SmartFusion2 SoC FPGA In-System Programming Using USB OTG Controller Interface - Libero SoC v11.4

Demo Guide

August 2014

Microsemi®

Revision History

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

Confidentiality Status

This is a non-confidential document.

Table of Contents

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

SmartFusion2 devices are used by:

- 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 USB OTG Controller Interface

Introduction

You can use in-system programming (ISP) to reprogram for design iterations and field upgrades. SmartFusion2 devices support ISP using universal serial bus (USB) on-the-go (OTG) controller interface. This document describes how to program the following using ISP through the USB OTG controller 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, refer to the *SmartFusion2 Programming User Guide*. For information on USB OTG controller, refer to the *SmartFusion2 Microcontroller Subsystem User Guide*.

Requirements and Details

Table 1 • **Reference Design Requirements and Details**

Reference Design Requirements and Details	**Description**
Hardware Requirements	
SmartFusion2 Development Kit • 12 V adapter • FlashPro4 programmer • USB A to Mini-B cable • USB cable with Micro-A end	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_usb_interface_demo_11p4_df*

The demo design files include:

- Sample programming files
- Libero SoC project
- STAPL programming file
- readme.txt

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

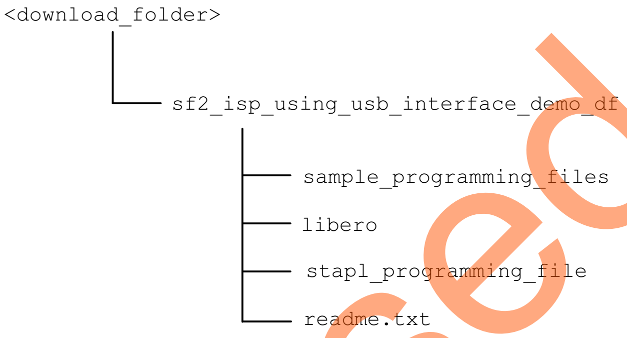


Figure 1 • **Demo Design Files Top Level Structure**

Figure 2 describes the demo. The SmartFusion2 device application configures the following:

- The MMUART_1 peripheral for serial communication with host PC.
- The USB OTG as mass storage class host, which can read or write files from the mass storage device connected to the SmartFusion2 device through USB cable with Micro A end. Refer to the "Appendix 7: Hardware Project Implementation Settings" section on page 31.

The SmartFusion2 device also initializes the system controller to run the ISP service. The SmartFusion2 device detects the connected USB mass storage device and accesses the programming files. The Cortex-M3 processor reads 512 byte blocks of the programming file data from the USB mass storage device and 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. Refer to the "Demo Design Description" section on page 8 for information on the various modes of operation.

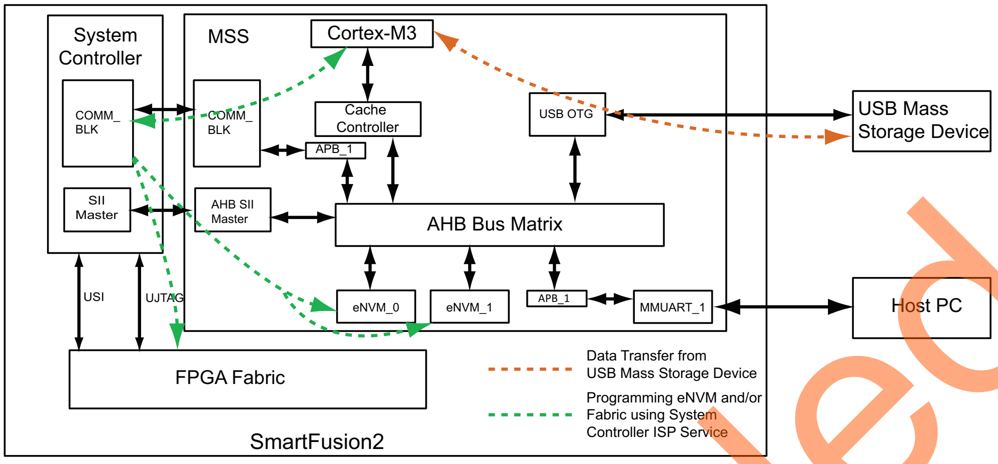


Figure 2 • **Top Level Demo Diagram**

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

- Using the USB-RS232 Serial adapter or the RS232 cable.
  - Refer to the "Running the Demo Design" section on page 12.
- Using the USB to UART (FTDI) interface.
  - Refer to the "Appendix 1: Connecting the SmartFusion2 Device to the Host PC Through the USB to UART (FTDI) Interface" on page 24.

Demo Design Features

This 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 USB mass storage device through the USB OTG controller 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 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 on page 31.

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_usb_interface_demo_df\sample_programming_files* contains the following sample programming files along with Libero design files.

- `envmonly.spi`: Programs only eNVM. The eNVM client has a simple message display program.
- `fabriconly.spi`: Programs only the FPGA fabric. The FPGA fabric has a light-emitting diode (LED) blinking logic.
- `fabenvm.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_usb_interface_demo_df\sample_programming_files\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_usb_interface_demo_df\stapl_programming_file*.

Note: For more information on generating `.spi` programming files refer to the "Appendix 6: Generating .spi Programming File using Libero" on page 29.

ISP Execution Flow

Figure 3 on page 10 describes the ISP flow. The SmartFusion2 Device application initially configures the USB OTG controller in Host mode and MMUART_1 for serial communication. It also initializes the system controller to start the ISP service in the selected operation mode.

On receiving the ISP operation mode and the programming file index, the application starts reading the input source programming file in a 512 byte blocks. The application stores the received data in a temporary buffer and sends the same data to the ISP service. The application requests the next block of 512 byte data until the entire file gets transferred from the USB mass storage device. The SmartFusion2 Device application 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.

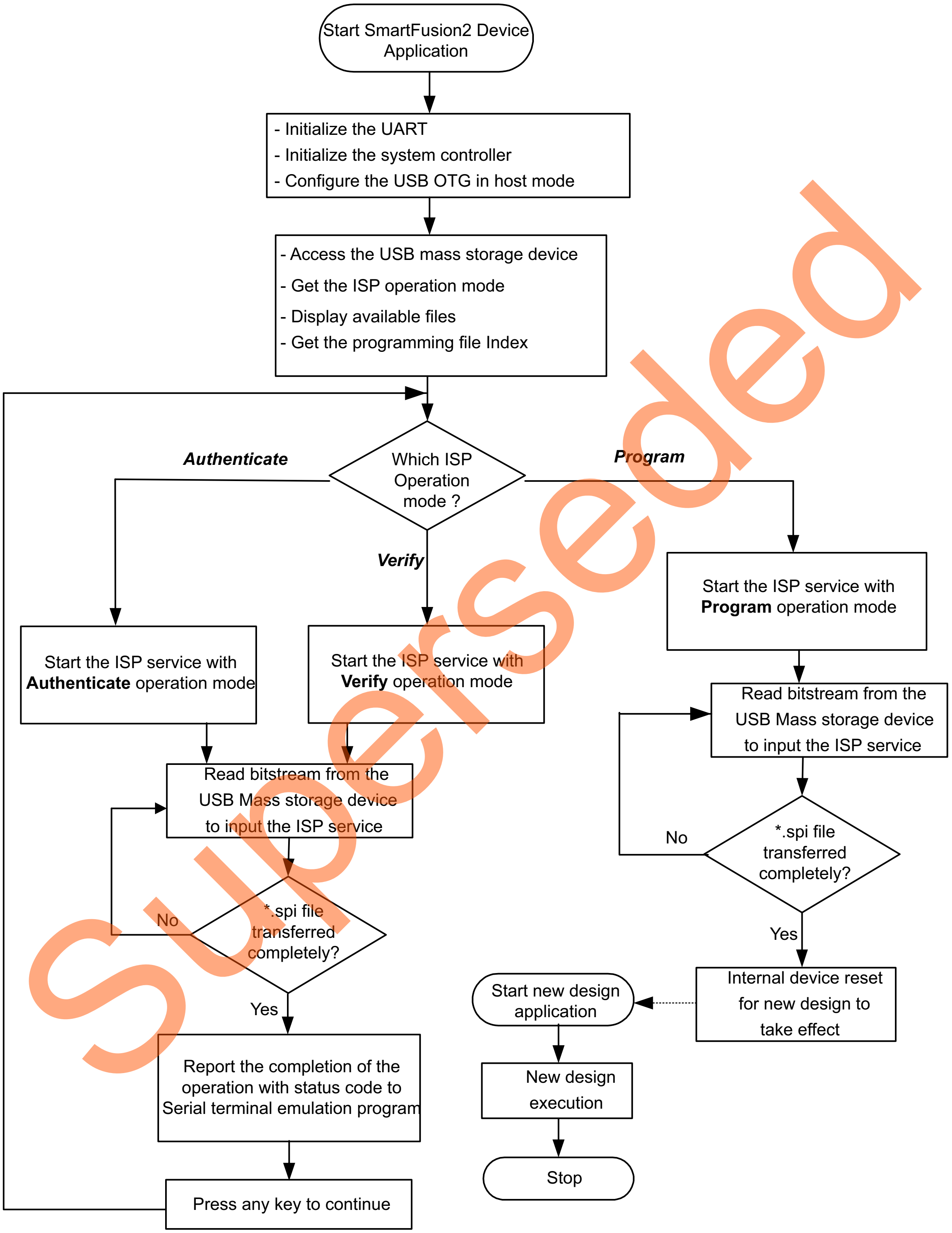

Start SmartFusion2 Device Application
- Initialize the UART
- Initialize the system controller
- Configure the USB OTG in host mode
- Access the USB mass storage device
- Get the ISP operation mode
- Display available files
- Get the programming file Index
Which ISP Operation mode ?
Authenticate
Verify
Program
Start the ISP service with Authenticate operation mode
Start the ISP service with Verify operation mode
Start the ISP service with Program operation mode
Read bitstream from the USB Mass storage device to input the ISP service
*.spi file transferred completely?
No
Yes
Report the completion of the operation with status code to Serial terminal emulation program
Press any key to continue
Read bitstream from the USB Mass storage device to input the ISP service
*.spi file transferred completely?
No
Yes
Internal device reset for new design to take effect
Start new design application
New design execution
Stop

Figure 3 • **Execution Flow of ISP Operation Mode**

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 provided on 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.

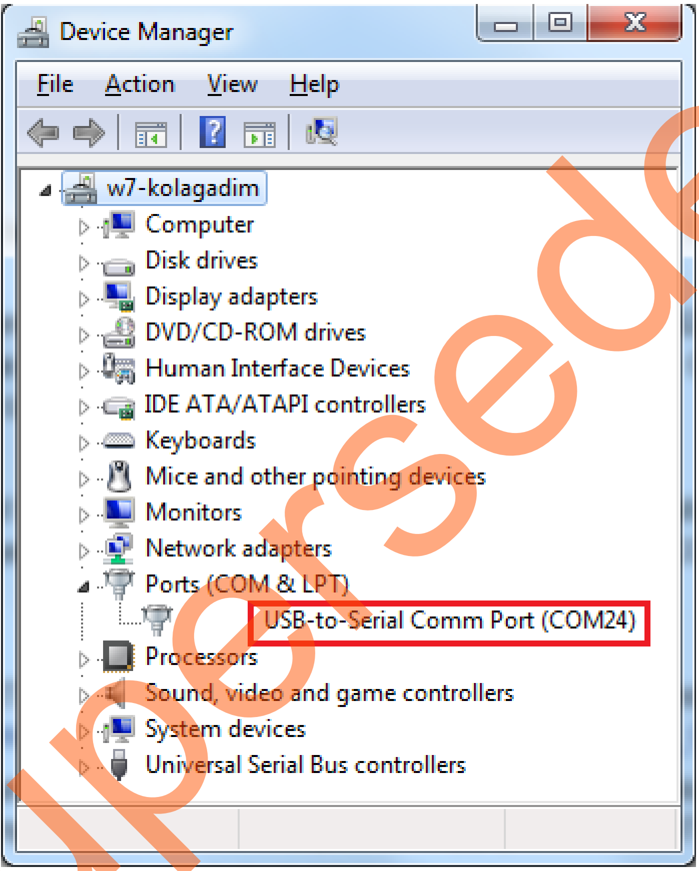


Figure 4 • **USB-to-Serial Communication Port**

3. Connect the jumpers on the SmartFusion2 Development Kit board, as described in Table 2. For information on jumper locations, refer to the "Appendix 4: Jumper Locations" on page 27.
   - **Caution**: Before making the jumper connections, switch off the power supply switch, SW7.

Table 2 • **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	
J139	1-2 closed	Jumper to select USB reset.
J163	1-2 closed	Jumper to select USB OTG mode of operation.
J164	1-2 closed	Jumper to provide the VBUS supply to USB when using in Host mode.
J188, 197	1-2 closed	• 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. Refer to "Appendix 3: Board Setup Through the USB to UART (FTDI) Interface using the USB A to Mini - B Cable".

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. Refer to "Appendix 1: Connecting the SmartFusion2 Device to the Host PC Through the USB to UART (FTDI) Interface" on page 24 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 25
- "Appendix 3: Board Setup Through the USB to UART (FTDI) Interface using the USB A to Mini - B Cable" on page 26

Running the Demo Design

1. Download the demo design from:
   http://soc.microsemi.com/download/rsc/?f=sf2_isp_using_usb_interface_demo_11p4_df.
2. Switch **ON** the SW7 power supply switch.
3. 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

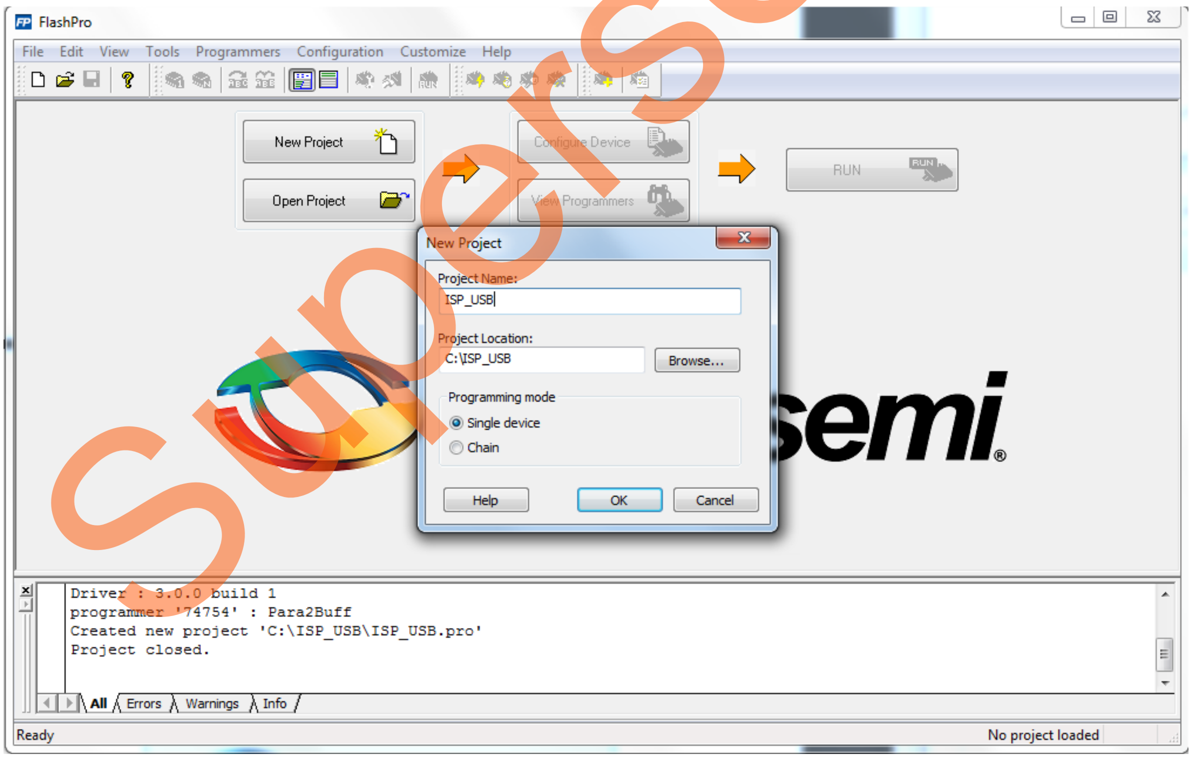
For information on configuring the serial terminal emulation programs, refer to the *Configuring Serial Terminal Emulation Programs Tutorial*.

4. Connect the USB cable Micro A end to the P1 connector of SmartFusion2 Development Kit board and other end to the USB mass storage device.

   The USB OTG controller in Host mode is tested to work with mass storage devices listed below:

   - Sandisc CruzeerBlade - 16Gb/8Gb/4Gb/1GB
   - Kingston Datatraveller - 4Gb/2Gb
   - Kingston DT109B - 8GB
   - Transcend JetFlash - 4Gb

   Make sure to connect preformatted USB Flash drive to the SmartFusion2 device with the sample programming files provided in *<download_folder>\sf2_isp_using_usb_interface_demo_df\sample_programming_files*.
5. Launch the FlashPro software.
6. Click **New Project**.
7. In the **New Project** window, type the project name.



New Project

Project Name: ISP_USB

Project Location: C:\ISP_USB

Programming mode

● Single device

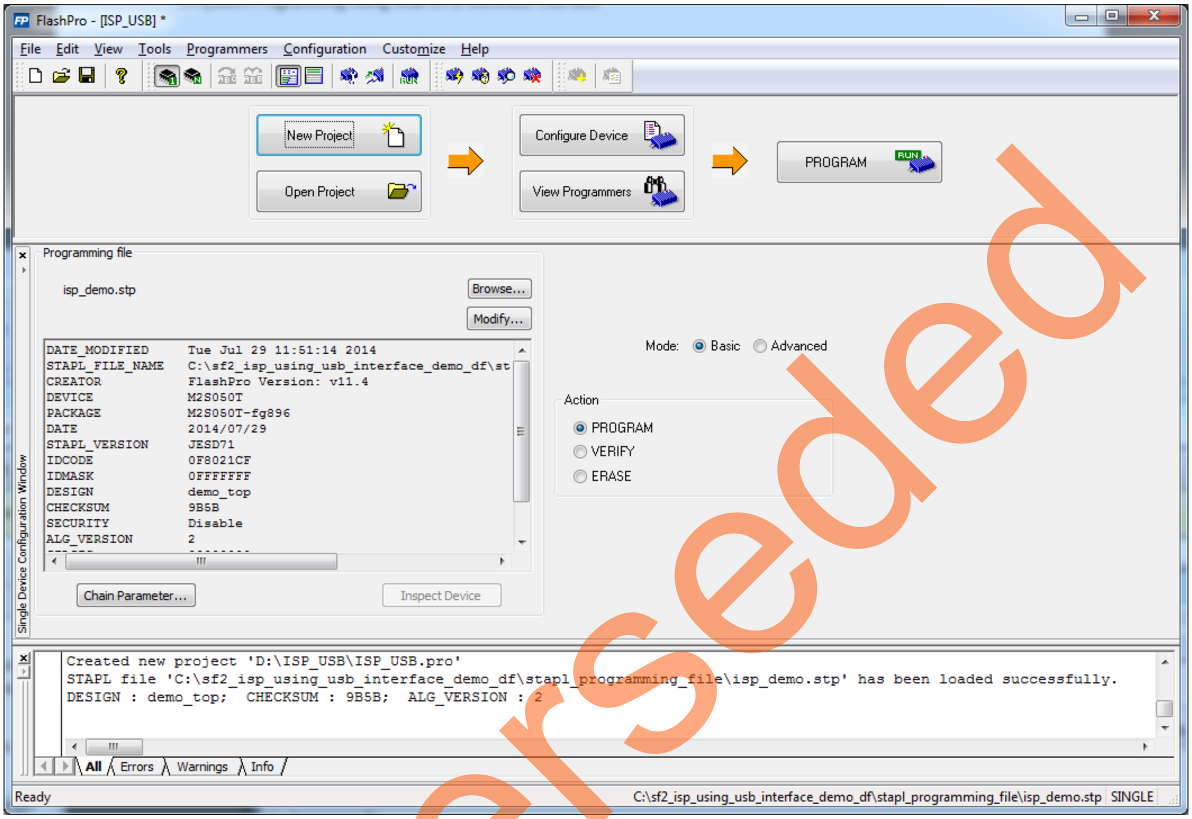
○ Chain

```
Driver : 3.0.0 build 1
programmer '74754' : Para2Buff
Created new project 'C:\ISP_USB\ISP_USB.pro'
Project closed.
```

Figure 5 • **FlashPro New Project**

8. Click **Browse** and navigate to the location where you want to save the project.
9. Select **Single device** as the **Programming mode**.
10. Click **OK** to save the project.
11. Click **Configure Device** on the FlashPro GUI.

12. 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_usb_interface_demo_df\stapl_programming_file*. The required programming file is selected and is ready to be programmed in the device.



Programming file

isp_demo.stp

```
DATE_MODIFIED      Tue Jul 29 11:51:14 2014
STAPL_FILE_NAME    C:\sf2_isp_using_usb_interface_demo_df\st
CREATOR            FlashPro Version: v11.4
DEVICE             M2S050T
PACKAGE            M2S050T-fg896
DATE               2014/07/29
STAPL_VERSION      JESD71
IDCODE             0F8021CF
IDMASK             0FFFFFFF
DESIGN             demo_top
CHECKSUM           9B5B
SECURITY           Disable
ALG_VERSION        2
```

Mode: ● Basic ○ Advanced

Action

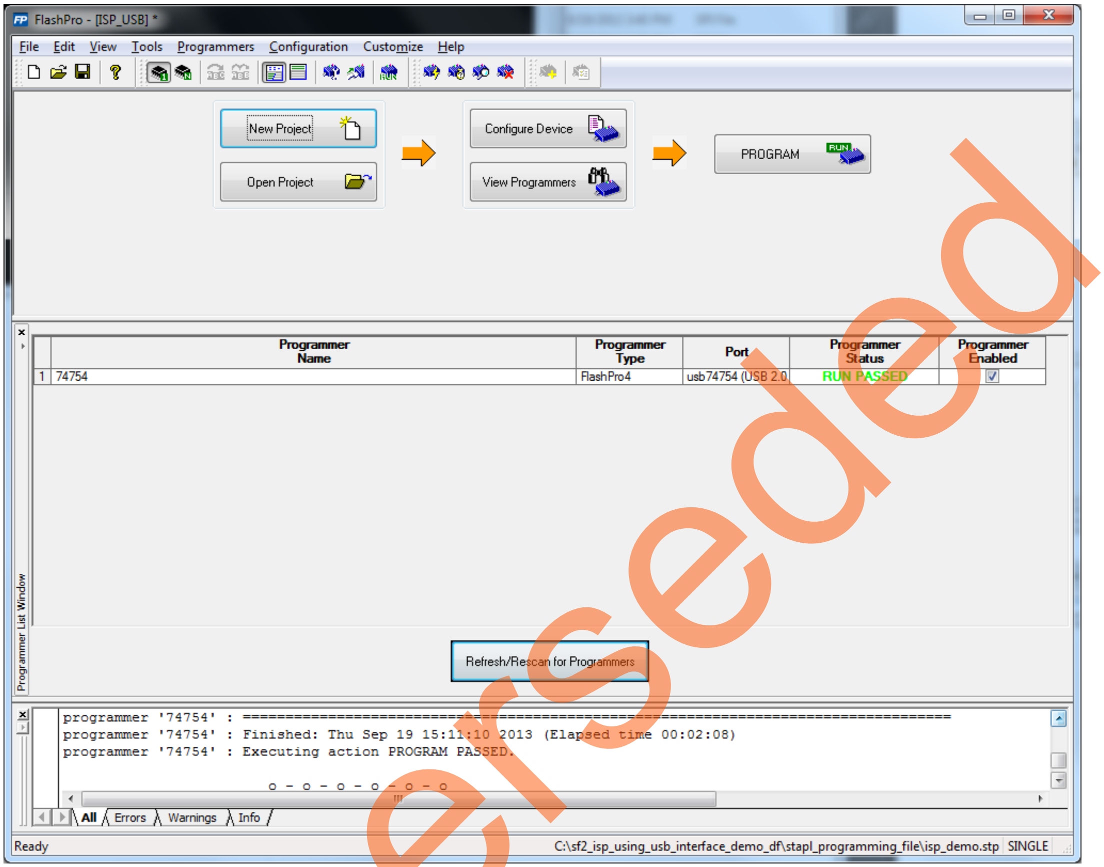
● PROGRAM
○ VERIFY
○ ERASE

```
Created new project 'D:\ISP_USB\ISP_USB.pro'
STAPL file 'C:\sf2_isp_using_usb_interface_demo_df\stapl_programming_file\isp_demo.stp' has been loaded successfully.
DESIGN : demo_top;  CHECKSUM : 9B5B;  ALG_VERSION : 2
```

Ready   C:\sf2_isp_using_usb_interface_demo_df\stapl_programming_file\isp_demo.stp   SINGLE

Figure 6 • **FlashPro Project Configured**

13. 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.



	Programmer Name	Programmer Type	Port	Programmer Status	Programmer Enabled
1	74754	FlashPro4	usb74754 (USB 2.0)	RUN PASSED	☑

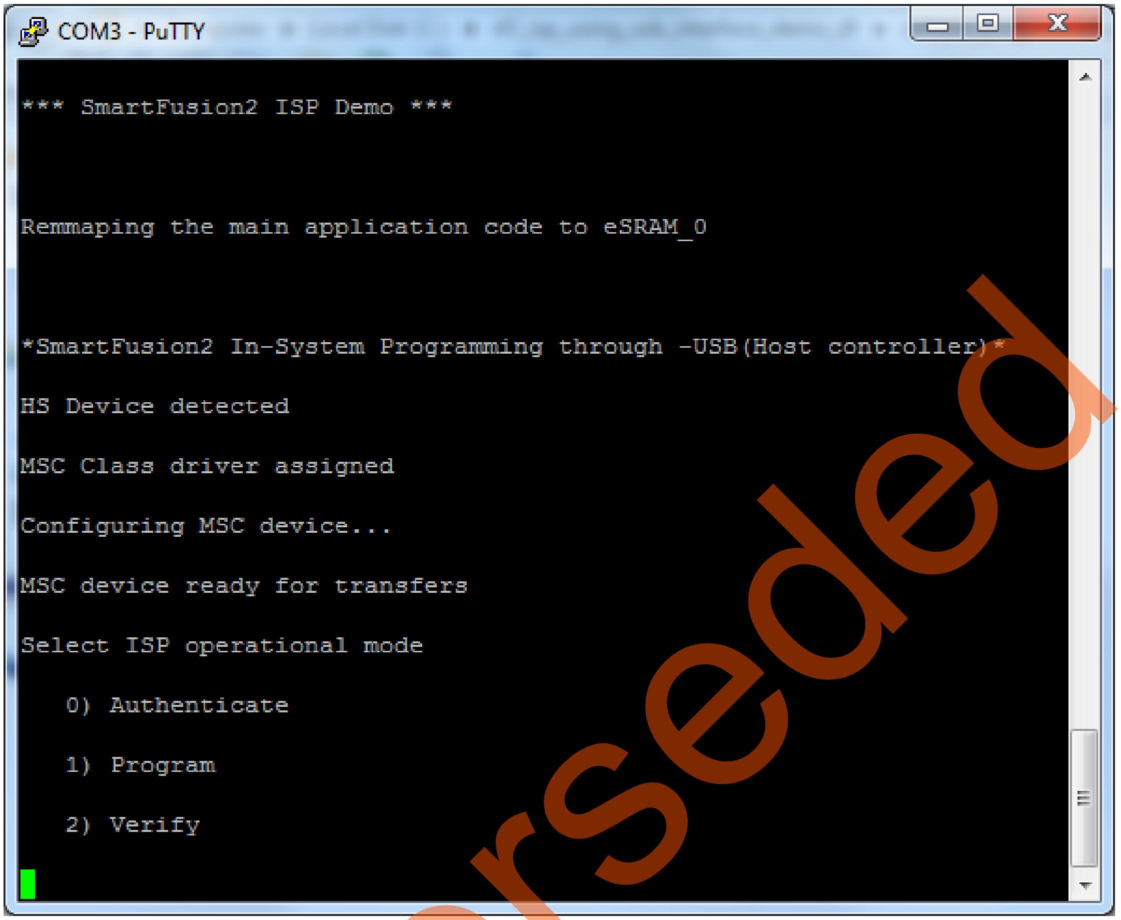
```
programmer '74754' : ===================================================================================
programmer '74754' : Finished: Thu Sep 19 15:11:10 2013 (Elapsed time 00:02:08)
programmer '74754' : Executing action PROGRAM PASSED.

                    o - o - o - o - o - o
```

Ready C:\sf2_isp_using_usb_interface_demo_df\stapl_programming_file\isp_demo.stp SINGLE

Figure 7 • **FlashPro Program Passed**

– LEDs 5 to 8 blinking in the board indicates that the SmartFusion2 device fabric is preprogrammed successfully.

On programming the SmartFusion2 device successfully using FlashPro, the serial terminal emulation program shows the initialization messages and ISP operation modes as shown in Figure 8 on page 16.

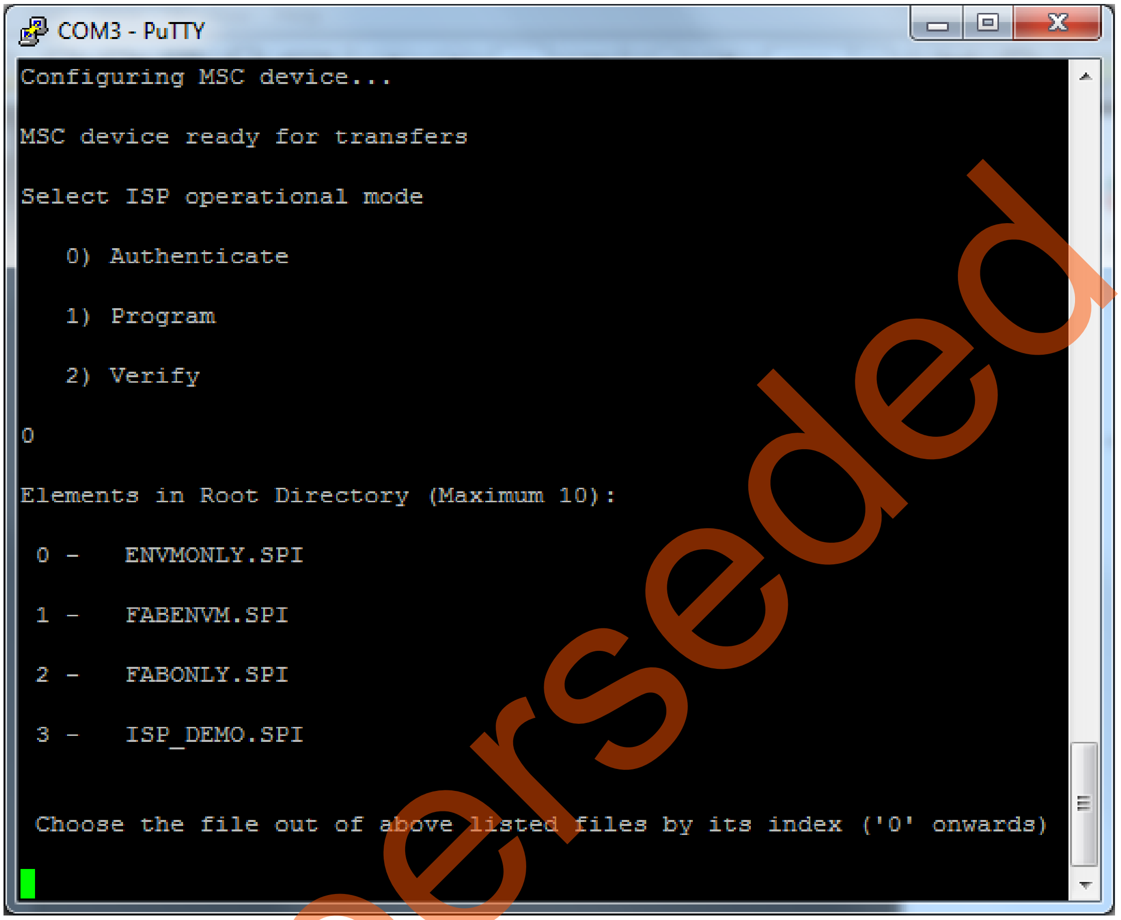


```
*** SmartFusion2 ISP Demo ***


Remmaping the main application code to eSRAM_0


*SmartFusion2 In-System Programming through -USB(Host controller)*

HS Device detected

MSC Class driver assigned

Configuring MSC device...

MSC device ready for transfers

Select ISP operational mode

   0) Authenticate

   1) Program

   2) Verify
```

Figure 8 • **ISP Operation Modes Selection**

14. On selecting the operation mode, the files in the USB storage device are displayed as shown in Figure 9.

    Note: Maximum of 10 files can be shown from the USB storage device.



```
Configuring MSC device...

MSC device ready for transfers

Select ISP operational mode

   0) Authenticate

   1) Program

   2) Verify

0

Elements in Root Directory (Maximum 10):

 0 -    ENVMONLY.SPI

 1 -    FABENVM.SPI

 2 -    FABONLY.SPI

 3 -    ISP_DEMO.SPI


 Choose the file out of above listed files by its index ('0' onwards)
```

Figure 9 • **Available Files in USB Mass Storage Device**

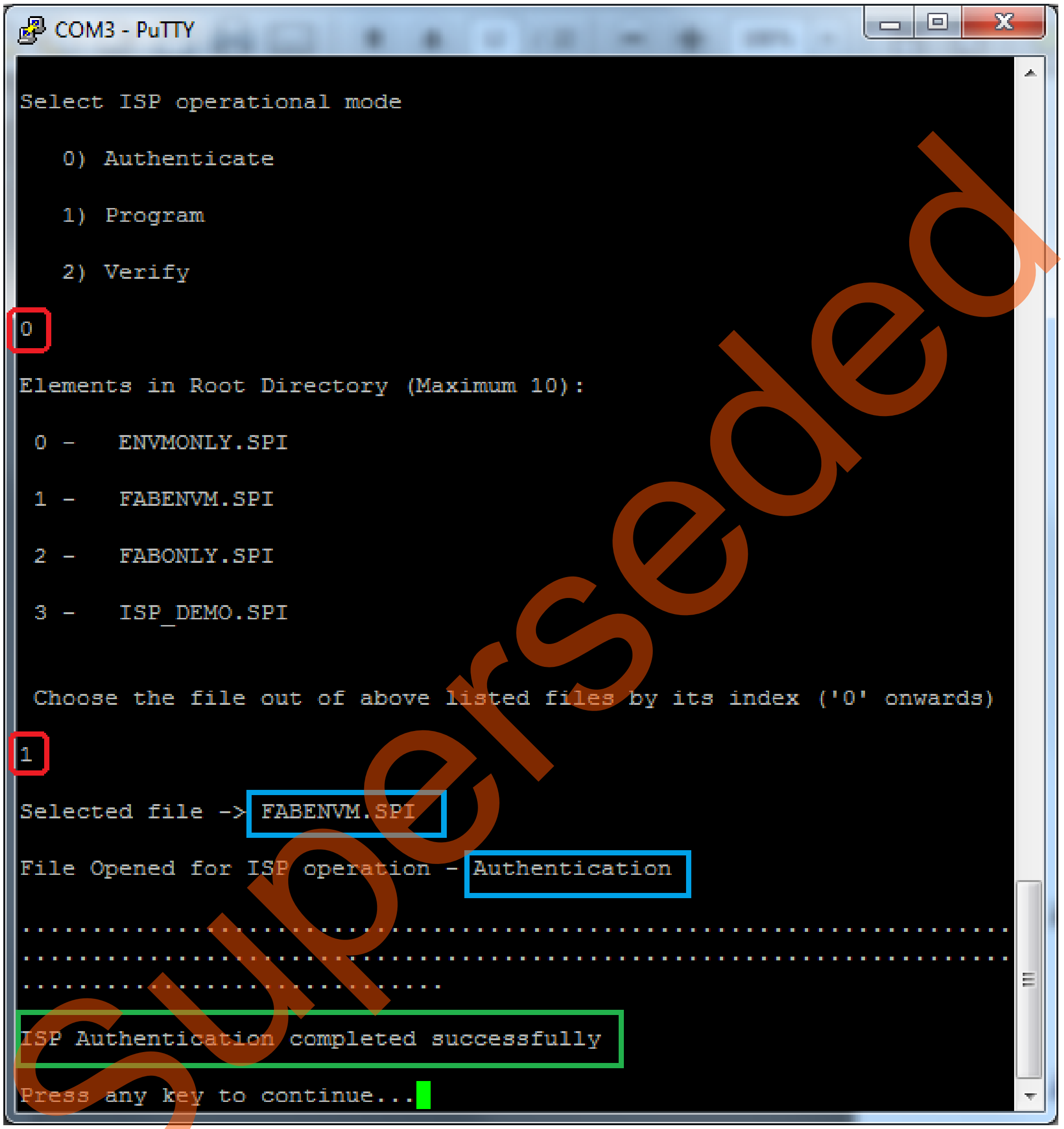
15. Select the programming file from the listed files by its index to perform the selected ISP operation mode.

Authenticate Operation Mode

To authenticate the data from `fabenvm.spi`, enter:

1. **0** to select **Authenticate** operation mode under **Select ISP operation mode**.
2. The corresponding index number to select `fabenvm.spi` programming file.

On selecting the programming file, the application starts reading the programming file from USB mass storage device to execute the ISP operation mode. On completion of the ISP authentication, the serial terminal emulation program displays an operation success message. Figure 10 shows the operation success message.



```
Select ISP operational mode

   0) Authenticate

   1) Program

   2) Verify

0

Elements in Root Directory (Maximum 10):

 0 -    ENVMONLY.SPI

 1 -    FABENVM.SPI

 2 -    FABONLY.SPI

 3 -    ISP_DEMO.SPI


 Choose the file out of above listed files by its index ('0' onwards)

1

Selected file -> FABENVM.SPI

File Opened for ISP operation - Authentication

......................................................................
......................................................................
..............................

ISP Authentication completed successfully

Press any key to continue...
```

Figure 10 • **ISP Authentication Results**

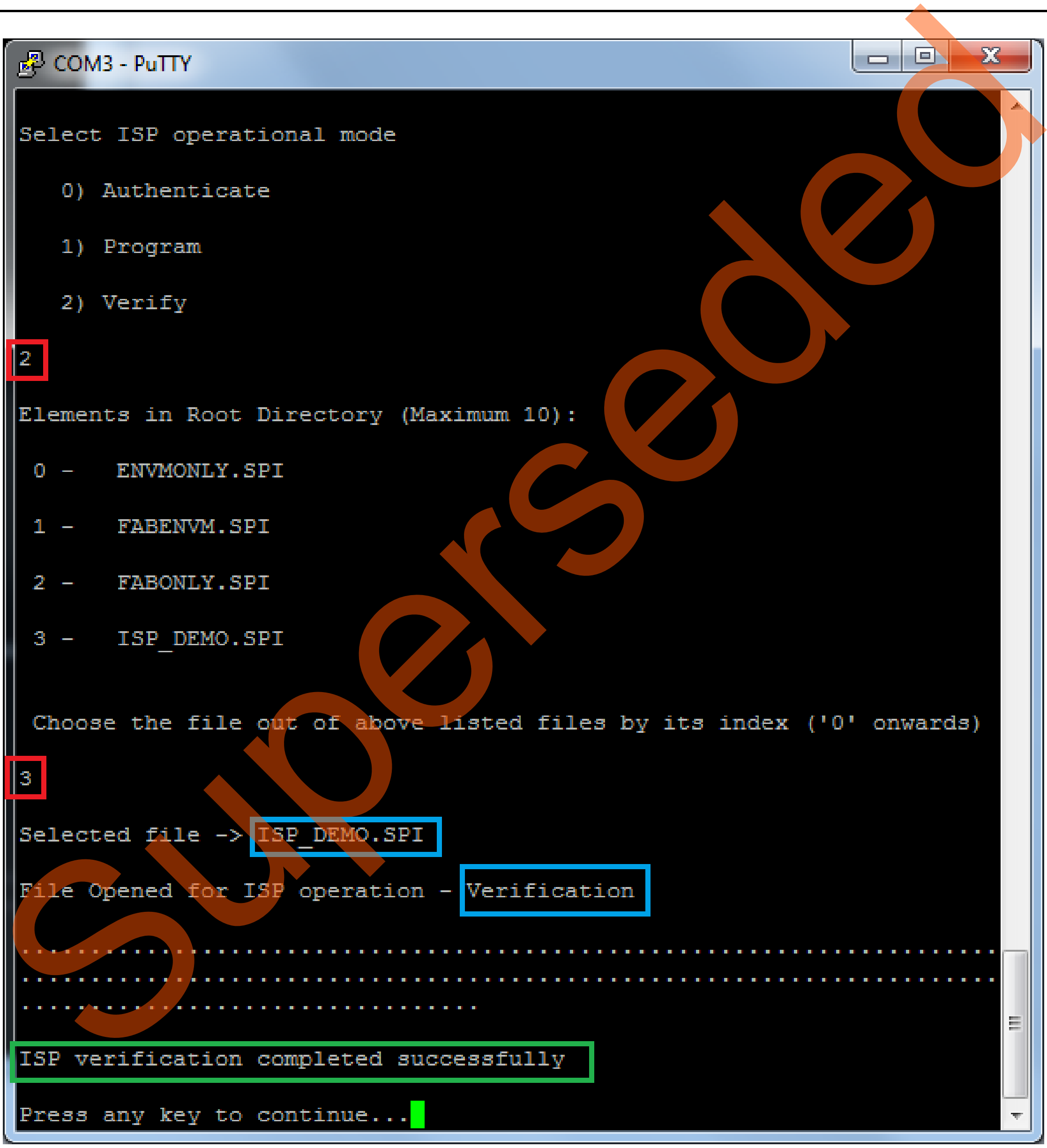
3. Press **SW9** to reset the SmartFusion2 Development Kit or Power Cycle the SmartFusion2 Development Kit to try other ISP operation modes. If the USB storage device files are not displayed on serial terminal emulation program, press **SW9** to reset the board.

Verify Operation Mode

To verify the device FPGA fabric and eNVM contents, enter:

1. **2** to select **Verify** operation mode under **Select ISP operation mode**.
2. The corresponding index number to select `isp_demo.spi` programming file.

   On selecting the programming file, the application starts reading the programming file from USB mass storage device to execute the ISP operation mode. On completion of the ISP verification, the serial terminal emulation program displays an operation success message. Figure 10 shows the operation success message.



```
Select ISP operational mode

   0) Authenticate

   1) Program

   2) Verify

2

Elements in Root Directory (Maximum 10):

 0 -    ENVMONLY.SPI

 1 -    FABENVM.SPI

 2 -    FABONLY.SPI

 3 -    ISP_DEMO.SPI


 Choose the file out of above listed files by its index ('0' onwards)

3

Selected file -> ISP_DEMO.SPI

File Opened for ISP operation - Verification

......................................................................
......................................................................
.................................

ISP verification completed successfully

Press any key to continue...
```

Figure 11 • **ISP Verification Results**

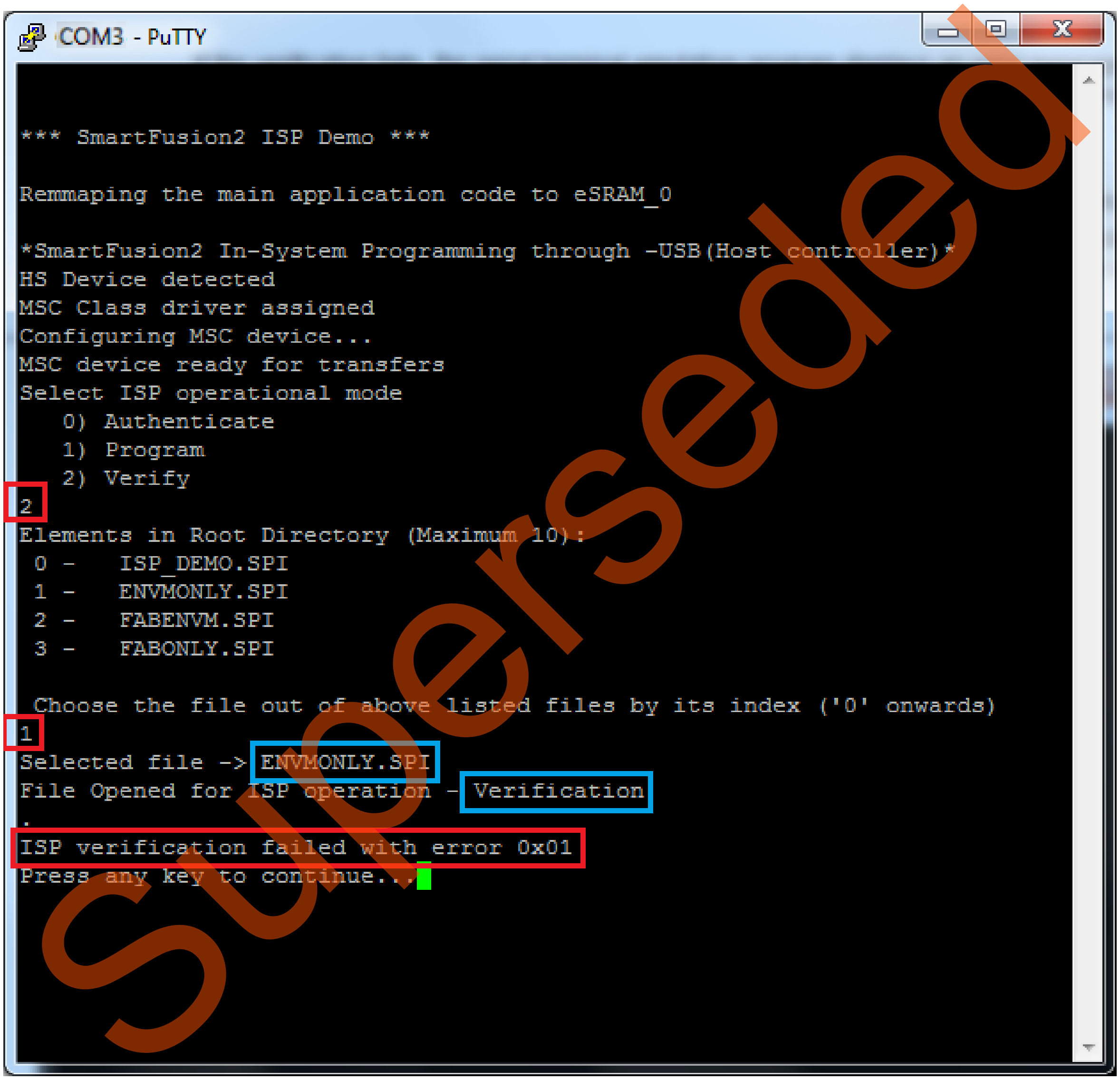
The verification operation demonstrated is for the `isp_demo.stp` file that is already running in the SmartFusion2 device. If any other `.spi` file is verified while the `isp_demo.stp` file is still running, that verification operation fails.

If the verification fails, the serial terminal emulation program displays an error message with an error code. Figure 12 shows an example error message. For more information on error codes, refer to the "Appendix 5: Error Codes" on page 28.

The programming files are at:

<download_folder>\sf2_isp_using_usb_interface_demo_df\sample_programming_files.

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.



```
*** SmartFusion2 ISP Demo ***

Remmaping the main application code to eSRAM_0

*SmartFusion2 In-System Programming through -USB(Host controller)*
HS Device detected
MSC Class driver assigned
Configuring MSC device...
MSC device ready for transfers
Select ISP operational mode
   0) Authenticate
   1) Program
   2) Verify
2
Elements in Root Directory (Maximum 10):
 0 -    ISP_DEMO.SPI
 1 -    ENVMONLY.SPI
 2 -    FABENVM.SPI
 3 -    FABONLY.SPI

 Choose the file out of above listed files by its index ('0' onwards)
1
Selected file -> ENVMONLY.SPI
File Opened for ISP operation - Verification
.
ISP verification failed with error 0x01
Press any key to continue...
```

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

3. Press **SW9** to reset the SmartFusion2 Development Kit or Power Cycle the SmartFusion2 Development Kit to try other ISP operation modes. If the USB storage device files are not displayed on the serial terminal emulation program, press **SW9** to reset the board.

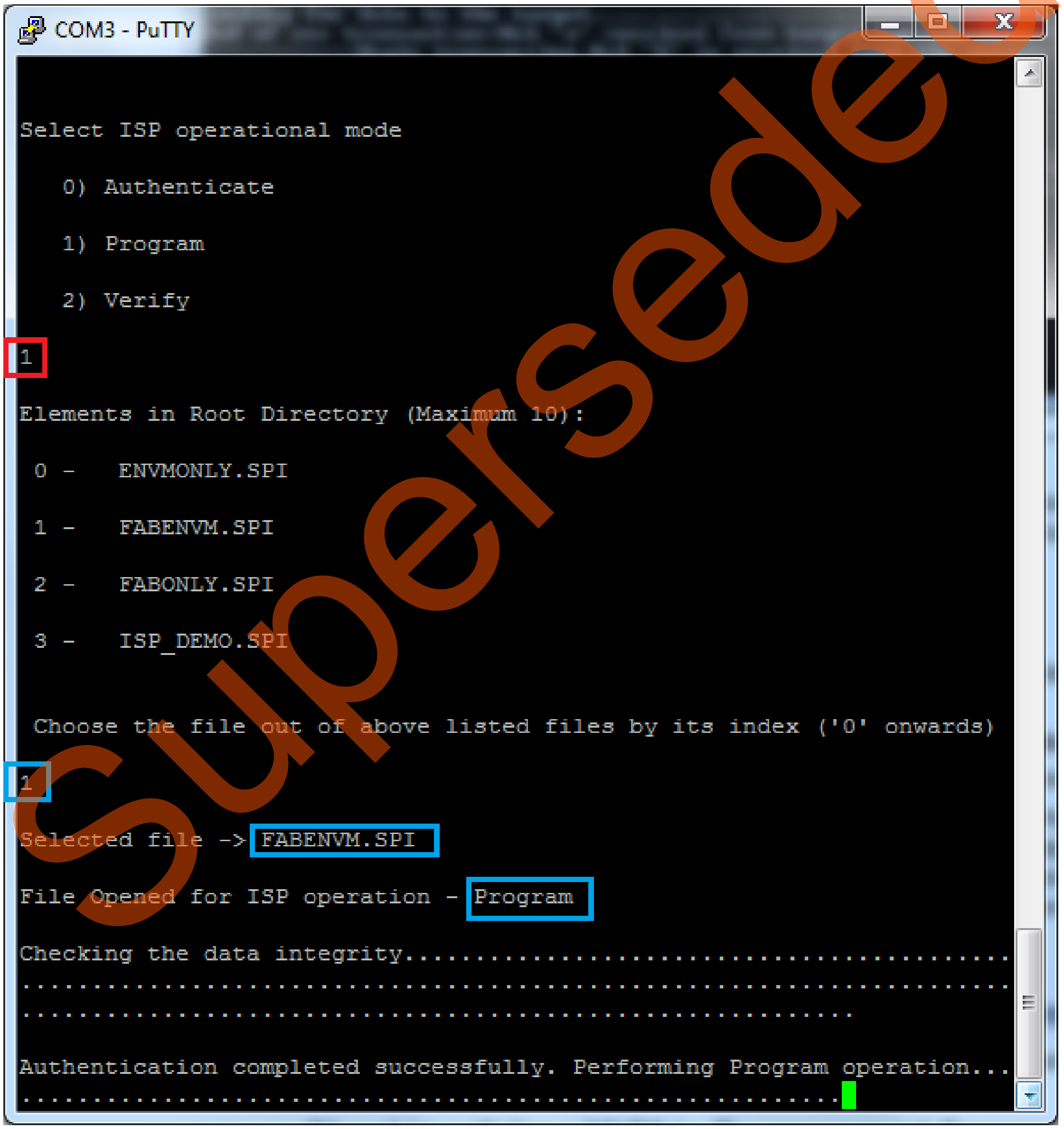
Program Operation Mode

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

1. **1** to select **Program** operation mode under **Select ISP operation mode**.
2. The corresponding index number to select `fabenvm.spi` programming file.

   On selecting the programming file, the application starts reading the programming file from the USB mass storage device to execute the ISP operation mode. The application checks the data integrity of the selected programming file prior to perform the ISP program operation. Once the programming operation is completed, an internal reset is generated for the new design to take effect.

Figure 13 shows selection of program operation mode for the `fabenvm.spi` programming file.



```
Select ISP operational mode

   0) Authenticate

   1) Program

   2) Verify

1

Elements in Root Directory (Maximum 10):

 0 -    ENVMONLY.SPI

 1 -    FABENVM.SPI

 2 -    FABONLY.SPI

 3 -    ISP_DEMO.SPI


 Choose the file out of above listed files by its index ('0' onwards)

1

Selected file -> FABENVM.SPI

File Opened for ISP operation - Program

Checking the data integrity..........................................
......................................................................
...........................................................

Authentication completed successfully. Performing Program operation...
..........................................................
```

Figure 13 • **ISP Program Operation Mode**

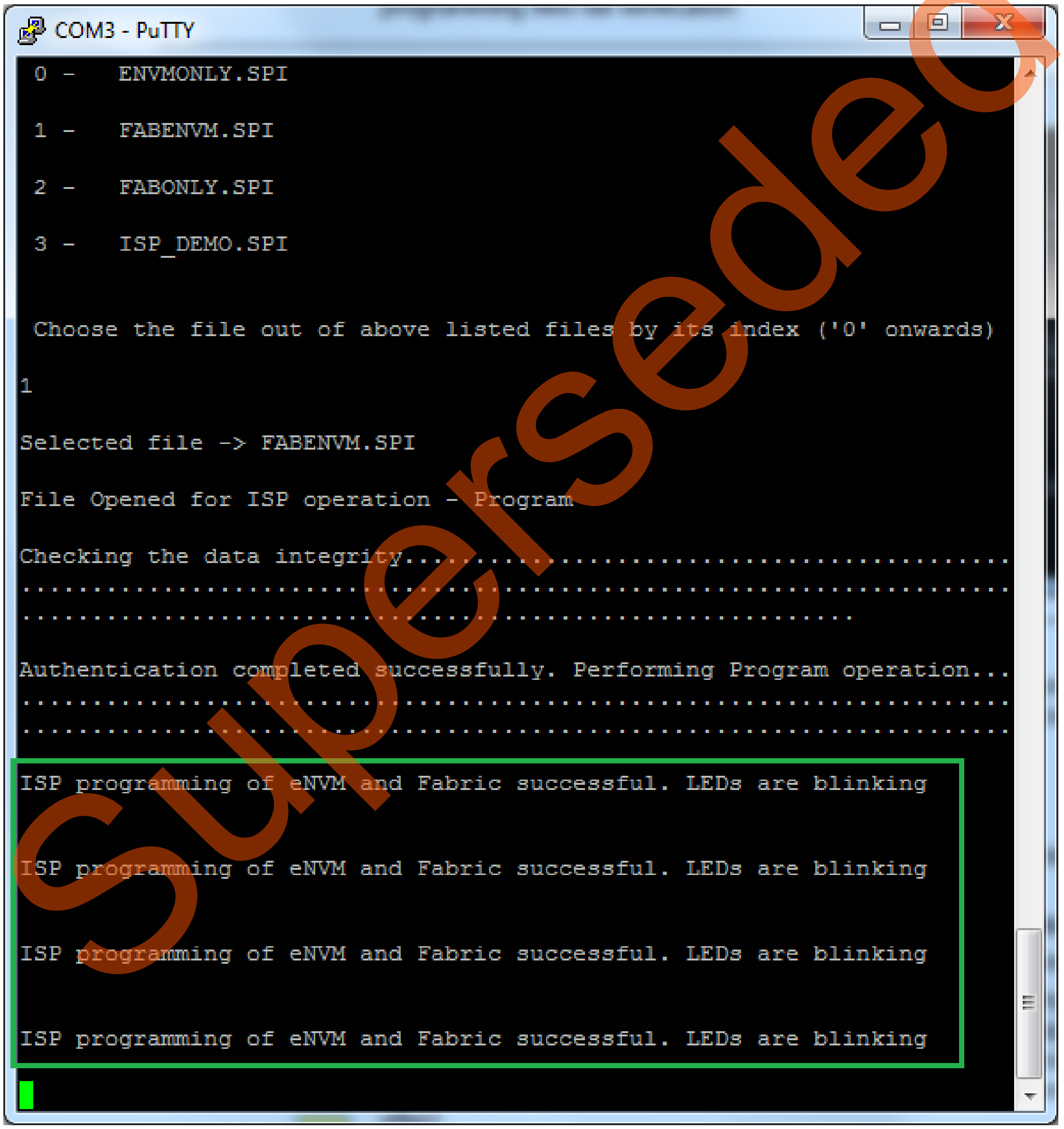
Press **SW9** to reset the SmartFusion2 Development Kit or Power Cycle the SmartFusion2 Development Kit.

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

The serial terminal emulation program displays the success message as shown in Figure 14 if the eNVM is programmed successfully.



```
 0 -    ENVMONLY.SPI

 1 -    FABENVM.SPI

 2 -    FABONLY.SPI

 3 -    ISP_DEMO.SPI


 Choose the file out of above listed files by its index ('0' onwards)

1

Selected file -> FABENVM.SPI

File Opened for ISP operation - Program

Checking the data integrity...........................................
......................................................................
...........................................................

Authentication completed successfully. Performing Program operation...
......................................................................
......................................................................

ISP programming of eNVM and Fabric successful. LEDs are blinking


ISP programming of eNVM and Fabric successful. LEDs are blinking


ISP programming of eNVM and Fabric successful. LEDs are blinking


ISP programming of eNVM and Fabric successful. LEDs are blinking

```

Figure 14 • **ISP Program Results**

Programming Results

The result shown in Figure 14 on page 22 is for the `fabenvm.spi` file. Table 3 shows the possible results for ISP Program operation mode for sample programming files provided in folder *<download_folder>\sf2_isp_using_usb_interface_demo_df\sample_programming_files*.
Not all `.spi` files listed in the table are demonstrated.

Table 3 • **ISP Programming Results**

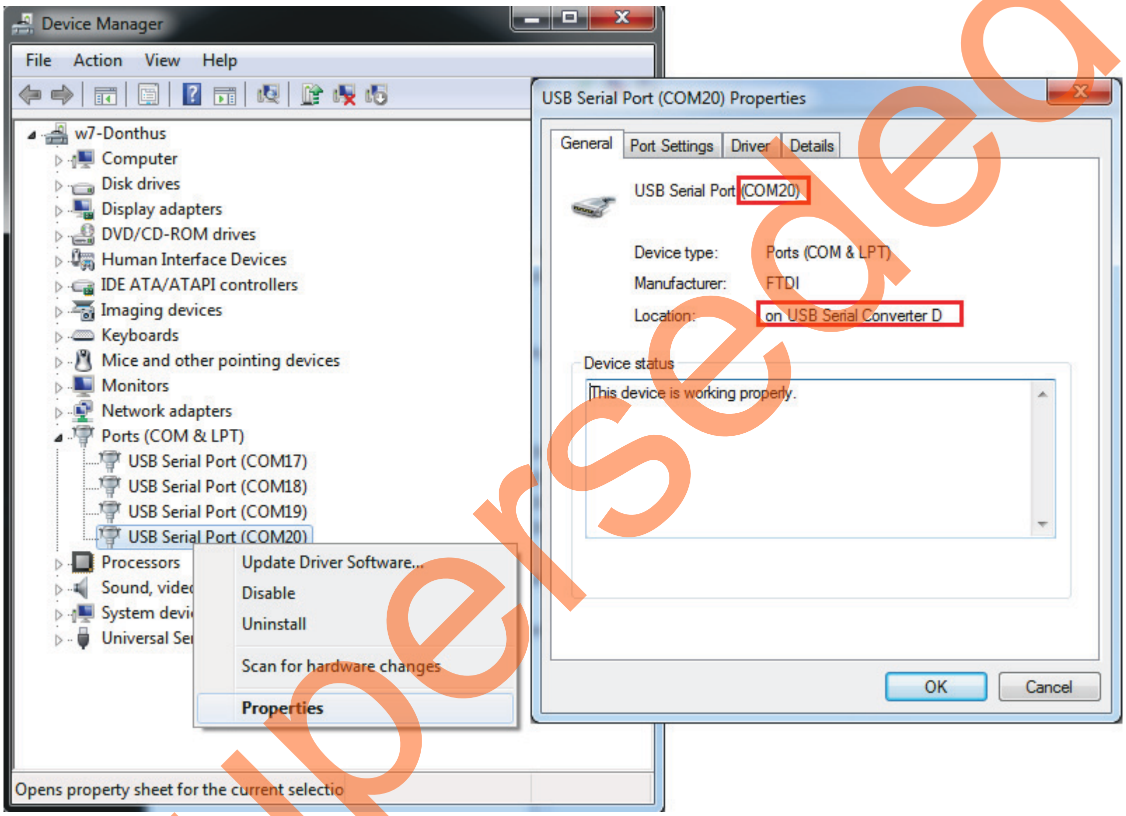
*.spi Programming File Name	eNVM Programming Result	FPGA Fabric Programming Result
`envmonly.spi`	The serial terminal emulation program shows successful eNVM program message	NA
`fabonly.spi`	NA	SmartFusion2 LEDs 1 to 4 blinks
`fabenvm.spi`	The serial terminal emulation program shows successful eNVM program message	SmartFusion2 LEDs 1 to 4 blinks

After successful ISP Program operation, the SmartFusion2 Development Kit board 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.



USB Serial Port (COM20)

Device type: Ports (COM & LPT)
Manufacturer: FTDI
Location: on USB Serial Converter D

Device status
This device is working properly.

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 26 shows these pin connections.
   - – Refer to Figure 18 on page 27 for the 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 the SmartFusion2 Development Kit board.

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

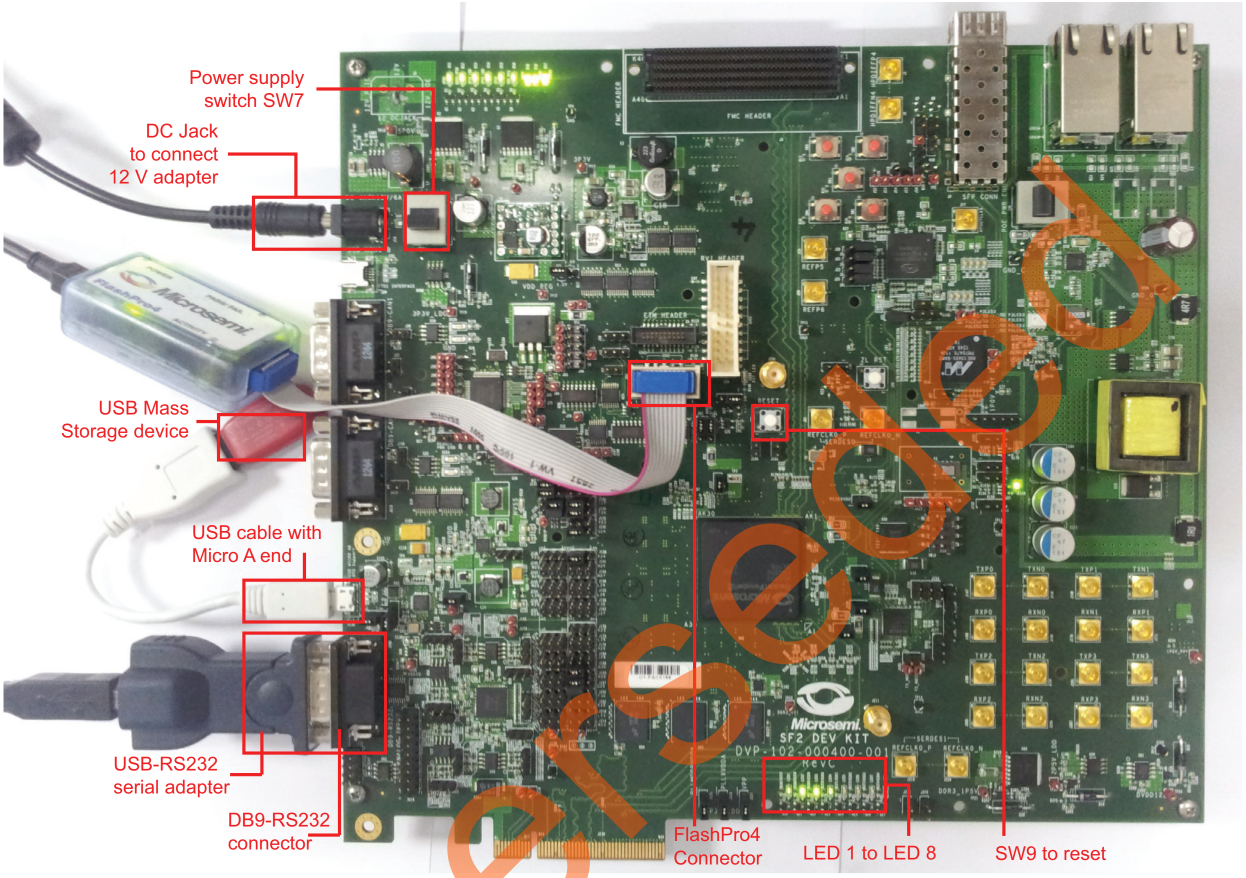


Figure 16 • **Board Setup when using the USB-RS232 Serial Adapter or RS232 Cable**

Appendix 3: Board Setup Through the USB to UART (FTDI) Interface using the USB A to Mini - B Cable

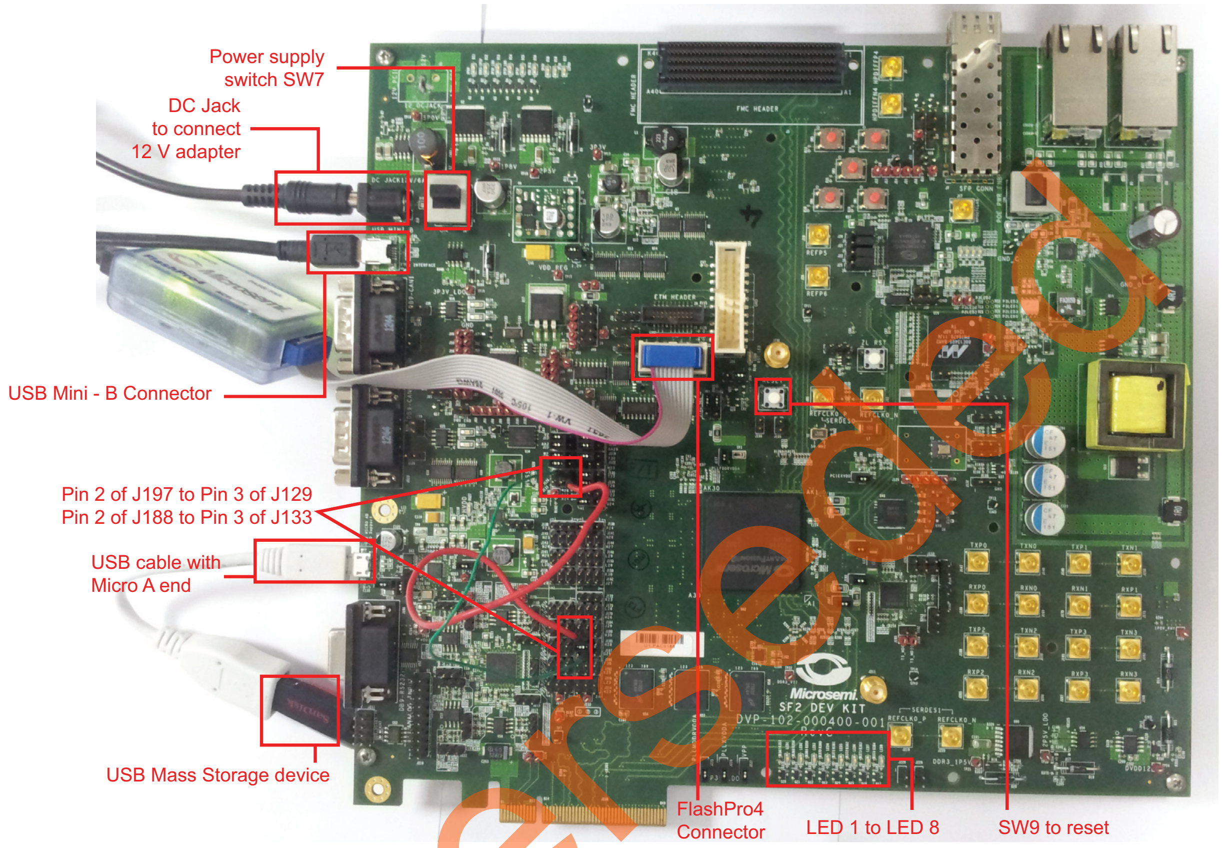
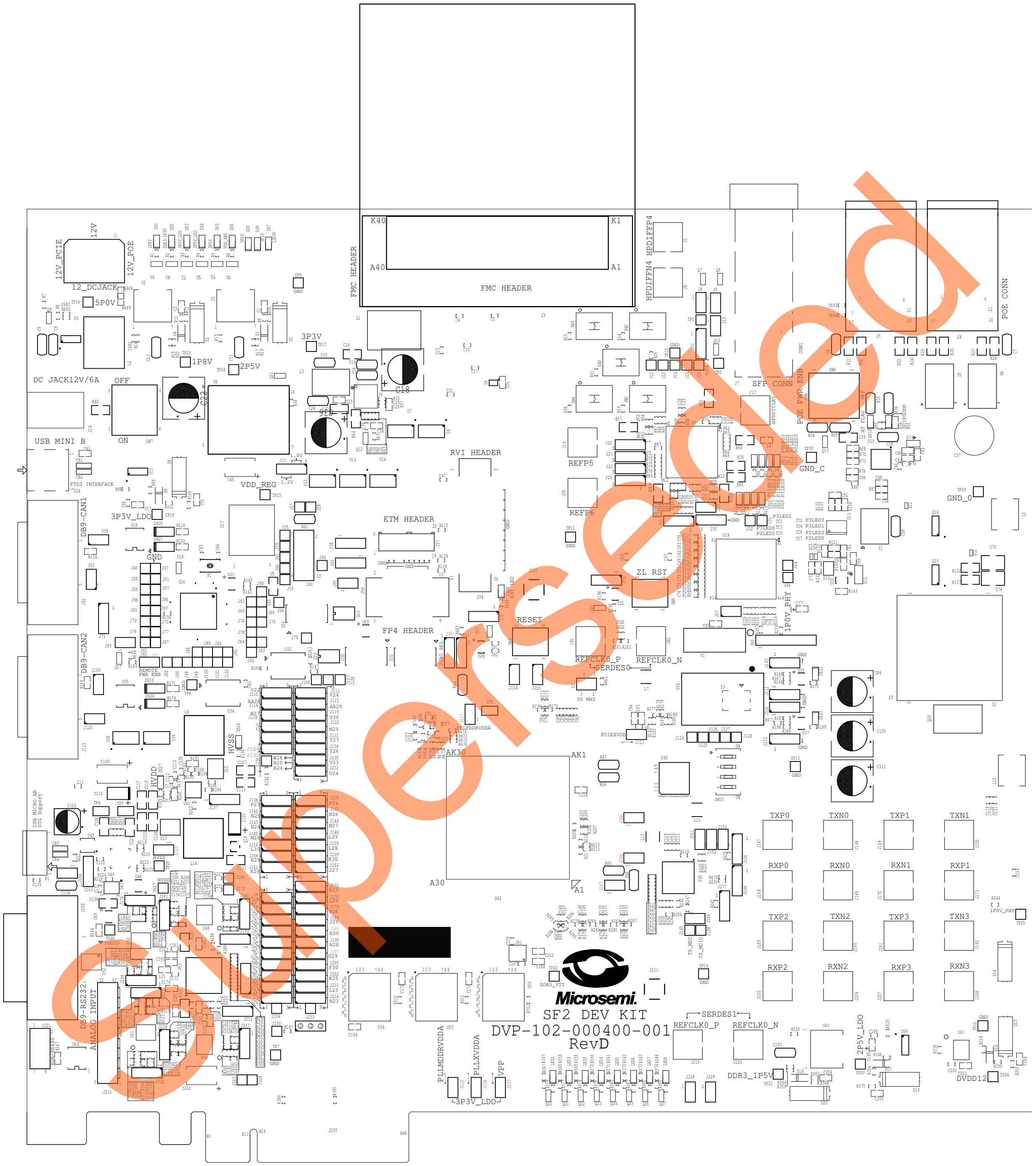

Power supply switch SW7
DC Jack to connect 12 V adapter
USB Mini - B Connector
Pin 2 of J197 to Pin 3 of J129
Pin 2 of J188 to Pin 3 of J133
USB cable with Micro A end
USB Mass Storage device
FlashPro4 Connector
LED 1 to LED 8
SW9 to reset

Figure 17 • **Board Setup when Through the USB to UART (FTDI) Interface using the USB A to Mini - B Cable**

Appendix 4: Jumper Locations



[redacted]

Figure 18 • **SmartFusion2 Development Kit Silkscreen Top View**

Figure 18 shows the jumper locations in the Development Kit board:

- Jumpers highlighted in red are set by default.
- Jumpers highlighted in green must be set manually.
- The location of the jumpers in Figure 18 on page 27 are searchable.

Appendix 5: Error Codes

Table 4 • **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 `fabenvm.spi` programming file. The Libero design file is provided in *<download_folder>\sf2_isp_using_usb_interface_demo_df\sample_programming_file\fabric_and_envm*.
2. Right-click **Bitstream** under **Handoff Design for Production** in the **Design Flow** tab, and click **Export...** from the context menu.

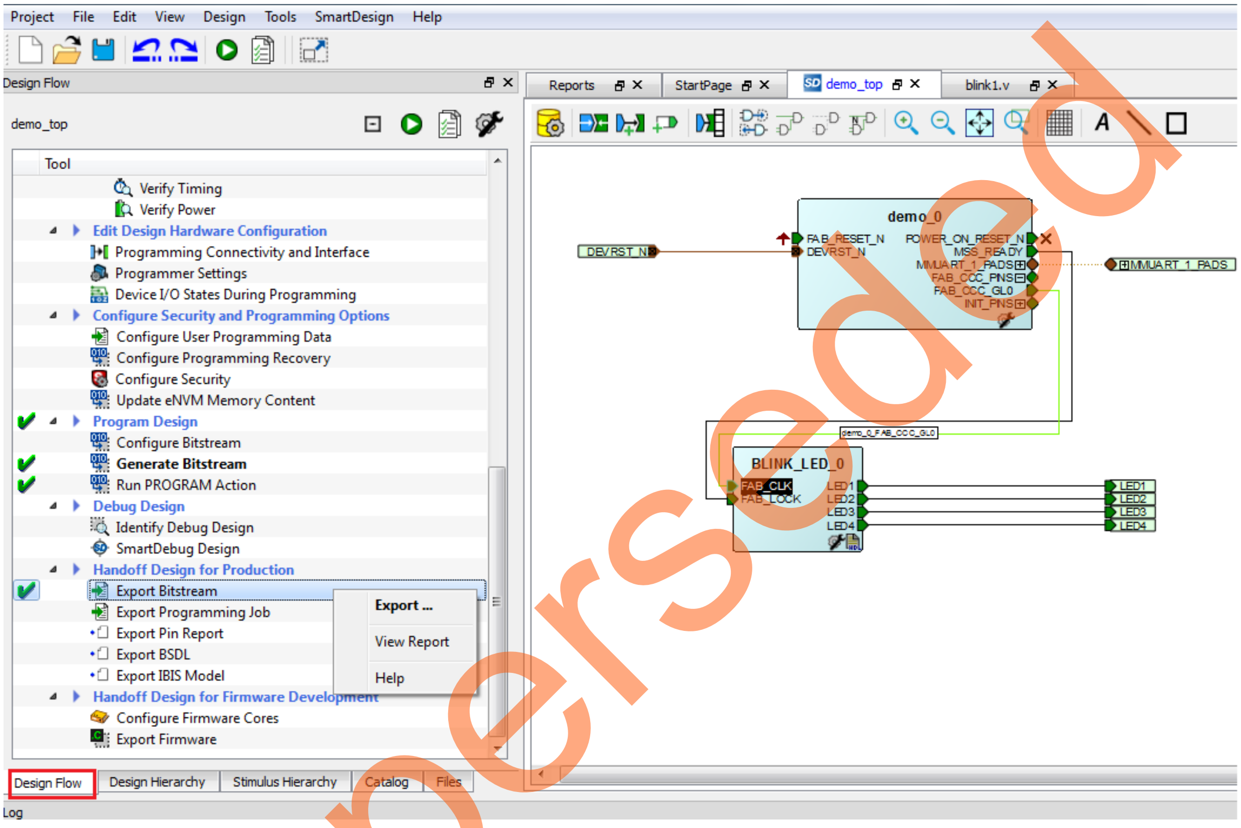
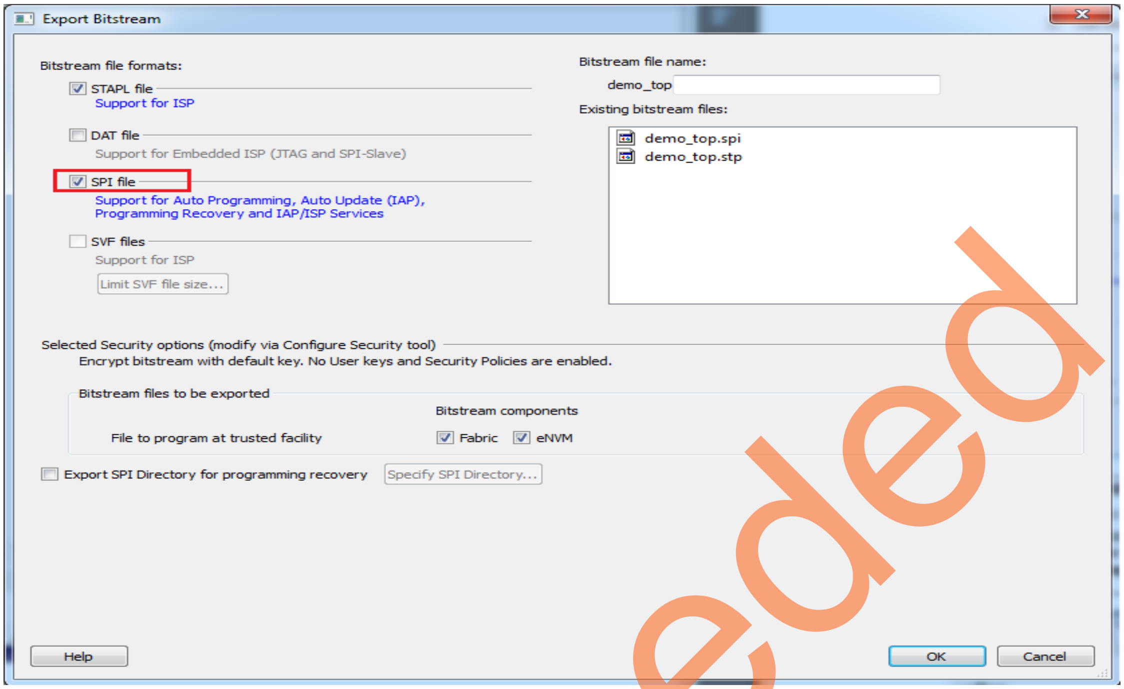


Figure 19 • **Configuring Export Bitstream**

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



Bitstream file formats:

☑ STAPL file
Support for ISP

☐ DAT file
Support for Embedded ISP (JTAG and SPI-Slave)

☑ SPI file
Support for Auto Programming, Auto Update (IAP), Programming Recovery and IAP/ISP Services

☐ SVF files
Support for ISP

Bitstream file name: demo_top

Existing bitstream files:
demo_top.spi
demo_top.stp

Selected Security options (modify via Configure Security tool)
Encrypt bitstream with default key. No User keys and Security Policies are enabled.

Bitstream files to be exported

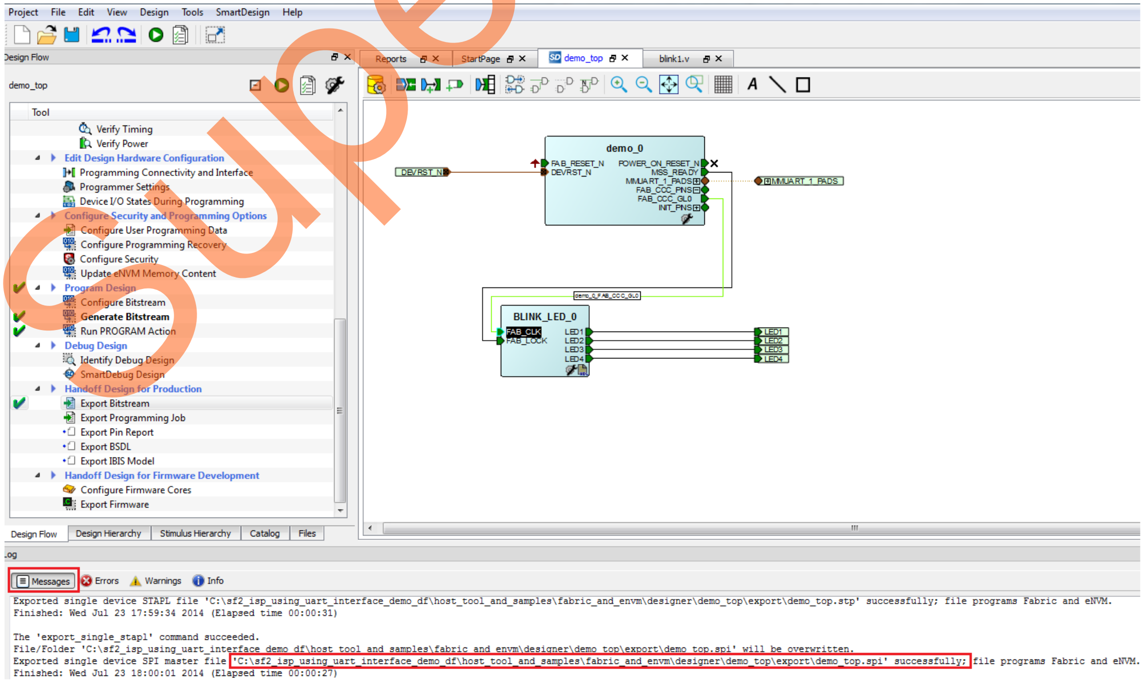
File to program at trusted facility

Bitstream components: ☑ Fabric ☑ eNVM

☐ Export SPI Directory for programming recovery

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 29). Figure 21 shows the `.spi` file location in **Messages** tab.



```
Exported single device STAPL file 'C:\sf2_isp_using_uart_interface_demo_df\host_tool_and_samples\fabric_and_envm\designer\demo_top\export\demo_top.stp' successfully; file programs Fabric and eNVM.
Finished: Wed Jul 23 17:59:34 2014 (Elapsed time 00:00:31)

The 'export_single_stapl' command succeeded.
File/Folder 'C:\sf2_isp_using_uart_interface_demo_df\host_tool_and_samples\fabric_and_envm\designer\demo_top\export\demo_top.spi' will be overwritten.
Exported single device SPI master file 'C:\sf2_isp_using_uart_interface_demo_df\host_tool_and_samples\fabric_and_envm\designer\demo_top\export\demo_top.spi' successfully; file programs Fabric and eNVM.
Finished: Wed Jul 23 18:00:01 2014 (Elapsed time 00:00:27)
```

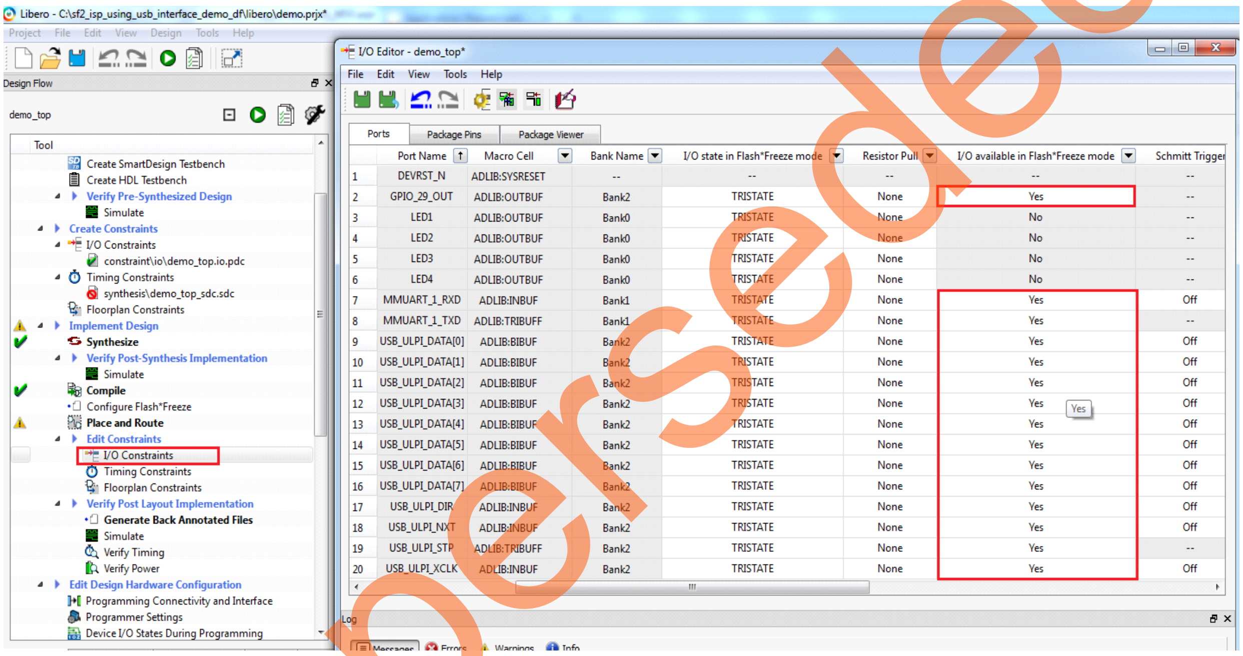
Figure 21 • **.SPI File Location**

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, one UART interface (MMUART_1) for serial communication and USB OTG as mass storage class host. 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 Figure 22. The user has to **Commit and Check** the settings from the File menu after configuring the ports.

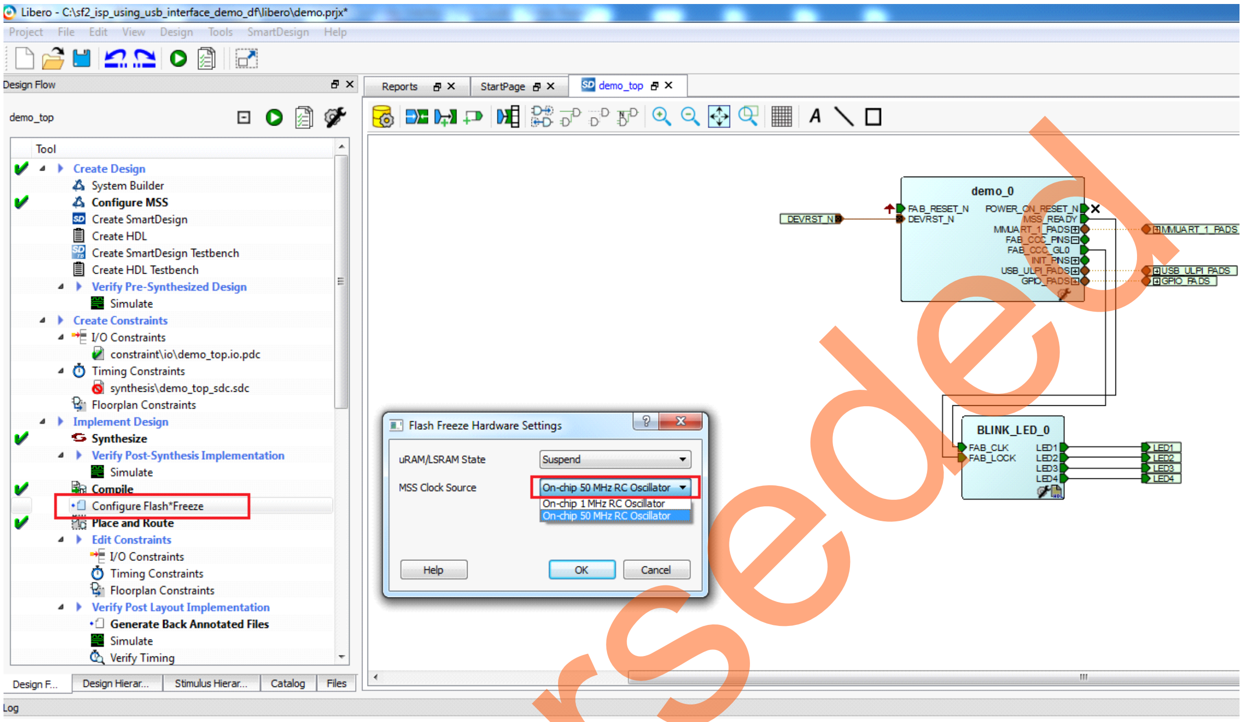

Libero - C:\sf2_isp_using_usb_interface_demo_df\libero\demo.prjx*
Project File Edit View Design Tools Help
Design Flow
demo_top
Tool
Create SmartDesign Testbench
Create HDL Testbench
Verify Pre-Synthesized Design
Simulate
Create Constraints
I/O Constraints
constraint\io\demo_top.io.pdc
Timing Constraints
synthesis\demo_top_sdc.sdc
Floorplan Constraints
Implement Design
Synthesize
Verify Post-Synthesis Implementation
Simulate
Compile
Configure Flash*Freeze
Place and Route
Edit Constraints
I/O Constraints
Timing Constraints
Floorplan Constraints
Verify Post Layout Implementation
Generate Back Annotated Files
Simulate
Verify Timing
Verify Power
Edit Design Hardware Configuration
Programming Connectivity and Interface
Programmer Settings
Device I/O States During Programming
I/O Editor - demo_top*
File Edit View Tools Help
Ports
Package Pins
Package Viewer
Log
Messages
Errors
Warnings
Info

	Port Name	Macro Cell	Bank Name	I/O state in Flash*Freeze mode	Resistor Pull	I/O available in Flash*Freeze mode	Schmitt Trigger
1	DEVRST_N	ADLIB:SYSRESET	--	--	--	--	--
2	GPIO_29_OUT	ADLIB:OUTBUF	Bank2	TRISTATE	None	Yes	--
3	LED1	ADLIB:OUTBUF	Bank0	TRISTATE	None	No	--
4	LED2	ADLIB:OUTBUF	Bank0	TRISTATE	None	No	--
5	LED3	ADLIB:OUTBUF	Bank0	TRISTATE	None	No	--
6	LED4	ADLIB:OUTBUF	Bank0	TRISTATE	None	No	--
7	MMUART_1_RXD	ADLIB:INBUF	Bank1	TRISTATE	None	Yes	Off
8	MMUART_1_TXD	ADLIB:TRIBUFF	Bank1	TRISTATE	None	Yes	--
9	USB_ULPI_DATA[0]	ADLIB:BIBUF	Bank2	TRISTATE	None	Yes	Off
10	USB_ULPI_DATA[1]	ADLIB:BIBUF	Bank2	TRISTATE	None	Yes	Off
11	USB_ULPI_DATA[2]	ADLIB:BIBUF	Bank2	TRISTATE	None	Yes	Off
12	USB_ULPI_DATA[3]	ADLIB:BIBUF	Bank2	TRISTATE	None	Yes	Off
13	USB_ULPI_DATA[4]	ADLIB:BIBUF	Bank2	TRISTATE	None	Yes	Off
14	USB_ULPI_DATA[5]	ADLIB:BIBUF	Bank2	TRISTATE	None	Yes	Off
15	USB_ULPI_DATA[6]	ADLIB:BIBUF	Bank2	TRISTATE	None	Yes	Off
16	USB_ULPI_DATA[7]	ADLIB:BIBUF	Bank2	TRISTATE	None	Yes	Off
17	USB_ULPI_DIR	ADLIB:INBUF	Bank2	TRISTATE	None	Yes	Off
18	USB_ULPI_NXT	ADLIB:INBUF	Bank2	TRISTATE	None	Yes	Off
19	USB_ULPI_STP	ADLIB:TRIBUFF	Bank2	TRISTATE	None	Yes	--
20	USB_ULPI_XCLK	ADLIB:INBUF	Bank2	TRISTATE	None	Yes	Off

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.


Libero - C:\sf2_isp_using_usb_interface_demo_df\libero\demo.prjx*

Flash Freeze Hardware Settings

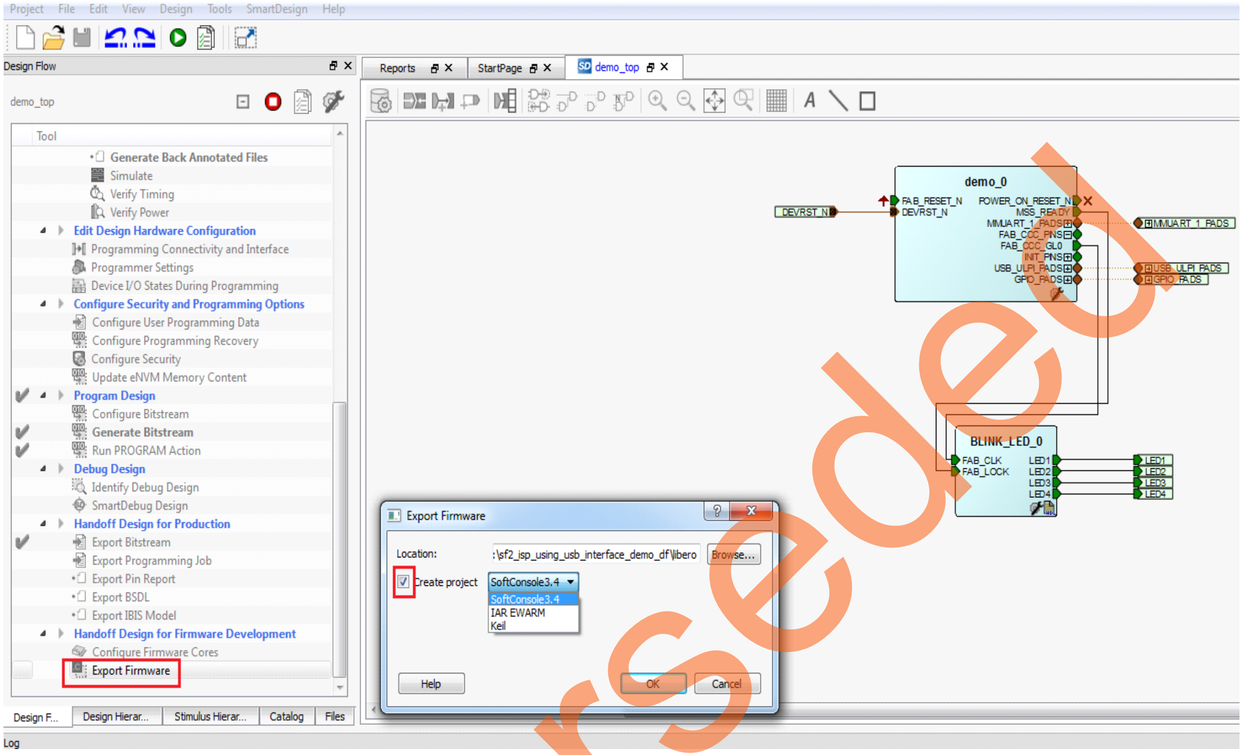
uRAM/LSRAM State: Suspend

MSS Clock Source: On-chip 50 MHz RC Oscillator

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.



Location: :\sf2_isp_using_usb_interface_demo_df\libero

☑ Create project SoftConsole3.4

Figure 24 • Export Firmware Options

On successful firmware generation, the firmware and SoftConsole folders are generated at *<download_folder>\sf2_isp_using_usb_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_usb_interface_demo_df\libero\SoftConsole\demo_MSS_CM3)* using SoftConsole IDE v3.4 SP1. Figure 25 shows **SoftConsole Project** workspace.

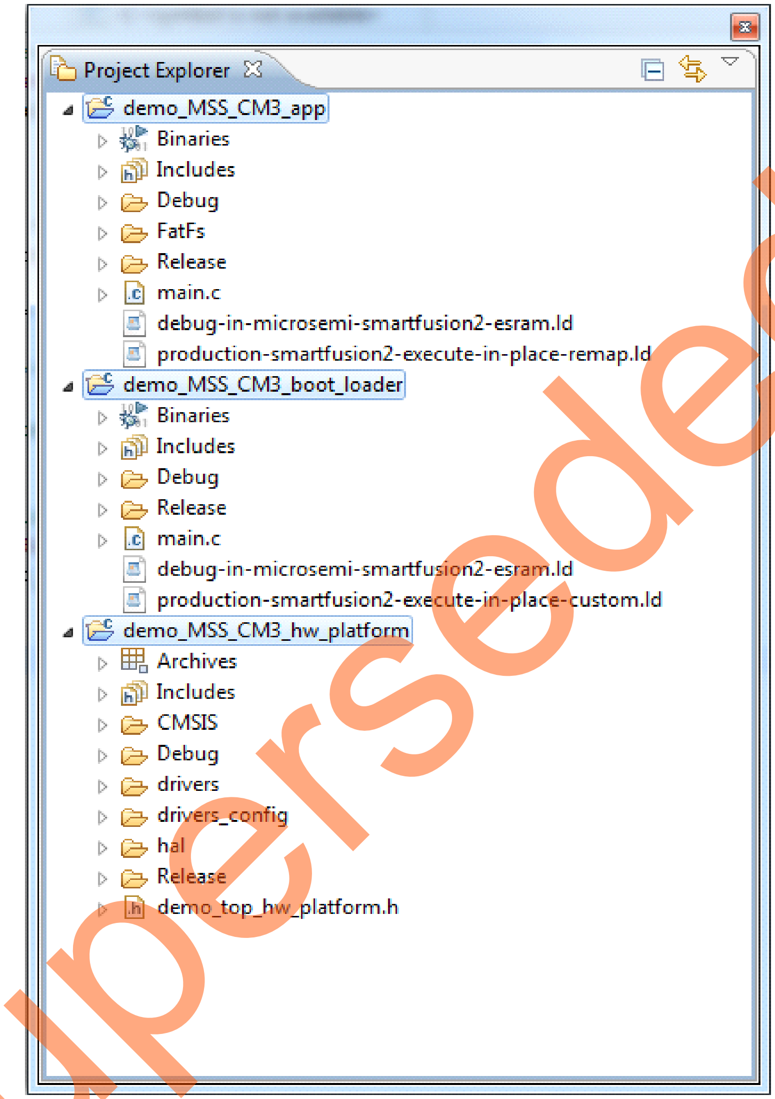


Figure 25 • **SoftConsole Project Workspace**

The SoftConsole workspace consists of three projects.

- demo_MSS_CM3_app
  This project displays the available programming files from USB mass storage device. 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 59740).	NA
Revision 3 (May 2014)	Updated the document for Libero v11.3 software release (SAR 56662).	NA
Revision 2 (December 2013)	Updated "Demo Design Description" section (SAR 53451).	8
Revision 1 (November 2013)	Updated the document for Libero v11.2 software release (SAR 52963).	NA
Revision 0 (October 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.

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