **write** to move the programming file to the external SPI Flash memory. The SPI Flash memory is erased according to the programming file size and the programming file bitstream is written to the SPI Flash memory. The status of the SPI Flash erase and SPI Flash program operations can be observed on the status bars and also using the messages on the log window, as shown in the following figure.

Figure 20 • SPI Flash Programming


2.9 IAP Step2: Initiating the IAP Services

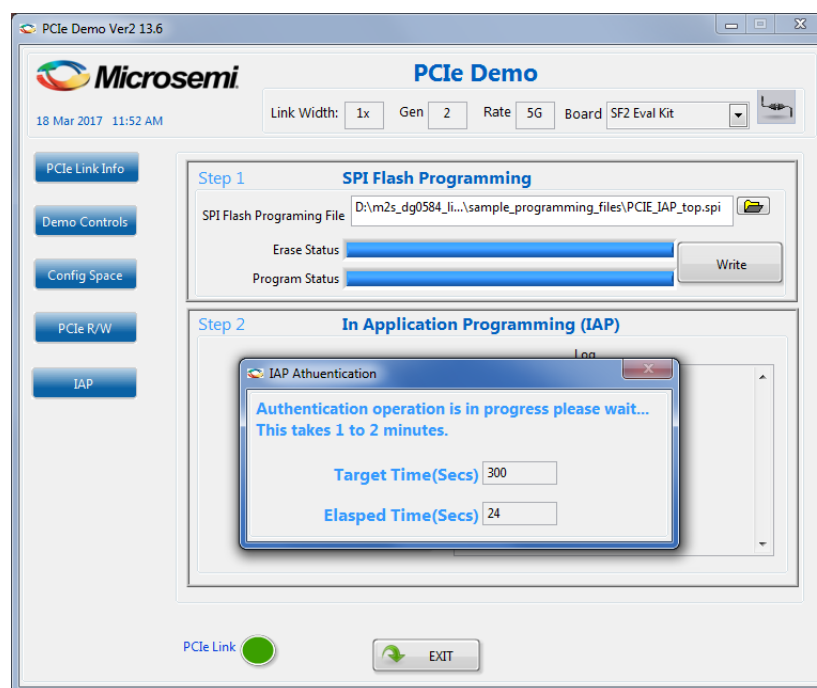
The IAP services can be executed by clicking Authenticate, Verify, or Program on the GUI.

2.9.1 Authenticate and Program Operation Mode

The following steps describe how to authenticate and program the operation mode:

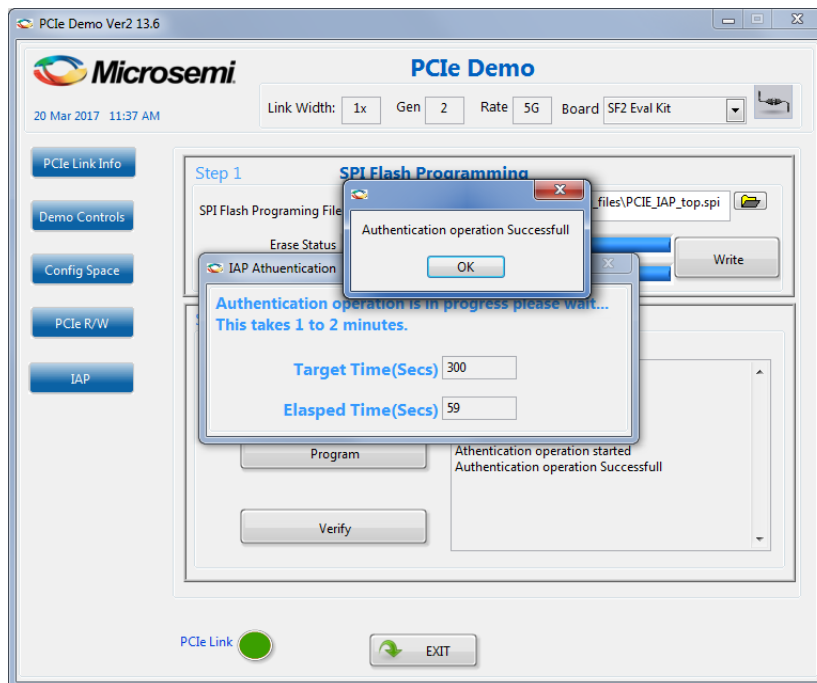
1. Click **Authenticate** in GUI to check the data integrity of the bitstream data stored in the SPI Flash.
The following figure shows the **IAP operation is in progress** message.

Figure 21 • IAP Authentication Status



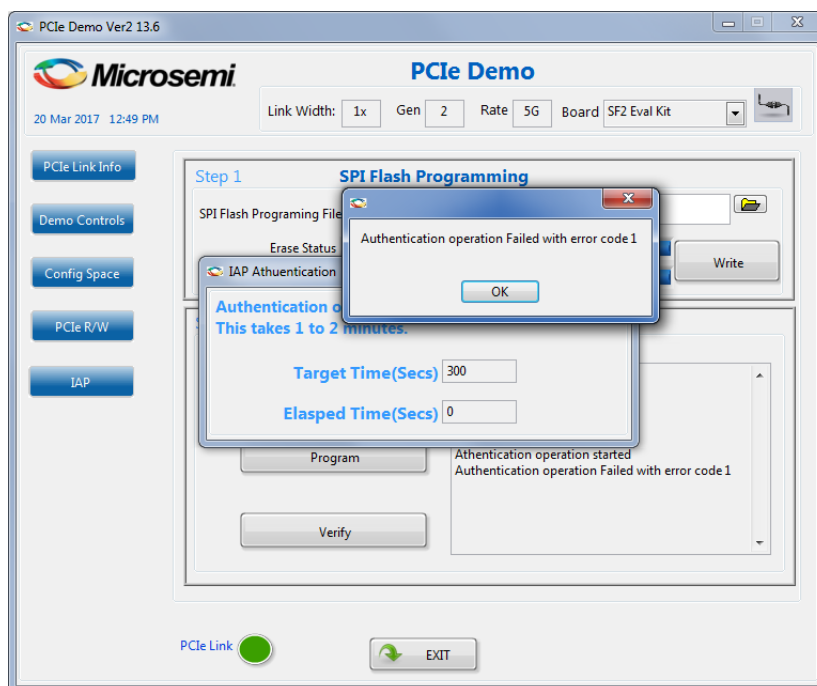
On completion of the IAP authentication, the GUI displays an **Authentication operation successful** message as shown in the following figure.

Figure 22 • Authentication Success Message in PCIe Demo GUI



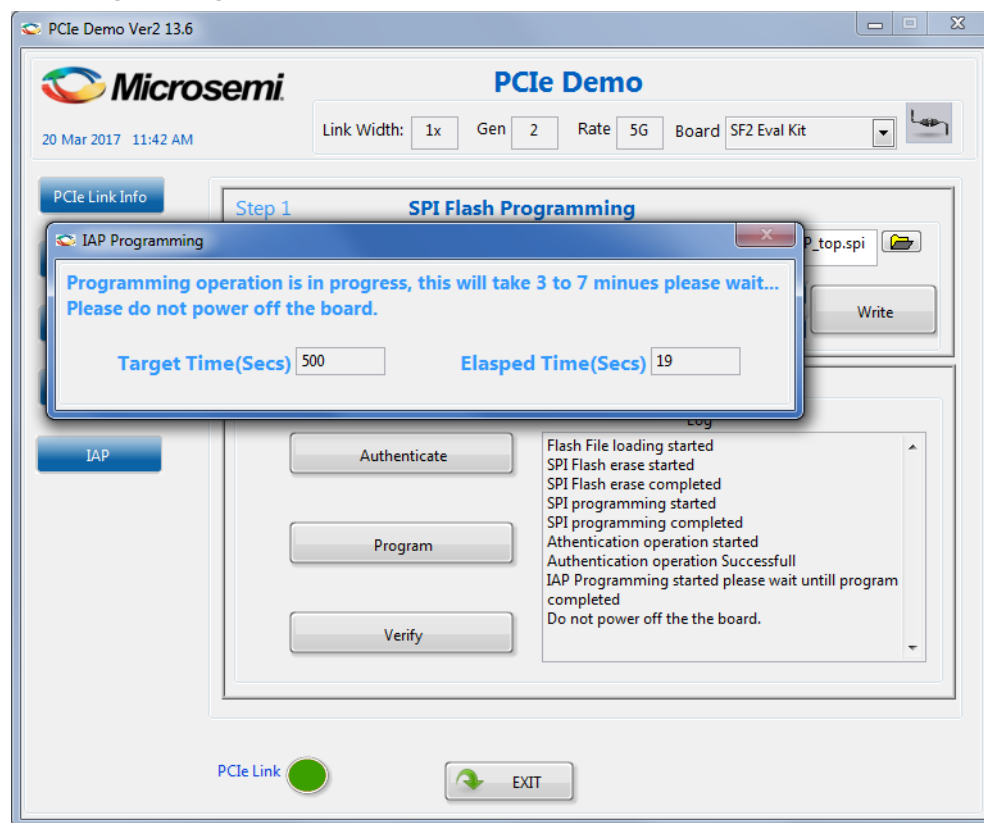
The demo GUI also provides an option to create an authentication failure scenario, which can be achieved by modifying the programming file (*.spi) contents and running the authenticate operation. The following figure shows a sample authentication failed error scenario message. For information about error codes, see [Appendix: Error Codes](#), page 24.

Figure 23 • Authentication Failed



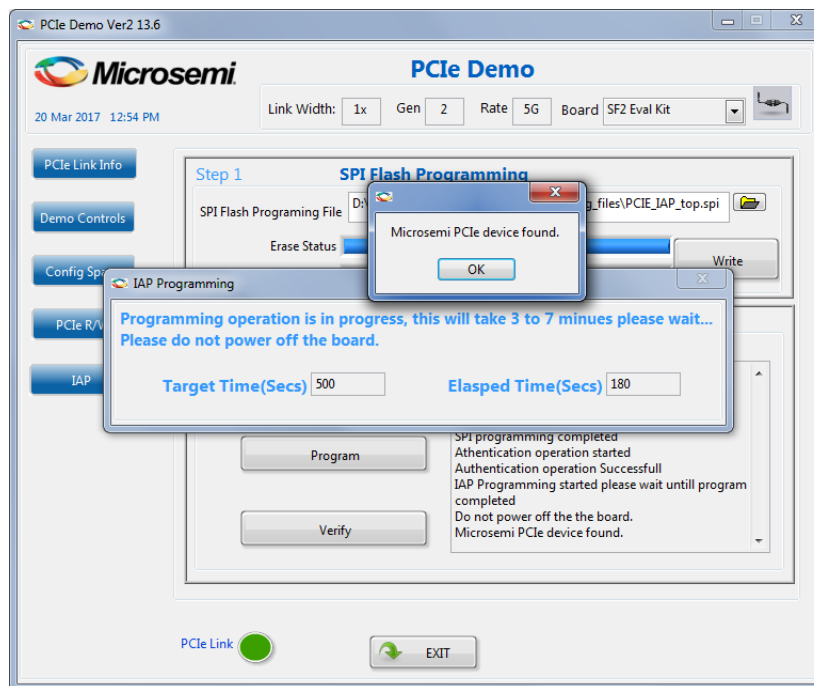
2. Click Program in the GUI to reprogram the FPGA fabric and the eNVM of the SmartFusion2 device. It takes about 3 to 5 minutes for the IAP service to complete and program the FPGA fabric and eNVM. The following figure shows the IAP programming status.

Figure 24 • IAP Programming Status



On completion of the IAP program, the GUI displays a Microsemi PCIe Device Found message as shown in the following figure. This indicates that the SmartFusion2 device is programmed with the PCIe enabled design and the PCIe communication link status is restored to the active state. If the SmartFusion2 device is programmed without the PCIe enabled design, the GUI displays a No Microsemi PCIe Device Found message. For more information on design type and programming results, see [Table 3](#), page 20.

Figure 25 • Microsemi PCIe Device Found Message in PCIe Demo GUI



2.9.1.1 Programming Results

The following table shows the possible results for the IAP program operation mode for sample programming files provided in the folder:

<download_folder>\sf2_iap_using_interface_demo_df\sample_programming_files.

All the .spi files listed in the following table are not demonstrated.

Table 3 • IAP Programming Results

*.spi Programming File Name	Design Type	eNVM Programming Result	FPGA Fabric Programming Result
iap_envm_only.spi	PCIe is not enabled	SmartFusion2 LED 4-bit counter logic (LEDs DS4 to DS7)	NA
iap_fabric_only.spi	PCIe is not enabled	NA	SmartFusion2 LEDs DS0 to DS3 blink
iap_fabric_and_envm.spi	PCIe is not enabled	SmartFusion2 LED 4-bit counter logic (LEDs DS4 to DS7)	SmartFusion2 LEDs DS0 to DS3 blink
pcie_iap_top.spi	PCIe is enabled	Retains original eNVM program.	Retains original FPGA fabric logic

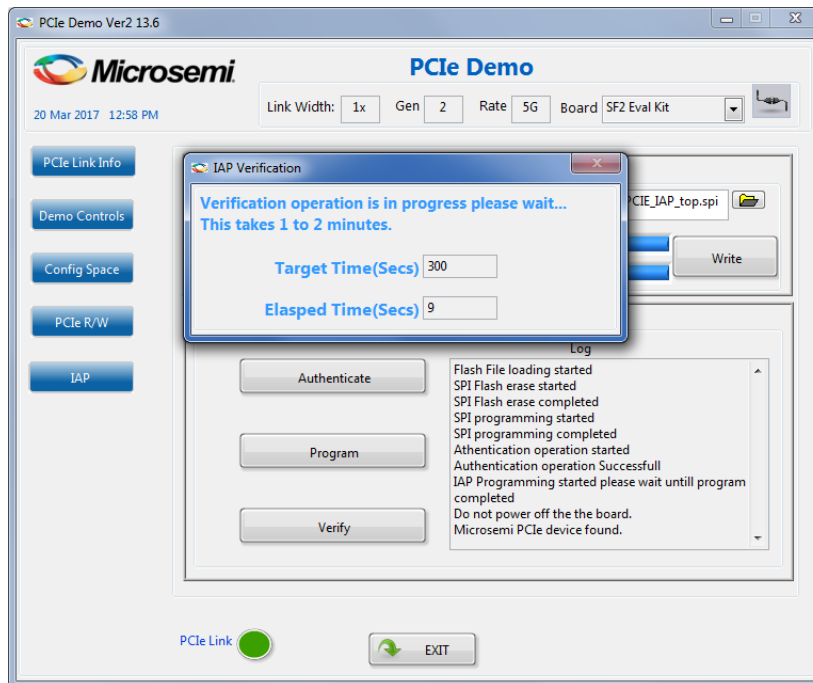
Note: After the successful IAP program operation, the SmartFusion2 Security Evaluation Kit must be reprogrammed with the original pcie_iap_top.stp file to try the IAP operation modes again, if the design type is not PCIe enabled.

2.9.2 Verify Operation Mode

The following steps describe how to verify operation mode:

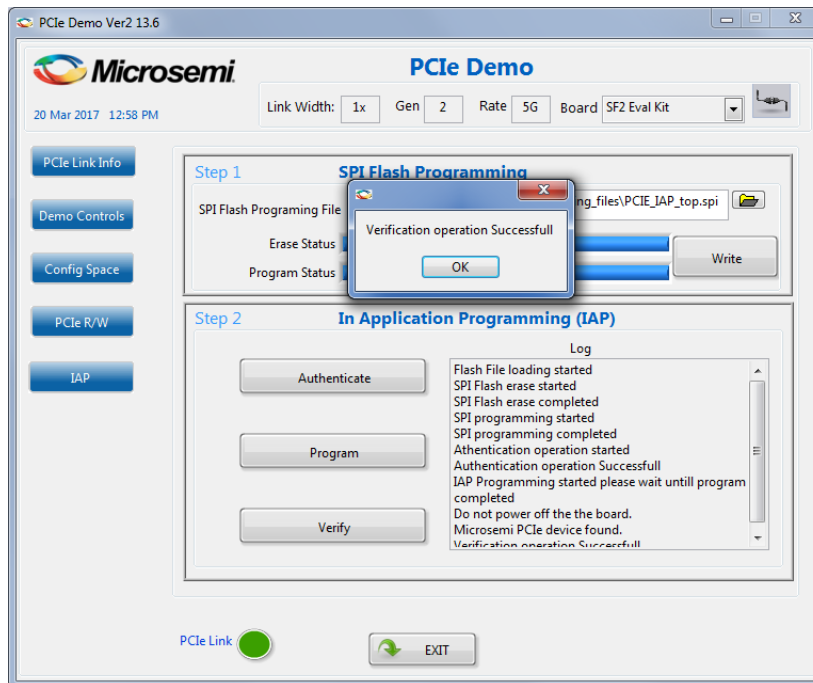
1. Click **Verify** in the GUI to verify the SmartFusion2 device FPGA fabric and eNVM contents. The following figure shows the **Verification operation is in progress** message.

Figure 26 • IAP Verification Status



The following figure shows the **Verification operation successful** message.

Figure 27 • IAP Verification Success



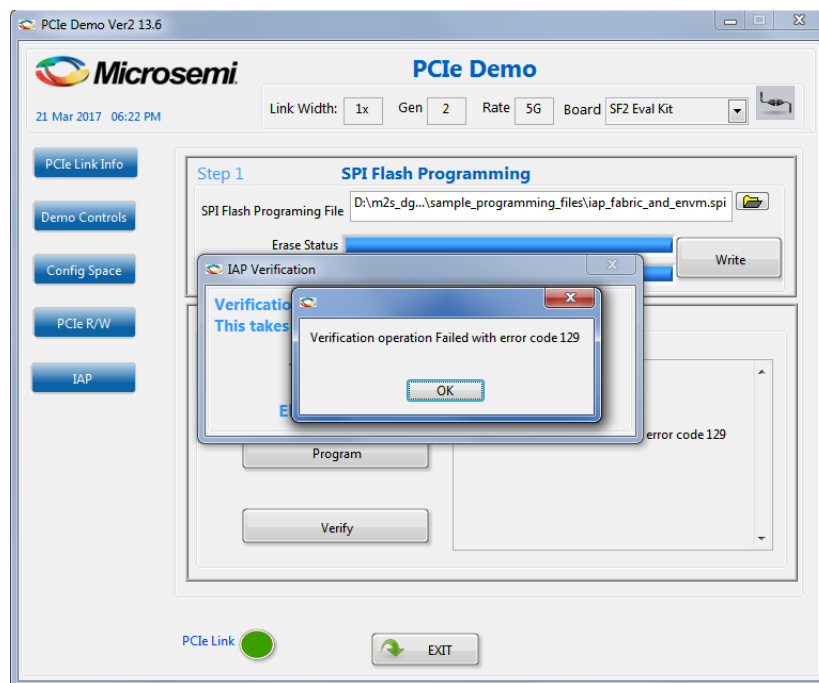
The verification operation is successful when the SmartFusion2 device contents match the programming bitstream data stored in the SPI Flash. If the verification fails, the GUI displays an error message with an error code. For information about error codes, refer to the [Appendix: Error Codes](#), page 24.

The sample programming files are at— `<download_folder>\sf2_iap_using_interface_demo_df\sample_programming_files`.

All the files do not pass the verification. Only the `pcie_iap_top.spi` file passes the **verification** operation as it matches the SmartFusion2 device contents (`pcie_iap_top.stp`). The other programming files fail verification.

The following figure shows the verification error message.

Figure 28 • IAP Verification Error Message



2.10 Known Issue

After successful completion of the two-step IAP or CM3 ISP, LSRAM read and write access fails from the fabric path. This is a known silicon issue, which is documented in the [ER0196: SmartFusion2 Device, Errata](#). The workaround for this problem is to reset the system after the IAP or ISP program operation. Microsemi recommends that this workaround is implemented for any design, which accesses LSRAM after IAP or ISP. For more information about how to implement this workaround, see [Appendix: Implementing Workaround to Access Fabric LSRAM after IAP/ISP Program Operation](#), page 31.

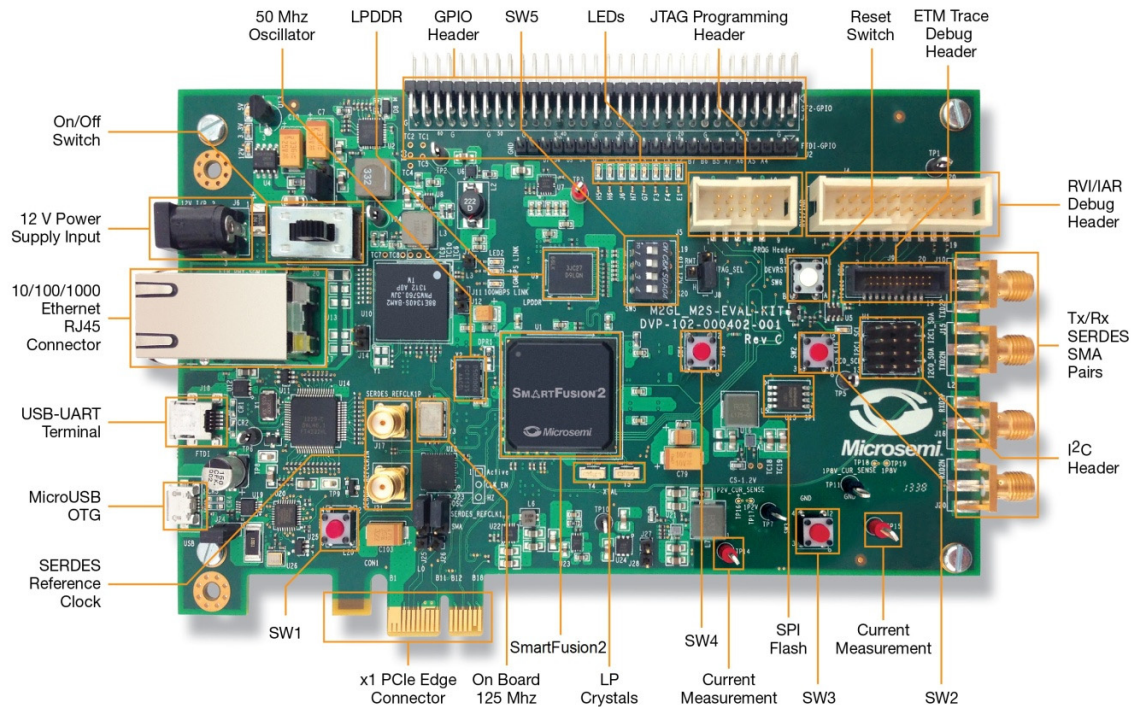
The design example provided in this demonstration implements the workaround for accessing LSRAM after implementing the IAP or ISP program operation in the Libero software. The design files are available in the following location:

`<download_folder>\sf2_iap_using_interface_demo_df\sample_programming_files\LSRAM_Workaround\PCIE_IAP_Tamper.rar`

3 Appendix: SmartFusion2 Security Evaluation Kit Board

The following figure shows the SmartFusion2 Security Evaluation Kit board.

Figure 29 • SmartFusion2 Security Evaluation Kit Board



4 Appendix: Error Codes

The following table lists the information about the error codes.

Table 4 • Error Codes

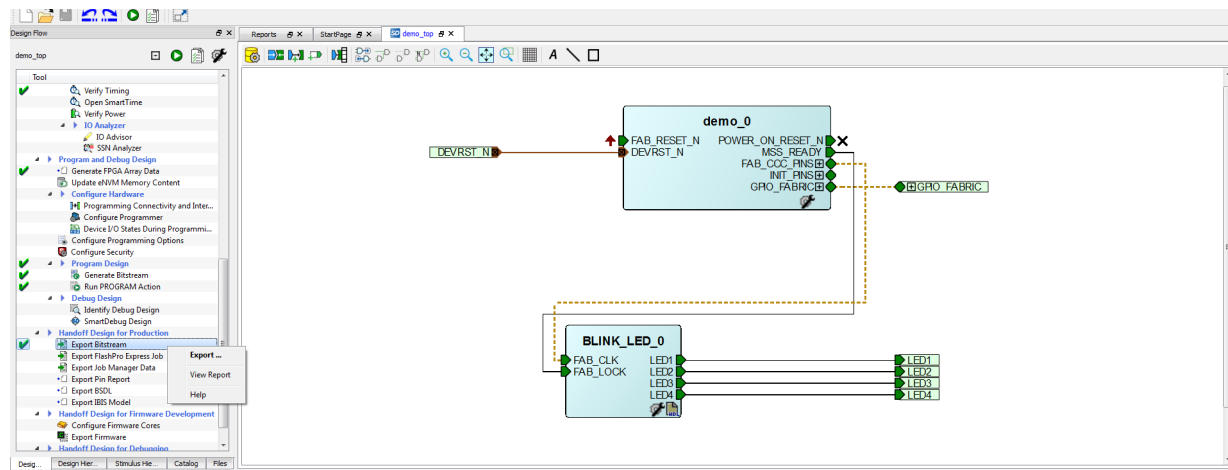
Define	Error Code	Description
#define MSS_SYS_CHAINING_MISMATCH	1	Device contents mismatch
#define MSS_SYS_UNEXPECTED_DATA_RECEIVED	2	Data is not supported
#define MSS_SYS_INVALID_ENCRYPTION_KEY	3	Invalid encryption key
#define MSS_SYS_INVALID_COMPONENT_HEADER	4	Invalid file
#define MSS_SYS_BACK_LEVEL_NOT_SATISFIED	5	Corrupted/ Invalid file
#define MSS_SYS_DSN_BINDING_MISMATCH	7	Corrupted/ Invalid file
#define MSS_SYS_ILLEGAL_COMPONENT_SEQUENCE	8	Corrupted/ Invalid file
#define MSS_SYS_INSUFFICIENT_DEV_CAPABILITIES	9	Invalid Device capabilities
#define MSS_SYS_INCORRECT_DEVICE_ID	10	Invalid Device ID
#define MSS_SYS_UNSUPPORTED_BITSTREAM_PROT_VER	11	Bitstream not supported
#define MSS_SYS_VERIFY_NOT_PERMITTED_ON_BITSTR	12	Verification not allowed for input bitstream
#define MSS_SYS_ABORT	127	Operation aborted
#define MSS_SYS_NVM_VERIFY_FAILED	129	eNVM verification failed
#define MSS_SYS_DEVICE_SECURITY_PROTECTED	130	Device secured
#define MSS_SYS_PROGRAMMING_MODE_NOT_ENABLED	131	Programming mode not enabled

5 Appendix: Generating .spi Programming File using Libero

The following steps describe how to generate .spi programming file using Libero:

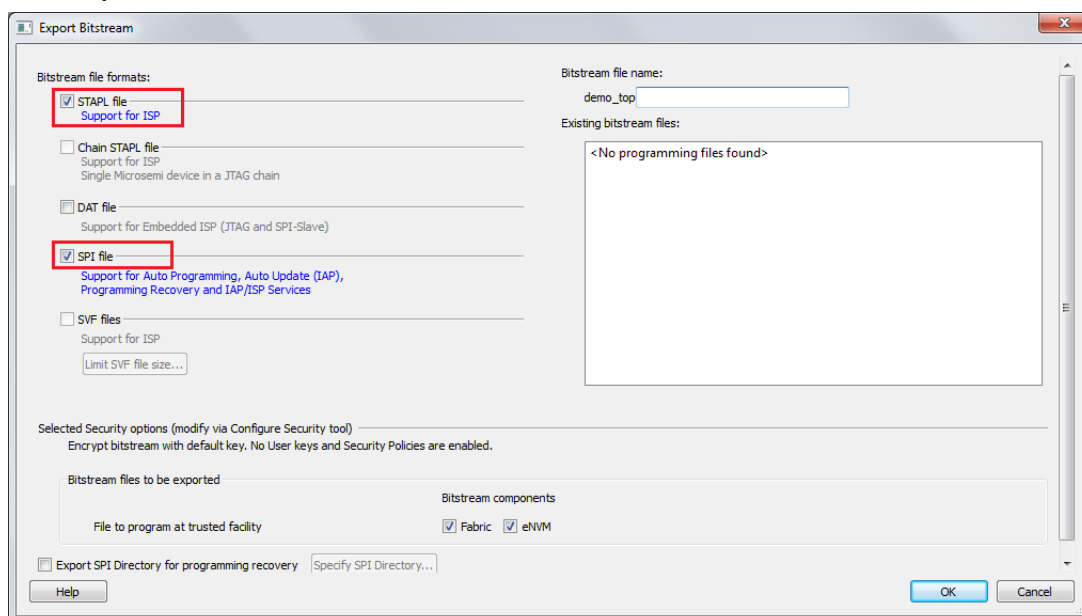
1. Launch the Libero SoC software to open a project for the iap_fabric_and_envm.spi programming file. The Libero design file is provided in
`<download_folder>\sf2_iap_using_interface_demo_df\smample_programming_files\fabric_and_envm`.
2. Right-click **Export Bitstream** under **Handoff Design for Production** in the **Design Flow** tab, and click **Export...** from the context menu.

Figure 30 • Configuring Export Bitstream



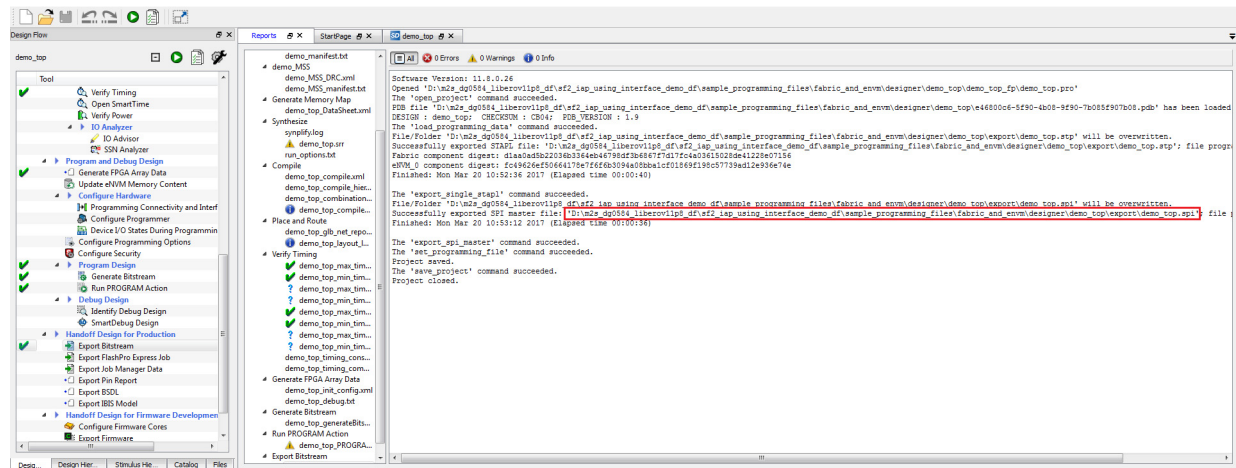
3. Select the **SPI file** check box in the **Export Bitstream** window.

Figure 31 • Export Bitstream Window



4. Click **OK**.

- Figure 32 • SPI File Location**

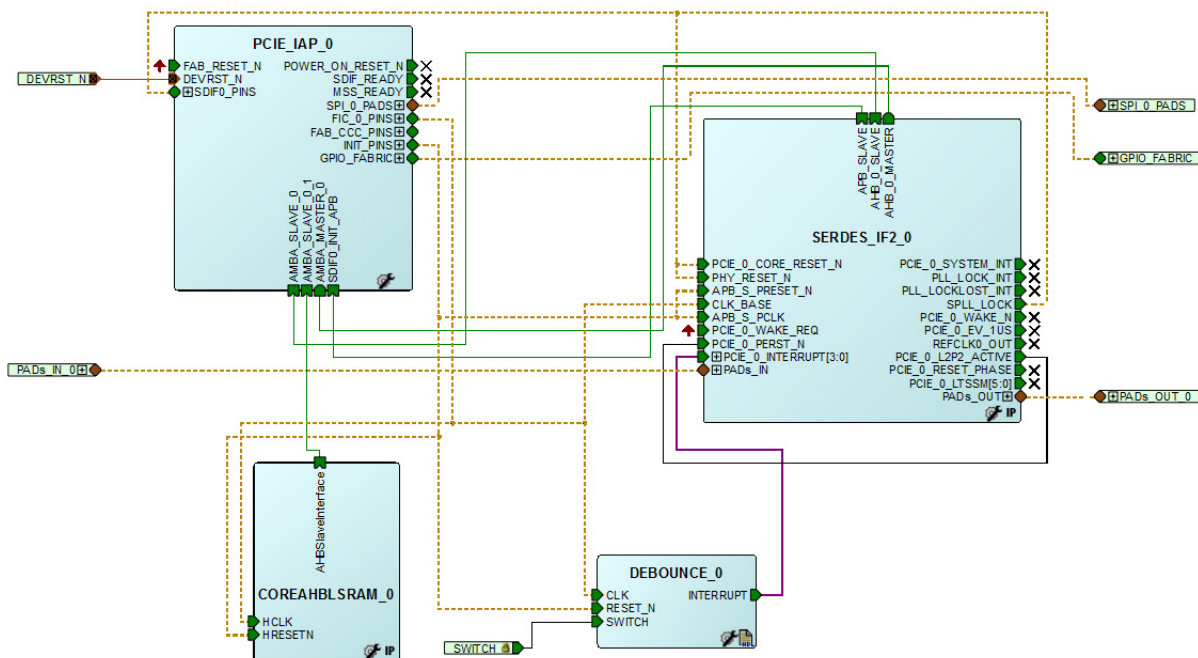


6 Appendix: Hardware Implementation

In this demo design, the following blocks are configured in Libero hardware project:

- The SERDES_IF_1 in the SmartFusion2 device is configured for PCIe 2.0, x4 lanes, and Gen2 rate.
- The CoreAHBLSRAM IP is configured to use the 4 KB of LSRAM.
- The advanced high-performance bus (AHB) master interface of SERDES_IF_1 is enabled and connected to the AHB slave interface of FIC_0 to access the MSS peripherals. The AHB slave interface of SERDES_IF_1 and CoreAHBLSRAM IP are connected to the AHB master interface of FIC_0 to access the PCIe interface and the fabric LSRAM from the MSS.
- BAR0 and BAR1 are configured in the 32-bit memory mapped memory mode. The advanced extensible interface (AXI) master window 0 is enabled and configured to map 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 map the BAR1 memory address space to the eSRAM address space to perform read and write operations from the PCIe interface. The AXI slave window 0 is enabled and configured to map the SmartFusion2 local address space to the host PC address space.
- The MSS GPIO block is enabled and configured: – GPIO_0 to GPIO_7 as outputs and connected to LEDs – GPIO_8 to GPIO_11 as inputs and connected to DIP switches.
- The M3_CLK clock and Serial Controller 1 clock base are configured to 50 MHz.
- The MSS SPI_0 controller is enabled to access the external SPI Flash memory.

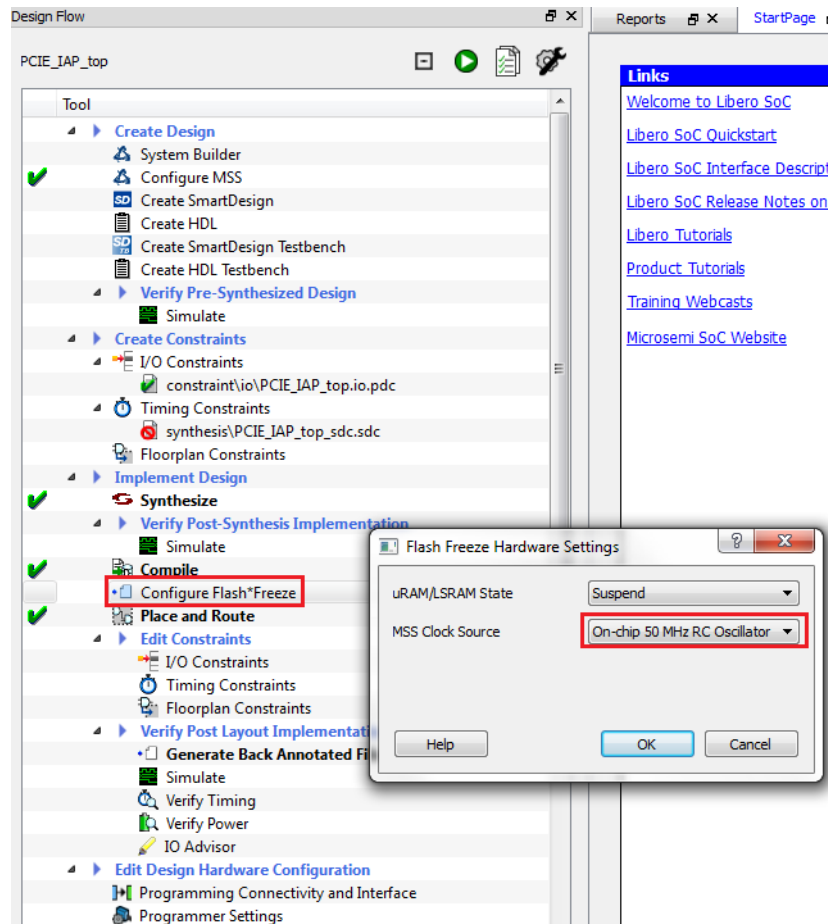
Figure 33 • Demo Design Libero Top-Level Diagram



6.1 Standby Clock Source Configuration

The standby clock source for the MSS in the F*F mode is configured to **On-chip 50 MHz RC Oscillator** using the **Flash*Freeze Hardware Settings** dialog box in the Libero SoC software, as shown in the following figure. A higher MSS clock frequency is required in the F*F mode to meet the SPI communication speed requirements.

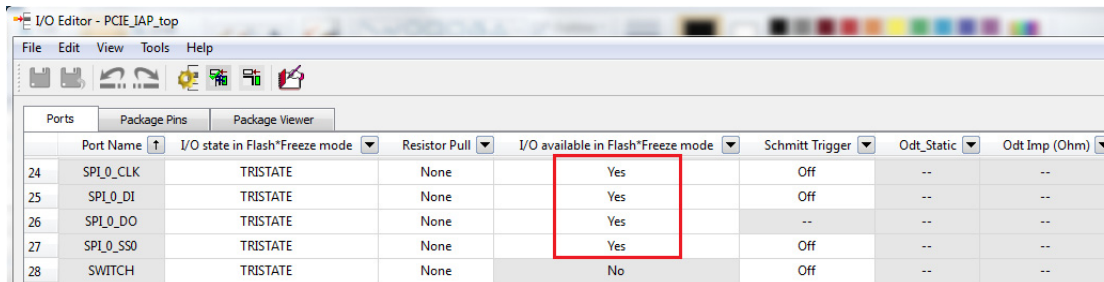
Figure 34 • Flash*Freeze Hardware Settings



6.2 Configuring I/Os for Flash*Freeze Mode

The FPGA fabric is not operational during the Program or Verify IAP operations as the device enters into the Flash*Freeze (F*F) mode. On the SmartFusion2 Security Evaluation Kit board, the SPI_0 is interfaced to the on-board SPI Flash memory for loading the programming bitstream data to the SPI Flash using the SPI interface. During the F*F mode, the fabric and I/Os are not available. Therefore, all the SPI_0 ports are configured using the I/O Editor to be available during the F*F mode, as shown in the following figure. **Commit and Check** the settings from the File menu after configuring the SPI_0 ports.

Figure 35 • Configuring SPI_0 Ports Available During F*F

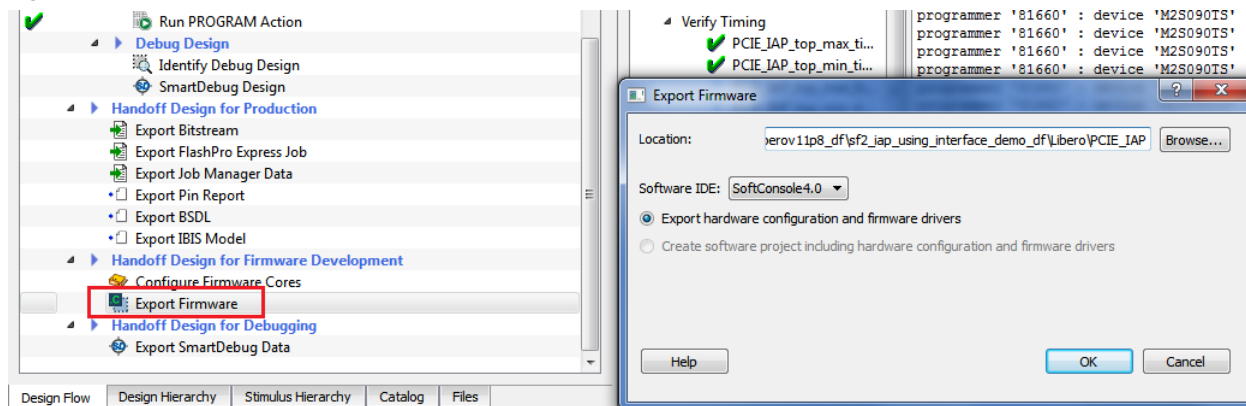


Port Name	I/O state in Flash*Freeze mode	Resistor Pull	I/O available in Flash*Freeze mode	Schmitt Trigger	Odt_Static	Odt Imp (Ohm)
24 SPI_0_CLK	TRISTATE	None	Yes	Off	--	--
25 SPI_0_DI	TRISTATE	None	Yes	Off	--	--
26 SPI_0_DO	TRISTATE	None	Yes	--	--	--
27 SPI_0_SS0	TRISTATE	None	Yes	Off	--	--
28 SWITCH	TRISTATE	None	No	Off	--	--

6.3 SoftConsole Project Generation

The firmware and SoftConsole project workspace can be generated by selecting the **Create project for selected Software Tool Chain** check box and selecting a **Software ToolChain** option from the drop-down list, as shown in the following figure.

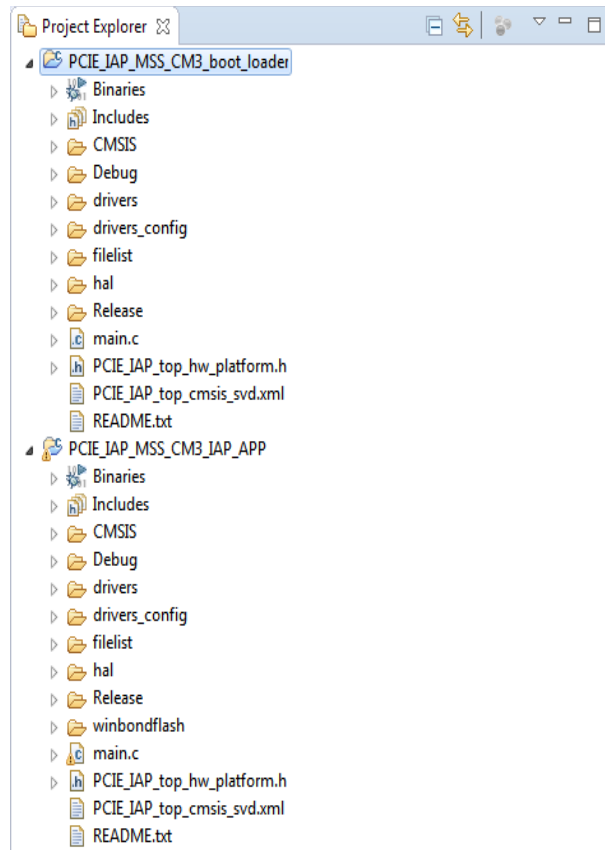
Figure 36 • Export Firmware Options



After successful firmware generation, the firmware and SoftConsole folders are generated at `<download_folder>\sf2_iap_using_interface_demo_df\libero` as specified in the **Location** field of **Export Firmware** dialog box shown in the above figure.

For software modifications, open the SoftConsole project workspace (located at `<download_folder>\sf2_iap_using_interface_demo_df\libero\PCIE_IAP\SoftConsole4.0`) using SoftConsole IDE v4.0. The following figure shows the SoftConsole project workspace.

Figure 37 • SoftConsole Project Workspace



The SoftConsole workspace consists the following projects:

- **PCIE_IAP_MSS_CM3_IAP_APP:**
Receives the bitstream from the host PC through the PCIe interface and invokes the system controller programming services.
- **PCIE_IAP_MSS_CM3_boot_loader:**
Implements remapping of eSRAM to the Cortex-M3 processor code space after copying the IAP code from eNVM to eSARM.
- Both the projects contain all the firmware and hardware abstraction layers of the hardware design.

7 Appendix: Implementing Workaround to Access Fabric LSRAM after IAP/ISP Program Operation

The LSRAM write and read accesses are denied after implementing the IAP or ISP program operation. The workaround for this problem is to apply System Reset after the IAP or ISP program operation.

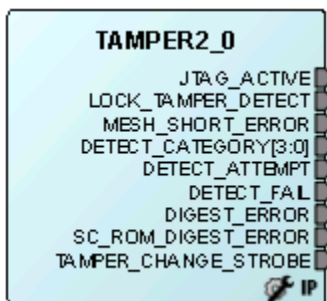
7.1 Changes Required in Libero Design

7.1.1 Option 1: Creating SmartDesign

The following steps describe how to apply System Reset:

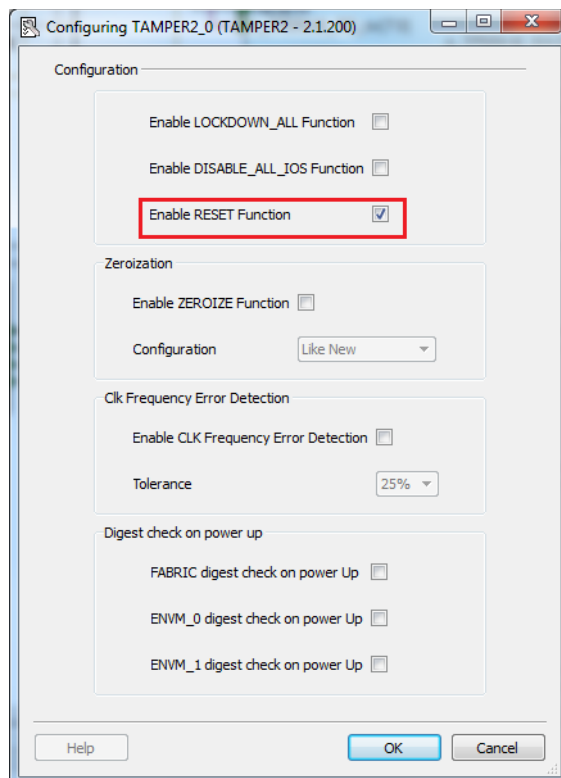
1. Choose **File > New > SmartDesign**.
2. Enter the **Name** as **Dev_Restart_after_IAP_blk** in the **Create New SmartDesign** window.
3. Navigate to **Libero Catalog** to open **Tamper Macro**.
 - a. Drag the **Tamper Macro** (shown in the following figure) available in **Libero Catalog** to the **Dev_Restart_after_IAP_blk** SmartDesign canvas.

Figure 38 • Tamper Macro - Before Configuration



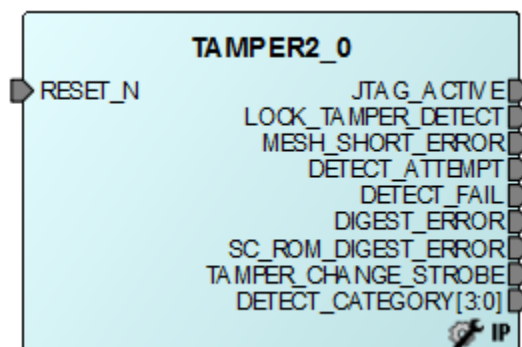
- b. Select the **Enable RESET Function** check box in the **Configuring Tamper 2_0** window, as shown in [Figure 39](#), page 32.
- c. Click **OK**. The **System Reset** is enabled.

Figure 39 • Tamper Macro Configuration Window



The following figure shows the TAMPER2_0 macro after configuration.

Figure 40 • Tamper Macro - After Configuration



4. Instantiate the **FSM Module** provided in the design files. This FSM logic performs three consecutive address writes to the two-port large SRAM with the known data pattern and then reads back data from those three consecutive address locations to compare. If the read back data pattern does NOT match the written data pattern, the FSM asserts the RESET_N input to Tamper Macro, which in turn causes a System Reset. If the read back data pattern matches the written data pattern, the FSM does not do anything.

Follow these steps to add the FSM logic to the PCIe IAP design:

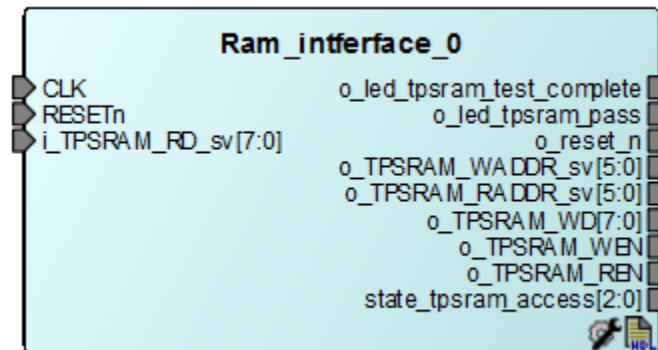
a. Choose **File > Import > HDL Source Files**.

b. Browse to the Ram_interface.v file location in the design files folder.

<download_folder>\sf2_iap_using_interface_demo_dfsample_programming_files\LSRAM_Work-around\Sourcefiles

- c. Click the **Dev_Restart_after_IAP_blk** tab and drag the **Ram_interface** component from the **Design Hierarchy** to the **Dev_Restart_after_IAP_blk SmartDesign** canvas. Figure 41 shows the **Ram_interface** component.

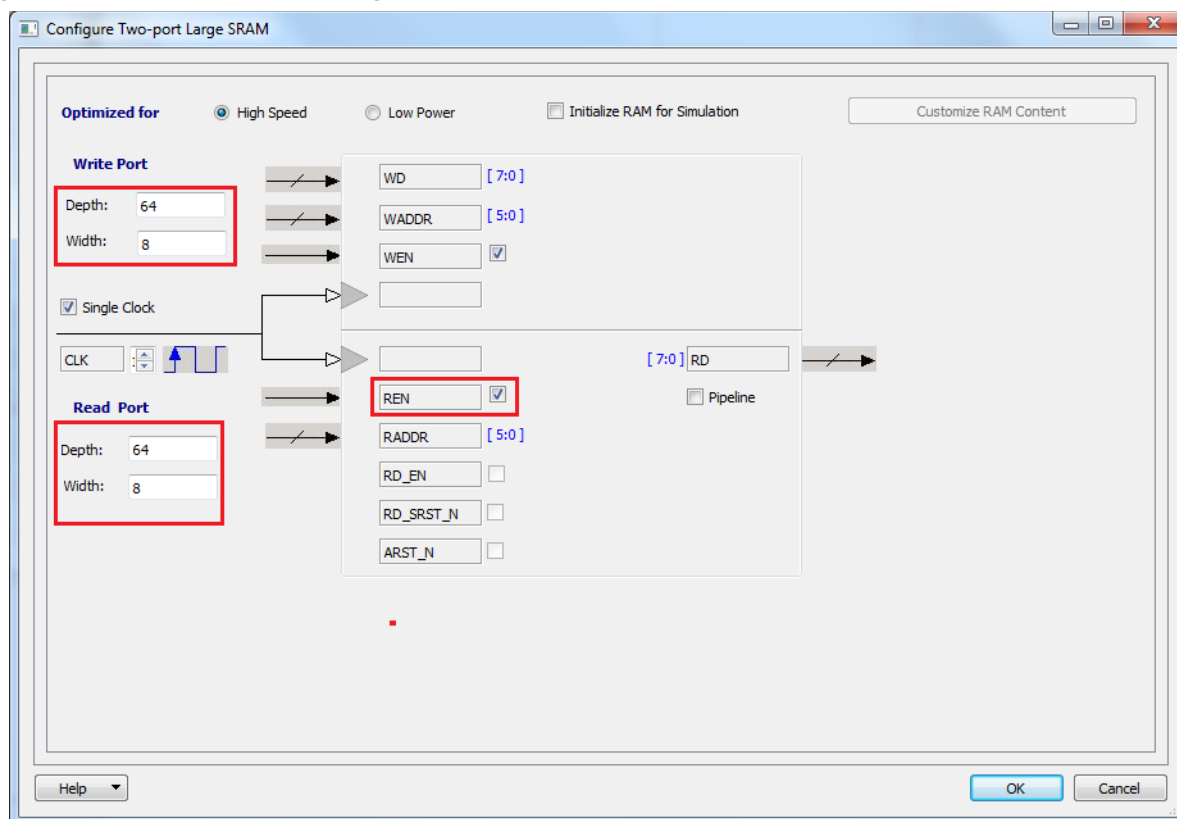
Figure 41 • Ram_interafce FSM Component



After the completion of the IAP programming, the System Controller asserts POWER_ON_RESET_n to the FPGA fabric. This triggers the RESETn signal and initiates the state machine in the FSM module.

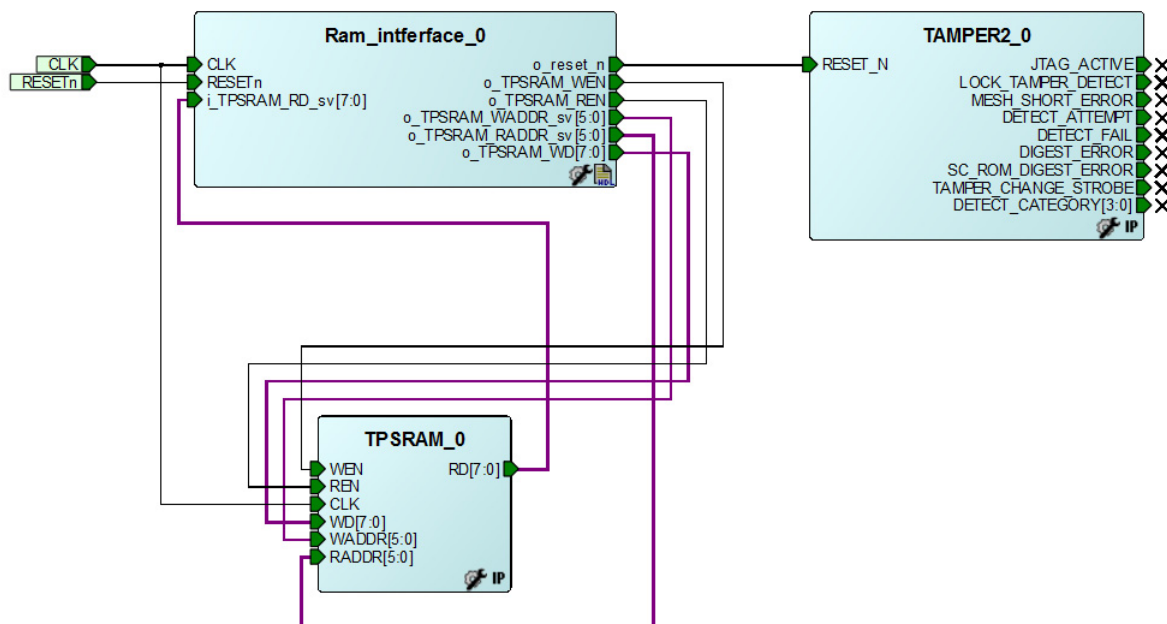
5. Drag the **Two-Port Large SRAM (TPSRAM)** available in the **Libero Catalog** to the **Dev_Restart_after_IAP_blk SmartDesign** canvas.
6. Configure the **TPSRAM** with the following settings:
 - Write Port
 - Depth: 64
 - Width: 8
 - Read Port
 - Depth: 64
 - Width: 8
 - Select **Check REN** check box

Figure 42 • Two-Port SRAM Configurator Window



7. Connect **Tamper Macro, FSM, and TPSRAM**, as shown in the following figure.

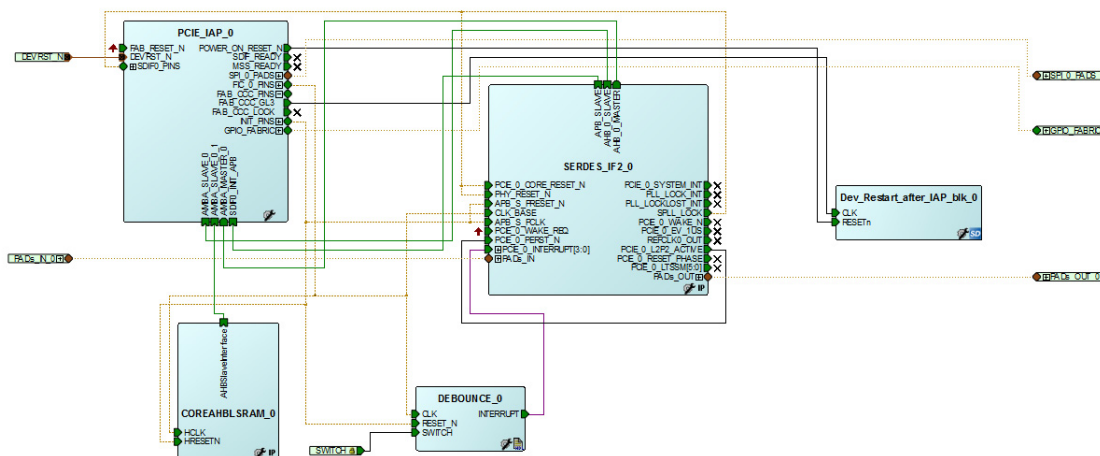
Figure 43 • Dev_Restart_after_IAP_blk SmartDesign



8. Click the **PCIE_IAP_top** tab and drag the **Dev_Restart_after_IAP_blk** component from the **Design Hierarchy** to the **PCIE_IAP_top** SmartDesign canvas.

- Connect the blocks as shown in the following figure and generate **PCIE_IAP_top** SmartDesign. This completes the implementation of the workaround.

Figure 44 • PCIE_IAP_top SmartDesign



Note: This workaround is applicable for v11.6 software release or later, and must be implemented in the Libero design, which is used to generate the .spi programming file. Older versions of Libero may prune Tamper Macro during synthesis. To avoid pruning, one of the recommended options is to promote the DETECT_ATTEMPT signal of Tamper Macro to the top-level.

7.1.2 Option 2: Importing the .cxf file in Libero Design

Another option to implement this workaround is to import the .cxf file for SmartDesign **Dev_Restart_after_IAP_blk**. The .cxf file is provided with the design files and it has all the component instantiations and connections mentioned in [Option 1: Creating SmartDesign](#), page 31 from Step 1 to Step 6.

The following steps describe how to import .cxf file.

- Extract the files.
<download_folder>\sf2_iap_using_interface_demo_df\sample_programming_files\LSRAM_Workaround\PCIE_IAP_Tamper.rar
- Choose **File > Import > Others**.
- Browse to the following **Dev_Restart_after_IAP_blk.cxf** file location in the design files folder.
<download_folder>\sf2_iap_using_interface_demo_df\sample_programming_files\LSRAM_Workaround\PCIE_IAP_Tamper\PCIE_IAP\component\work\Dev_Restart_after_IAP_blk
- Browse to the **Ram_interface.v** file location in the design files folder.
<download_folder>\sf2_iap_using_interface_demo_df\sample_programming_files\LSRAM_Workaround\Sourcefiles.
- Repeat Step 7 and Step 8 of Option 1 to instantiate **Dev_Restart_after_IAP_blk** in **PCIE_IAP_top SmartDesign**.