SmartFusion2 and IGLOO2 - Accessing eNVM and eSRAM from FPGA Fabric - Libero SoC v11.4

Table of Contents

Purpose

This application note describes how to access the embedded nonvolatile memory (eNVM) and embedded static random access memory (eSRAM) from FPGA fabric in SmartFusion®2 system-on-chip (SoC) FPGA and IGLOO®2 FPGA devices.

Introduction

This application note describes the following:

- Accessing the eNVM from FPGA fabric
- Accessing the eSRAM from FPGA fabric

Accessing eNVM from FPGA Fabric

SmartFusion2 SoC FPGA and IGLOO2 FPGA devices have a maximum of two on-chip 256 KB flash memories called eNVM. The eNVM stores the application code image or data required to be stored by the end application. The eNVM block is interfaced through the eNVM controller to the AHB bus matrix.

In SmartFusion2 SoC FPGA and IGLOO2 FPGA devices, the eNVM can be initialized by custom logic in FPGA fabric (Fabric Master).

In this application note, the Fabric master writes and reads from 25th page (address starting from 0x60000C80 to 0x60000CFC) of the eNVM.

For more information about eNVM initialization methods, refer to Application Note AC391.

Accessing eSRAM from FPGA Fabric

SmartFusion2 SoC FPGA and IGLOO2 FPGA devices have two eSRAM blocks, each of 32 Kbytes, for data read and write operations. These eSRAM blocks are interfaced through eSRAM controllers to the AHB bus matrix.

In SmartFusion2 SoC FPGA and IGLOO2 FPGA devices, the eSRAM can be accessed by custom logic in FPGA fabric (Fabric Master).

In this application note, Fabric master write and reads from 32 eSRAM locations (0x20000000 to 0x20000080).

References

The following list of references is used in this document.

Microsemi Publications

- *IGLOO2 System Builder User Guide*
- *IGLOO2 High Performance Memory Subsystem User Guide*
- *IGLOO2 FPGA Programming User Guide*
- *SmartFusion2 MSS Embedded Nonvolatile Memory (eNVM) Simulation*
- *SmartFusion2 Microcontroller Subsystem User Guide*
- *SmartFusion2 System Builder User Guide*

Design Requirements

Table 1 lists the design requirements.

Table 1 • **Design Requirements**

Design Requirements	Description
Hardware Requirements	
SmartFusion2 Evaluation Kit (M2S025T - FGG484): • 12 V adapter (provided along with the kit) • FlashPro4 programmer (provided along with the kit)	Rev C or later
IGLOO2 Evaluation Kit: • 12 V adapter (provided along with the kit) • FlashPro4 programmer (provided along with the kit)	Rev C or later
Host PC or Laptop	Any 64-bit Windows Operating System
Software Requirements	
Libero® SoC (System-on-Chip)	11.4

Design Description

The design examples included with this application note uses the following DIP switches:

- **SW5-1 DIP switch**: Used to start the eNVM write and read operations. An incremental data pattern starting from 0x0000000 to 0x0000001F is written to 25th page of the eNVM.
- **SW5-2 DIP switch**: Used to start the eSRAM write and read operations. An incremental data starting from 0xA1B2C300 to 0xA1B2C31F is written to the eSRAM locations starting from 0x20000000 to 0x20000080 respectively.

During the eSRAM or eNVM read operation, the read data from the eNVM or eSRAM is stored in the Fabric SRAM. The SmartDebug tool in Libero SoC verifies the write and read operations performed on the eNVM and eSRAM.

The design example uses two RTL FSMs—one FSM provides the eNVM or eSRAM write and read commands. The other FSM is an AHB master which receives these commands and communicates with the selected memory using AHB bus matrix through FIC_0 (Fabric Interface Controller) interface.

The read operations of the eNVM and the read and write operations of the eSRAM are simple AHB transactions. eNVM write requires a separate set of command sequences. For more information about this, see "Appendix B" on page 26

Hardware Implementation

The hardware implementation involves configuring the Device Features, Memory, Peripheral, and Clocks pages using System Builder. Configuring the TPSRAM IP and adding fabric logic are done at the top level using Smart Design.

Figure 1 shows the top level hardware design in SmartDesign for IGLOO2.

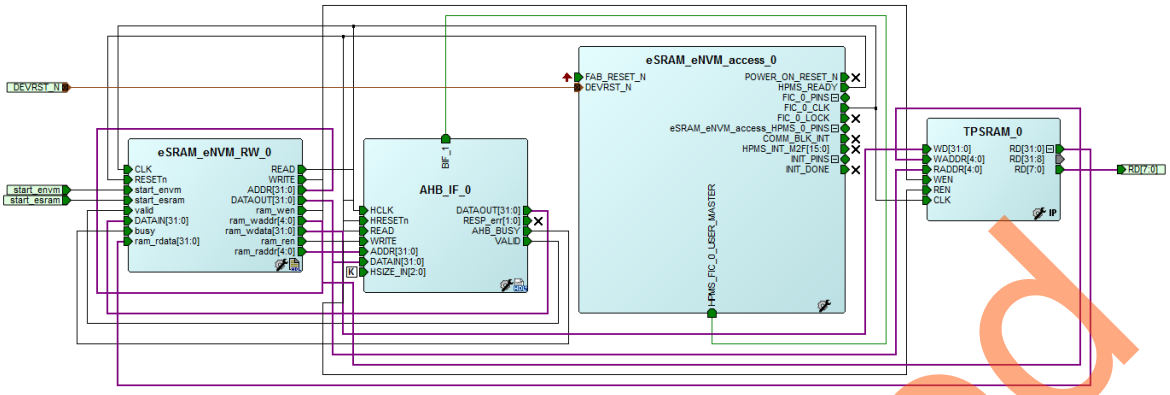

***Figure 1* • Top level SmartDesign for IGLOO2**

Figure 2 shows the top level hardware design in SmartDesign for SmartFusion2.

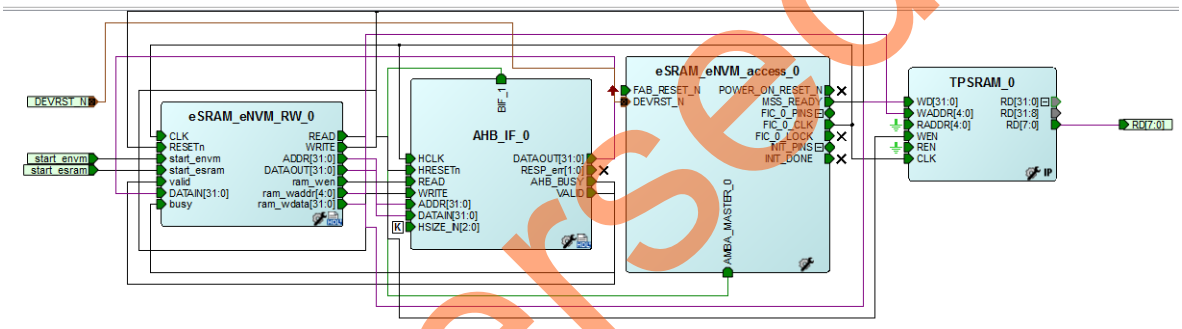

***Figure 2* • Top level SmartDesign for SmartFusion2**

SmartDesign Components

The top level SmartDesign has four components (as shown in Figure 1 and Figure 2):

- **eSRAM_eNVM_access_0**: System Builder generated component.
- **AHB_IF_0**: User generated RTL FSM, which performs AHB master function. This FSM interacts with the eNVM and eSRAM controller using AHB switch matrix through the FIC_0 interface.
- **eSRAM_eNVM_RW_0**:User generated RTL FSM, which takes inputs from the user and provides the required commands to AHB_IF_0 (AHB master).
- **TPSRAM_0**: Fabric IP core. Stores the data read from the eNVM or eSRAM.

System Builder Configuration for IGLOO2

1. In the **Device Features** page, ensure that **HPMS On-chip Flash Memory (eNVM)** and **On-chip SRAM (eSRAM)** check boxes are checked, as shown in Figure 3.

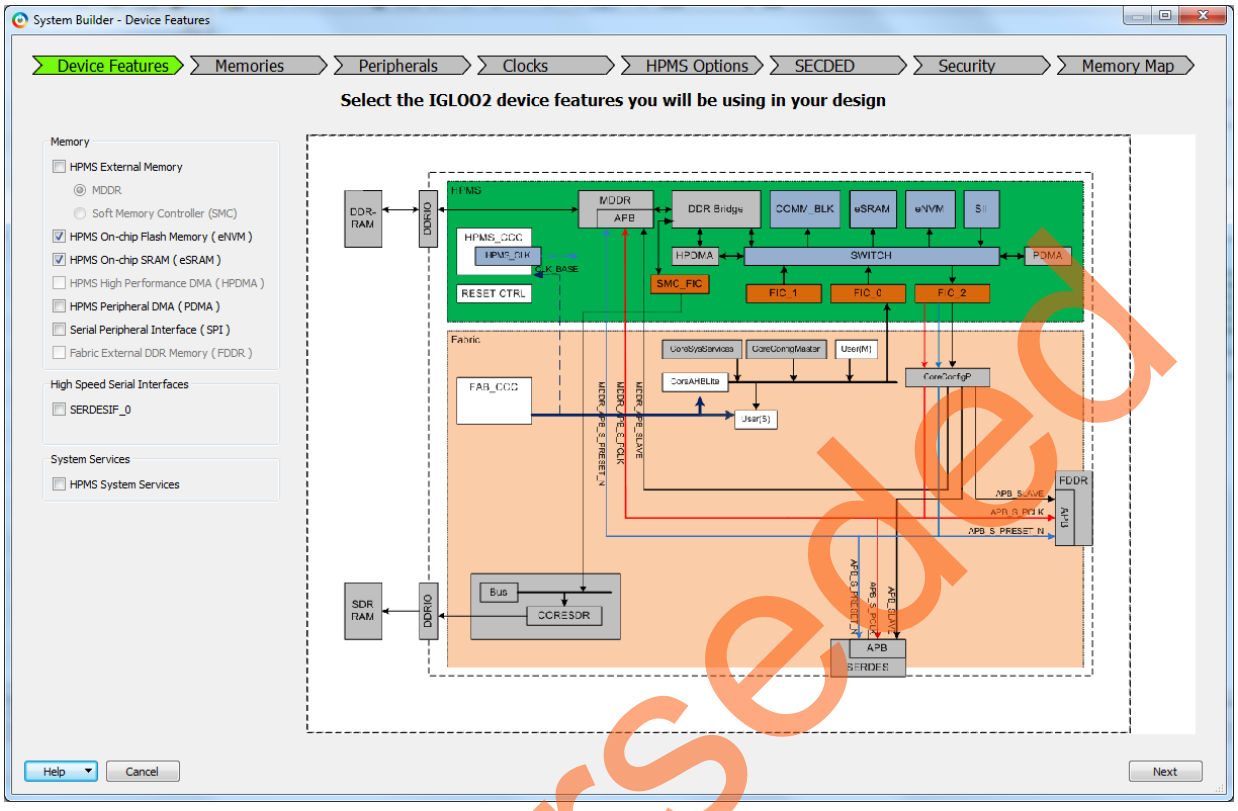

Figure 3 • **IGLOO2 - Device Features Page**

2. In the **Memories** page, add **zeros_client** to initialize the eNVM with zeros. Figure 4 shows the **Memories** page with zeros client.

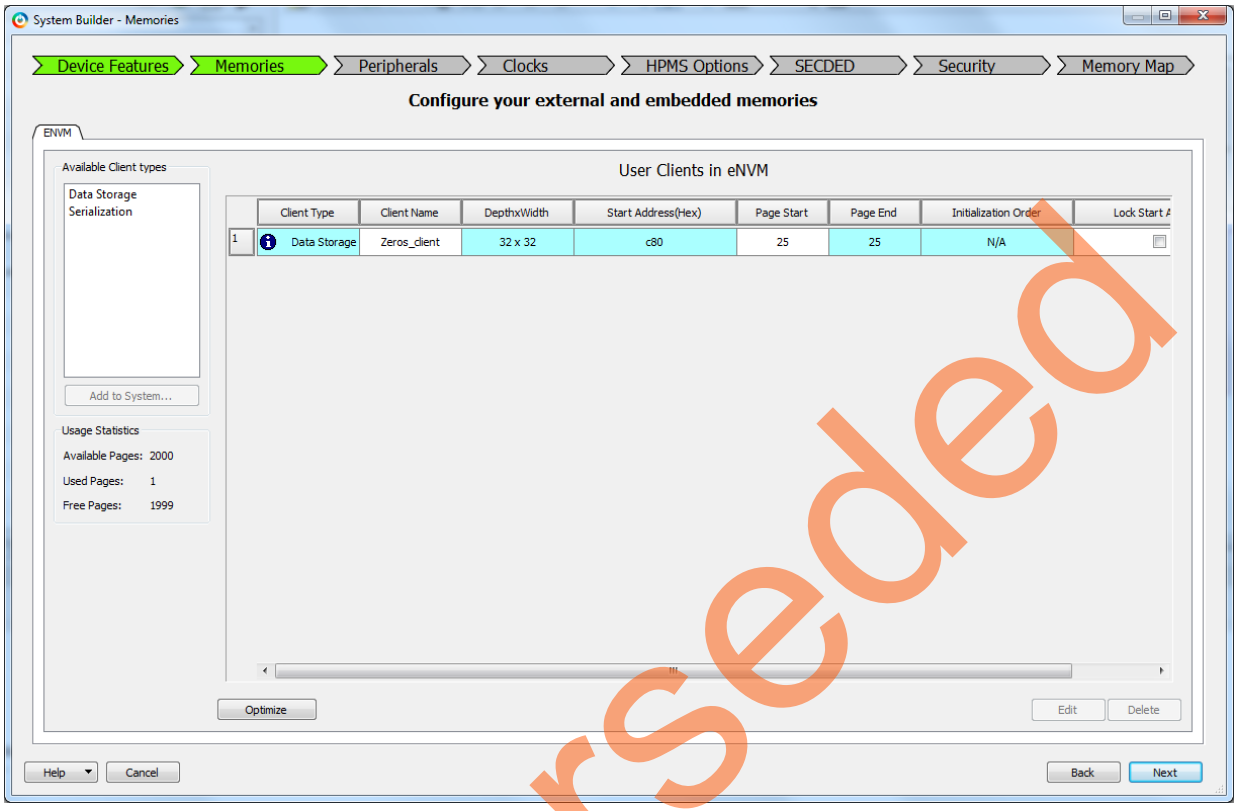

***Figure 4* • IGLOO2 - Memories Page**

3. In the **Peripherals** page, add **HPMS FIC_0-Fabric Master Subsystem** to provide the AHBL master interface to the user logic, as highlighted in Figure 5.

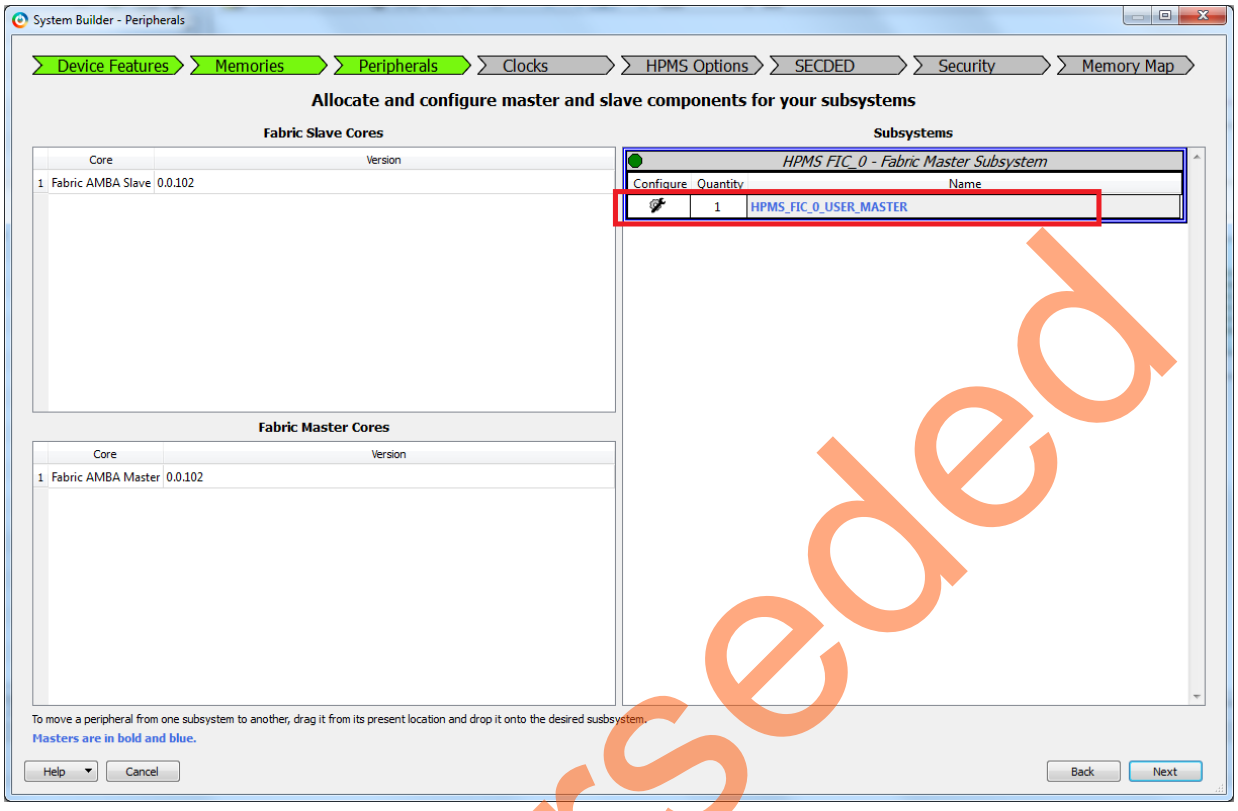

Figure 5 • **IGLOO2 - Peripherals Page**

Figure 6 shows the clocks configuration page for IGLOO2.

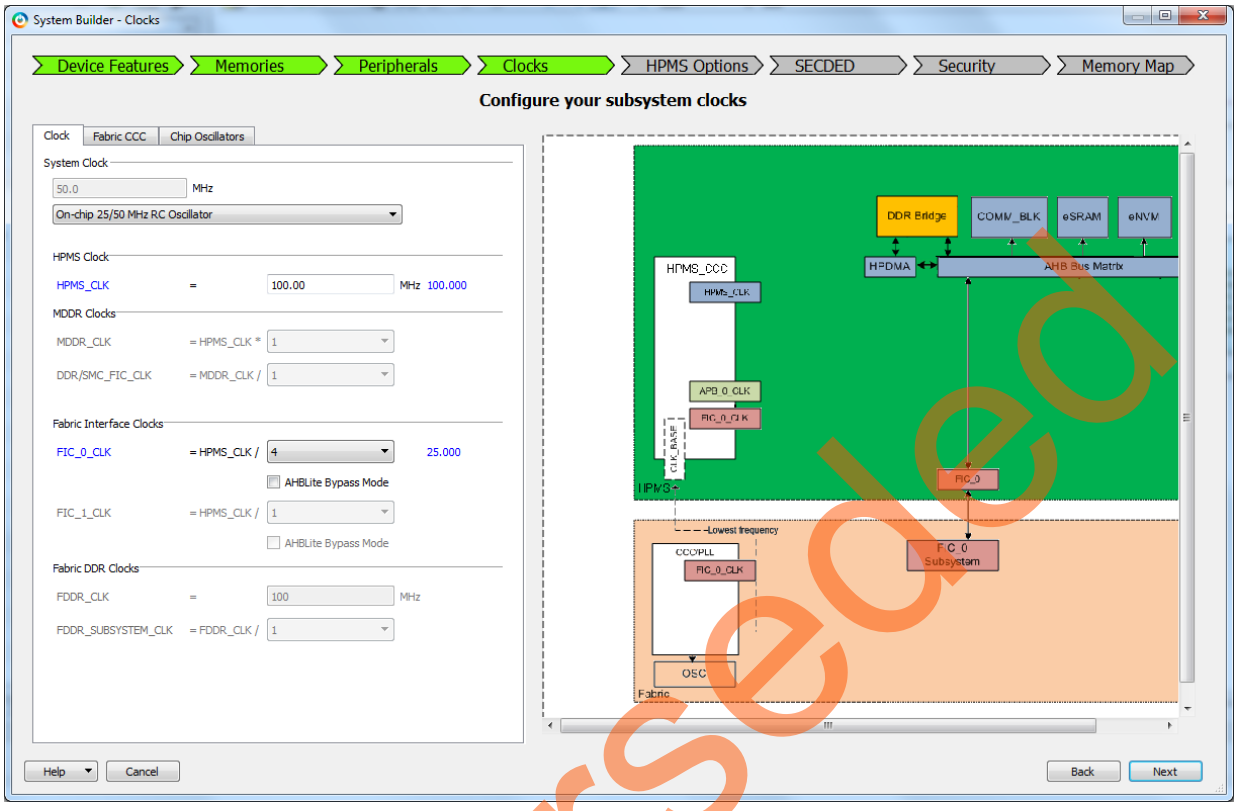

Figure 6 • **IGLOO2 - Clocks Page**

For more information about how to generate a complete system builder component for IGLOO2, refer to *IGLOO2 System Builder User Guide*.

System Builder Configuration for SmartFusion2

1. In the **Device Features** page, ensure that **MSS On-chip Flash Memory (eNVM)** check box is checked, as shown in Figure 7.

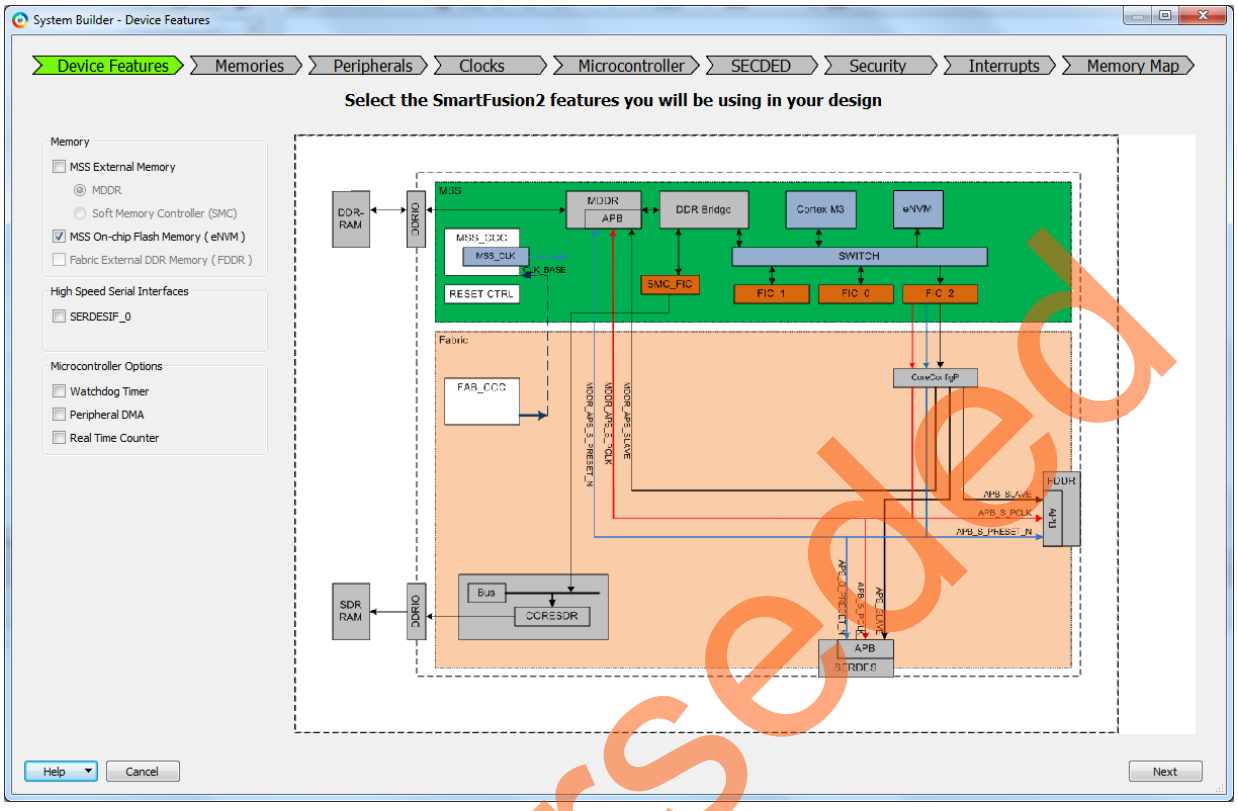

Figure 7 • **SmartFusion2 - Device Features Page**

2. In the **Memories** page, add **zeros_client** and **dummy_client**, as shown in Figure 8.

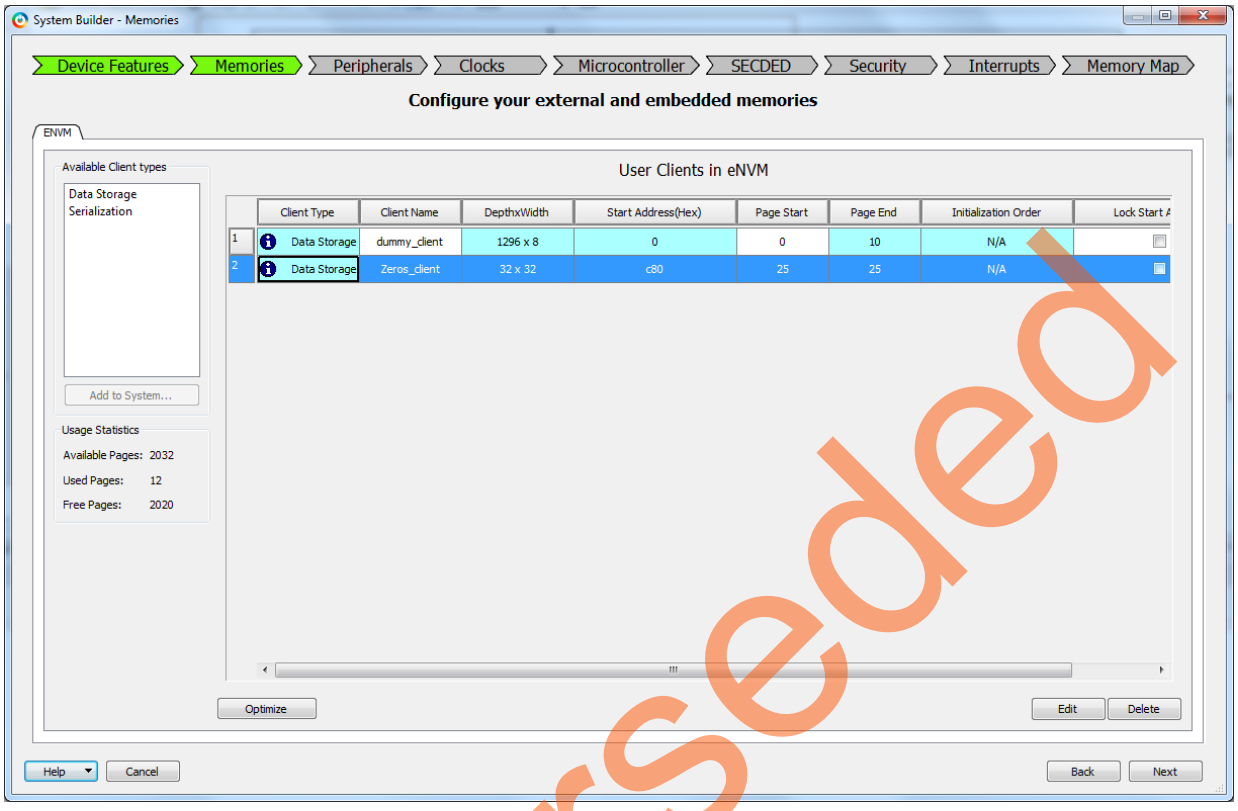

Figure 8 • **SmartFusion2 - Memories Page**

3. In the **Peripherals** page, add **MSS FIC_0 - Fabric Master Subsystem** to provide the AHBL master interface to the user logic, as highlighted in Figure 9.

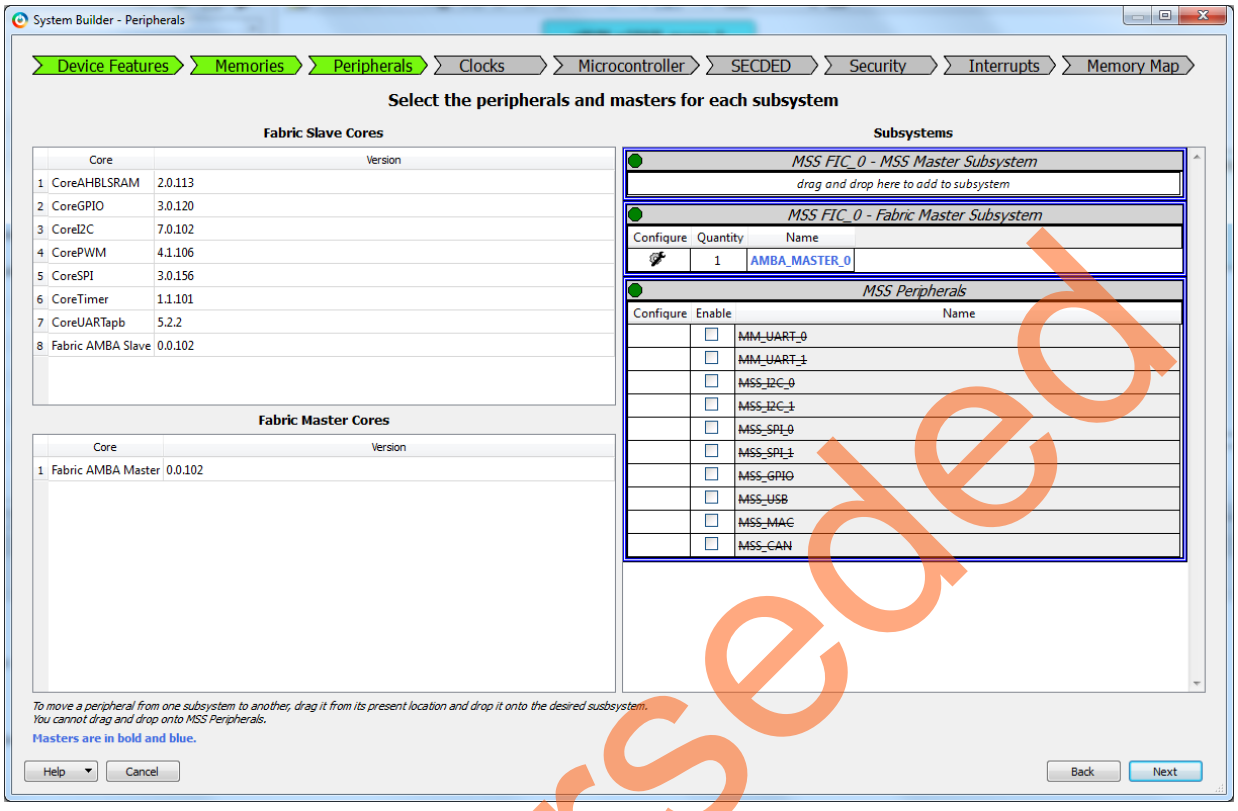

***Figure 9* • SmartFusion2 - Peripherals Page**

Figure 10 shows the clocks configuration page for SmartFusion2.

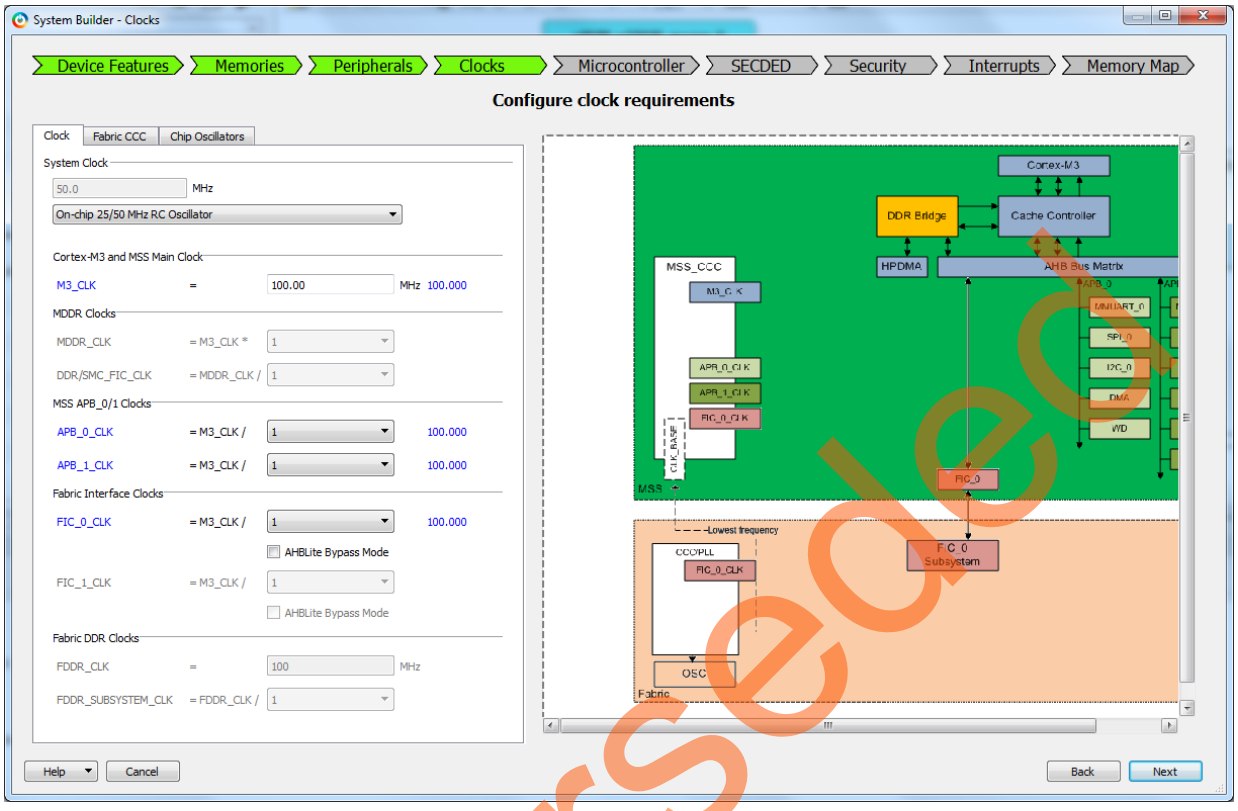

***Figure 10* • SmartFusion2 - Clocks Page**

For more information about how to generate a complete system builder component for SmartFusion2, refer to *SmartFusion2 System Builder User Guide*.

TPSRAM IP Configuration

In SmartDesign, TPSRAM IP is configured as Write Port 32 (Depth) X 32 (Width) and Read port as 32 (Depth) X 32 (Width). Figure 11 shows the TPSRAM IP configuration.

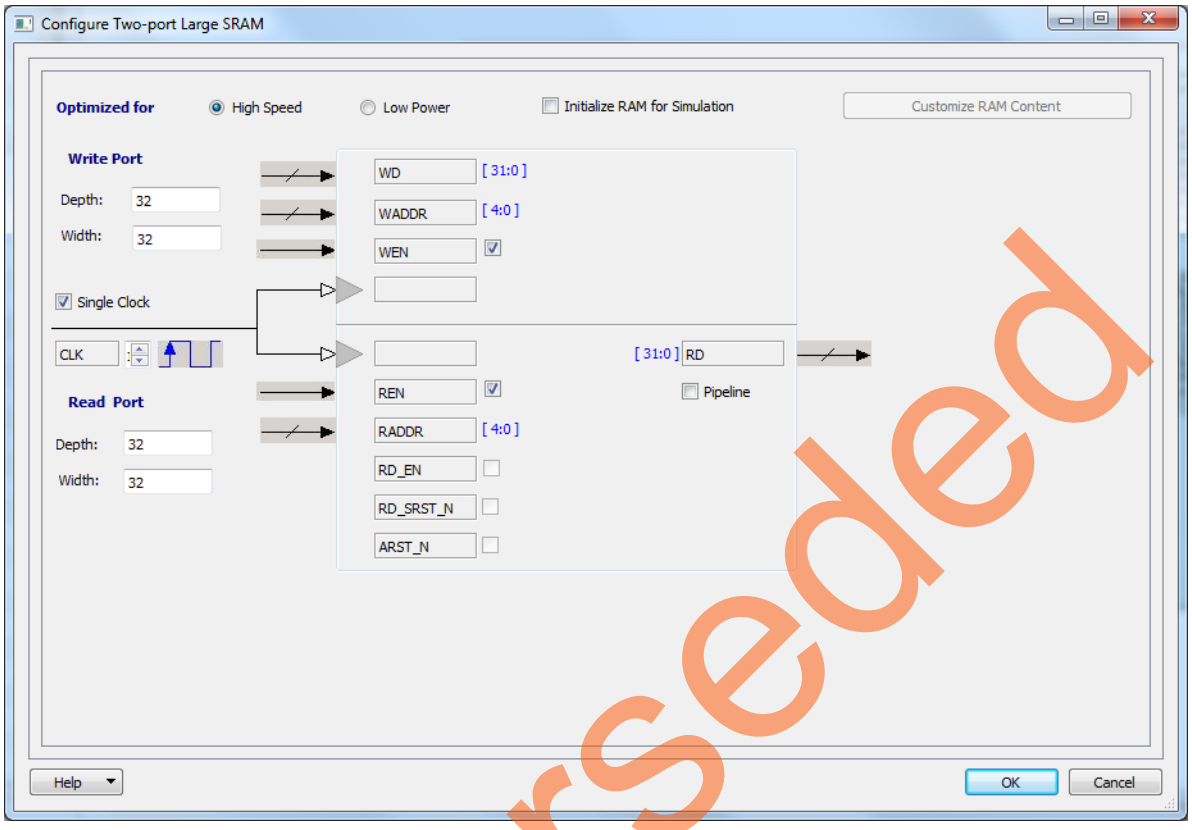

***Figure 11* • TPSRAM Configuration**

Simulation

This section describes the design simulation.

eNVM Simulation

To perform the eNVM write and read simulation operation, provide "1" to "0" transition on the start_envm input signal.

Figure 12 shows the eNVM write command sequence. For more information about this, see "Appendix B" on page 26.

.

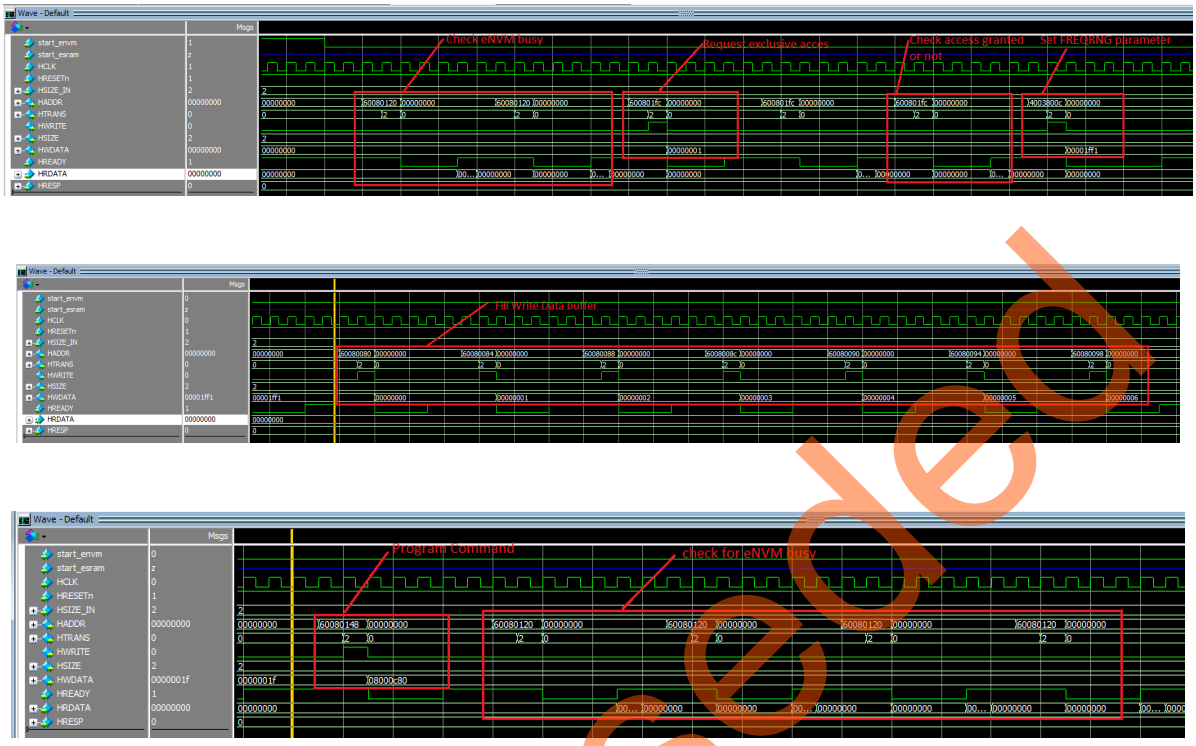

Figure 12 • **eNVM Write Command Sequence**

Figure 13 shows the eNVM read simulation (AHB read operation).

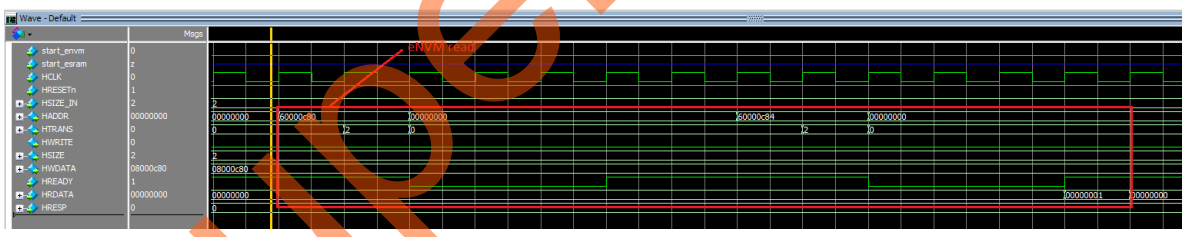

Figure 13 • **eNVM Read Simulation**

Figure 14 shows the release exclusive access to eNVM.

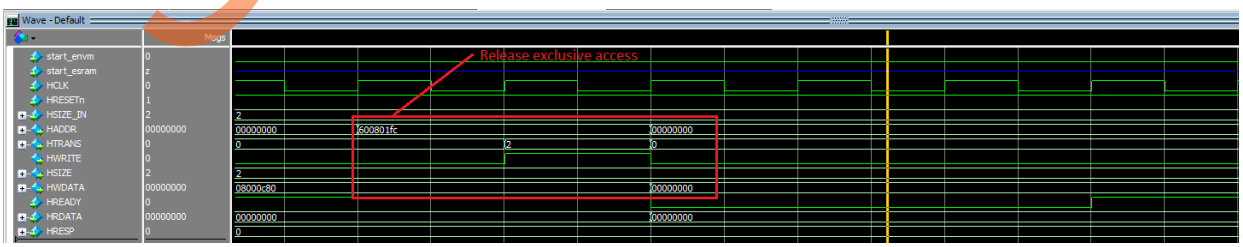

Figure 14 • **Release Exclusive Access to eNVM**

eSRAM Simulation

To perform the eSRAM write and read simulation operation, provide "1" to "0" transition on the start_esram input signal.

Figure 15 shows the eSRAM write simulation (AHB write operation).

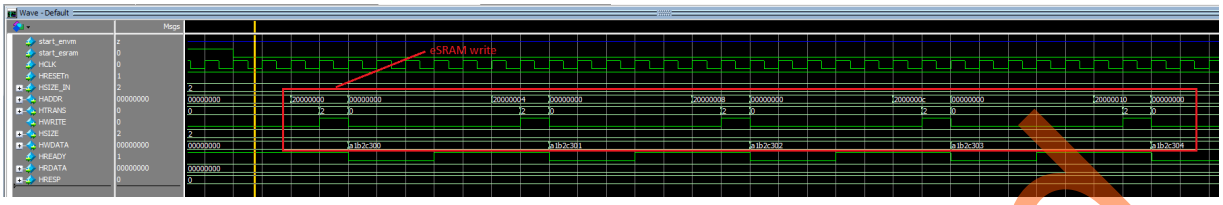

Figure 15 • **eSRAM Write Simulation**

Figure 16 shows the eSRAM read simulation (AHB read operation).

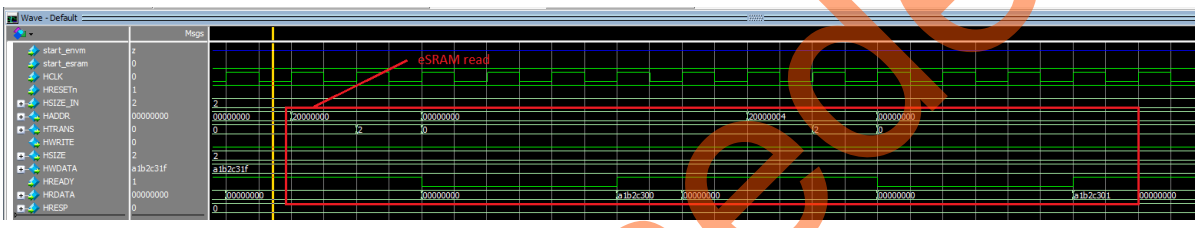

Figure 16 • **eSRAM Read Simulation**

Setting Up the Design

The following steps describe how to setup the hardware demo for IGLOO2 Evaluation Kit board:

1. Connect the jumpers on the IGLOO2 Evaluation Kit board as perTable 2.

Table 2 shows the jumper settings.

Table 2 • **Jumper Settings**

Jumper	Pin (From)	Pin (To)	Comments
J22	1	2	default
J23	1	2	default
J24	1	2	default
J8	1	2	default
J3	1	2	default

Note: Ensure that the power supply switch SW7 is switched off while connecting the jumpers on the IGLOO2 Evaluation Kit.

2. Connect the power supply to the J6 connector.
3. Switch on the power supply switch SW7.
4. Connect the FlashPro4 programmer to the J5 connector of the IGLOO2 Evaluation Kit board.

The following steps describe how to setup the hardware demo for SmartFusion2 Evaluation Kit board:

1. Connect the jumpers on the SmartFusion2 Evaluation Kit board as perTable 3.

Table 3 shows the jumper settings.

Table 3 • **Jumper Settings**

Jumper	Function	Default Setting
J23	Selects switch-side Mux inputs of A or B to the line side	—
	Pin 1-2 (Input A to the line side) that is on board 125 MHz differential clock oscillator output is routed to the line side	Closed
	Pin 2-3 (Input B to the line side) that is external clock required to source through SMA connectors to the line side	Open
J22	Selects the output enables control for the line side outputs	—
	Pin 1-2 (line side output enabled)	Closed
	Pin 2-3 (line side output disabled)	Open
J24	Provides VBUS supply to USB when using in Host mode	Open
J8	JTAG selection jumper to select between RVI header or FP4 header for application debug	—
	Pin 1-2 FP4 for SoftConsole/FlashPro	Closed
	Pin 2-3 RVI for Keil ULINK™/IAR J-Link®	Open
	Pin 2-4 to remotely toggle JTAG_SEL signal using GPIO capability of FT4232 chip	Open
J3	Selects SW2 input or ENABLE_FT4232 signal from FT4232H chip	—

Note: Ensure that the power supply switch SW7 is switched off while connecting the jumpers on the SmartFusion2 Evaluation Kit.

2. Connect the power supply to the J6 connector.
3. Switch on the power supply switch SW7.
4. Connect the FlashPro4 programmer to the J5 connector of the SmartFusion2 Evaluation Kit board.

Figure 17 shows the demo setup for SmartFusion2 Evaluation Kit / IGLOO2 Evaluation Kit.

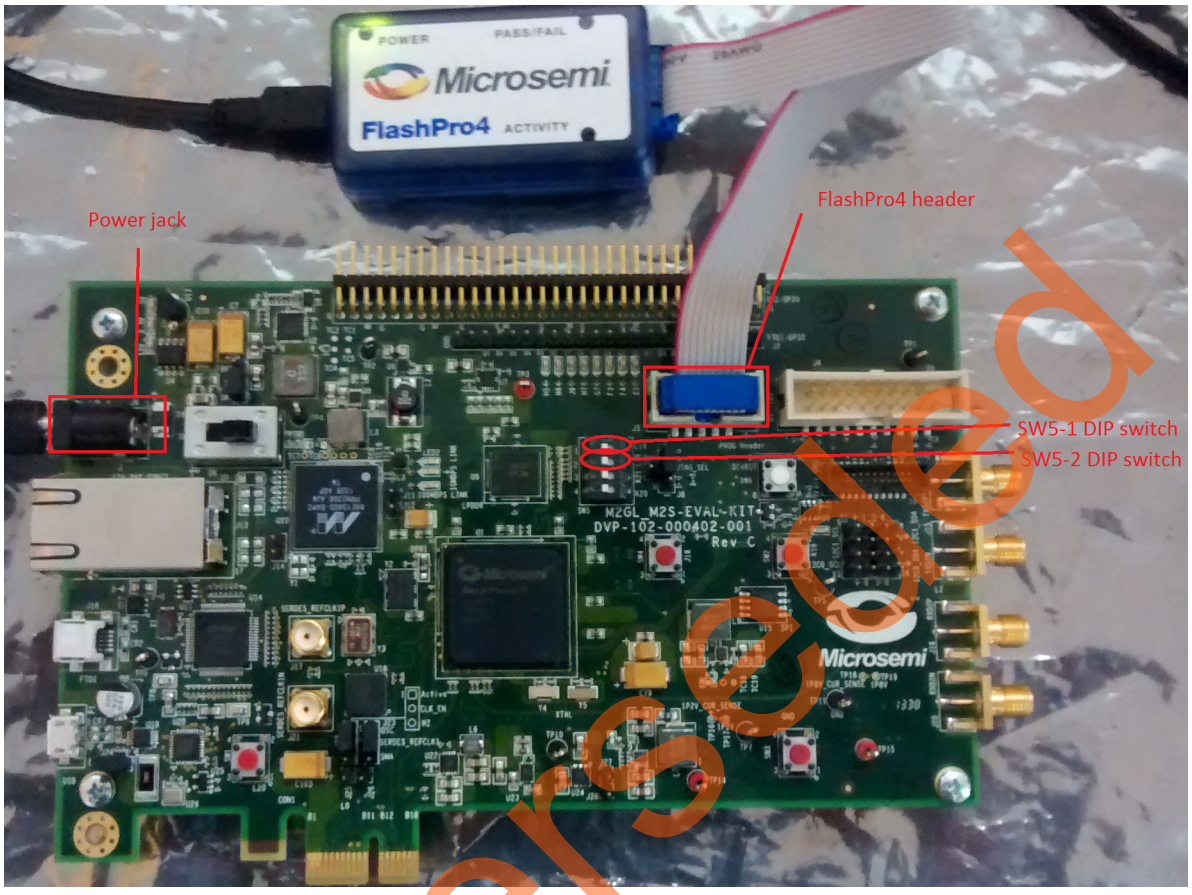

Figure 17 • **Accessing eNVM and eSRAM Demo Setup**

Programming the Demo Design

The following steps describe how to program the demo design:

1. Download the demo design from the following link:
   http://soc.microsemi.com/download/rsc/?f=m2s_m2gl_ac429_accessing_envm_esram_fpga_liberov11p4_an_df
2. Switch on the power supply switch SW7.
3. Launch the FlashPro software.
4. Click **New Project** (see Figure 18).
5. In the **New Project** window, enter the project name as **eNVM_eSRAM_RW**.
6. Click **Browse** and navigate to the location where the project needs to be saved.
7. Select **Single device** as the Programming mode.
8. Click **OK** to save the project.

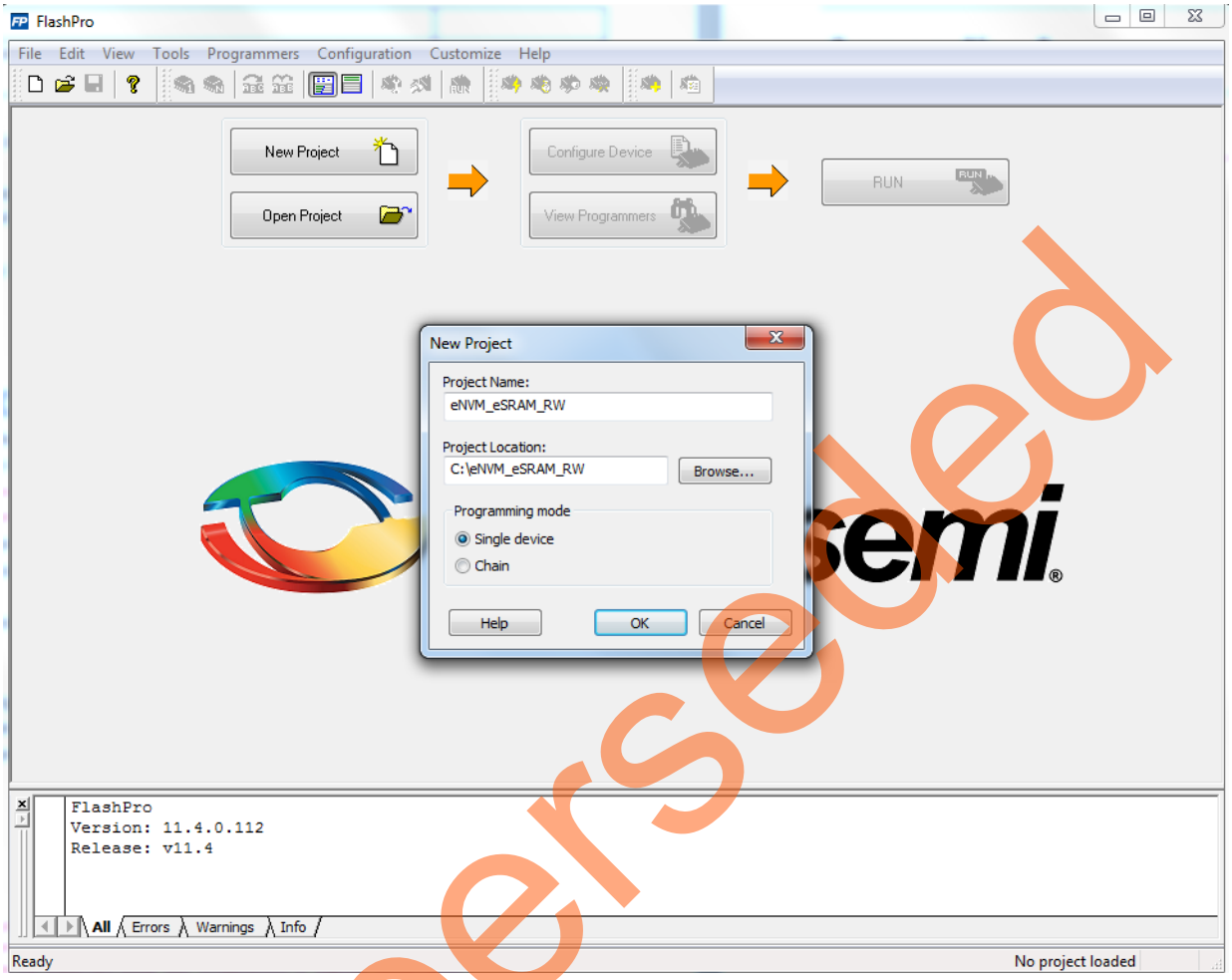

Figure 18 • **FlashPro - New Project**

Configuring the Device

The following steps describe how to configure the device:

1. Click **Configure Device** on the FlashPro GUI (see Figure 19).
2. Click **Browse** and navigate to the location where the **ENVM_RW_Fabric_Top.stp** programming file is located, and select the file.

   The default location of the programming file is shown below:

   – For SmartFusion2: *<download folder>\eSRAM_eNVM_RW_Fabric\Programming_file\SF2\eSRAM_eNVM_access_top.stp*

   – For IGLOO2: *<download folder>\eSRAM_eNVM_RW_Fabric\Programming_file\IGL2\eSRAM_eNVM_access_top.stp*

3. Select **Basic** as Mode and **PROGRAM** as Action (highlighted in Figure 19).

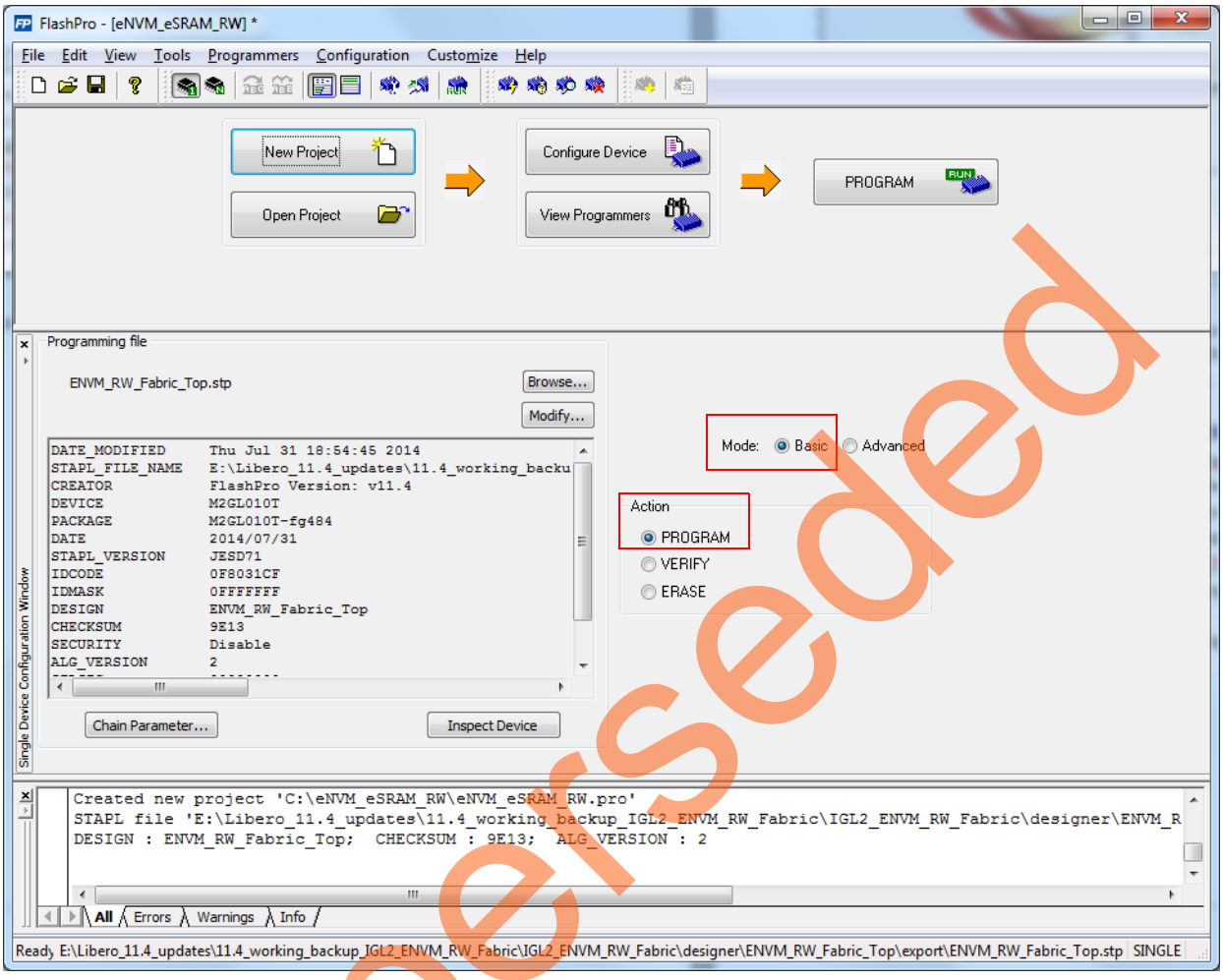

***Figure 19* • FlashPro - Project Configuration**

4. Click **PROGRAM** to start programming the device. Wait until the Programmer Status is changed to **RUN PASSED** (highlighted in Figure 20).

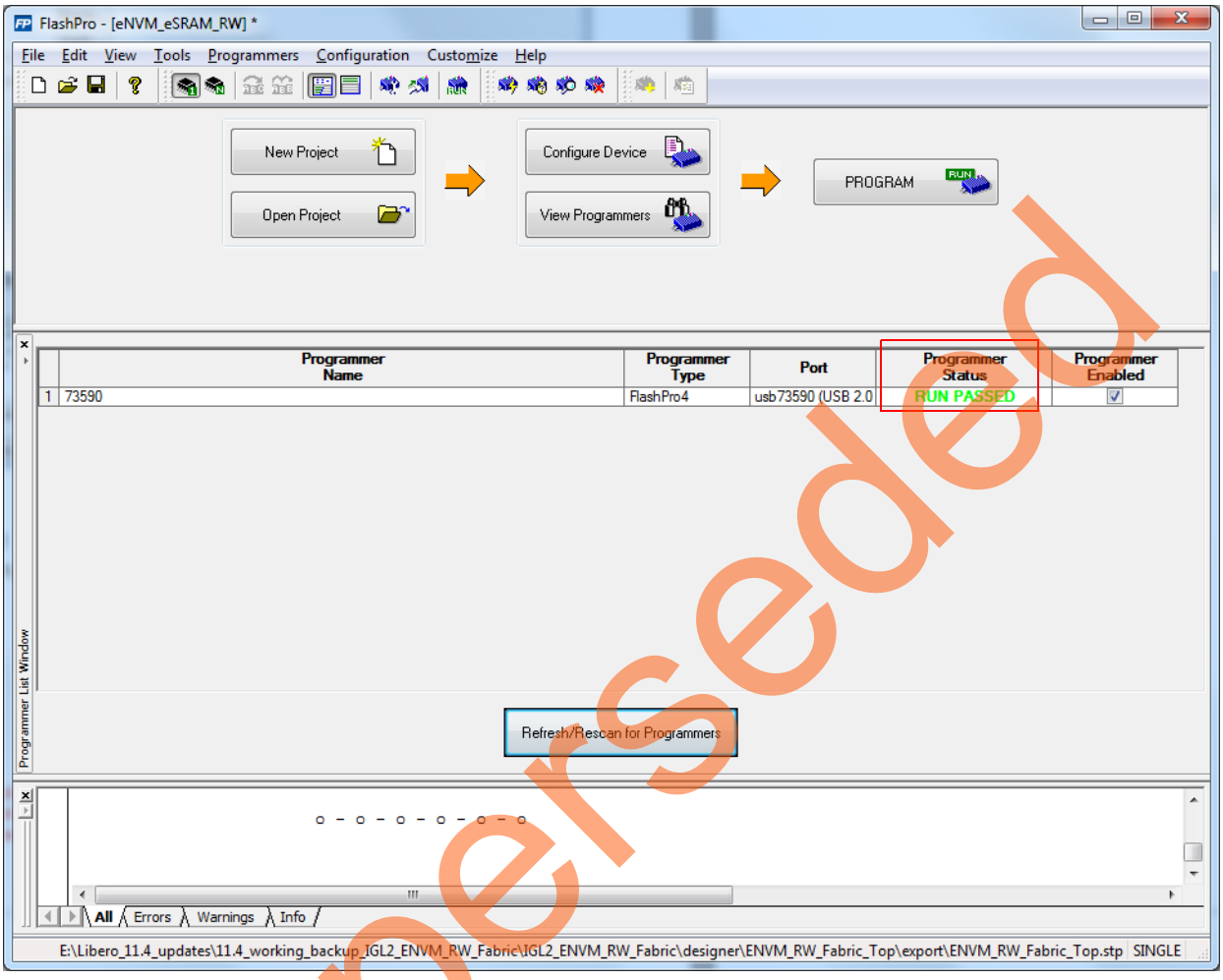

Figure 20 • **FlashPro - RUN PASSED**

Running the Design

eNVM Write and Read Operations

The following steps describe how to perform read and write operations from the eNVM:

1. Make "**1**" to "**0**" transition using the SW5-1 DIP switch (FPGA pin number: L19) to perform write and read operation from the eNVM. An incremental data starting from 0x00000000 to 0x0000001F is written to page 25 of the eNVM, and the data is read back from the eNVM and stored in the Fabric SRAM.
2. The SmartDebug tool in Libero SoC verifies the write and read operations.
   The following steps describe how to verify the write and read operations:

a. In Libero SoC, go to **Designflow > DebugDesign > SmartDebugDesign**. Right-click and open the **SmartDebug** window (see Figure 21).

Figure 21 shows the **SmartDebug** window.

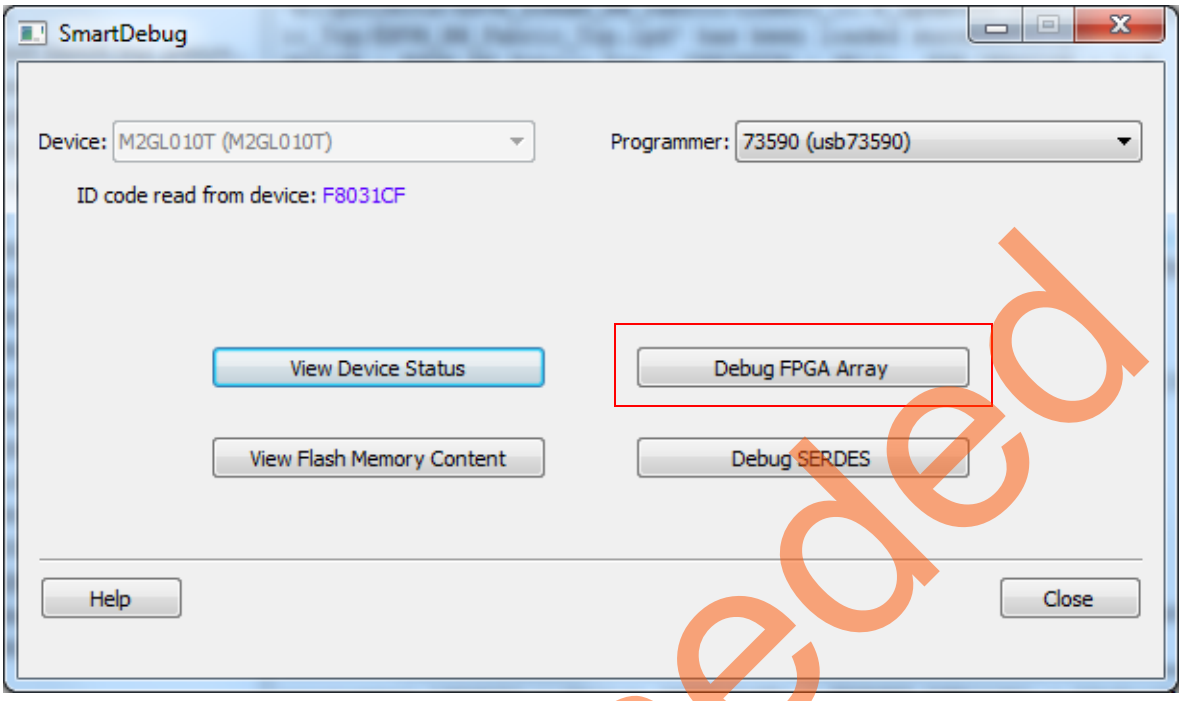

Figure 21 • **SmartDebug Window**

b. Click **Debug FPGA Array** (highlighted in Figure 21) and open the **Debug FPGA Array** window (see Figure 22).

c. Browse for **eNVM_eSRAM_RW_Top_debug.txt** file and click **Read Block** (highlighted in Figure 22) in **Memory Blocks** tab.

Figure 22 shows the **Debug FPGA Array** window.

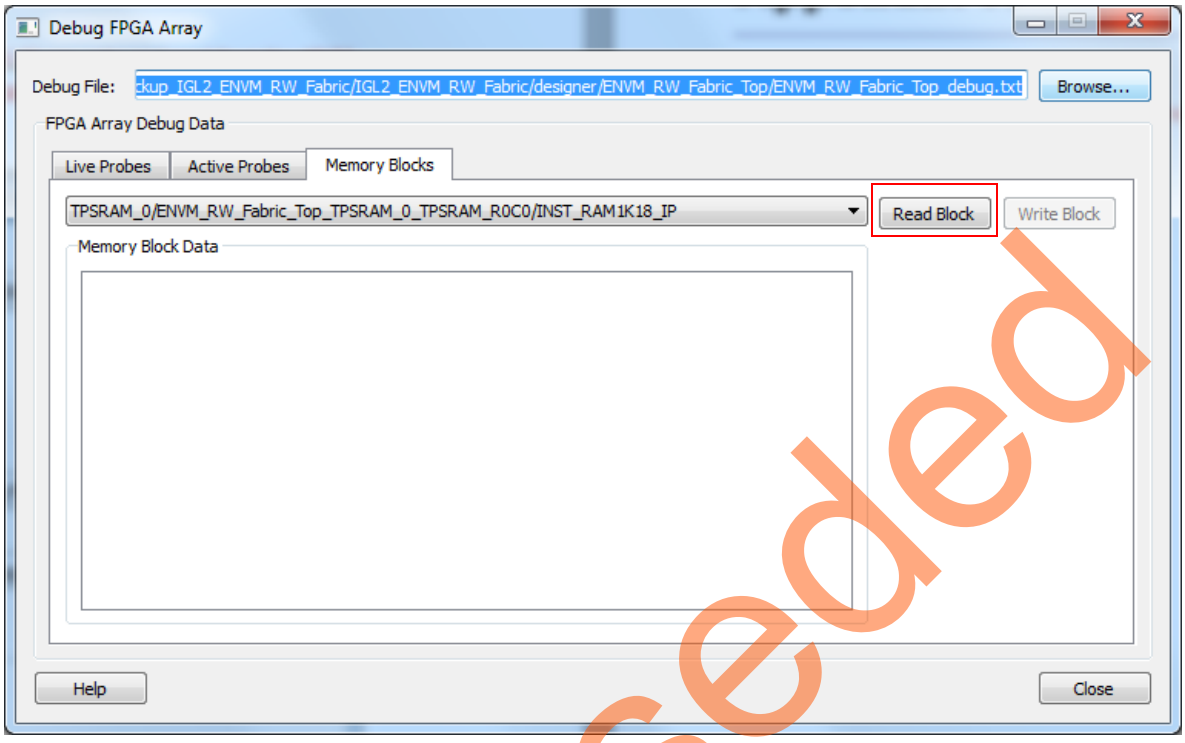

***Figure 22* • Debug FPGA Array Window**

Figure 23 shows the displayed fabric SRAM (TPSRAM) memory content. It shows that eNVM Write and Read is successful.

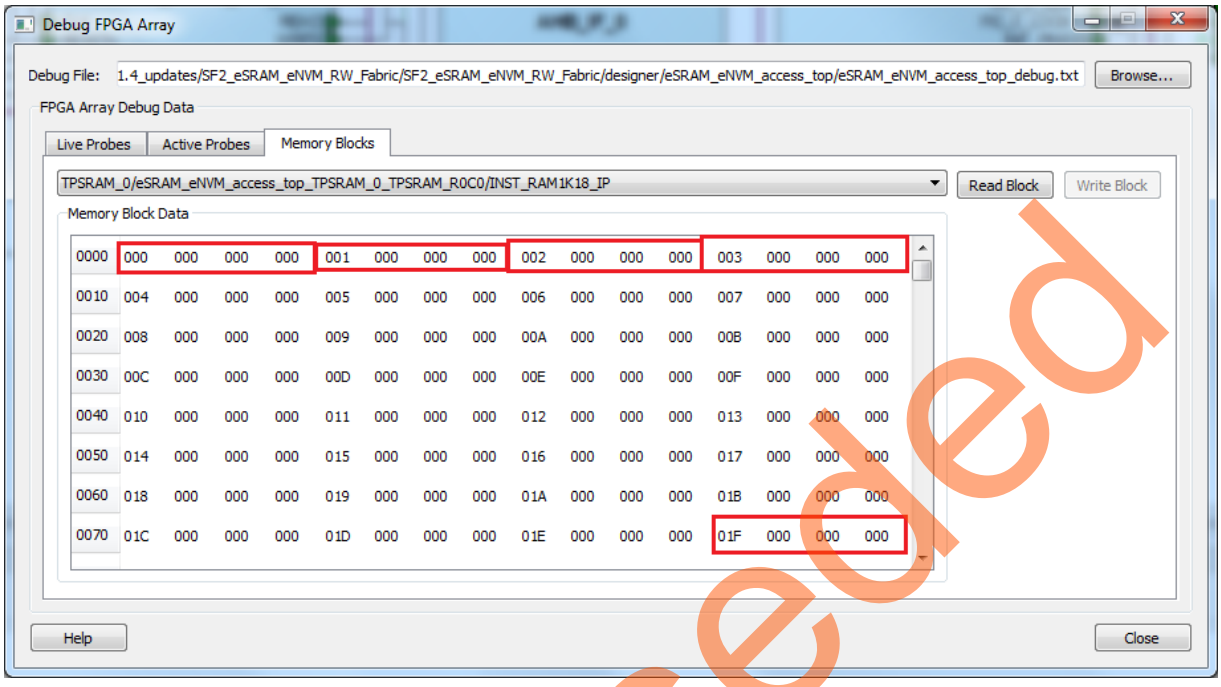

***Figure 23* • Debug FPGA Array - Memory Blocks**

d. Close the **Debug FPGA Array** window.

e. Close the **SmartDebug** window.

For more information about how to interpret the Memory Block Data, go to **Help > Help Topics > Debug Design > SmartFusion2 and IGLOO2 SmartDebug > Debug FPGA Array - Memory Blocks Tab**.

eSRAM Write and Read Operation

The following steps describe how to perform read and write operations from the eSRAM:

1. Make "**1**" to "**0**" transition using the SW5-2 DIP switch (FPGA pin number: L18) to write and read from the eSRAM.An incremental data pattern starting from 0xA1B2C300 to 0xA1B2C31F is written to 32 locations in eSRAM address starting from 0x20000000 to 0x20000080, and the data is read back from that eSRAM and stored in the Fabric SRAM.
2. To verify the write and read operations using the SmartDebug tool in Libero SoC, follow the steps (Step a to Step e) from the eNVM write and read operation.

Figure 24 shows the displayed fabric SRAM (TPSRAM) memory content. It shows that eSRAM write and read operations are successful.

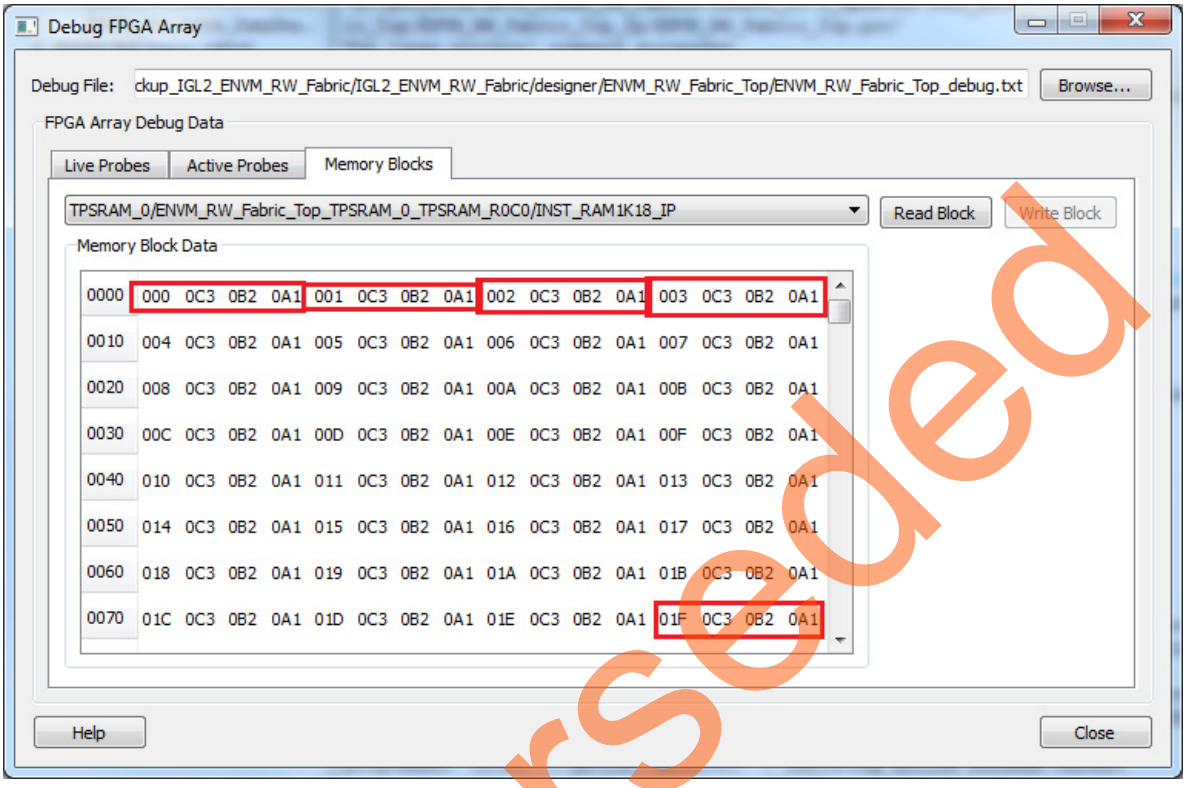

***Figure 24* • Debug FPGA Array - Memory Blocks**

For more information about how to interpret the Memory Block Data, go to **Help > Help Topics > Debug Design > SmartFusion2 and IGLOO2 SmartDebug > Debug FPGA Array - Memory Blocks Tab**.

Conclusion

This application note describes how to write and read to the eNVM and eSRAM from FPGA fabric. This application note also describes the usage of SmartDebug tool to verify the design functionality.

Appendix A – Design Files

Download the design files from the Microsemi SoC Products Group website:

http://soc.microsemi.com/download/rsc/?f=m2s_m2gl_ac429_accessing_envm_esram_fpga_liberov11p4_an_df

The design file consists of Libero Verilog projects and programming files (*.stp) for SmartFusion2 Evaluation Kit and IGLOO2 FPGA Evaluation Kit. Refer to the Readme.txt file included in the design file for the directory structure and description.

Appendix B

eNVM Write Operation

The following steps describe how to perform the eNVM write operation:

1. Wait for Bit 0 of status register (address: 0x60080120) to become 1. If this bit is 0, it implies that the eNVM is busy.
2. Request exclusive access to the eNVM. This is required to ensure that no two Masters can write to the eNVM at the same time. This is done by writing 0x1 to the REQACCESS register (address: 0x600801FC).
3. Check if request has been granted, by reading back from the REQACCESS register. On read back, check for Bit 2 (counting up from 0):
   – if Bit 2 is 1, it implies that the request is successful.
   – If Bit 2 is 0, the request for exclusive access is denied. The eNVM cannot be written at this time
4. Write 0x000001FF1 to ENVM_CR register (address: 0x4003800C). This changes the FREQRNG register field to 15 decimal.
5. Writes to the eNVM are buffered. First, write data into the write data buffer (WDB — It is a byte addressable 1024-bit buffer. Its base address is 0x60080080 for eNVM_0 and 0x600C0080 for eNVM_1), and then use a single command to commit (aka program) data into one page of the eNVM. Write the data into the WDB.
6. Compute the values of bits that needs to be written into the eNVM command register:
   – Bits 31-20 should be 0x080.
   – Bit 19 should be 0x0.
   – Bits 18-7 corresponds to the number of page to be written (for 25th page, the bit field is 000 0000 1100 1).
   – Bits 6-0 should be 0x0.
7. Write the eNVM command register (address: 0x60080148) with the data computed in Step 6. eNVM does not respond to further commands until the write is complete.
8. Release exclusive access to the eNVM by writing 0x0 to the REQACCESS register (address: 0x600801FC).

eNVM Read, eSRAM Write and Read Operations

No special command sequences are required for the eNVM read, eSRAM read, and eSRAM write operations. eNVM read and eSRAM read are performed using AHB read operation, and eSRAM writes are performed using AHB write operations.

List of Changes

The following table lists critical changes that were made in each revision of the document.

Revision*	Changes	Page
Revision 1 (September 2014)	Initial release	NA

*Note: *The revision number is located in the part number after the hyphen. The part number is displayed at the bottom of the last page of the document. The digits following the slash indicate the month and year of publication.*

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**.