
***SmartFusion2 SoC FPGA In-System
Programming Using UART Interface -
Libero SoC v11.4***

Demo Guide

Superseded

August 2014

Revision History

Date	Revision	Change
August 18, 2014	4	Fifth release
May 3, 2014	3	Fourth release
16 December 2013	2	Third release
02 December 2013	1	Second release
27 August 2013	0	First release

Confidentiality Status

This document is a non-confidential.

Table of Contents

Preface	4
About this document	4
Intended Audience	4
References	4
Microsemi Publications	4
In-System Programming Using UART Interface	5
Introduction	5
Requirements and Details	5
Demo Design	5
Introduction	5
Demo Design Features	7
Demo Design Description	7
Setting Up the Demo Design	10
Alternate Setup	11
Board Setup Snapshot	11
Running the Demo Design	12
Example command	14
Resetting the board	15
Authenticate Operation Mode	15
Verify Operation Mode	16
Program Operation Mode	17
Checking if the fabric is programmed successfully	18
Checking if the eNVM is programmed successfully	18
Programming Results	19
Appendix 1: Connecting the SmartFusion2 Device to the Host PC Through the USB to UART (FTDI) Interface ..	20
Appendix 2: Board Setup when Using the USB-RS232 Serial Adapter or RS232 Cable	21
Appendix 3: Board setup through the USB to UART (FTDI) interface using the USB A to Mini - B Cable	22
Appendix 4: Jumper Locations	23
Appendix 5: Error Codes	24
Appendix 6: Generating .spi Programming File using Libero	25
Appendix 7: Hardware Project Implementation Settings	28
Configuring the I/Os for Flash*Freeze Mode	28
Standby Clock Source Configuration	29
Softconsole Project Generation	30
A List of Changes	32
B Product Support	33
Customer Service	33
Customer Technical Support Center	33
Technical Support	33
Website	33
Contacting the Customer Technical Support Center	33
Email	33
My Cases	34
Outside the U.S.	34
ITAR Technical Support	34

Preface

About this document

This demo is for SmartFusion[®]2 system-on-chip (SoC) field programmable gate array (FPGA) devices. It provides instructions on how to use the corresponding reference design.

Intended Audience

The following designers using the SmartFusion2 devices:

- FPGA designers
- Embedded designers
- System-level designers

References

Microsemi Publications

- SmartFusion2 Programming User Guide
- SmartFusion2 System Controller User Guide
- SmartFusion2 Microcontroller Subsystem User Guide
- SmartFusion2 SoC FPGA Remapping eNVM, eSRAM, and DDR/SDR SDRAM Memories Application Notes
- Configuring Serial Terminal Emulation Programs Tutorial

See the following web page for a complete and up-to-date listing of SmartFusion2 device documentation:
<http://www.microsemi.com/products/fpga-soc/soc-fpga/smartfusion2#documents>.

In-System Programming Using UART Interface

Introduction

In-system programming (ISP) can be used to reprogram for design iterations and field upgrades. SmartFusion2 devices support ISP using 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 *SmartFusion2 Programming User Guide*. For information on system controller programming services, see the *SmartFusion2 System Controller User Guide*.

Requirements and Details

Table 1 • Reference Design Requirements and Details

Reference Design Requirements and Details	Description
Hardware Requirements	
SmartFusion2 Development Kit <ul style="list-style-type: none">• 12 V adapter• FlashPro4 programmer• USB A to Mini-B cable	Rev D or later
USB-RS232 Serial adapter or RS232 cable	-
Host PC or Laptop	Windows 64-bit Operating System
Software Requirements	
Libero [®] System-on-Chip (SoC) for viewing the design files	v11.4
FlashPro Programming Software	v11.4
Host PC Drivers	USB to UART drivers

Demo Design

Introduction

The demo design files are available for download from the following path in the Microsemi[®] website: http://soc.microsemi.com/download/rsc/?f=sf2_isp_using_uart_interface_demo_11p4_df. The demo design files include:

- Libero SoC software project
- STAPL programming files
- UART Host PC Loader application (M2S_UARHost_Loader.exe)
- Sample programming files

Figure 1 shows the top-level structure of the design files. For further details, see the `readme.txt` file.

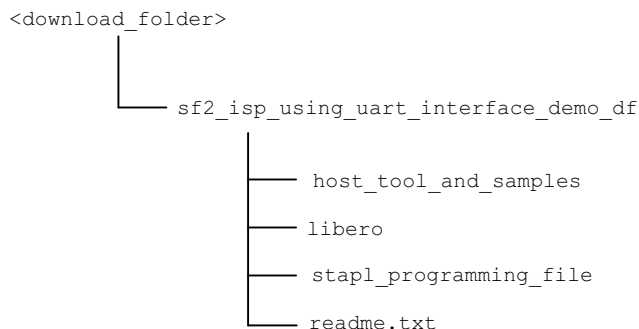


Figure 1 • Demo design files top level structure

Figure 2 describes the 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 bitstreams to the ARM® Cortex™-M3 processor. Refer to the "Appendix 7: Hardware Project Implementation Settings" section.

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 "Demo Design Description" on page 7 for information on modes of operation.

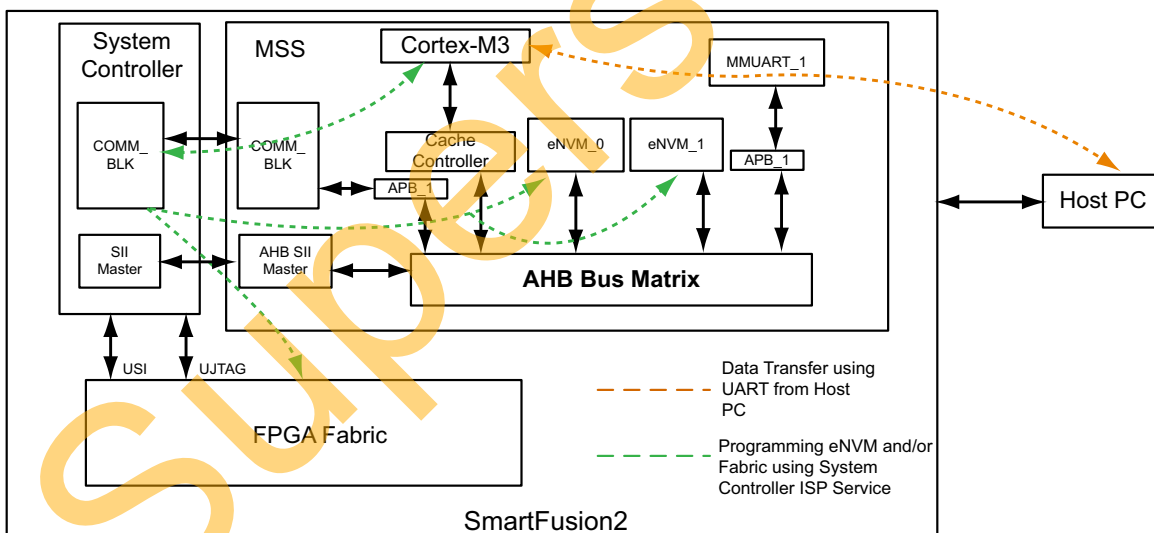


Figure 2 • Top-Level demo diagram

There are two ways to connect the Host PC to the SmartFusion2 device:

- Using the USB-RS232 Serial adapter or the RS232 cable.
 - See "Running the Demo Design" section on page 12.
- Using the USB to UART (FTDI) interface.
 - See "Appendix 1: Connecting the SmartFusion2 Device to the Host PC Through the USB to UART (FTDI) Interface" on page 20.

Demo Design 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.

Demo Design 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. Refer to the [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. Refer to the [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 since the eNVM programming is not taking place. Refer to the ["Appendix 7: Hardware Project Implementation Settings" section](#).

UART Host PC Loader

UART Host PC Loader (M2S_UARHost_Loader.exe) is an executable program that transfers the programming files (*.spi) from the Host PC to the SmartFusion2 Development Kit board. The M2S_UARHost_Loader.exe file is executed from the command prompt. It is located at: `<download_folder>\sf2_isp_using_uart_interface_demo_dfhost_tool_and_samples`.

The syntax is:

```
M2S_UARHost_Loader.exe <*.spi> <COM Port number> <Operation Mode>
```

Arguments:

- *.spi programming file.
- COM Port number.
- Operation Mode. See [Table 2](#).

For more information, see ["Running the Demo Design" on page 12](#).

[Table 2](#) 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

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_dfhost_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_dfhost_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_dfstapl_programming_file`.

Note: For more information on generating .spi programming files, refer to the ["Appendix 6: Generating .spi Programming File using Libero" section on page 25](#).

ISP Execution Flow

[Figure 3 on page 9](#) describes 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 starts transferring 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 was received and requests 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.

Figure 3 shows the ISP execution flow.

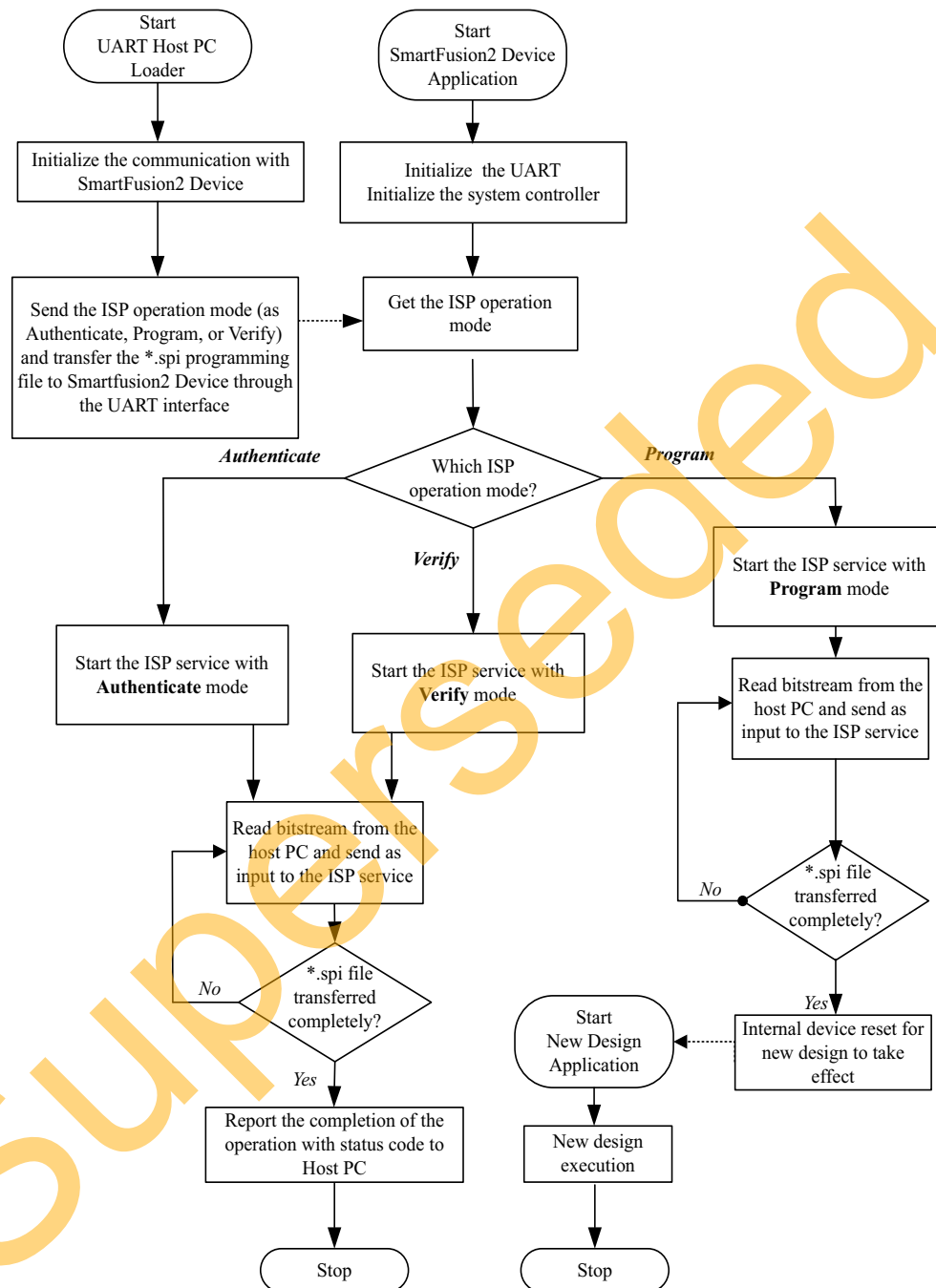


Figure 3 • ISP Execution Flow

Setting Up the Demo Design

1. Connect the FlashPro4 programmer to the J59 connector of the SmartFusion2 Development Kit board.
2. Connect the Host PC to the DB9-RS232 connector of the SmartFusion2 Development Kit board using the USB-RS232 serial adapter cable or the RS232 cable.
When using USB-RS232 serial adapter cable, make sure that the USB-RS232 serial adapter drivers are automatically detected. [Figure 4](#) shows an example **Device Manager** window that has the **USB-to-Serial Comm Port** listed under **Ports (Comm & LPT)**. COM port number is required to run the demo design, so make a note of it.

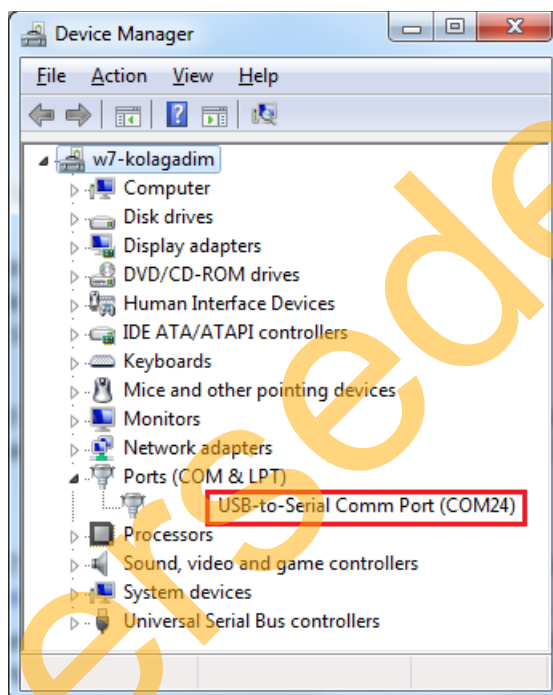


Figure 4 • Device Manager window showing the USB-to-Serial Communication Port

3. Connect the jumpers on the SmartFusion2 Development Kit board, as described in [Table 3](#) on [page 11](#). For information on jumper locations, see "[Appendix 4: Jumper Locations](#)" section on [page 23](#).
 - **Caution:** Before making the jumper connections, switch off the power supply switch, SW7.

Table 3 • SmartFusion2 Development Kit Jumper Settings

Jumper Number	Settings	Notes
J70, J93, J94, J117, J123, J142, J157, J160, J167, J225, J226, J227	1-2 closed	These are the default jumper settings of the Development Kit board. Make sure these jumpers are set accordingly.
J2	1-3 closed	
J23	2-3 closed	
J188, 197	1-2 closed	<ul style="list-style-type: none">• Jumper settings when using MMUART_1. These are not set by default and must be set manually.• Change these jumper settings if the USB to UART (FTDI) interface is used. See "Appendix 1: Connecting the SmartFusion2 Device to the Host PC Through the USB to UART (FTDI) Interface" on page 20

4. Connect the power supply to the J18 DC jack.

Alternate Setup

This demo can also be run using the USB to UART (FTDI) interface without using the USB-RS232 serial adapter or the RS232 cable. See ["Appendix 1: Connecting the SmartFusion2 Device to the Host PC Through the USB to UART \(FTDI\) Interface"](#) section on page 20 for information on how to connect the Host PC to the SmartFusion2 Development Kit board for serial communication through the FTDI USB interface using the USB A to Mini - B cable.

Board Setup Snapshot

Snapshots of the SmartFusion2 Development Kit board with all the setup made in both types of connections are given in the following appendices:

- ["Appendix 2: Board Setup when Using the USB-RS232 Serial Adapter or RS232 Cable"](#) on page 21
- ["Appendix 3: Board setup through the USB to UART \(FTDI\) interface using the USB A to Mini - B Cable"](#) on page 22

Running the Demo Design

1. Download the demo design from:
http://soc.microsemi.com/download/rsc/?f=sf2_isp_using_uart_interface_demo_11p4_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.

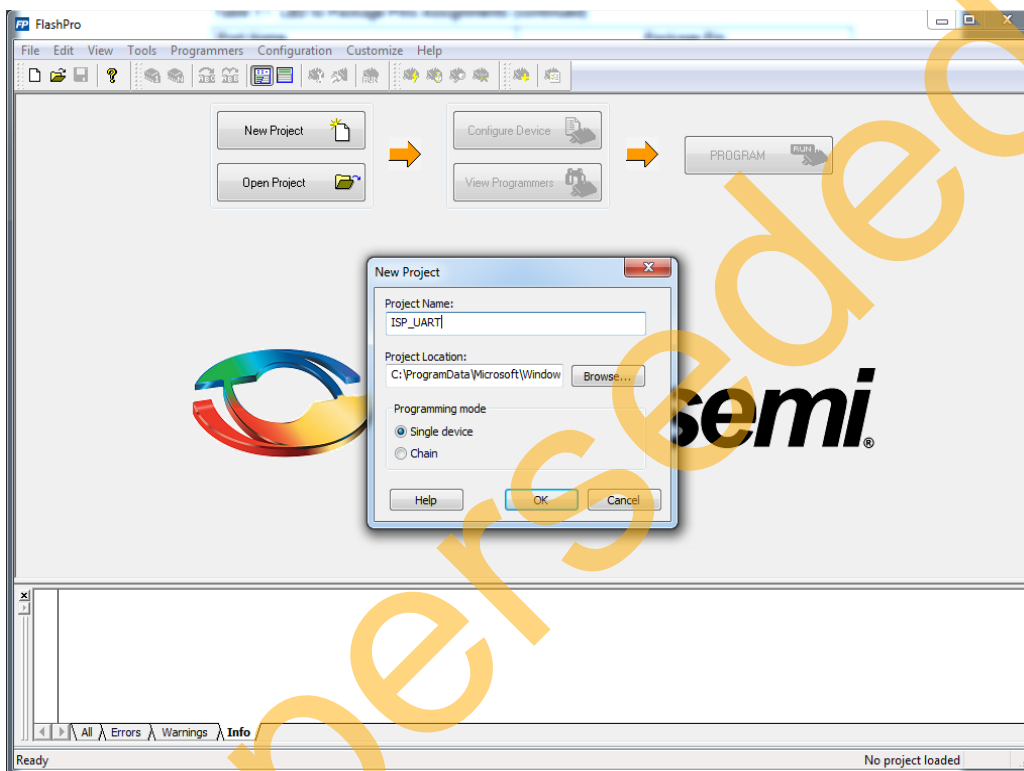


Figure 5 • FlashPro New Project

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.

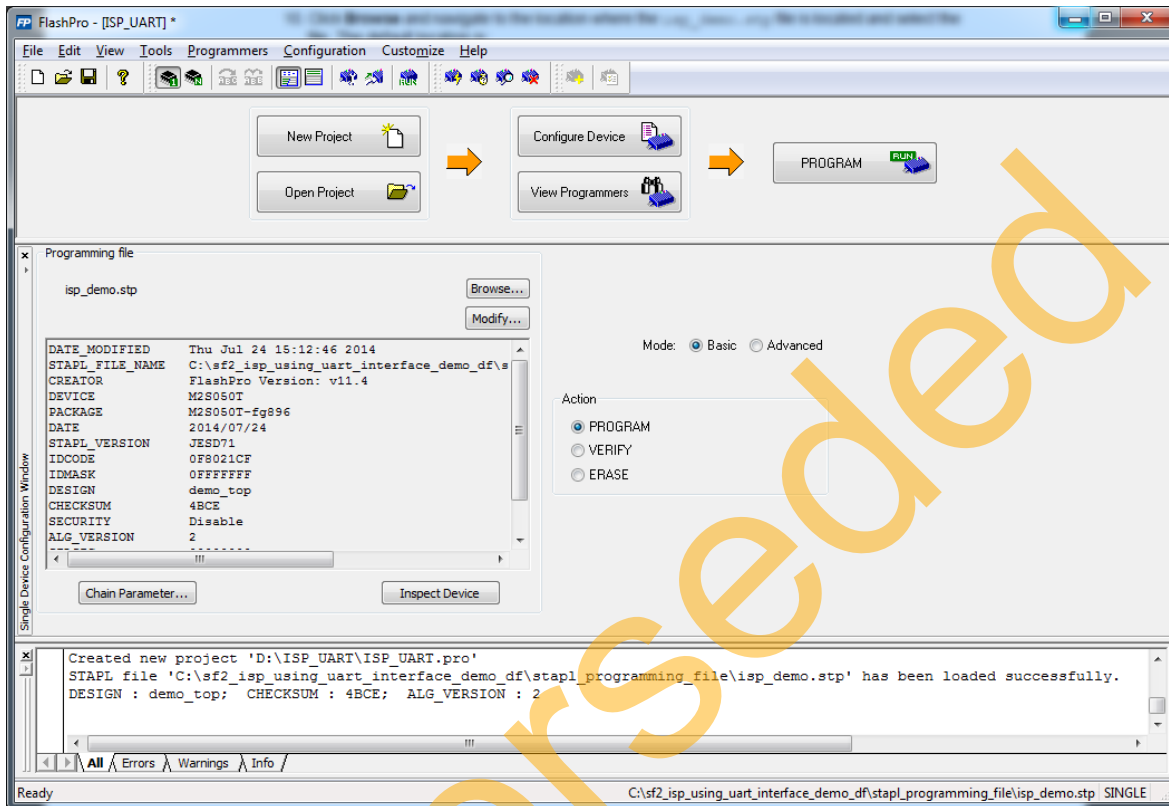


Figure 6 • FlashPro project configured

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 5 to 8 blinking in the board indicates that the SmartFusion2 Device fabric is preprogrammed successfully.

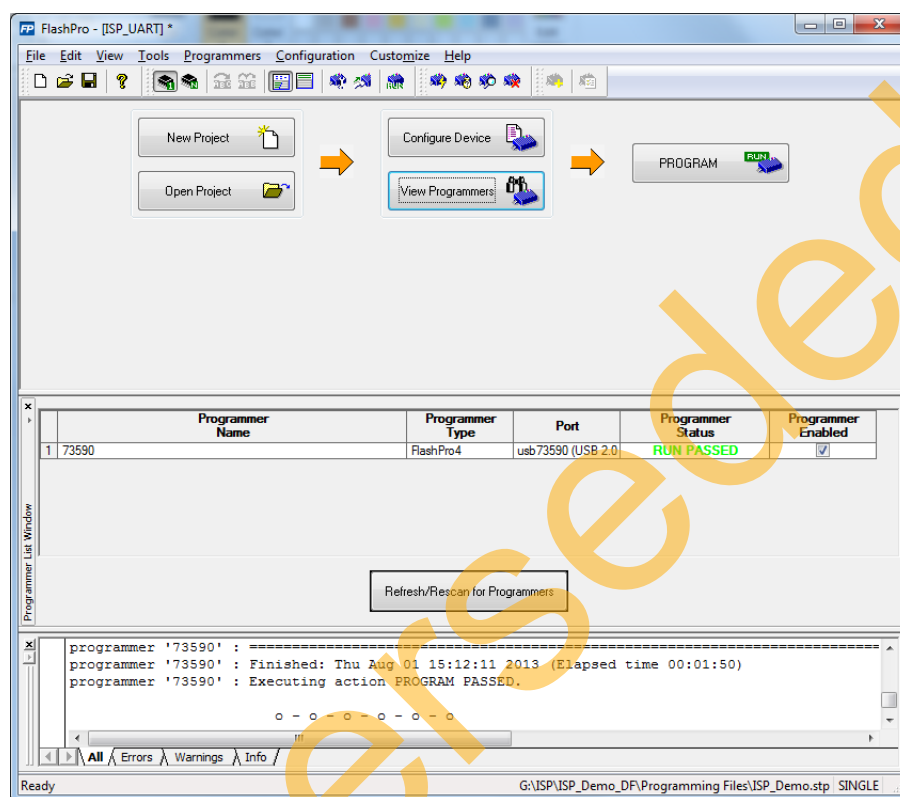


Figure 7 • FlashPro program passed

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

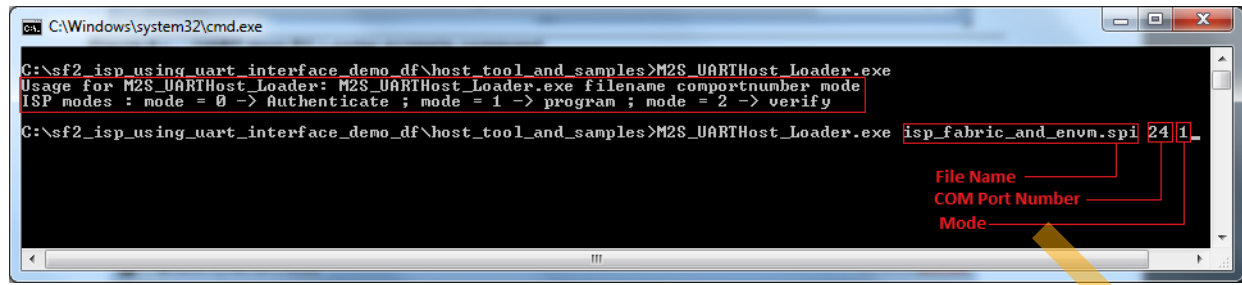
Example command

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

Figure 8 shows the UART Host PC Loader example command.



```

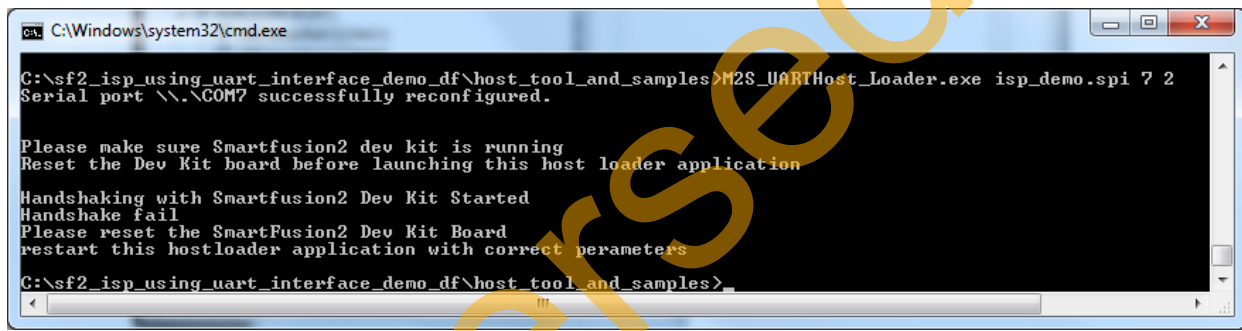
C:\Windows\system32\cmd.exe
C:\sf2_isp_using_uart_interface_demo_df\host_tool_and_samples>M2S_UARTHost_Loader.exe
Usage for M2S_UARTHost_Loader: M2S_UARTHost_Loader.exe filename comportnumber mode
ISP modes : mode = 0 -> Authenticate ; mode = 1 -> program ; mode = 2 -> verify
C:\sf2_isp_using_uart_interface_demo_df\host_tool_and_samples>M2S_UARTHost_Loader.exe isp_fabric_and_envm.spi 24 1
  
```

Figure 8 • UART Host PC Loader example command

Resetting the board

If the UART Host PC Loader is not connected to the SmartFusion2 Development Kit board, press the switch, **SW9** to reset the board.

Figure 9 shows an example message that instructs to reset the board.



```

C:\Windows\system32\cmd.exe
C:\sf2_isp_using_uart_interface_demo_df\host_tool_and_samples>M2S_UARTHost_Loader.exe isp_demo.spi 7 2
Serial port \\.\COM7 successfully reconfigured.

Please make sure Smartfusion2 dev kit is running
Reset the Dev Kit board before launching this host loader application

Handshaking with Smartfusion2 Dev Kit Started
Handshake fail
Please reset the SmartFusion2 Dev Kit Board
restart this hostloader application with correct parameters
C:\sf2_isp_using_uart_interface_demo_df\host_tool_and_samples>
  
```

Figure 9 • UART Host PC Loader reset

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. Figure 10 shows the operation success message.

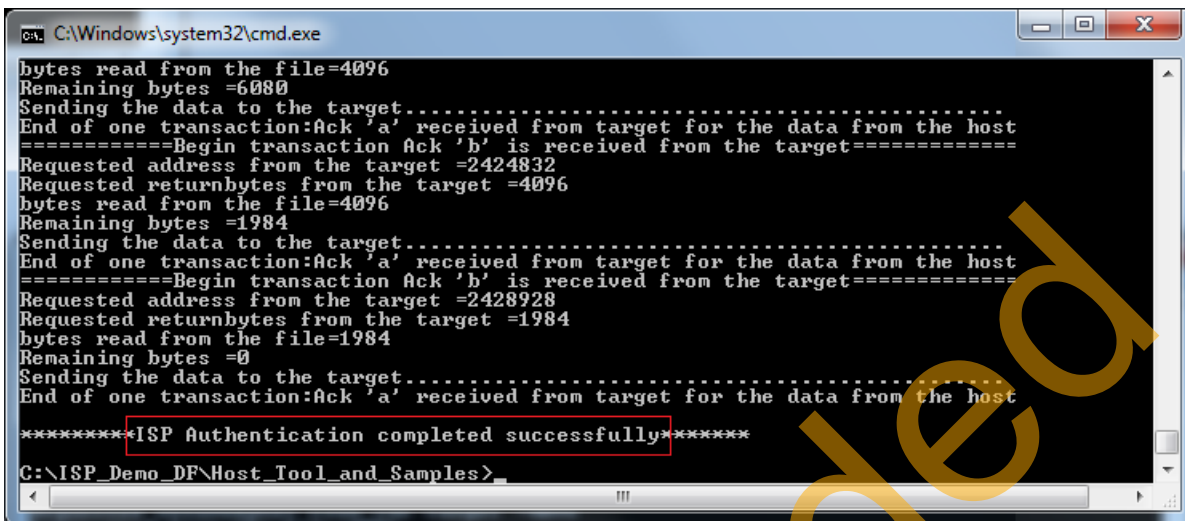


Figure 10 • ISP Authentication Status

Press the switch, **SW9** to reset the SmartFusion2 Development Kit and try other ISP operation modes.

Verify Operation Mode

To verify the device FPGA fabric and eNVM contents, type the command:

```
M2S_UARHost_Loader.exe isp_demo.spi 24 2
```

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

Figure 11 shows a successful verification message.

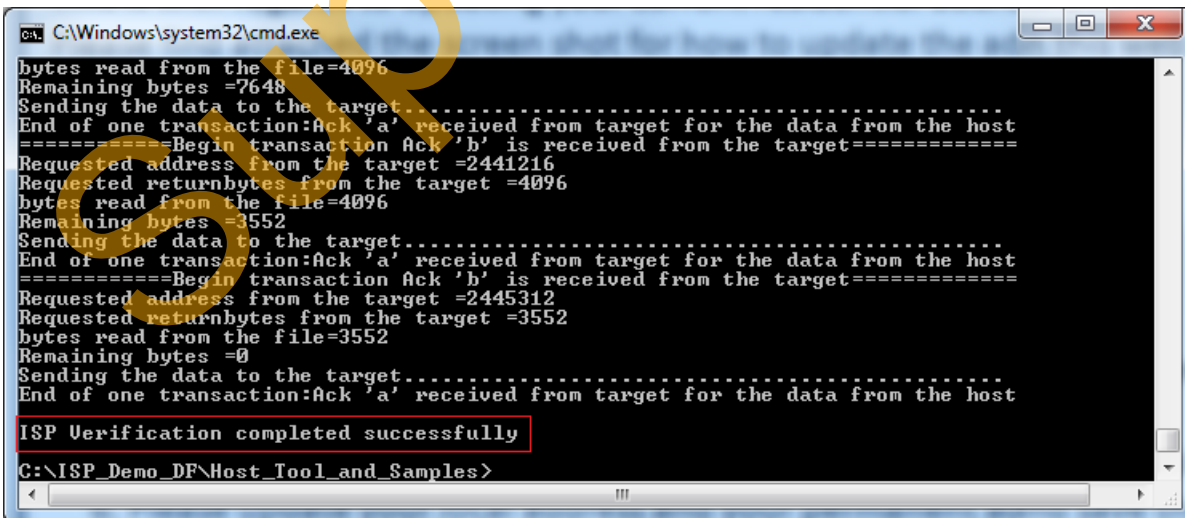


Figure 11 • ISP Verification Status

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.

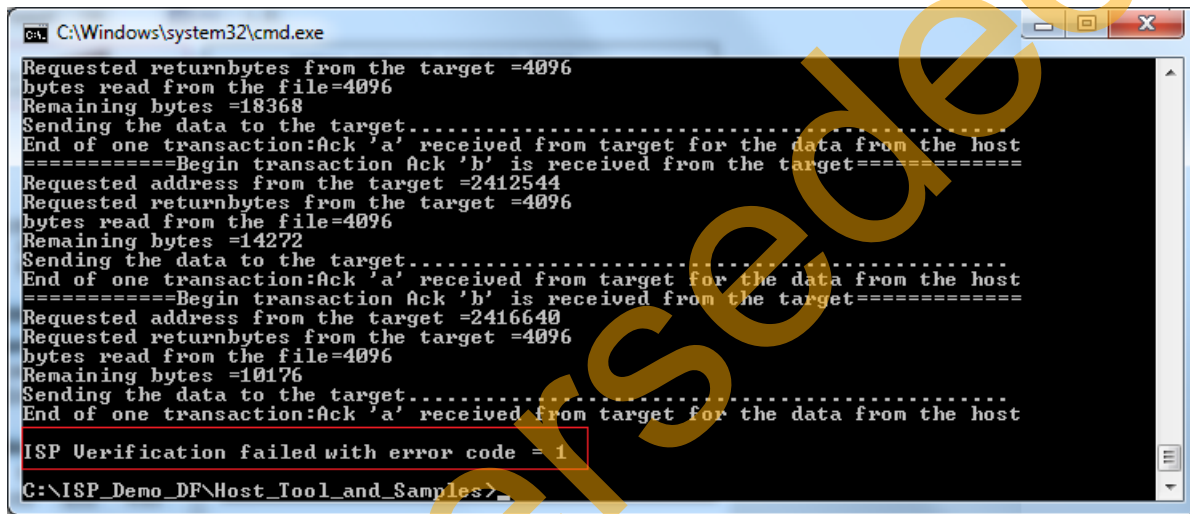
If the verification fails, the command prompt displays an error message with an error code. Figure 12 shows an example error message. For more information on error codes, see "Appendix 5: Error Codes" on page 24.

The programming files are at:

`<download_folder>\sf2_isp_using_uart_interface_demo_dfhost_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 **SW9** to reset the SmartFusion2 Development Kit to try other ISP operation modes from CMD prompt window.



```

C:\Windows\system32\cmd.exe
Requested returnbytes from the target =4096
bytes read from the file=4096
Remaining bytes =18368
Sending the data to the target.....
End of one transaction:Ack 'a' received from target for the data from the host
=====Begin transaction Ack 'b' is received from the target=====
Requested address from the target =2412544
Requested returnbytes from the target =4096
bytes read from the file=4096
Remaining bytes =14272
Sending the data to the target.....
End of one transaction:Ack 'a' received from target for the data from the host
=====Begin transaction Ack 'b' is received from the target=====
Requested address from the target =2416640
Requested returnbytes from the target =4096
bytes read from the file=4096
Remaining bytes =10176
Sending the data to the target.....
End of one transaction:Ack 'a' received from target for the data from the host
ISP Verification failed with error code = 1
C:\ISP_Demo_DF\Host_Tool_and_Samples>
  
```

Figure 12 • ISP verification failure error message

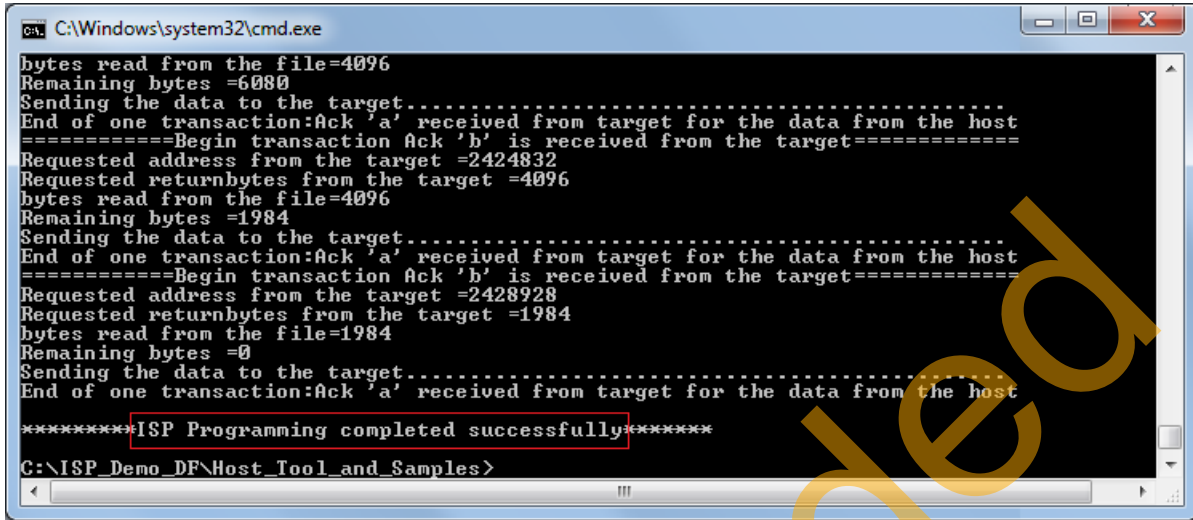
Program Operation Mode

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.
Figure 13 shows a successful ISP programming result.



```
C:\Windows\system32\cmd.exe
bytes read from the file=4096
Remaining bytes =6080
Sending the data to the target.....
End of one transaction:Ack 'a' received from target for the data from the host
=====Begin transaction Ack 'b' is received from the target=====
Requested address from the target =2424832
Requested returnbytes from the target =4096
bytes read from the file=4096
Remaining bytes =1984
Sending the data to the target.....
End of one transaction:Ack 'a' received from target for the data from the host
=====Begin transaction Ack 'b' is received from the target=====
Requested address from the target =2428928
Requested returnbytes from the target =1984
bytes read from the file=1984
Remaining bytes =0
Sending the data to the target.....
End of one transaction:Ack 'a' received from target for the data from the host
*****ISP Programming completed successfully*****
C:\ISP_Demo_DF\Host_Tool_and_Samples>
```

Figure 13 • ISP Program Status

Press **SW9** to reset the SmartFusion2 Development Kit or Power Cycle the SmartFusion2 Development Kit.

Checking if the fabric is programmed successfully

LEDs 1 to 4 blinking in the board indicates that the fabric is programmed successfully.

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
- Tera Term

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. Figure 14 shows an operation success message for eNVM programming in the PuTTY window.

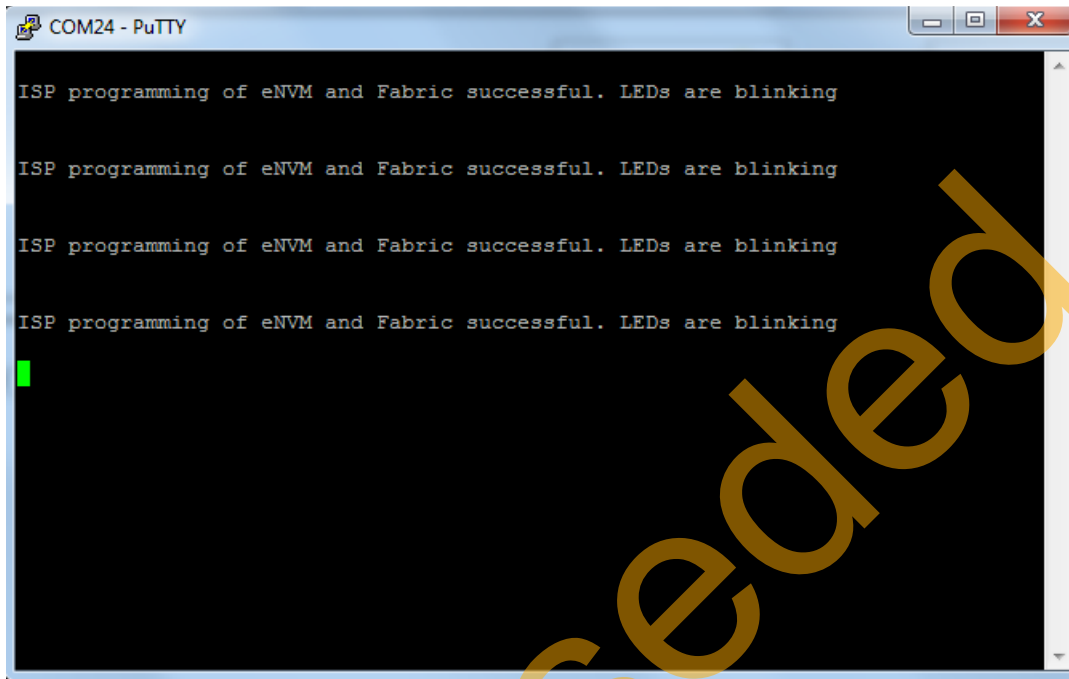


Figure 14 • ISP Program Successful

Programming Results

The result shown in Figure 14 is for the `isp_fabric_and_envm.spi` file. Table 4 shows the possible results for ISP Program operation mode for sample programming files provided in folder `<download_folder>\sf2_isp_using_uart_interface_demo_d\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
<code>isp_envm_only.spi</code>	The serial terminal emulation program shows successful eNVM program message	NA
<code>isp_fabric_only.spi</code>	NA	SmartFusion2 LEDs 1 to 4 blinks
<code>isp_fabric_and_envm.spi</code>	The serial terminal emulation program shows successful eNVM program message	SmartFusion2 LEDs 1 to 4 blinks

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

Appendix 1: Connecting the SmartFusion2 Device to the Host PC Through the USB to UART (FTDI) Interface

The following procedure describes how to connect the SmartFusion2 device to the Host PC through the USB to UART (FTDI) interface using a USB A to Mini - B Cable for serial communication:

1. Connect the host PC to the J24 connector using the USB A to Mini-B cable.
2. Make sure that the *USB to UART bridge drivers* are automatically detected. Of the four COM ports, select the one with **Location** as *on USD Serial Converter D*. [Figure 15](#) shows an example **Device Manager** window that has the **USB Serial Port** and its properties showing the port number and location. COM port number is required to run the demo design, so make a note of it.

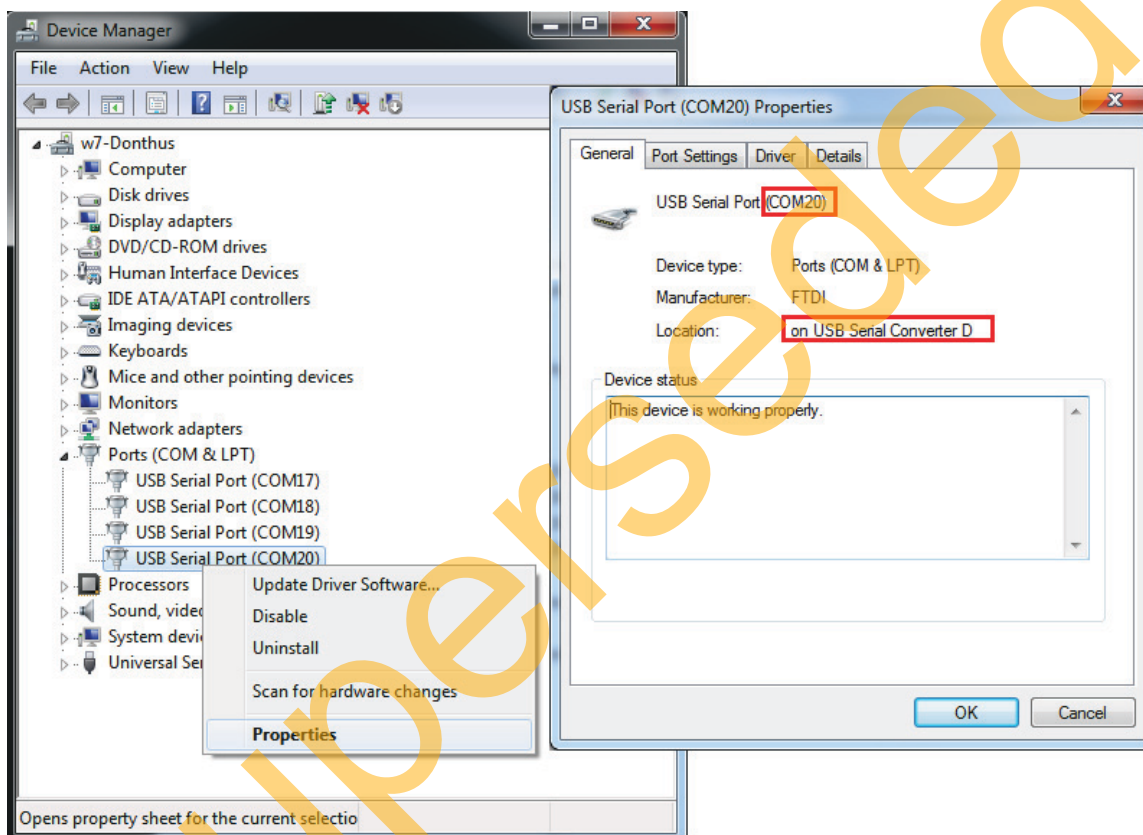


Figure 15 • Device Manager window showing the USB Serial Port

3. Connect the jumpers as follows:
 - Pin 2 of J197 to pin 3 of J129
 - Pin 2 of J188 to pin 3 of J133
 - [Figure 17 on page 22](#) shows these pin connections.
 - See [Figure 18 on page 23](#) for location of the jumpers. These connections are required for connecting the MMUART_1 TXD and RXD signals to the FTDI USB to UART bridge available in SmartFusion2 Development Kit board.

Appendix 2: Board Setup when Using the USB-RS232 Serial Adapter or RS232 Cable

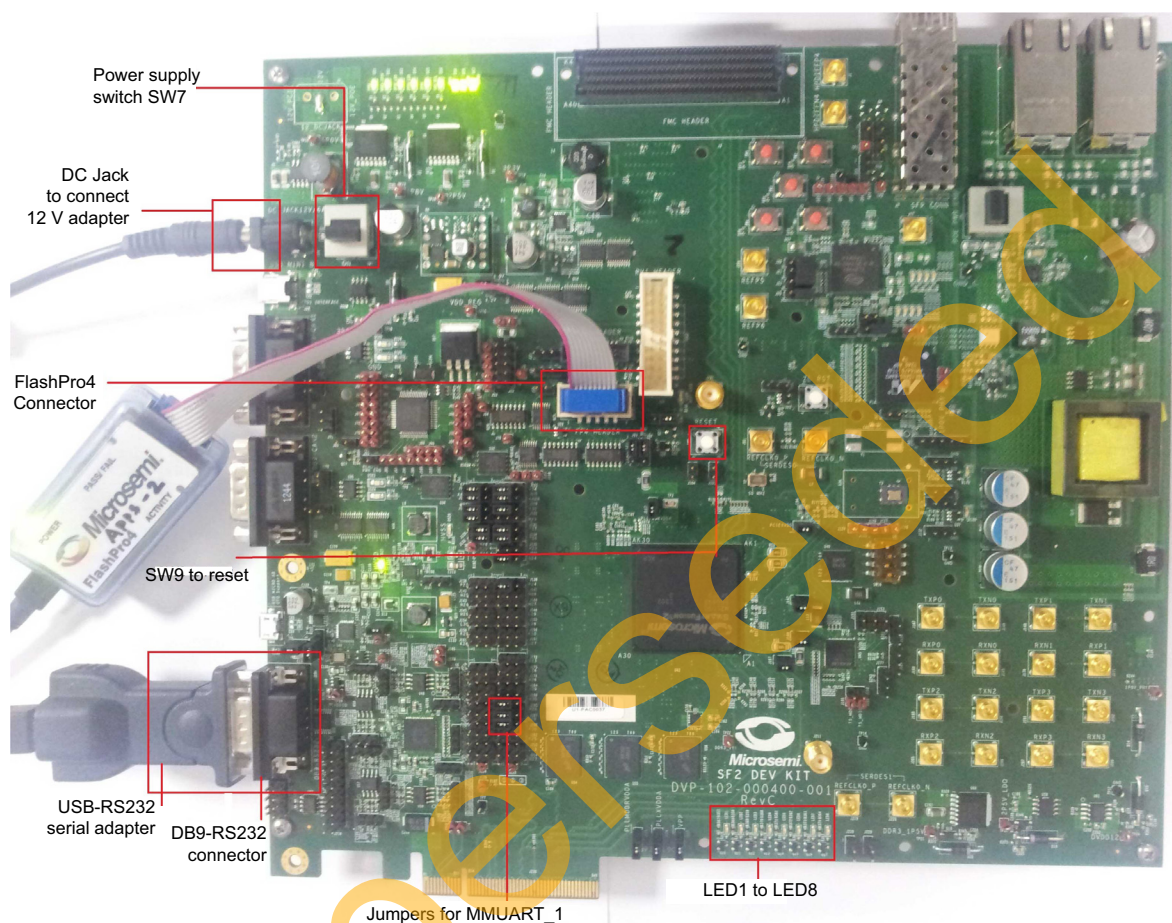


Figure 16 • Board setup when using the USB-RS232 Serial adapter or RS232 cable

Appendix 3: Board setup through the USB to UART (FTDI) interface using the USB A to Mini - B Cable

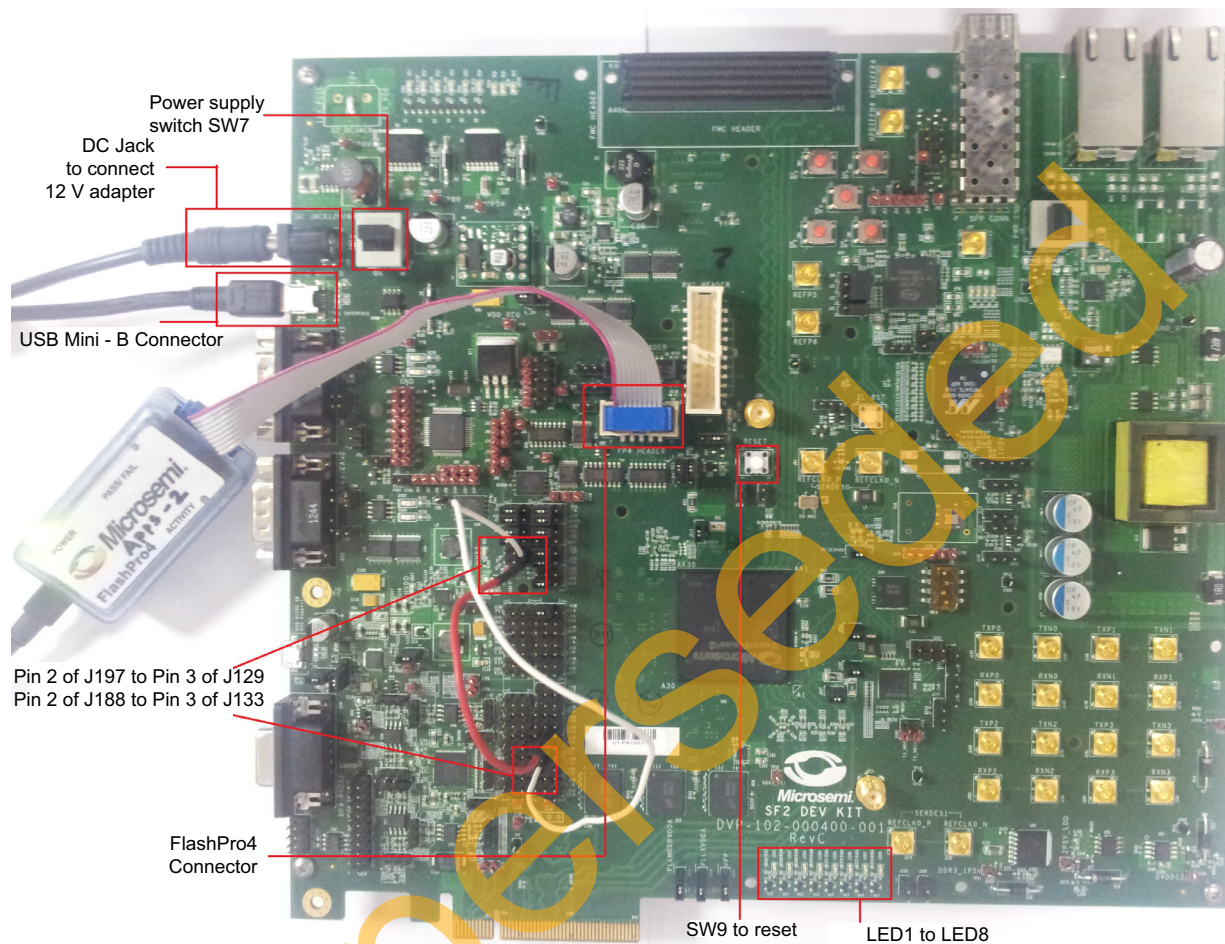


Figure 17 • Board setup when using the USB to UART (FTDI) interface using the USB A to Mini - B Cable

Appendix 4: Jumper Locations

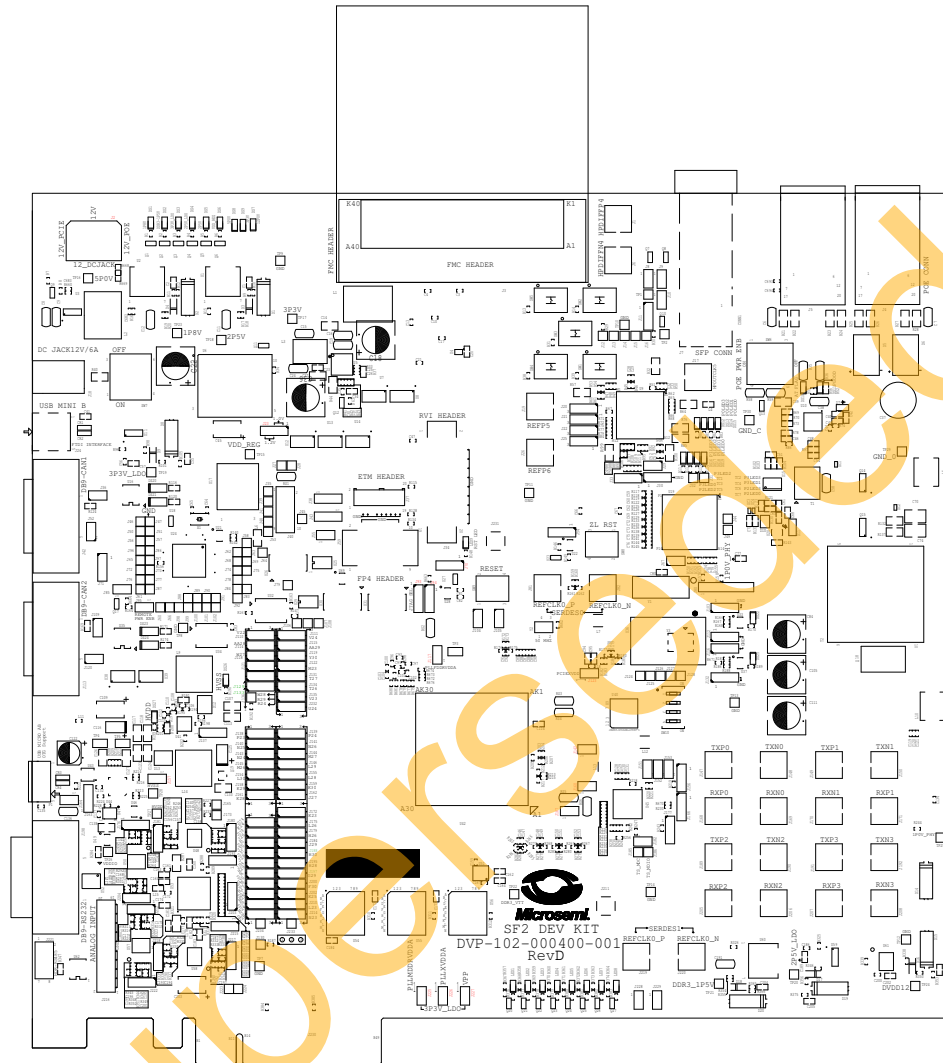


Figure 18 • SmartFusion2 Development Kit Silkscreen Top View

Figure 18 shows the jumper locations in Development Kit board.

Notes:

- Jumpers highlighted in red are set by default.
- Jumpers highlighted in green must be set manually.
- The location of the jumpers in Figure 18 are searchable.

Appendix 5: Error Codes

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.

Appendix 6: Generating .spi Programming File using Libero

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..

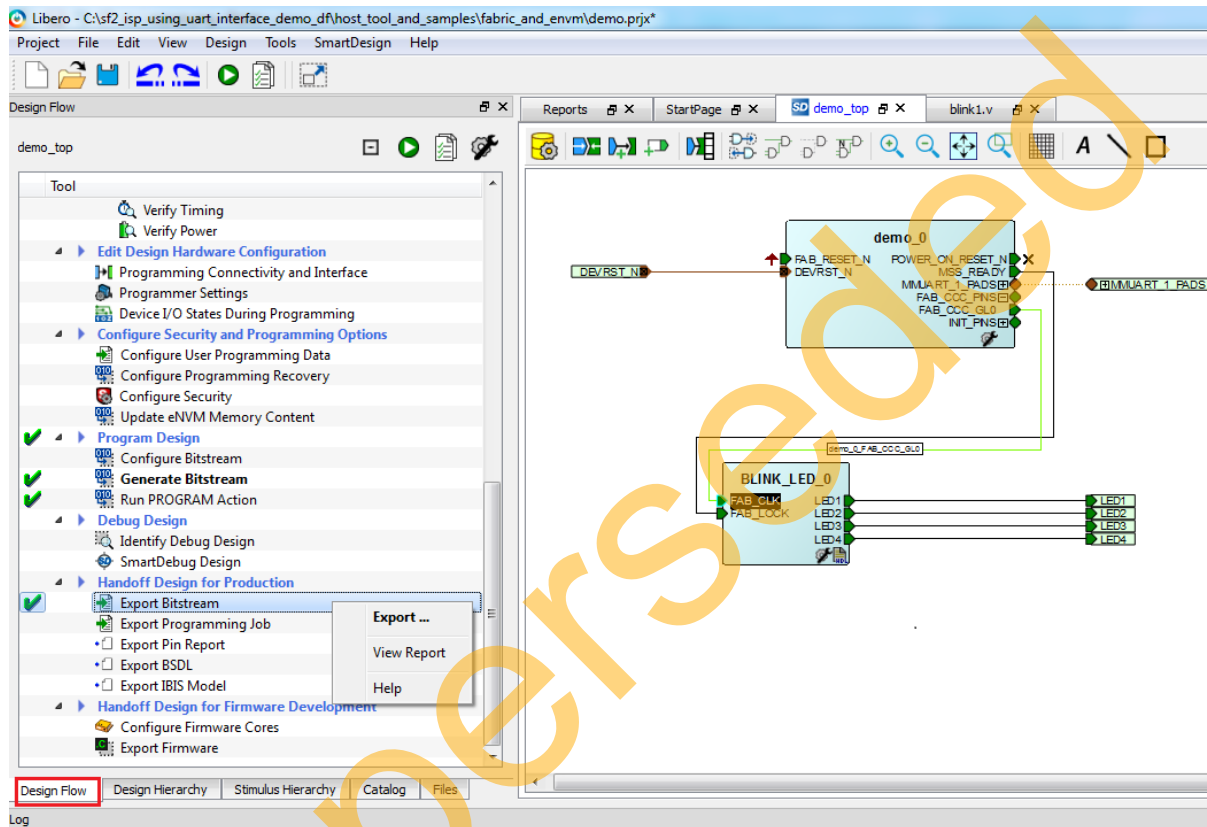


Figure 19 • Configuring Export Bitstream

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

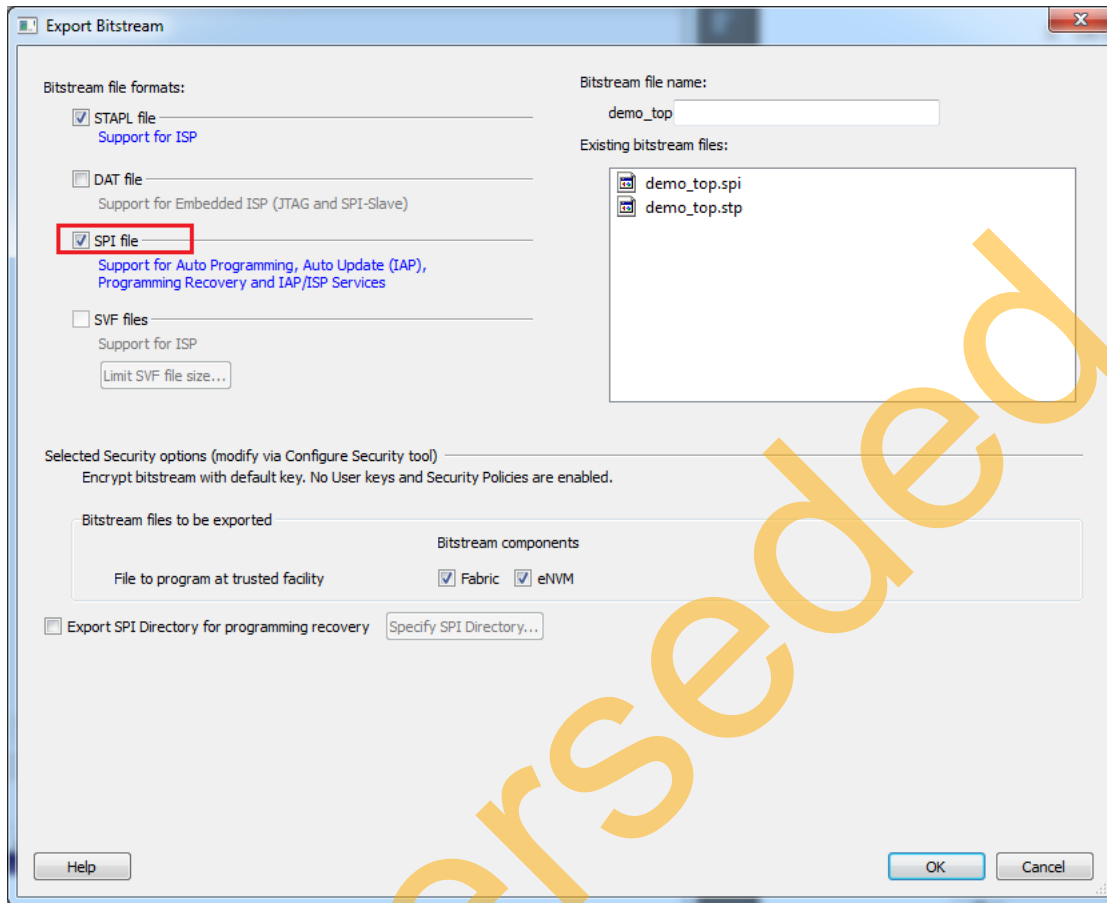


Figure 20 • Export Programming File Options window

4. Click **OK**.
5. Double-click **Export Bitstream** under **Handoff Design for Production** in the **Design Flow** tab to generate the `.spi` file (Figure 19 on page 25). Figure 21 shows the `.spi` file location in **Messages** tab.

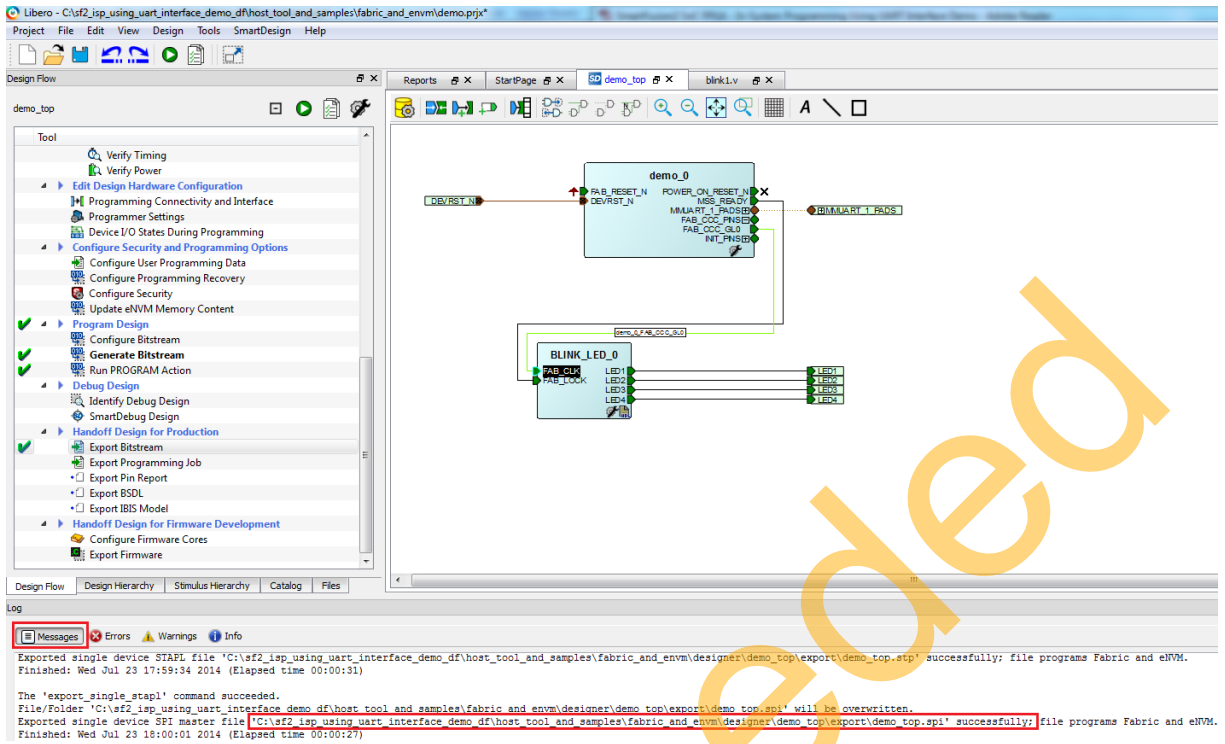


Figure 21 • .SPI File Location

Appendix 7: Hardware Project Implementation Settings

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

Configuring the I/Os for Flash*Freeze Mode

The Libero demo design configures M3_CLK to operate at 50 MHz 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 Development 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 Figure 22. The user has to commit and check the settings from the File menu after configuring the ports.

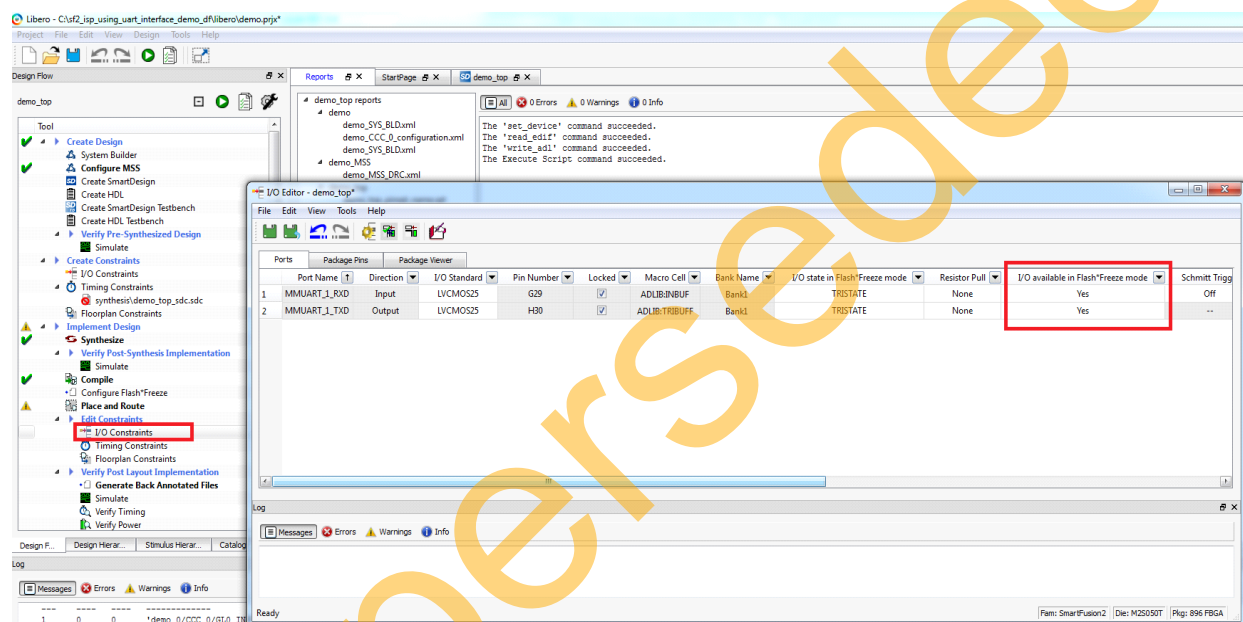


Figure 22 • Configuring MMUART_1 Ports to be Available During F*F

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 Figure 23. A higher MSS clock frequency is required in F*F mode to meet the MMUART baud rate requirements.

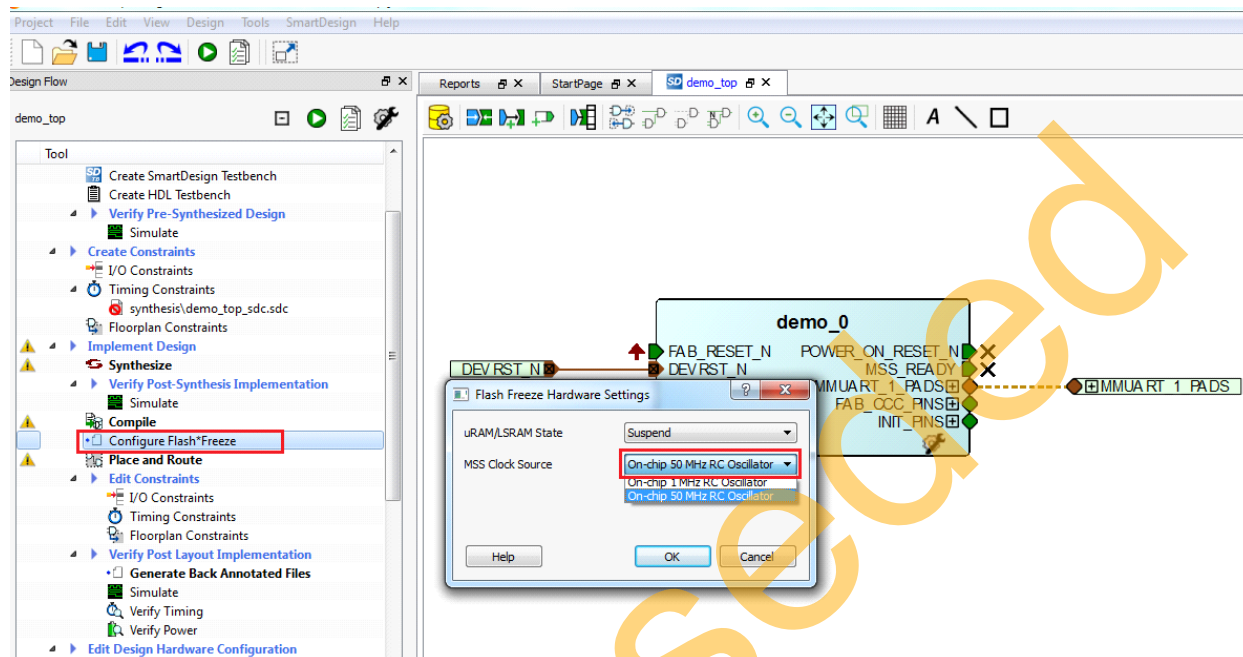


Figure 23 • Flash*Freeze Hardware Settings Dialog

SoftConsole Project Generation

The firmware and SoftConsole project workspace can be generated by checking the Create Project and selecting a Software IDE option in Libero project as shown in Figure 24.

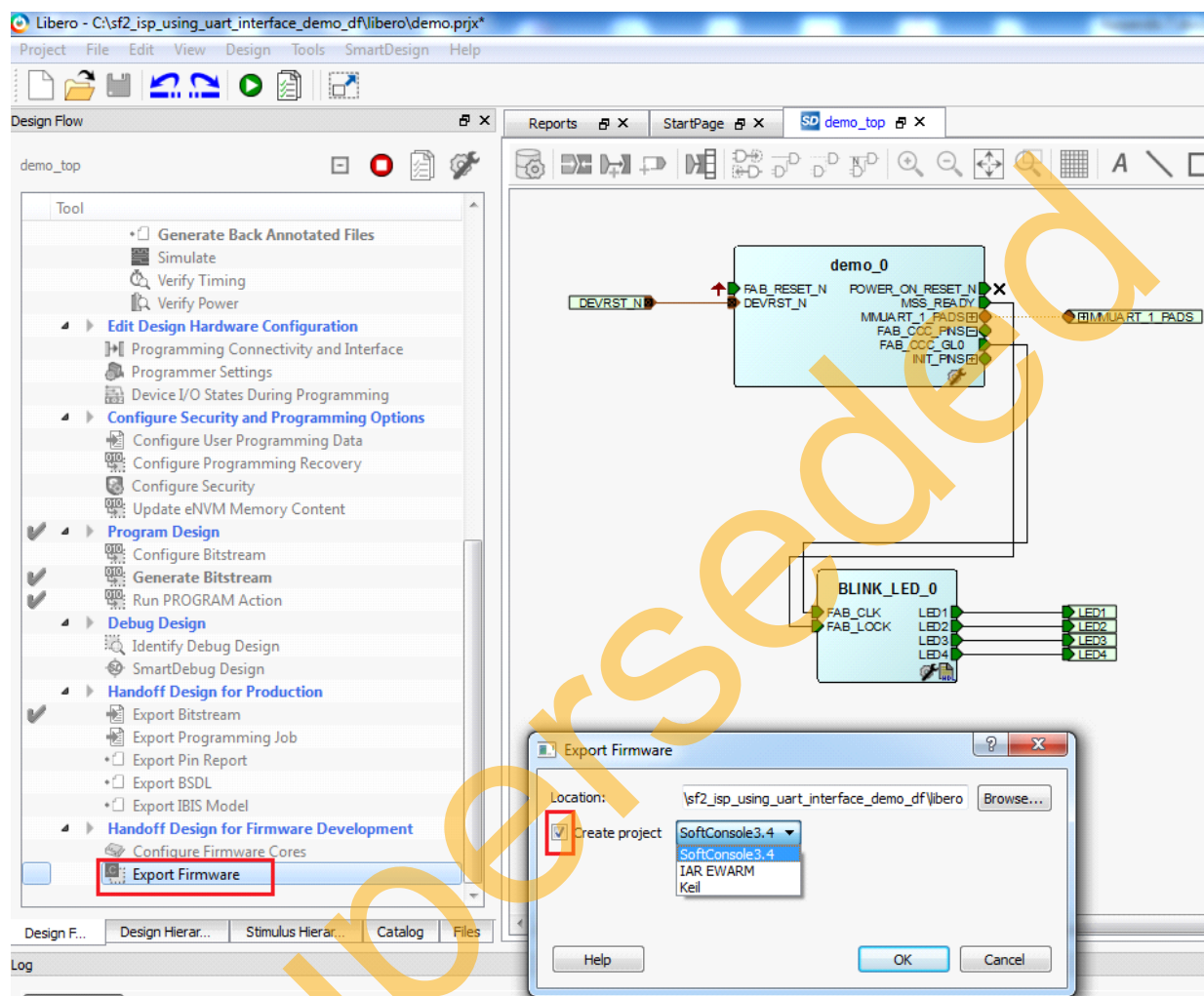


Figure 24 • Export Firmware Options

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 Figure 24.

For software modifications, open the **Softconsole Project** workspace (located at <download_folder>\sf2_isp_using_uart_interface_demo_d\libero\SoftConsole\demo_MSS_CM3) using SoftConsole IDE v3.4 SP1. [Figure 25](#) shows **SoftConsole Project** workspace.

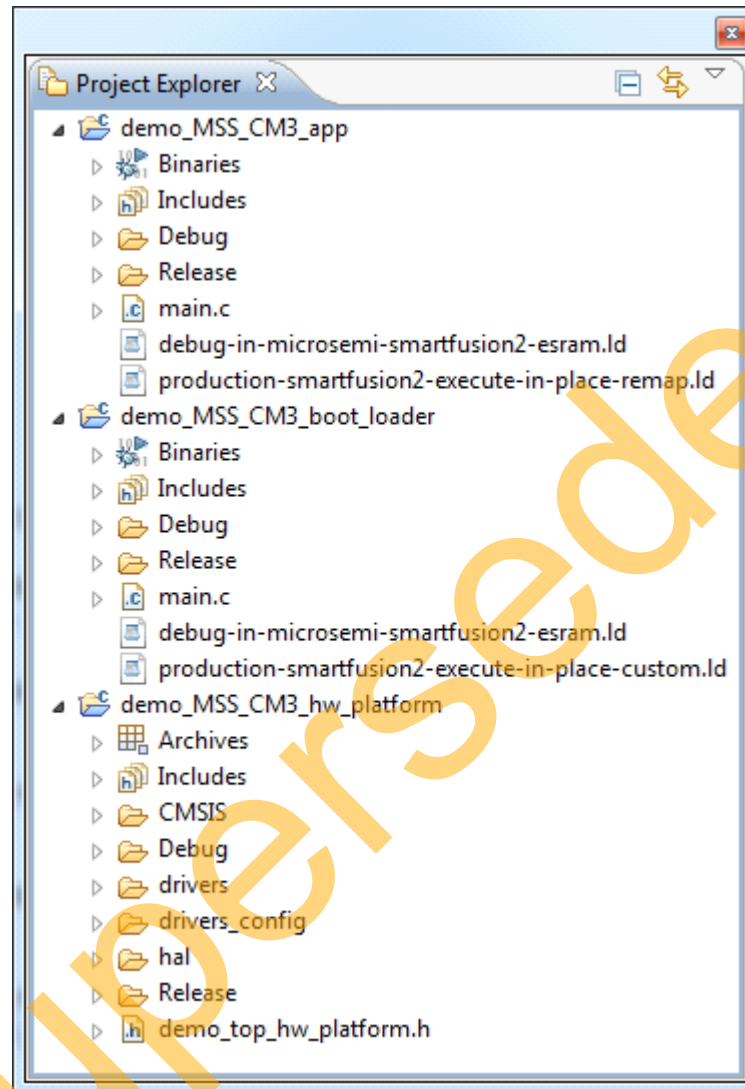


Figure 25 • SoftConsole Project Workspace

The SoftConsole workspace consists of three projects.

- **demo_MSS_CM3_app**
This project receives the bitstream from HostPC through UART interface and invokes the system controller programming services.
- **demo_MSS_CM3_boot_loader**
This project implements the remapping 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. The contents of this folder get over-written every time the firmware is exported as shown in [Figure 24](#).

A – List of Changes

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

Date	Changes	Page
Revision 4 (August 2014)	Updated the document for Libero v11.4 software release (SAR 59742).	NA
Revision 3 (May 2014)	Updated the document for Libero v11.3 software release (SAR 56619).	NA
Revision 2 (December 2013)	Updated " Demo Design Description " section (SAR 53451).	7
Revision 1 (December 2013)	Updated the document for Libero v11.2 software release (SAR 52962).	NA
Revision 0 (July 2013)	Initial release	NA

B – Product Support

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

Customer Service

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

From North America, call 800.262.1060

From the rest of the world, call 650.318.4460

Fax, from anywhere in the world, 408.643.6913

Customer Technical Support Center

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

Technical Support

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

Website

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

Contacting the Customer Technical Support Center

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

Email

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

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

My Cases

Microsemi SoC Products Group customers may submit and track technical cases online by going to [My Cases](#).

Outside the U.S.

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

ITAR Technical Support

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

Superseded

Superseded



Microsemi

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 and 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; Power-over-Ethernet ICs and midspans; as well as custom design capabilities and services. Microsemi is headquartered in Aliso Viejo, Calif. and has approximately 3,400 employees globally. Learn more at www.microsemi.com.

© 2014 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.