DG0454
Demo Guide
SmartFusion2 SoC FPGA In-System Programming Using UART Interface - Libero SoC v11.8

Power Matters.™

Power Matters.™
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

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.

50200454. 8.0 4/17

Contents

Figures

Tables

1 Revision History

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

1.1 Revision 8.0

Updated the document for Libero v11.8 software release.

1.2 Revision 7.0

Updated the design files for Libero v11.6 software release (SAR 72612).

1.3 Revision 6.0

Updated the design files for Libero v11.5 software release (SAR 68427) and (SAR 68139).

1.4 Revision 5.0

Updated the document for Libero v11.5 software release (SAR 65132).

1.5 Revision 4.0

Updated the document for Libero v11.4 software release (SAR 59742).

1.6 Revision 3.0

Updated the document for Libero v11.3 software release (SAR 56619).

1.7 Revision 2.0

Updated Description section (SAR 53451).

1.8 Revision 1.0

In revision 1.0, updated the document for Libero v11.2 software release (SAR 52962).

1.9 Revision 0

Revision 0 was the first publication of this document.

2 In-System Programming Using UART Interface

In-system programming (ISP) allows to reprogram the design iterations and field upgrades. SmartFusion®2 devices support ISP through the universal asynchronous receiver/transmitter (UART) interface. This document describes how to program the following using ISP through the UART interface:

- Embedded nonvolatile memory (eNVM)
- FPGA fabric
- Both the eNVM and the FPGA fabric

For information on different programming modes supported by SmartFusion2 SoC FPGAs, see the *UG0451: IGLOO2 and SmartFusion2 Programming User Guide*. For information on 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 requirements.

Table 1 • **Design Requirements**

Design Requirements	Description
Hardware	
SmartFusion2 Security Evaluation Kit: - 12 V adapter - FlashPro4 programmer - USB A to Mini-B cable	Rev D or later
Host PC or Laptop	Windows 64-bit Operating System
Software	
Libero® System-on-Chip (SoC)	v11.8
FlashPro Programming Software	v11.8
Host PC Drivers	*USB to UART*

2.2 Demo Design

The demo design files are available for download at:
http://soc.microsemi.com/download/rsc/?f=m2s_dg0454_liberov11p8_df

The demo design files include:

- Libero SoC software project
- STAPL programming files
- UART Host PC Loader application (`M2S_UARTHost_Loader.exe`)
- Sample programming files

The following figure shows the top-level structure of the design files. For further details, see the `readme.txt` file.

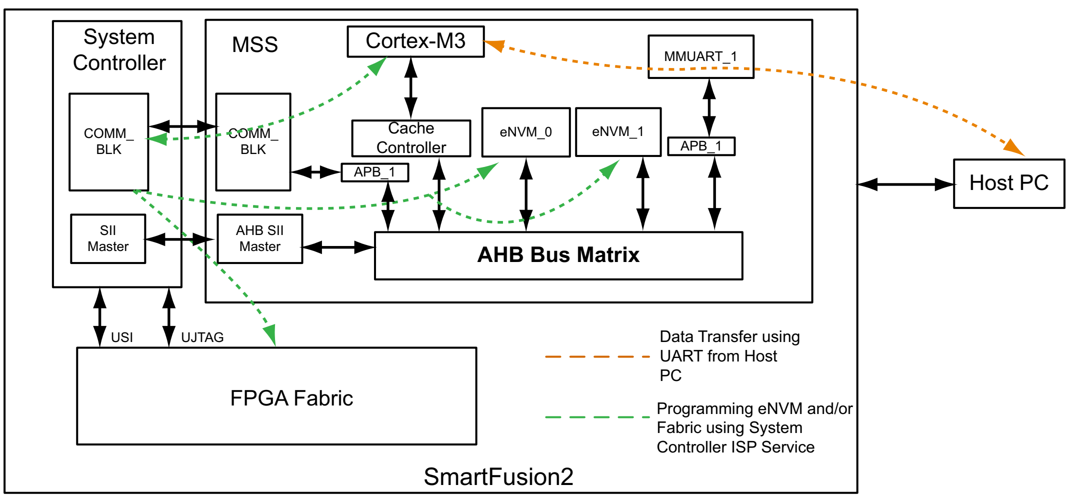
Figure 1 • **Demo Design Top-Level Structure**

```
<download_folder>
 └── sf2_isp_using_uart_interface_demo_df
      ├── host_tool_and_samples
      ├── libero
      ├── stapl_programming_file
      └── readme.txt
```

The following figure describes the top-level demo. The SmartFusion2 device application configures the MMUART_1 peripheral for serial communication and initializes the system controller to run the ISP service. The UART Host PC Loader initiates the communication with the SmartFusion2 device through the UART interface and sends the data bitstream to the ARM® Cortex®-M3 processor. See the Appendix: Hardware Project Implementation Settings, page 20.

The Cortex-M3 processor sends the received blocks of data to the system controller ISP service. The system controller ISP service executes the ISP operation in the requested mode and reports the status to the Cortex-M3 processor. See Description, page 4 for information on modes of operation.

Figure 2 • **Top-Level Demo Diagram**



2.2.1 Features

The demo design performs three types of programming based on the input provided by the programming file.

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

The ISP in SmartFusion2 devices is performed by the Cortex-M3 processor and the system controller. The system controller manages the SmartFusion2 device programming and handles the system service requests. The SmartFusion2 device allows the Cortex-M3 processor to directly provide a bitstream to the system controller for programming. The Cortex-M3 processor initializes the system controller and receives the programming bitstream from the Host PC through the UART interface. The received bitstream is sent to the system controller to execute the ISP service in one of the following modes of operation:

- **Authenticate**: System controller ISP service validates the integrity of the input data bitstream and reports the status information to the Cortex-M3 processor.
  - For security and reliability reasons, Microsemi recommends that the bitstream is authenticated before the program is executed, using the Authenticate operation mode. The SmartFusion2 device application must commit only the bitstream for programming, after successful authentication and the integrity of the bitstream is validated.
- **Program**: System controller ISP service programs the following depending on the input data bitstream:
  - eNVM
  - FPGA fabric
  - Both the eNVM and the FPGA fabric
- **Verify**: System controller ISP service verifies the contents of the SmartFusion2 device against the input data bitstream and reports the status information to the Cortex-M3 processor.

The system controller ISP service utilizes the COMM_BLK interface to receive the entire programming data bitstream as a continuous stream of bytes. See the *UG0331: SmartFusion2 Microcontroller Subsystem User Guide* for more information on communication block (COMM_BLK).

The Cortex-M3 processor in the SmartFusion2 device can execute an application image from embedded SRAM (eSRAM), eNVM or DDR/SDR memories. See the *AC390: SmartFusion2 SoC FPGA Remapping eNVM, eSRAM, and DDR/SDR SDRAM Memories Application Notes* for more information on remapping techniques. In this demo design, the Cortex-M3 processor executes the ISP application image from eSRAM while the eNVM programming taking place, that is during Program operation mode. In order to execute the application image from eSRAM, the Cortex-M3 processor copies the ISP application image (resides in eNVM data client) to the eSRAM and remaps the eSRAM to the Cortex-M3 processor code region. For Verify and Authenticate operation modes, the application image can be executed from either eNVM or eSRAM as the eNVM programming is not initiated. See the Appendix: Hardware Project Implementation Settings, page 20.

2.2.2.1 UART Host PC Loader

UART Host PC Loader (M2S_UARTHost_Loader.exe) is an executable program that transfers the programming files (`*.spi`) from the Host PC to the SmartFusion2 Security Evaluation Kit board. The `M2S_UARTHost_Loader.exe` file is executed from the command prompt. It is located at: *<download_folder>\sf2_isp_using_uart_interface_demo_df\host_tool_and_samples.*

The syntax is:

```
M2S_UARTHost_Loader.exe <*.spi> <COM Port number> <Operation Mode>
```

Arguments:

- `*.spi` programming file.
- COM Port number.
- Operation Mode. See Table 2, page 5.

For more information, see Running the Demo Design, page 8.

The following table shows the ISP operation modes and the values that are supplied in the command for the modes.

Table 2 • **ISP Operation Modes**

Mode	**Value**
Authenticate	0
Program	1
Verify	2

2.2.2.2 Programming Files

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

- eNVM
- FPGA fabric
- Both the eNVM and the FPGA fabric

The folder *<download_folder>\sf2_isp_using_uart_interface_demo_df\host_tool_and_samples* contains the following sample programming files.

- `isp_envm_only.spi`: Programs only eNVM. The eNVM client has a simple message display program.
- `isp_fabric_only.spi`: Programs only the FPGA fabric. The FPGA fabric has a light-emitting diode (LED) blinking logic.
- `isp_fabric_and_envm.spi`: Programs both the FPGA fabric and eNVM. The eNVM client has a message display program and the FPGA fabric has an LED blinking logic. The folder *<download_folder>\sf2_isp_using_uart_interface_demo_df\host_tool_and_samples\fabric_and_envm* contains the Libero design to generate this sample programming file.
- `isp_demo.spi`: This is the `.spi` file format version of `isp_demo.stp` file provided in *<download_folder>\sf2_isp_using_uart_interface_demo_df\stapl_programming_file*.

Note: For more information on generating `.spi` programming files, see the Appendix: Generating .spi Programming File using Libero, page 18.

2.2.2.3 ISP Execution Flow

The following figure shows the ISP flow. The UART Host PC Loader starts the communication with the SmartFusion2 device through the UART interface. On connecting with the SmartFusion2 device, the UART Host PC Loader sends the programming file size and the ISP operation mode to the target SmartFusion2 device. The SmartFusion2 device initializes the system controller and starts the ISP service in the chosen operation mode.

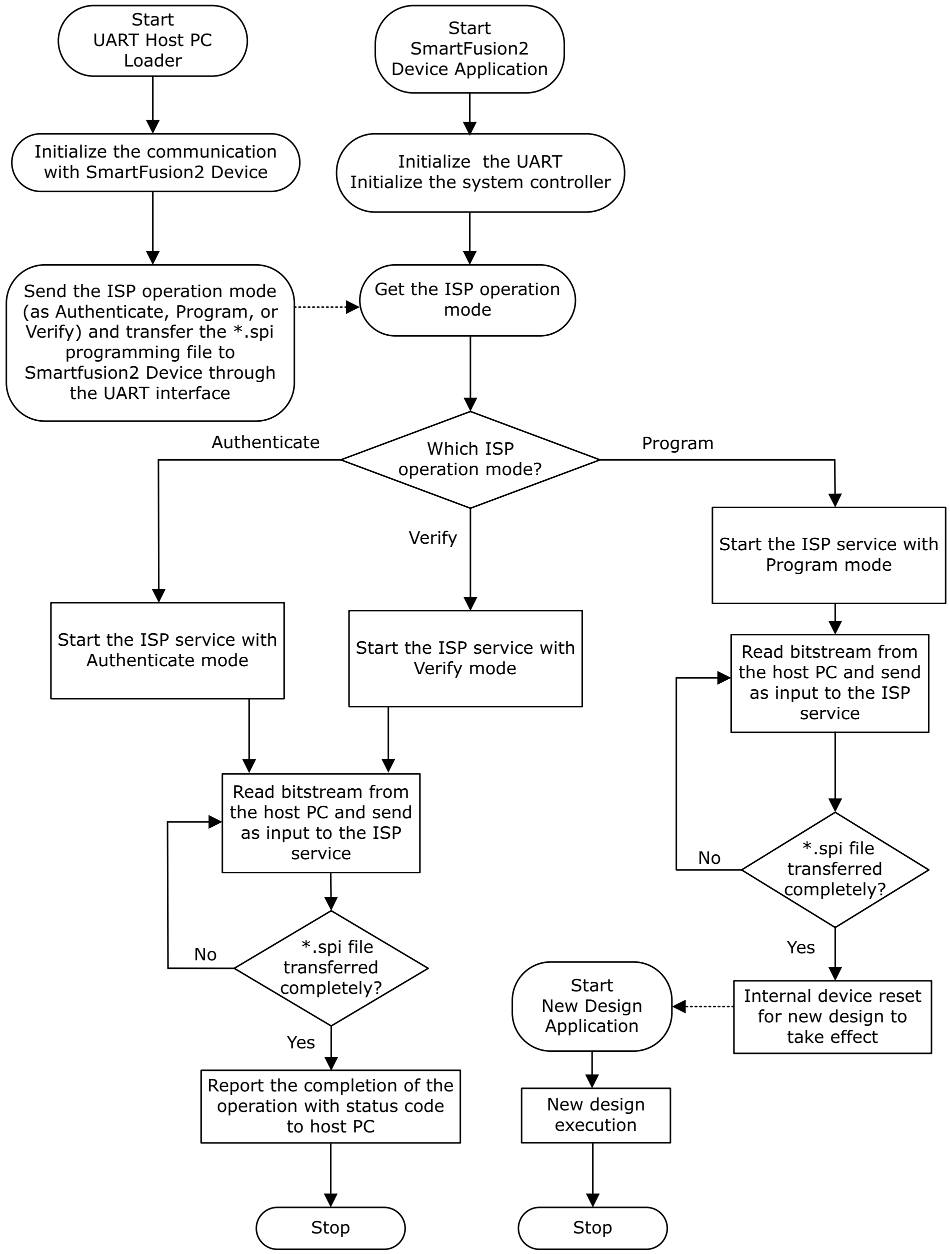
On receiving the data request from the SmartFusion2 device, the UART Host PC Loader transfers the input source programming file in blocks of 4 Kb data with cyclic redundancy check (CRC). The SmartFusion2 device:

- Stores the received 4 Kb data in a temporary buffer.
- Checks the CRC.
- Inputs the same data to the ISP service.
- Sends acknowledgment to the UART Host PC Loader for the 4 Kb data that is received and requested to send the next block of 4 Kb data.

This operation repeats until the UART Host PC Loader transfers the entire file. The UART Host PC Loader is notified with a status code when the ISP service completes the authentication or the verification process. When the operation mode is Program, an internal device reset is generated for the new design to take effect.

The following figure shows the ISP execution flow.

Figure 3 • **ISP Execution Flow**



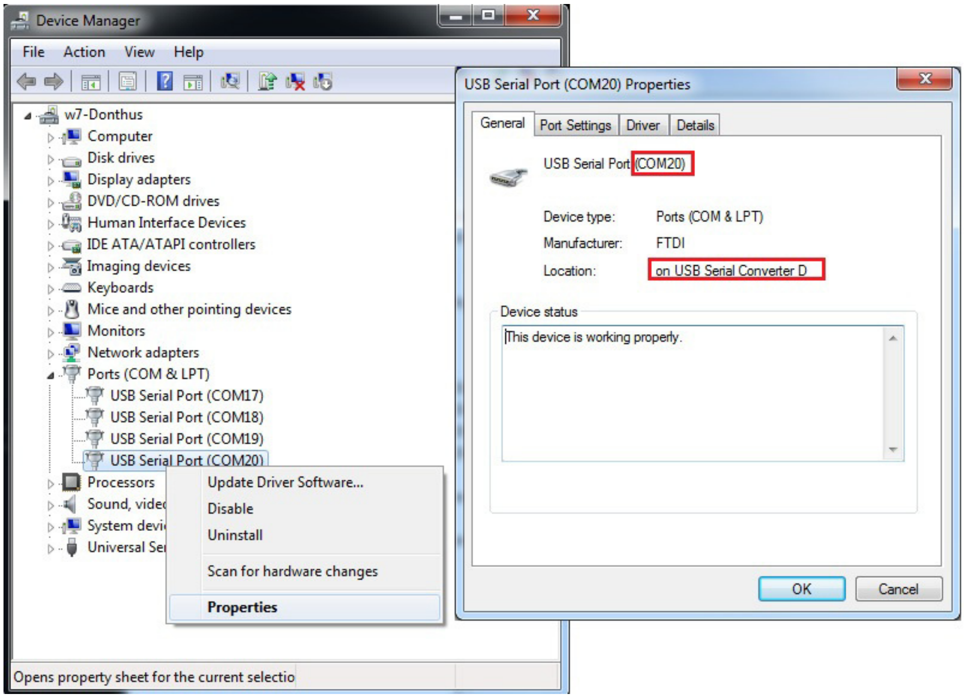
2.3 Setting Up the Demo Design

The following steps describe how to set up the demo design:

1. Connect the FlashPro4 programmer to the J5 connector of the SmartFusion2 Security Evaluation Kit board.
2. Connect the host PC to the J18 connector using the USB Mini-B cable. The USB to UART bridge drivers are automatically detected.

3. Of the four COM ports, select the one with **Location** as **on USB Serial Converter D**, as shown in the following figure.

Figure 4 • Device Manager Window



4. Connect the jumpers on the SmartFusion2 Security Evaluation Kit board as listed in the following table.
Caution: Switch off the SW7 switch on the board while making the jumper connections.

Table 3 • **SmartFusion2 Security Evaluation Kit Jumper Settings**

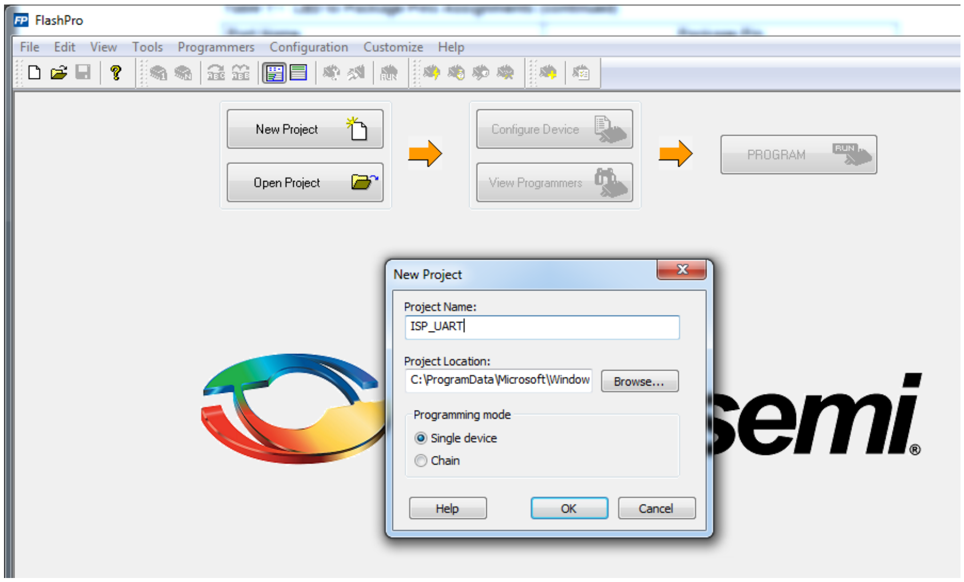
Jumper Number	Pin From	Pin To	Comments
J22, J23, J24, J8, J3	1	2	These are the default jumper settings of the SmartFusion2 Security Evaluation Kit board. Ensure that these jumpers are set properly.

5. Connect the power supply to the J6 connector.
6. Switch on the power supply switch, SW7. See Appendix: Board Setup for Running the Demo, page 15 for information on board setup.

2.4 Running the Demo Design

1. Download the demo design from:
   http://soc.microsemi.com/download/rsc/?f=m2s_dg0454_liberov11p8_df
2. Switch **ON** the SW7 power supply switch.
3. Launch the FlashPro software.
4. Click **New Project**.
5. In the **New Project** window, type the project name.

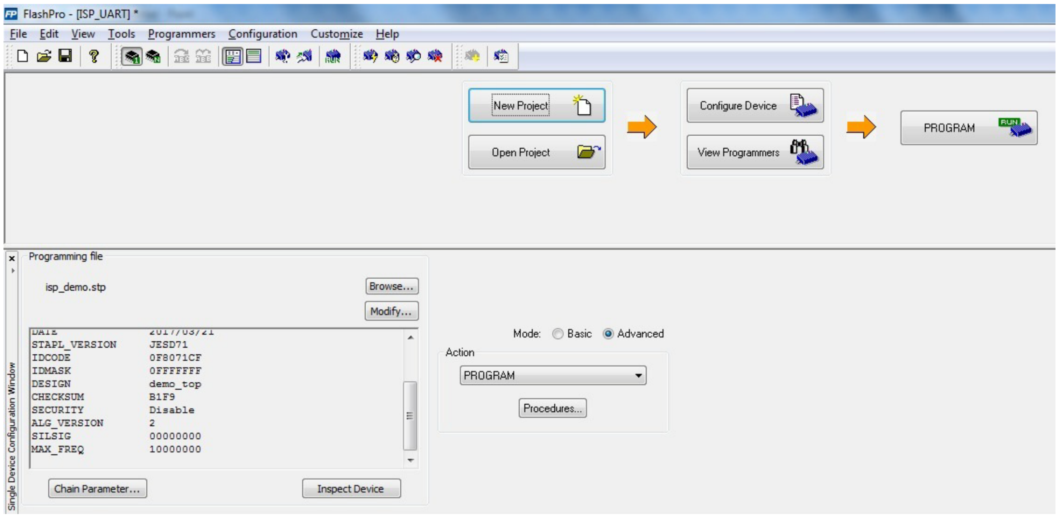
Figure 5 • **FlashPro New Project**



6. Click **Browse** and navigate to the location where you want to save the project.
7. Select **Single device** as the **Programming mode**.
8. Click **OK** to save the project.
9. Click **Configure Device**.

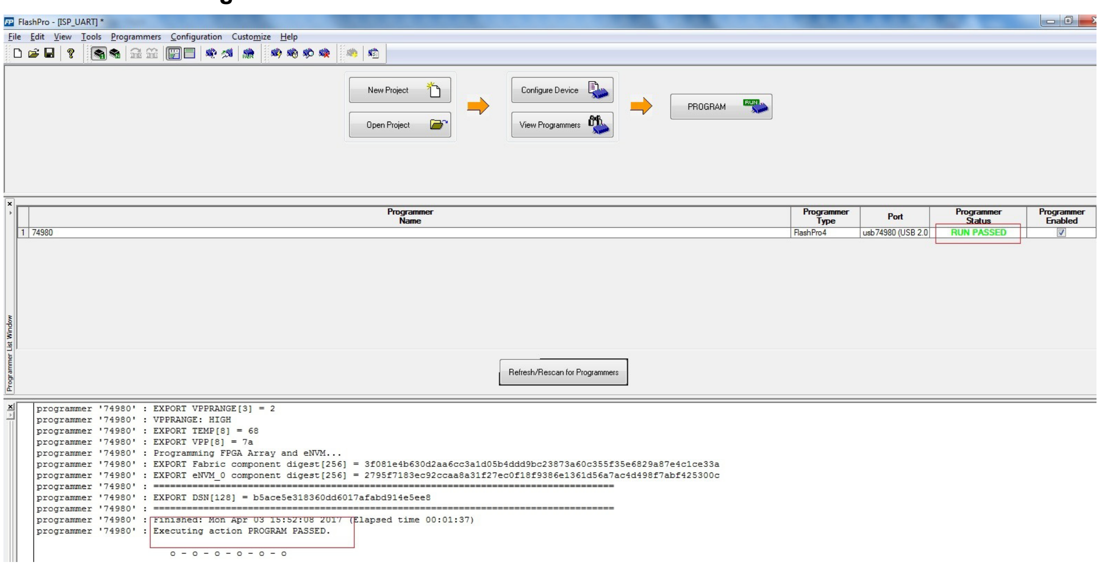
10. Click **Browse** and navigate to the location where the `isp_demo.stp` file is located and select the file. The default location is: *<download_folder>\sf2_isp_using_uart_interface_demo_df\stapl_programming_file*. The required programming file is selected and is ready to be programmed in the device.

Figure 6 • **FlashPro Project Configured**



11. Click **PROGRAM** to start programming the device. Wait until you get a message indicating that the program passed. ISP requires the SmartFusion2 device to be preprogrammed with the application code to activate the ISP service. So, the SmartFusion2 device is preprogrammed with the `isp_demo.stp` using FlashPro software.
LEDs 4 to 7 (H5, H6, J6, H7) blinking in the board indicates that the SmartFusion2 Device fabric is preprogrammed successfully.

Figure 7 • **FlashPro Program Passed**



12. Open the Command Prompt in the host PC.
13. Navigate to the directory, where the UART Host PC Loader (`M2S_UARTHost_Loader.exe`) is located. The default location is: *<download_folder>\sf2_isp_using_uart_interface_demo_df\host_tool_and_samples.*
14. Execute the `M2S_UARTHost_Loader.exe` file and launch the UART Host PC Loader to program the:
    - FPGA fabric
    - eNVM
    - FPGA fabric and eNVM

2.4.1 Example command

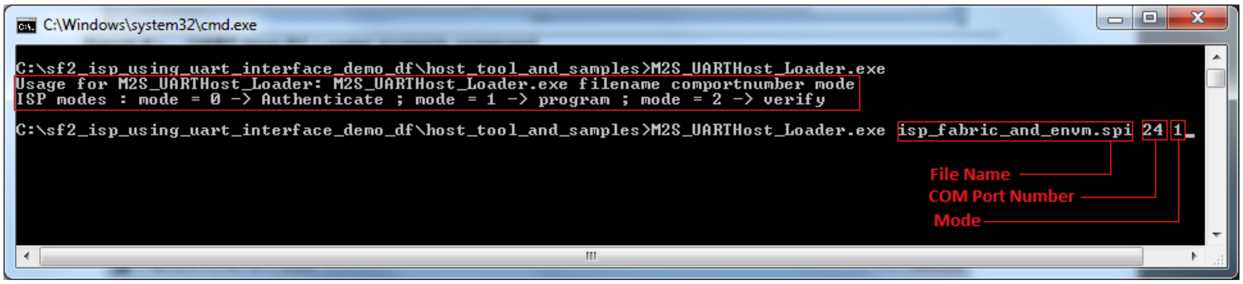
Example command for programming both the FPGA fabric and eNVM using the `isp_fabric_and_envm.spi` file:

```
M2S_UARTHost_Loader.exe isp_fabric_and_envm.spi 24 1
```

Where, 24 is the Com port number and 1 is the Operation Mode: Program

The following figure shows the UART Host PC Loader example command.

Figure 8 • **UART Host PC Loader Example Command**

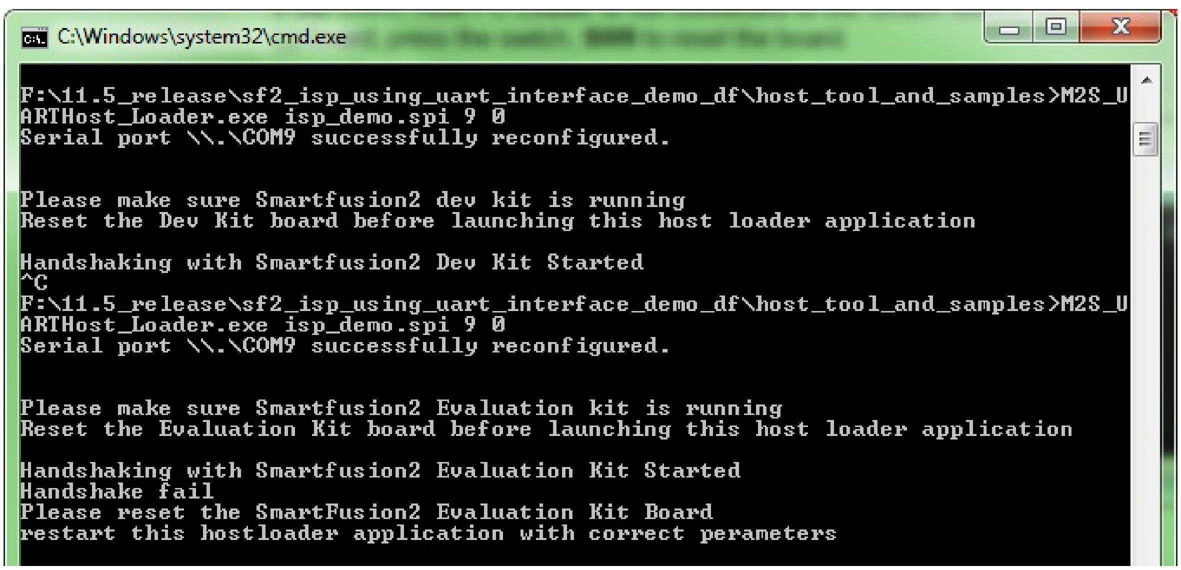


2.4.2 Resetting the board

If the UART Host PC Loader is not connected to the SmartFusion2 Security Evaluation Kit board, press the switch, SW6 to reset the board.

The following figure shows an example message that instructs to reset the board.

Figure 9 • **UART Host PC Loader Reset**



2.5 Authenticate Operation Mode

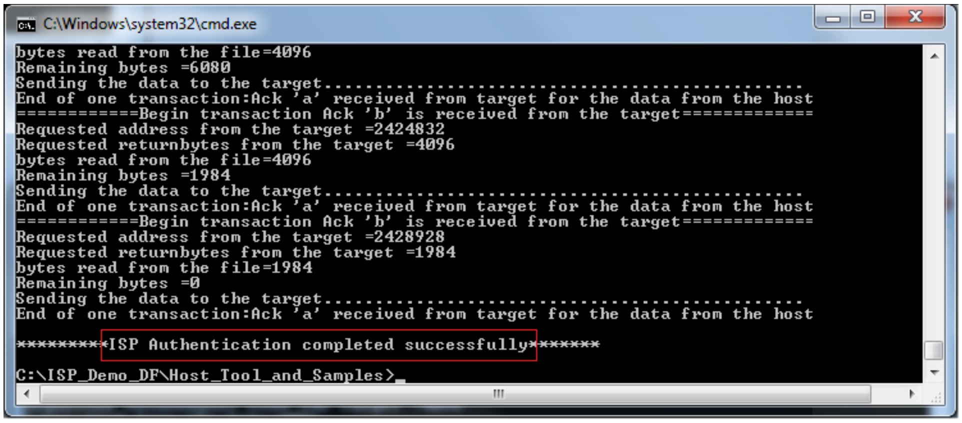
To authenticate the data from `isp_fabric_and_envm.spi`, type:

```
M2S_UARTHost_Loader.exe isp_fabric_and_envm.spi 24 0
```

Where, 24 is the Com port number and 0 is the Operation Mode: Authenticate.

On completion of the ISP authentication, the command prompt displays an operation success message. The following figure shows the operation success message.

Figure 10 • **ISP Authentication Status**



Press the switch, SW6 to reset the SmartFusion2 Security Evaluation Kit and try other ISP operation modes.

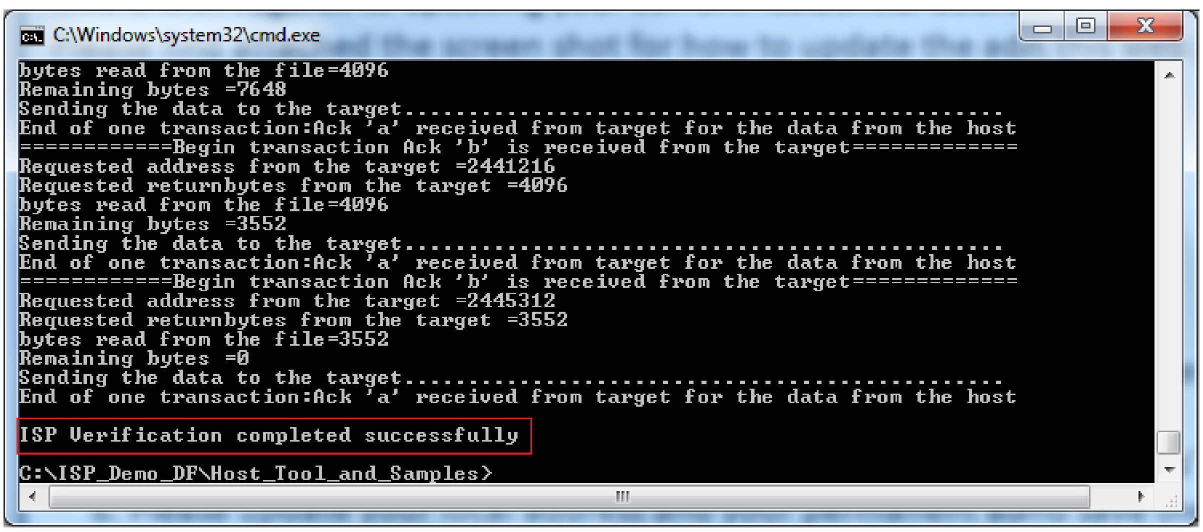
2.6 Verify Operation Mode

To verify the device FPGA fabric and eNVM contents, type the command:
`M2S_UARTHost_Loader.exe isp_demo.spi 24 2`

Where, 24 is the Com port number and 2 is the Operation Mode: Verify.

The following figure shows a successful verification message.

Figure 11 • **ISP Verification Status**



The verification operation demonstrated is for the `isp_demo.stp` file that is already running in the SmartFusion2 device. If any other `.spi` file is verified while the `isp_demo.stp` file is still running, that verification operation fails.

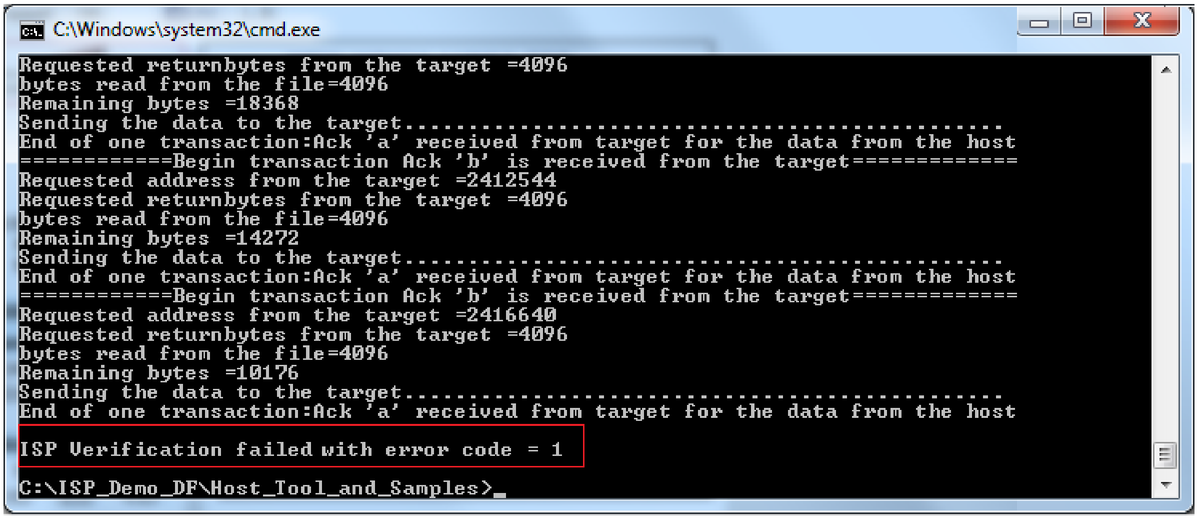
If the verification fails, the command prompt displays an error message with an error code. Figure 12, page 12 shows an example error message. For more information on error codes, see "Appendix: Error Codes" on page 17.

The programming files are at:
<download_folder>*\sf2_isp_using_uart_interface_demo_df\host_tool_and_samples.*

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

Press SW6 to reset the SmartFusion2 Security Evaluation Kit to try other ISP operation modes from CMD prompt window.

Figure 12 • **ISP Verification Failure Error Message**



2.7 Program Operation Mode

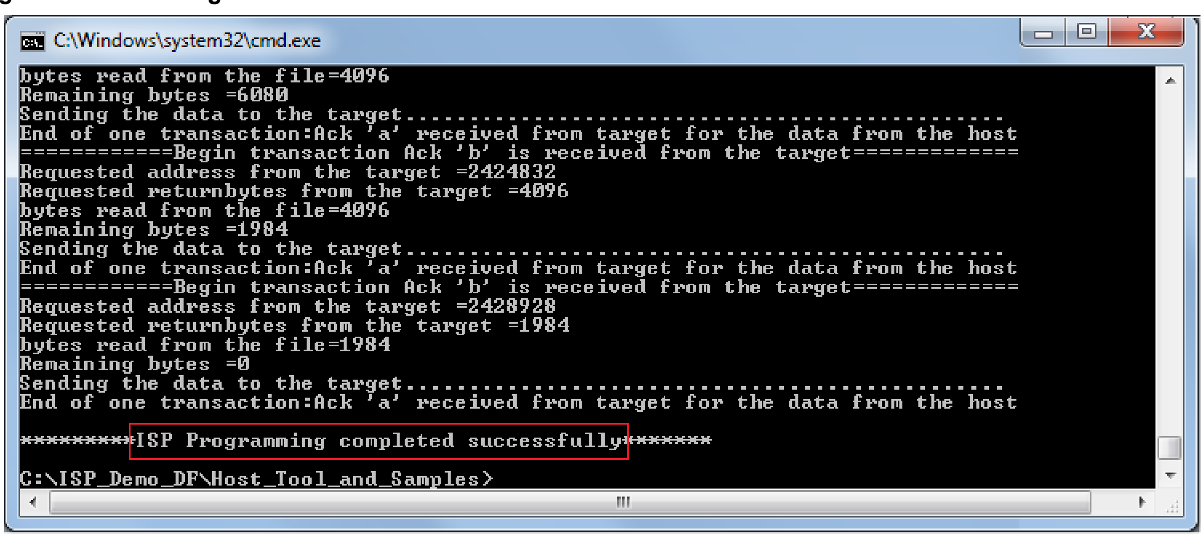
To program the FPGA fabric and the eNVM of the SmartFusion2 device using the `isp_fabric_and_envm.spi` file, type:

`M2S_UARTHost_Loader.exe isp_fabric_and_envm.spi 24 1`

Where, 24 is the Com port number and 1 is the Operation Mode: Program.

It takes a few minutes for the ISP service to complete and the FPGA fabric and eNVM are programmed. The following figure shows a successful ISP programming result.

Figure 13 • **ISP Program Status**



Press SW6 to reset the SmartFusion2 Security Evaluation Kit or power cycle the SmartFusion2 Security Evaluation Kit.

2.7.1 Checking if the Fabric is Programmed Successfully

LEDs 0 to 3 (G7, F3, F4, E1) blinking in the board indicates that the fabric is programmed successfully.

2.7.2 Checking if the eNVM is Programmed Successfully

To check if the eNVM is programmed successfully, start any serial terminal emulation program such as:

- HyperTerminal
- PuTTY
- TeraTerm

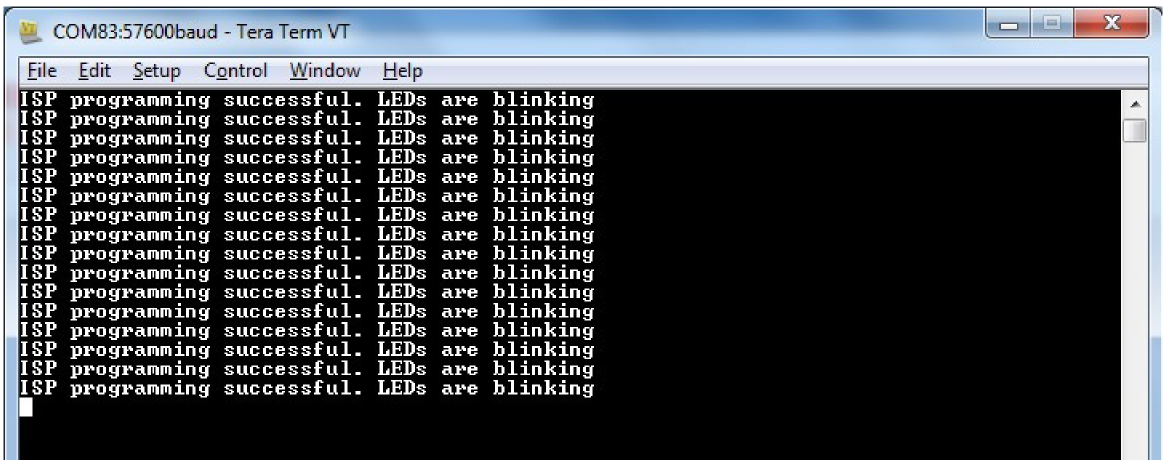
The configuration for the program is:

- Baud Rate: 57600
- 8 Data bits
- 1 Stop bit
- No Parity
- No Flow Control

For information on configuring the serial terminal emulation programs, see the *Configuring Serial Terminal Emulation Programs Tutorial*.

If the eNVM is programmed successfully, the serial terminal emulation program displays an operation success message. The following figure shows an operation success message for eNVM programming in the PuTTY window.

Figure 14 • ISP Program Successful



2.7.3 Programming Results

The result shown in the previous figure is for the `isp_fabric_and_envm.spi` file. The following table lists the possible results for ISP Program operation mode for sample programming files provided in folder *<download_folder>\sf2_isp_using_uart_interface_demo_df\host_tool_and_samples*. Not all `.spi` files listed in the table are demonstrated.

Table 4 • **ISP Programming Results**

*.spi Programming File Name	eNVM Programming Result	FPGA fabric Programming Result
isp_envm_only.spi	The serial terminal emulation program shows successful eNVM program message	NA
isp_fabric_only.spi	NA	SmartFusion2 LEDs 0 to 3 blinks
isp_fabric_and_envm.spi	The serial terminal emulation program shows successful eNVM program message	SmartFusion2 LEDs 0 to 3 blinks

Note: After successful ISP Program operation, the Security Evaluation Kit must be reprogrammed with the original `isp_demo.stp` file to try the ISP operation modes again.

2.8 Known Issue

After successful completion of the two-step IAP or 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 put the device in Flash *Freeze and exit from Flash *Freeze 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 23.

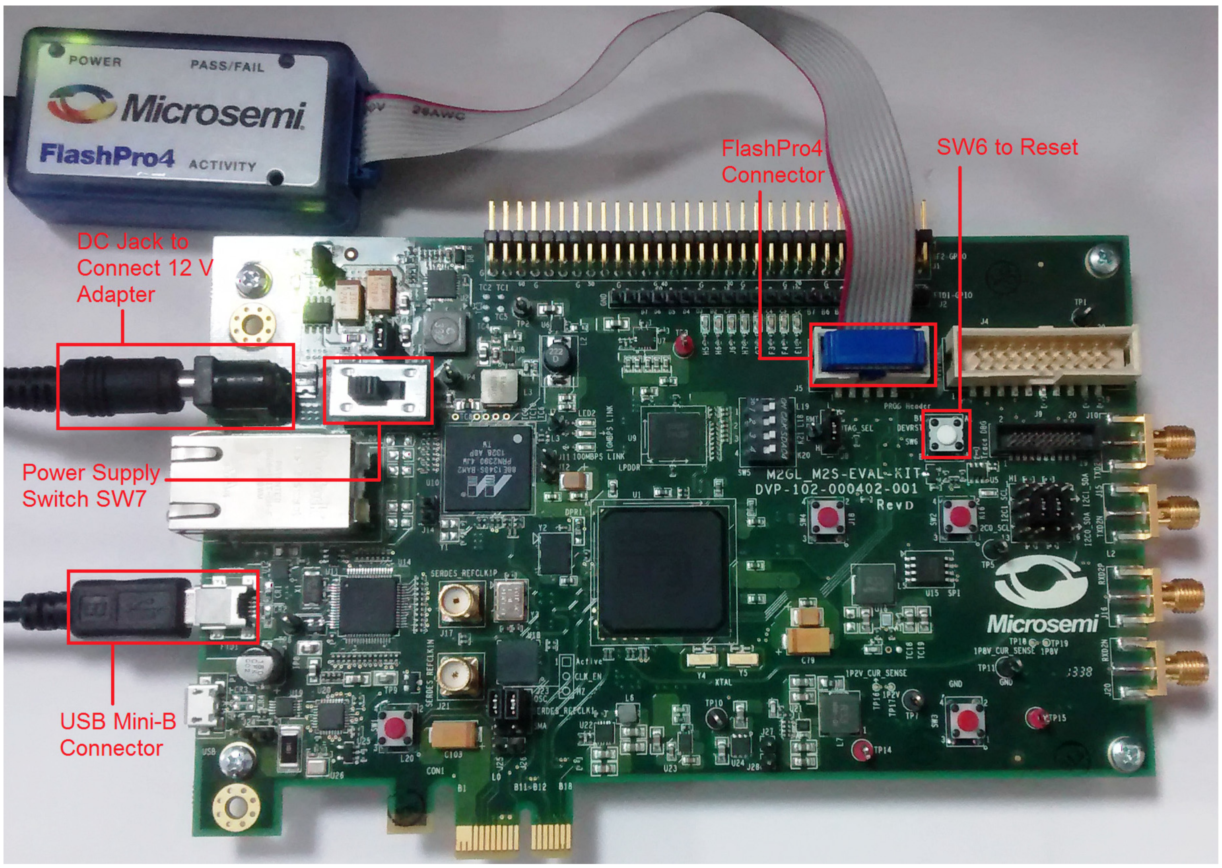
The design example provided in this demonstration implements the workaround for accessing LSRAM after implementing the IAP or ISP program operation in Libero software, and the design files are available in the following location:
<download_folder>\sf2_isp_using_uart_interface_demo_df\host_tool_and_samples\LSRAM_Workaroud

The LSRAM write and read accesses are denied after implementing IAP or ISP program operation. The workaround for this problem is to put the device in Flash *Freeze and exit from Flash *Freeze after IAP or ISP program operation.

3 Appendix: Board Setup for Running the Demo

The following figure shows the board setup for running the demo on the SmartFusion2 Security Evaluation Kit board.

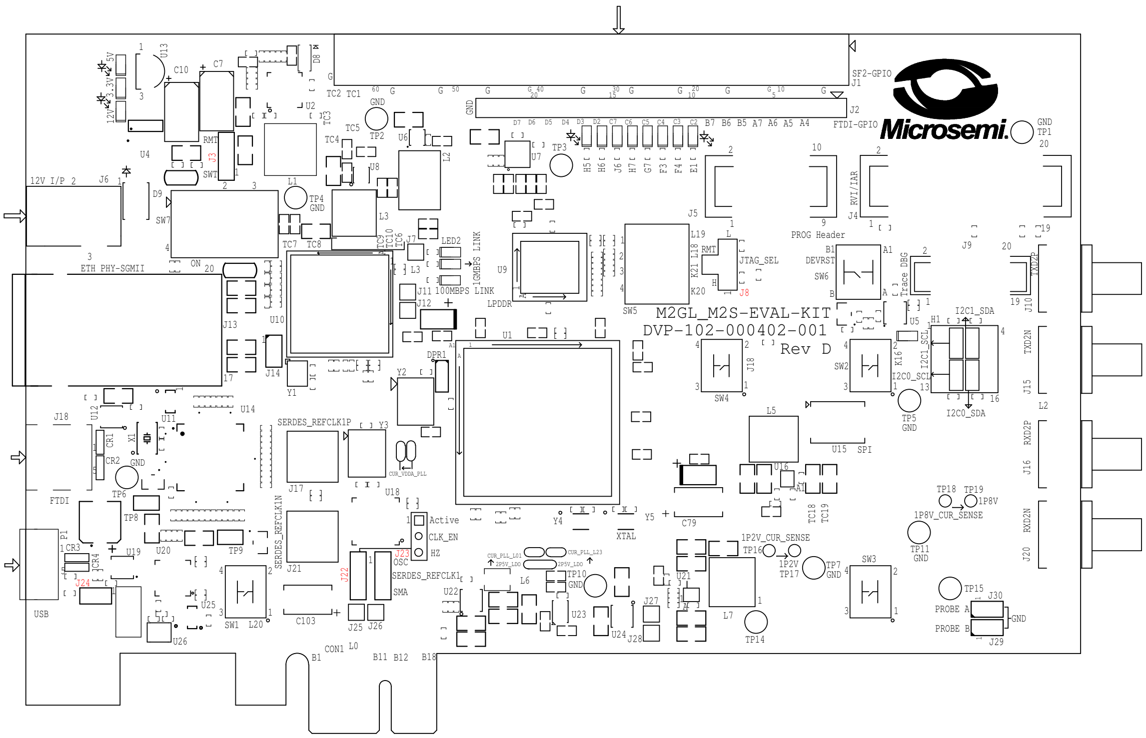
Figure 15 • **Board Setup for Running the Demo**



4 Appendix: Jumper Locations

The following figure shows the jumper locations in SmartFusion2 Security Evaluation Kit board.

Figure 16 • **SmartFusion2 Security Evaluation Kit Silkscreen Top View**



Note: Jumpers highlighted in red are set by default.

Note: The location of the jumpers in Figure 16 are searchable.

5 Appendix: Error Codes

The following table lists the error codes in SmartFusion2 Security Evaluation Kit board.

Table 5 • Error Codes

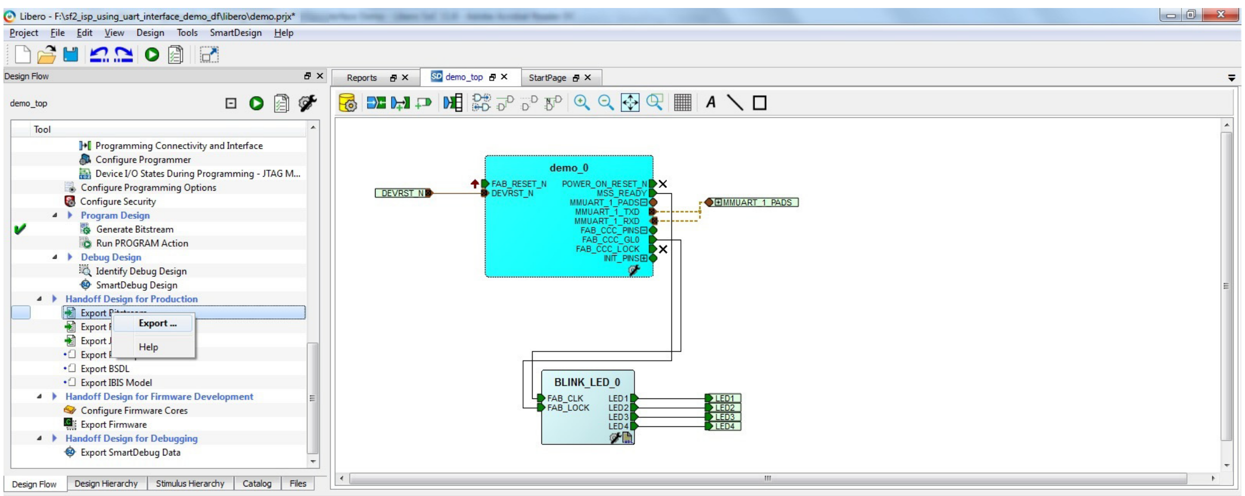
Define	Error Code	Description
#define MSS_SYS_CHAINING_MISMATCH	1u	Device contents mismatch
#define MSS_SYS_UNEXPECTED_DATA_RECEIVED	2u	Data is not supported
#define MSS_SYS_INVALID_ENCRYPTION_KEY	3u	Invalid encryption key
#define MSS_SYS_INVALID_COMPONENT_HEADER	4u	Invalid file header
#define MSS_SYS_BACK_LEVEL_NOT_SATISFIED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

6 Appendix: Generating .spi Programming File using Libero

The following steps describe how to generate `.spi` programming file using the Libero SoC software:

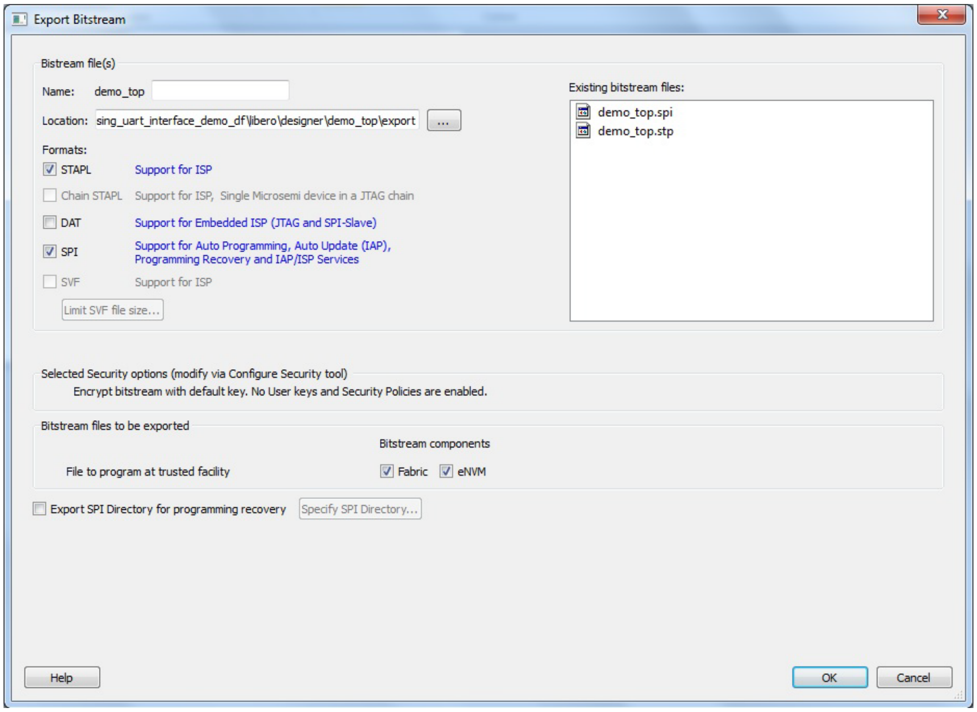
1. Launch the Libero SoC software to open a Libero project for `isp_fabric_and_envm.spi` programming file. The Libero design file is provided in *<download_folder>\sf2_isp_using_uart_interface_demo_df\host_tool_and_samples\fabric_and_envm*.
2. Right-click **Export Bitstream** under **Handoff Design for Production** in the **Design Flow** tab, and click **Export ...** from the context menu.

Figure 17 • **Configuring Export Bitstream**



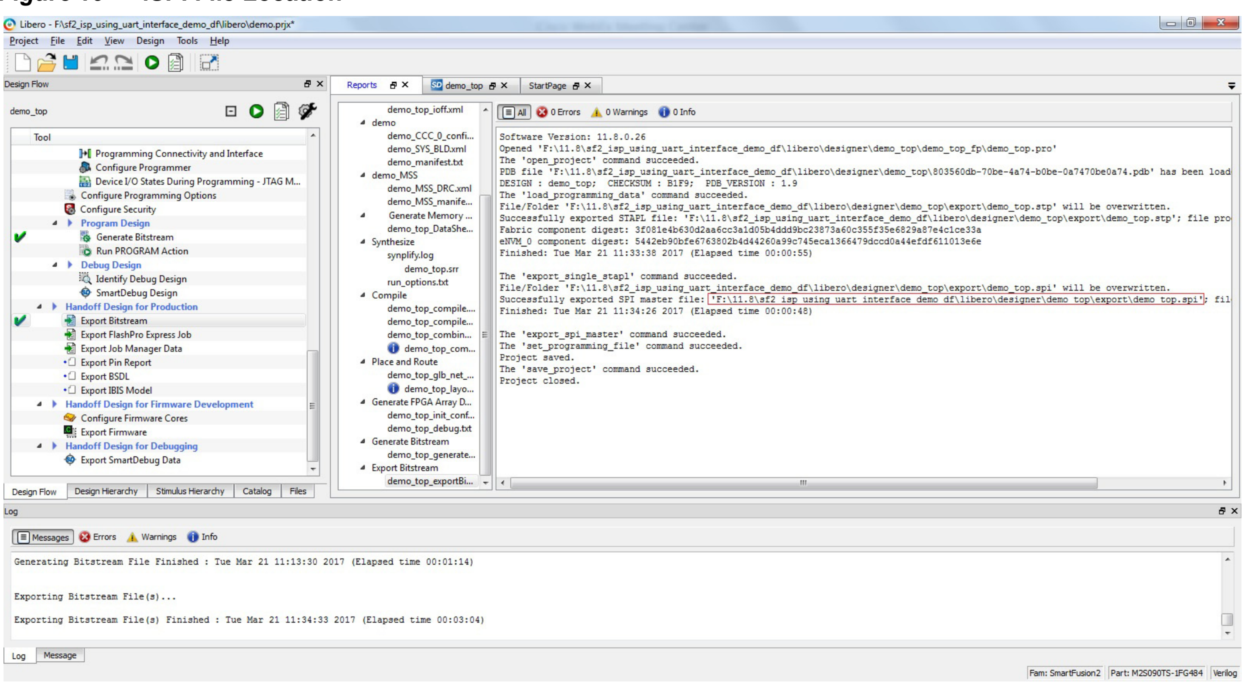
3. On the **Export Bitstream** window, select the **SPI file** check box.

Figure 18 • **Export Programming File Options Window**



4. Click **OK**.
5. Double-click **Export Bitstream** under **Handoff Design for Production** in the **Design Flow** tab to generate the `.spi` file (Figure 17, page 18). The following figure shows the `.spi` file location in **Reports** tab

Figure 19 • **.SPI File Location**



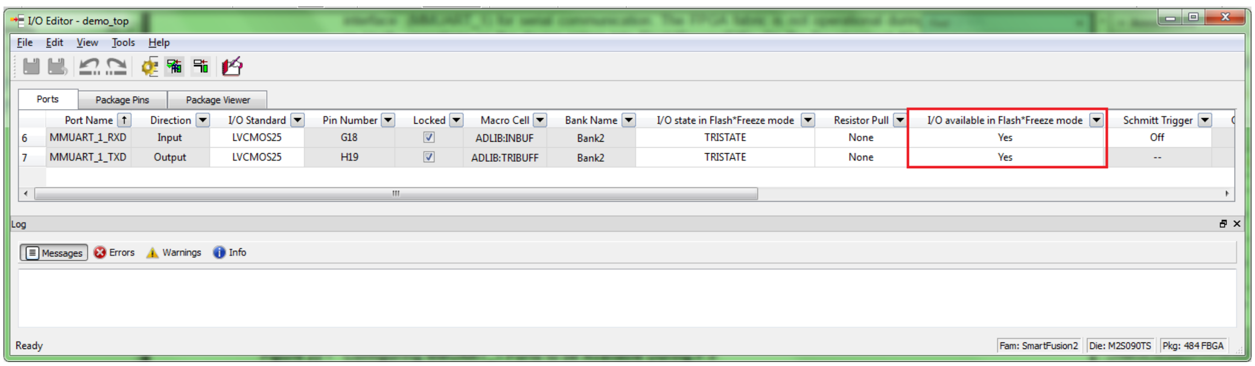
7 Appendix: Hardware Project Implementation Settings

The following hardware project settings are required to build the demo design.

7.1 Configuring the I/Os for Flash*Freeze Mode

The Libero demo design configures M3_CLK to operate at 50MHz and one UART interface (MMUART_1) for serial communication. The FPGA fabric is not operational during Program or Verify operations as the device enters into Flash*Freeze (F*F). On the Security Evaluation Kit board, the MMUART_0 TX and RX are connected to the mini-B USB through the fabric and fabric I/Os. During F*F mode, the fabric and I/Os are not available. So the MMUART_0 cannot be used as the serial communication interface. As such, MMUART_1 is used, and the RXD and TXD ports are configured using the I/O Editor to be available during F*F mode, as shown in the following figure. The user has to Check the settings from the File menu after configuring the ports.

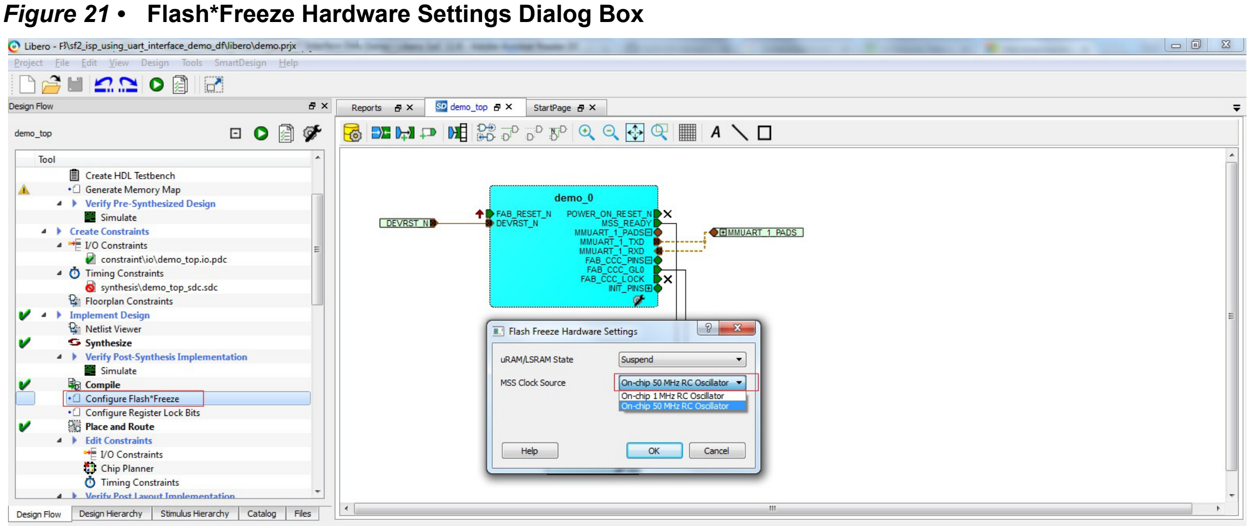
Figure 20 • **Configuring MMUART_1 Ports to be Available During F*F**



7.2 Standby Clock Source Configuration

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

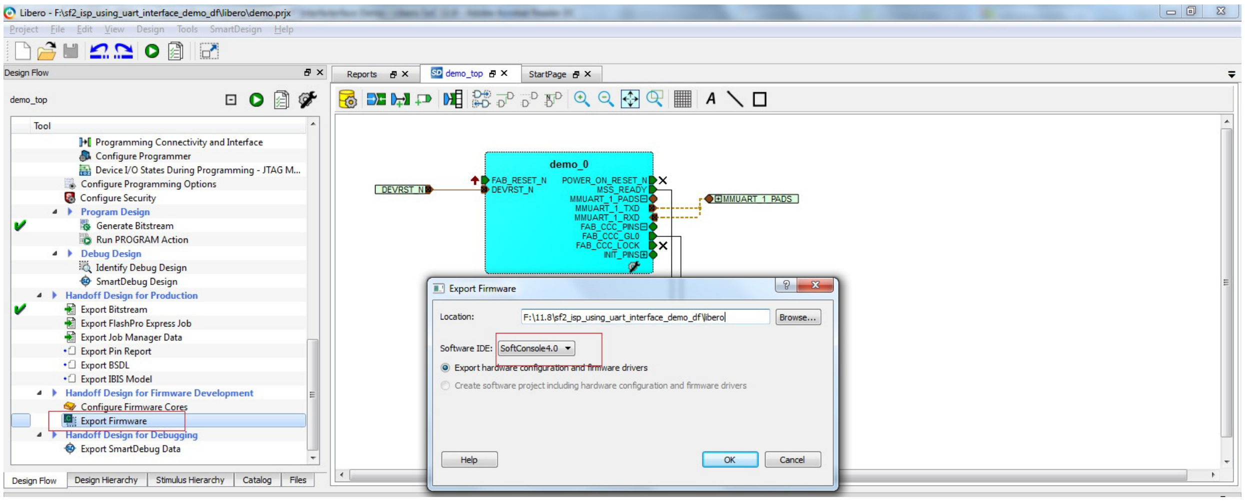
Figure 21 • **Flash*Freeze Hardware Settings Dialog Box**



7.3 SoftConsole Project Generation

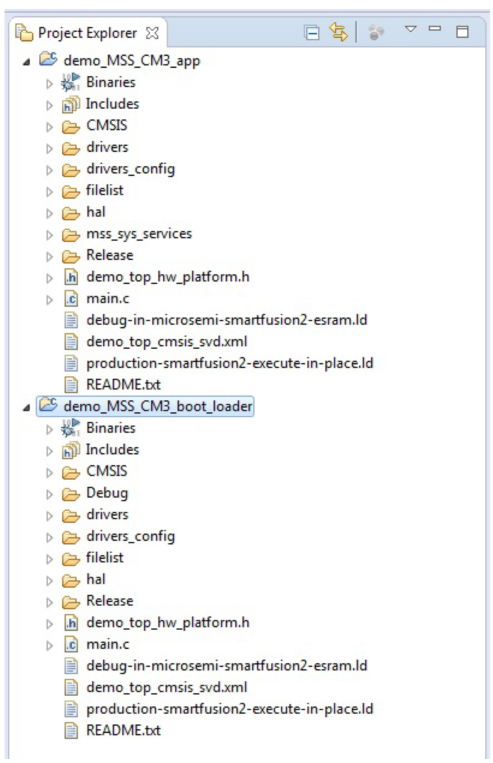
The firmware can be generated by checking the Create Project and selecting a Software IDE option in Libero project as shown in the following figure.

Figure 22 • Export Firmware Options



On successful firmware generation, the firmware and SoftConsole folders are generated at *<download_folder>\sf2_isp_using_uart_interface_demo_df\libero* as specified in Location field of Export Firmware dialog box, as shown in the following figure.

Figure 23 • SoftConsole Project Workspace



The SoftConsole workspace consists of three projects.

- demo_MSS_CM3_app
  This project receives the bitstream from the Host PC through UART interface and invokes the system controller programming services.

- demo_MSS_CM3_boot_loader
  This project implements the remapping of the eSRAM to Cortex-M3 processor code space after copying the ISP code to eSARM from eNVM.
- demo_MSS_CM3_hw_platform
  This project contains all the firmware and hardware abstraction layers that correspond to the hardware design. This project is configured as a library and is referenced by `demo_MSS_CM3_app` and `demo_MSS_CM3_boot_loader` application projects.

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

The LSRAM write and read accesses are denied after implementing IAP or ISP program operation. The workaround for this problem is to apply System Reset after IAP or ISP program operation. This workaround can be implemented by one of the following ways.

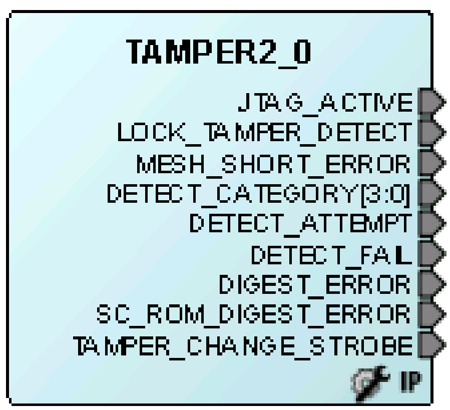
- Using SmartDesign
- Importing the `.cxf` file

8.1 Using SmartDesign

The following steps describe how to apply System Reset:

1. Go to **File** > **New** > **SmartDesign**.
2. Enter **Name** as **Dev_Restart_after_ISP_blk** in the **Create New SmartDesign** window.
3. Navigate to **Libero Catalog** to open **Tamper Macro**.
   a. Drag-and-drop the **Tamper Macro** available in **Libero Catalog** to the **Dev_Restart_after_ISP_blk SmartDesign** canvas, as shown in the following figure.

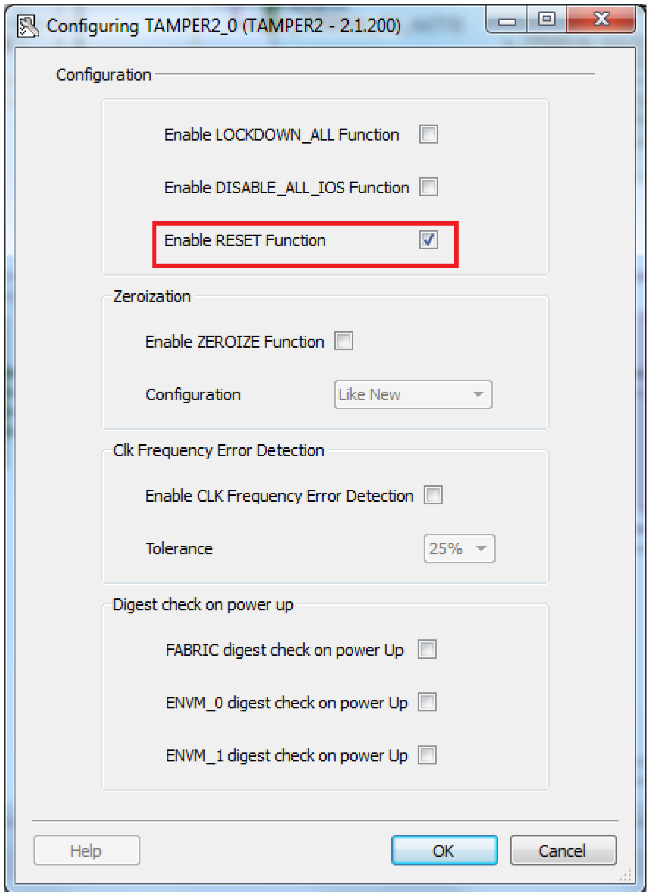
Figure 24 • **Tamper Macro**



b. Select the **Enable RESET Function** check box in the **Configuring Tamper 2_0** window.

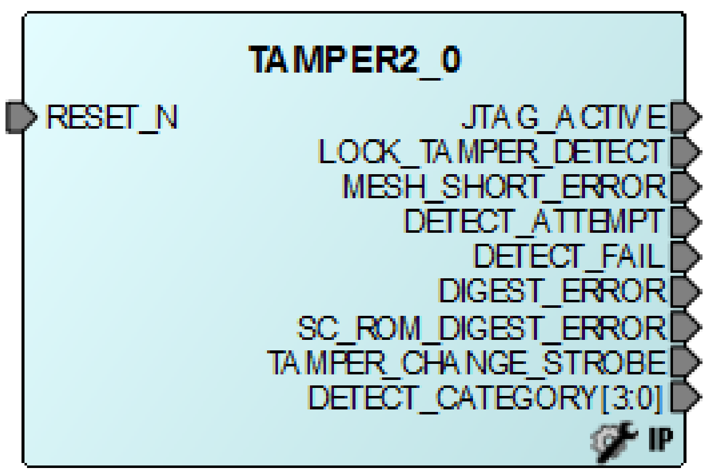
c. Click **OK**. The **System Reset** is enabled.

Figure 25 • **Tamper Macro Configuration Window**



The following figure shows the TAMPER2_0 macro after configuration.

Figure 26 • **Tamper Macro**

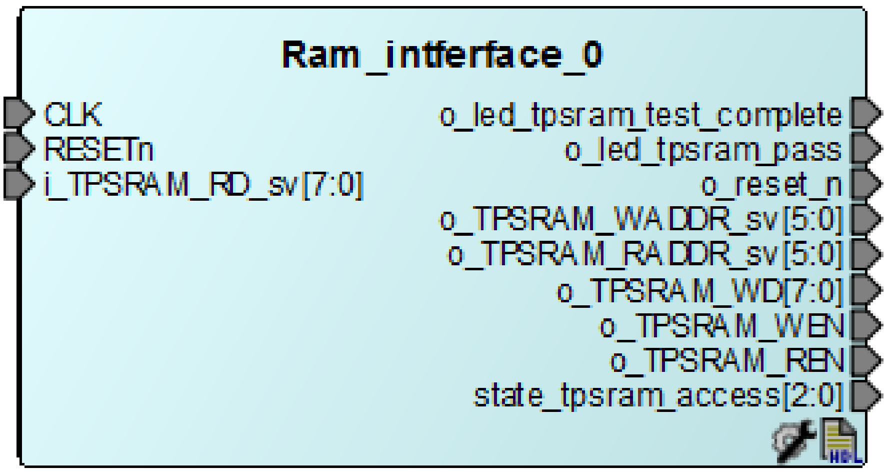


4. Instantiate the **FSM Module** provided in the design files. This FSM Logic performs 3 consecutive address writes to the Two-Port Large SRAM with the known data pattern and then reads back data from those 3 consecutive address locations to compare. If the read back data pattern does NOT match with the written data pattern, then the FSM asserts the RESET_N input to Tamper Macro, which in turn causes a System Reset. If the read back data pattern matches with the written data pattern, then the FSM does not do anything. Follow the steps to add the FSM logic to the PCIe IAP design:
   a. Choose **File** > **Import** > **HDL Source Files**.

   b. Browse to the following **Ram_interface.v file** location in the design files folder.
   <download_folder>\sf2_isp_using_uart_interface_demo_df\Source_files

   c. Click the **Dev_Restart_after_ISP_blk** tab and drag-and-drop the **Ram_interface** component from the **Design Hierarchy** to the **Dev_Restart_after_ISP_blk SmartDesign** canvas. The following figure shows the **Ram_interface** component.

Figure 27 • **Ram_interafce FSM Component**



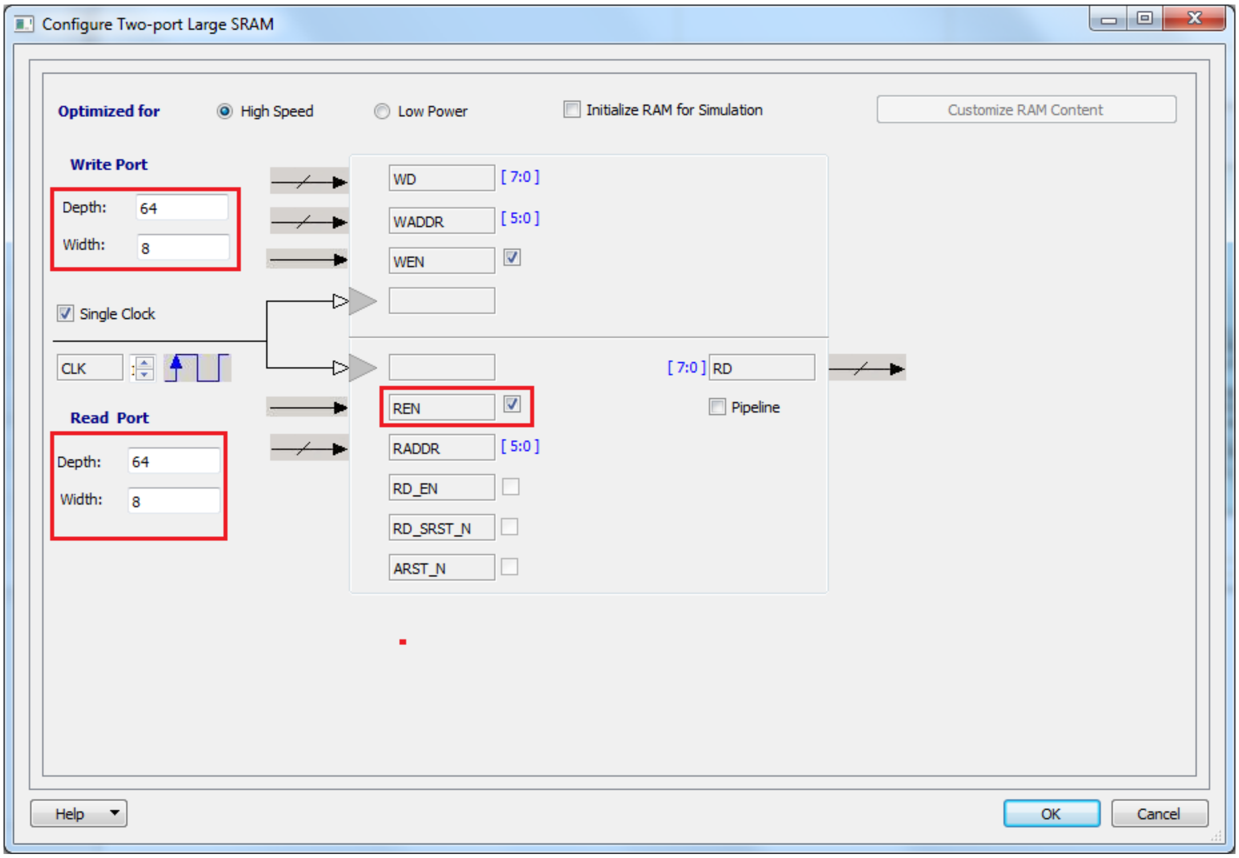
Upon completion of IAP programming, the System Controller asserts POWER_ON_RESET_n to FPGA fabric. This triggers the RESETn signal and initiates the state machine in the FSM module.

5. Drag-and-drop the **Two-Port Large SRAM (TPSRAM)** available in the **Libero Catalog** to the **Dev_Restart_after_ISP_blk SmartDesign** canvas. Configure the **TPSRAM** with the following settings:
   – Write Port
   - Depth: 64
   - Width: 8

   – Read Port
   - Depth: 64
   - Width: 8

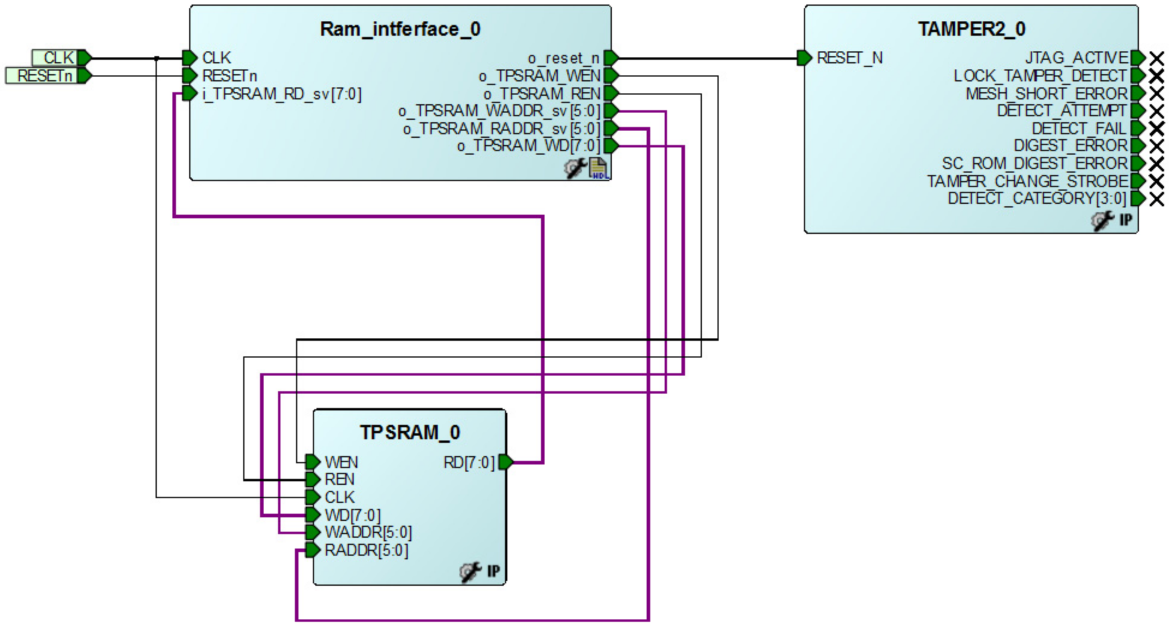
   – Select **Check REN** check box

Figure 28 • **Two-Port SRAM Configurator Window**



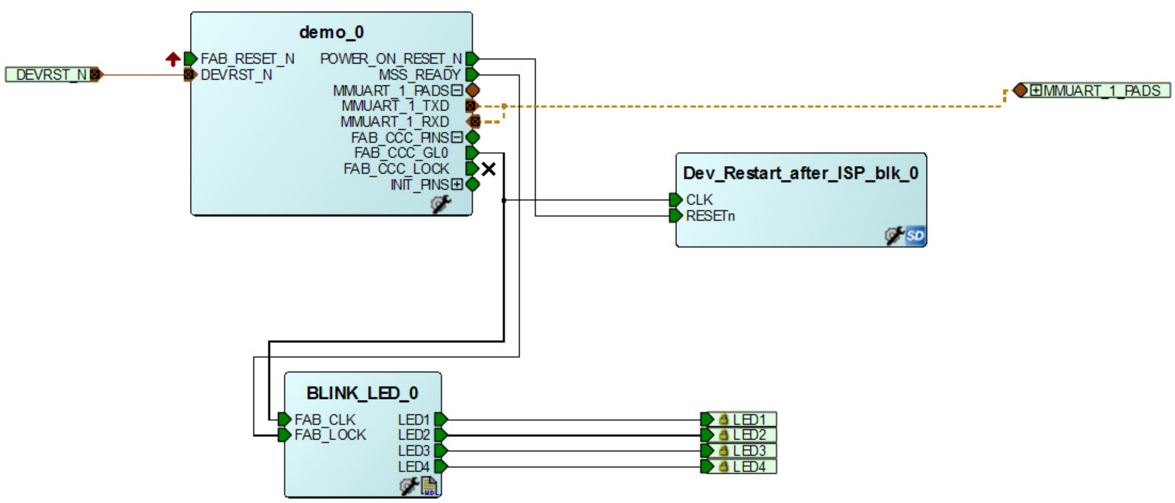
6. Make the connections for **Tamper Macro**, **FSM**, and **TPSRAM**, as shown in the following figure.

Figure 29 • **Dev_Restart_after_ISP_blk SmartDesign**



7. Click the **demo_top** tab and drag-and-drop the **Dev_Restart_after_ISP_blk** component from the **Design Hierarchy** to the **demo_top SmartDesign** canvas.
8. Make the connection as shown in the following figure and generate **demo_top SmartDesign**. This completes the implementation of the workaround.

Figure 30 • **demo_top SmartDesign**



Note: This workaround is applicable for v11.5 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 might 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.

8.2 Importing the .cxf File

Import the `.cxf` file for SmartDesign Dev_Restart_after_ISP_blk. This .cxf file is provided with the design files and it has all the component instantiations and connections mentioned in Using SmartDesign, page 23 from step 1 to 6.

The following steps describe how to import `.cxf` file to Libero project:

1. Choose **File** > **Import** > **Others**.
2. Browse to the following **Dev_Restart_after_ISP_blk.cxf** file location in the design files folder.
   <download_folder>\sf2_isp_using_uart_interface_demo_df\host_tool_and_samples\LSRAM_Workaround\component\work\Dev_Restart_after_ISP_blk
3. Browse to the following ***Ram_interface.v*** *file* location in the design files folder.
   <download_folder>\sf2_isp_using_uart_interface_demo_df\Source_files
   Repeat **Step 7** and **Step 8** to instantiate **Dev_Restart_after_ISP_blk** in **demo_top SmartDesign**.