AC429
Application Note
SmartFusion2 and IGLOO2 - Accessing eNVM and eSRAM from FPGA Fabric

Microsemi
a Microchip company

Microsemi 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
Email: sales.support@microsemi.com
www.microsemi.com

About Microsemi

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

51900429. 6.0 4/21

Contents

Figures

Tables

1 Revision History

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

1.1 Revision 6.0

Updated the document for Libero SoC v12.6.

1.2 Revision 5.0

The following is a summary of the changes made in this revision.

- Updated the document for Libero SoC v12.5.
- Removed the references to Libero version numbers.

1.3 Revision 4.0

The document was updated for Libero SoC v11.8 software release.

1.4 Revision 3.0

The document was updated for Libero SoC v11.7 software release (SAR 77431).

1.5 Revision 2.0

The following is a summary of the changes in revision 2.0 of this document.

- Updated the document for Libero SoC v11.6 software release (SAR 71803).
- Updated Simulation, page 11 (SAR 64030).

1.6 Revision 1.0

Revision 1.0 was the first publication of this document.

2 SmartFusion2 and IGLOO2 - Accessing eNVM and eSRAM from FPGA Fabric

This application note describes how to access the embedded Non-Volatile Memory (eNVM) and embedded Static Random Access Memory (eSRAM) from the FPGA fabric in SmartFusion®2 System-on-Chip (SoC) FPGA and IGLOO®2 FPGA devices.

2.1 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.

The eNVM can be initialized by the custom logic in the FPGA fabric (fabric master).

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

For more information about eNVM initialization methods, refer to the *AC391: SmartFusion2 SoC FPGA - eNVM Initialization Application Note*.

2.2 Accessing eSRAM from FPGA Fabric

SmartFusion2 SoC FPGA and IGLOO2 FPGA devices have two eSRAM blocks, each of 32 KB, 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 the FPGA fabric (fabric master).

In this application note, the fabric master writes to and reads from 32 eSRAM locations (0x20000000 to 0x20000080).

2.3 Design Requirements

The following table lists the resources required for this application:

Table 1 • **Design Requirements**

Requirement	Version
Operating System	64 bit Windows 7 and 10
Hardware	
SmartFusion2 Security Evaluation Kit or IGLOO2 Evaluation Kit • 12 V adapter • USB A to mini-B cable	• SmartFusion2: Rev D or later • IGLOO2: Rev C or later
Software	
FlashPro Express	**Note:** Refer to the `readme.txt` file provided in the design files for the software versions used with this reference design.
Libero® System-on-Chip (SoC)	
SoftConsole	
Host PC drivers	USB to UART drivers

Note: Libero SmartDesign and configuration screen shots shown in this guide are for illustration purpose only. Open the Libero design to see the latest updates.

2.4 Prerequisites

Before you begin:

1. Download and install Libero SoC (as indicated in the website for this design) on the host PC from the following location:

   https://www.microsemi.com/product-directory/design-resources/1750-libero-soc

2. For demo design files download link:

   http://soc.microsemi.com/download/rsc/?f=m2s_m2gl_ac429_df

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

2.5 Design Description

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

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

During 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 that receives these commands and communicates with the selected memory using the AHB bus matrix through the FIC_0 (fabric interface controller) interface.

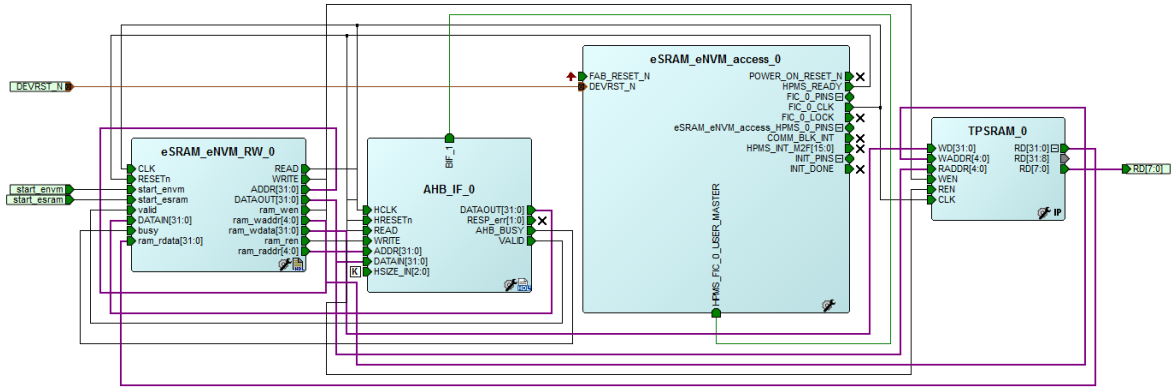
The read operations of the eNVM and the read and write operations of the eSRAM are simple AHB transactions. eNVM write requires a separate set of command sequences. For more information about this, refer to Appendix 2: eNVM and eSRAM Write/Read Operations, page 22.

2.6 Hardware Implementation

The hardware implementation involves configuring the device features, memory, peripherals, and clock pages using System Builder. Configuring the TPSRAM IP and adding fabric logic are done at the top-level using SmartDesign.

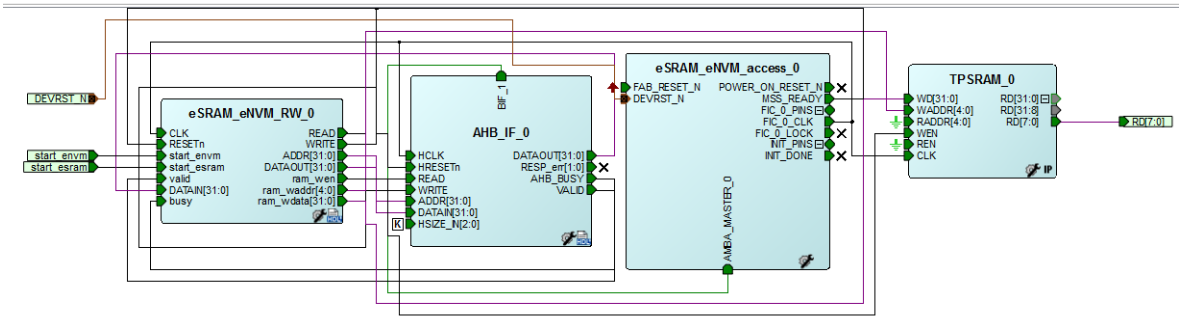
The following figure shows the top-level hardware design in SmartDesign for IGLOO2.

Figure 1 • **Top-Level SmartDesign for IGLOO2**



The following figure shows the top-level hardware design in SmartDesign for SmartFusion2.

Figure 2 • **Top-Level SmartDesign for SmartFusion2**



2.6.1 SmartDesign Components

The top-level SmartDesign has four components (as shown in the preceding figures):

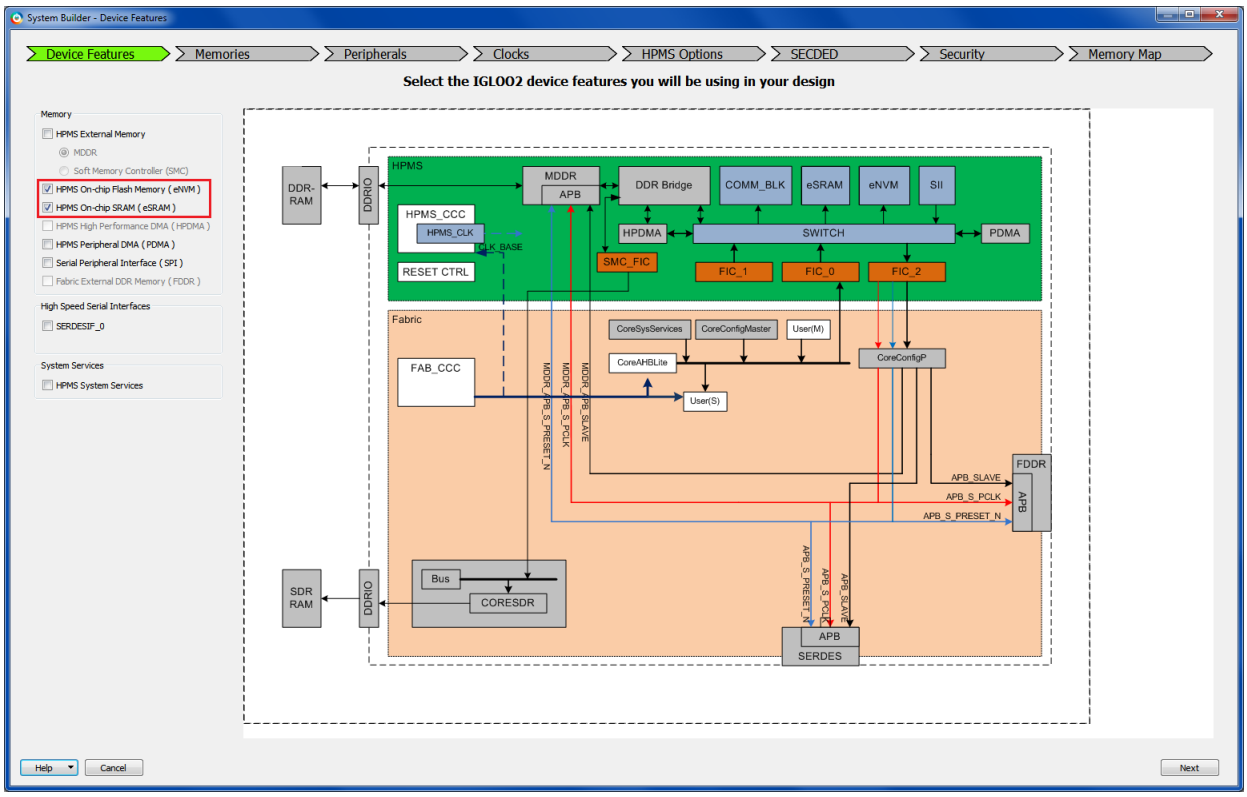
- eSRAM_eNVM_access_0: System Builder generated component.
- AHB_IF_0: User generated RTL FSM, which performs AHB master function. This FSM interacts with the eNVM and eSRAM controller using the AHB switch matrix through the FIC_0 interface.
- eSRAM_eNVM_RW_0: User generated RTL FSM, which takes inputs 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.

2.6.1.1 System Builder Configuration for IGLOO2

For configuring IGLOO2 using System Builder, follow these steps:

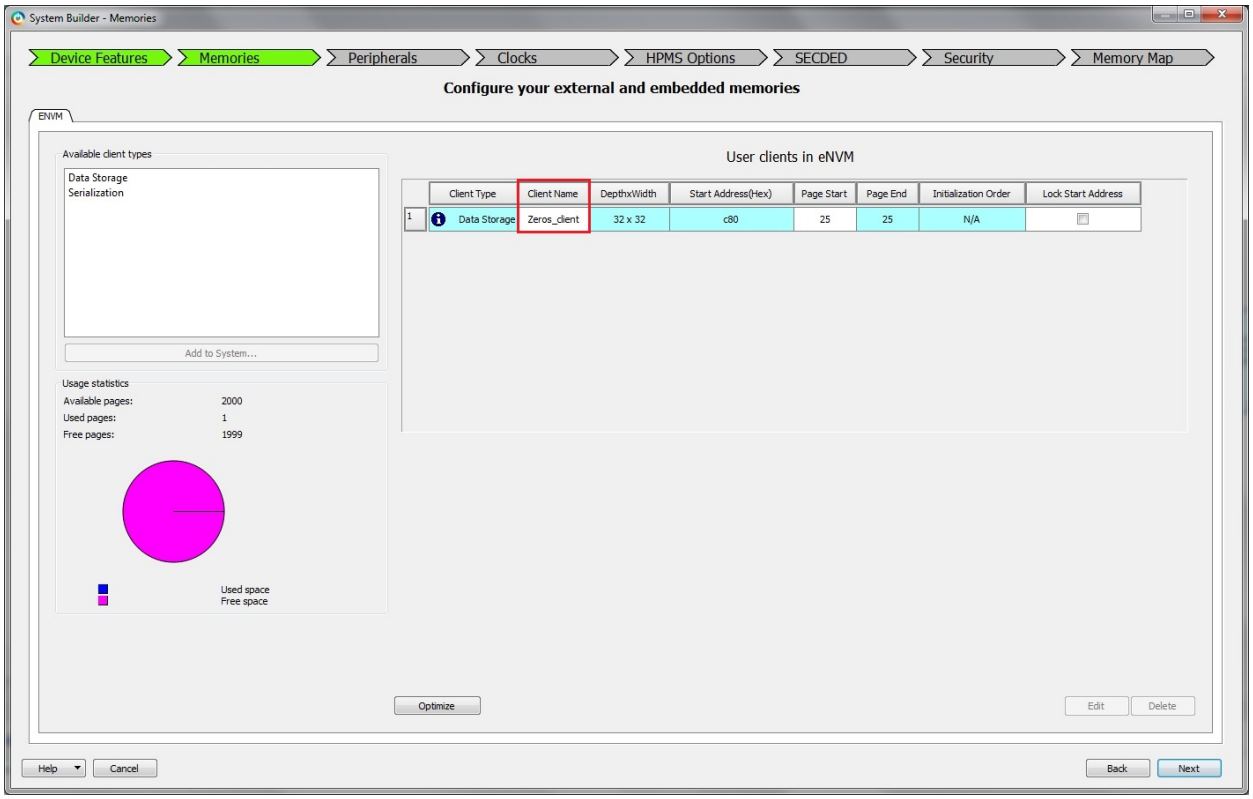
1. In the **Device Features** page, ensure **HPMS On-chip Flash Memory (eNVM)** and **On-chip SRAM (eSRAM)** checkboxes are checked, as shown in the following figure.

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



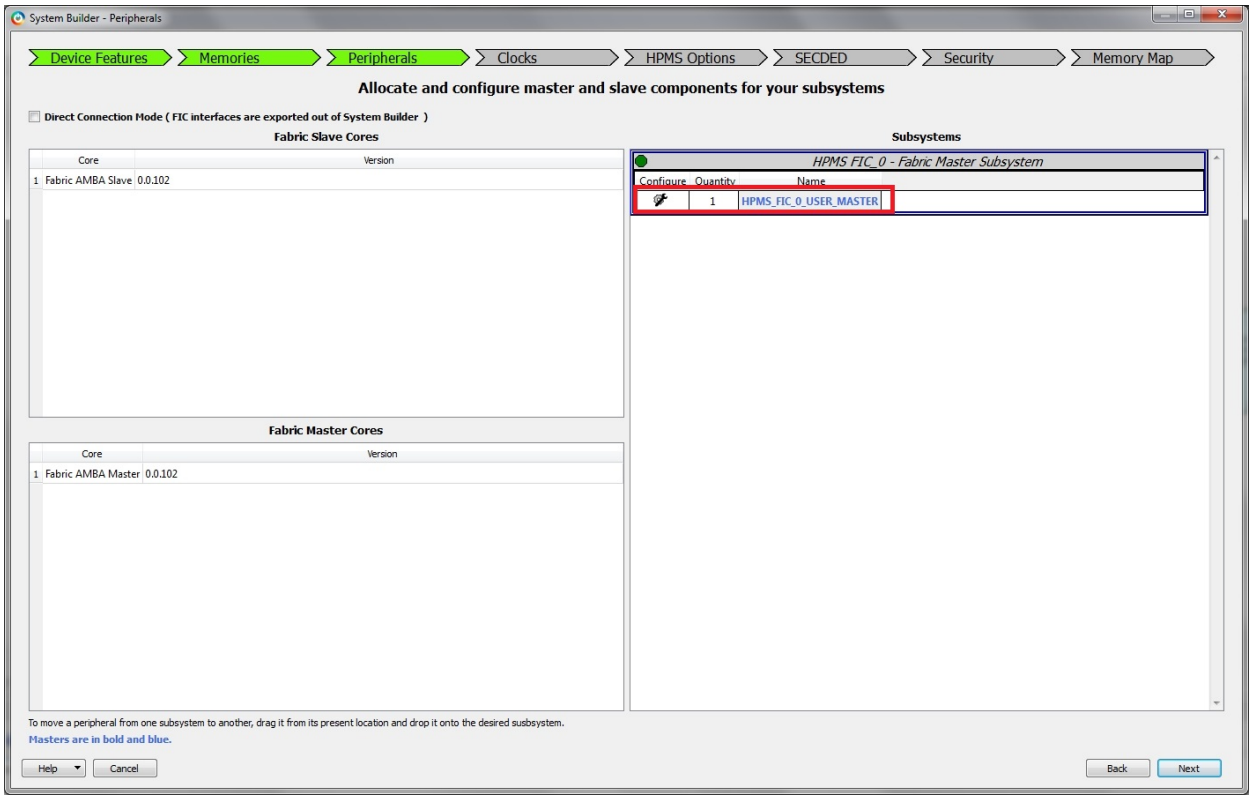
2. In the **Memories** page, add **Zeros_client** to initialize the eNVM with zeros, as shown in the following figure.

Figure 4 • **IGLOO2—Memories Page**



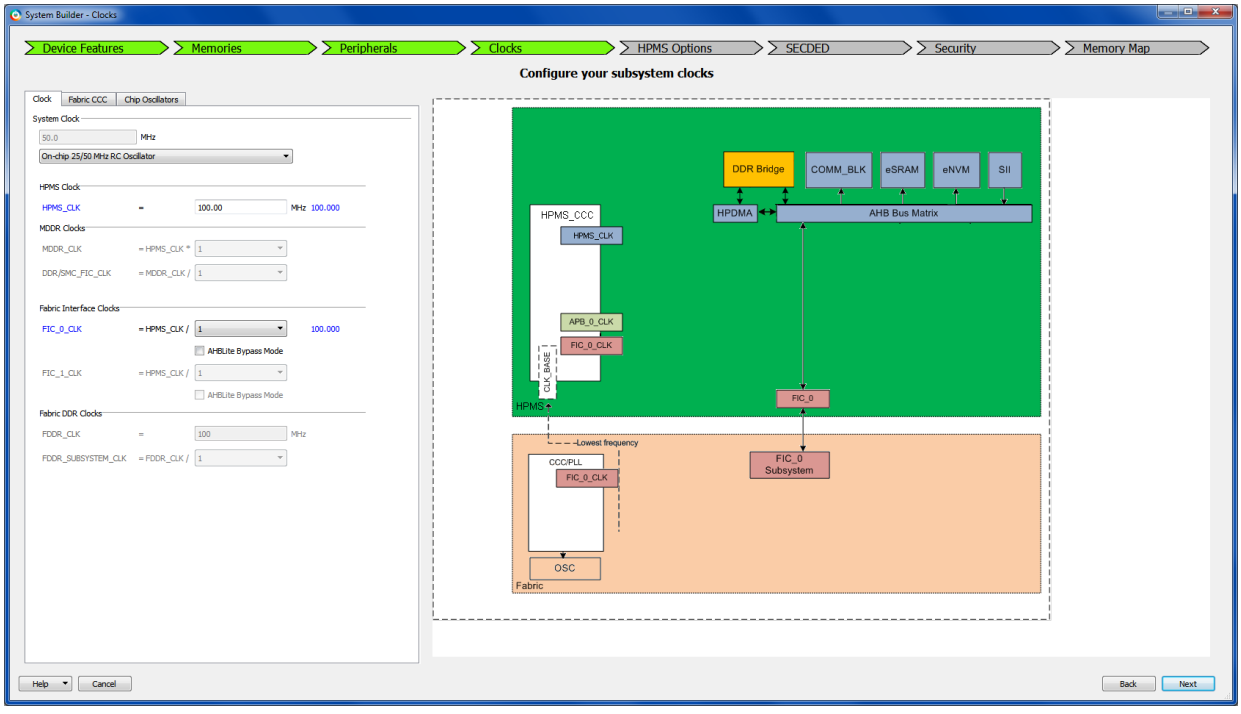
3. In the **Peripherals** page, add **HPMS FIC_0_USER_MASTER** under **Subsystems** to provide the AHBL master interface to the user logic, as shown in the following figure.

Figure 5 • **IGLOO2—Peripherals Page**



4. In the **Clocks** page, select **On-chip 25/50 MHz RC Oscillator** as **System Clock** and **100** MHz as **HPMS_CLK** frequency, as shown in the following figure.

Figure 6 • **IGLOO2—Clocks Page**



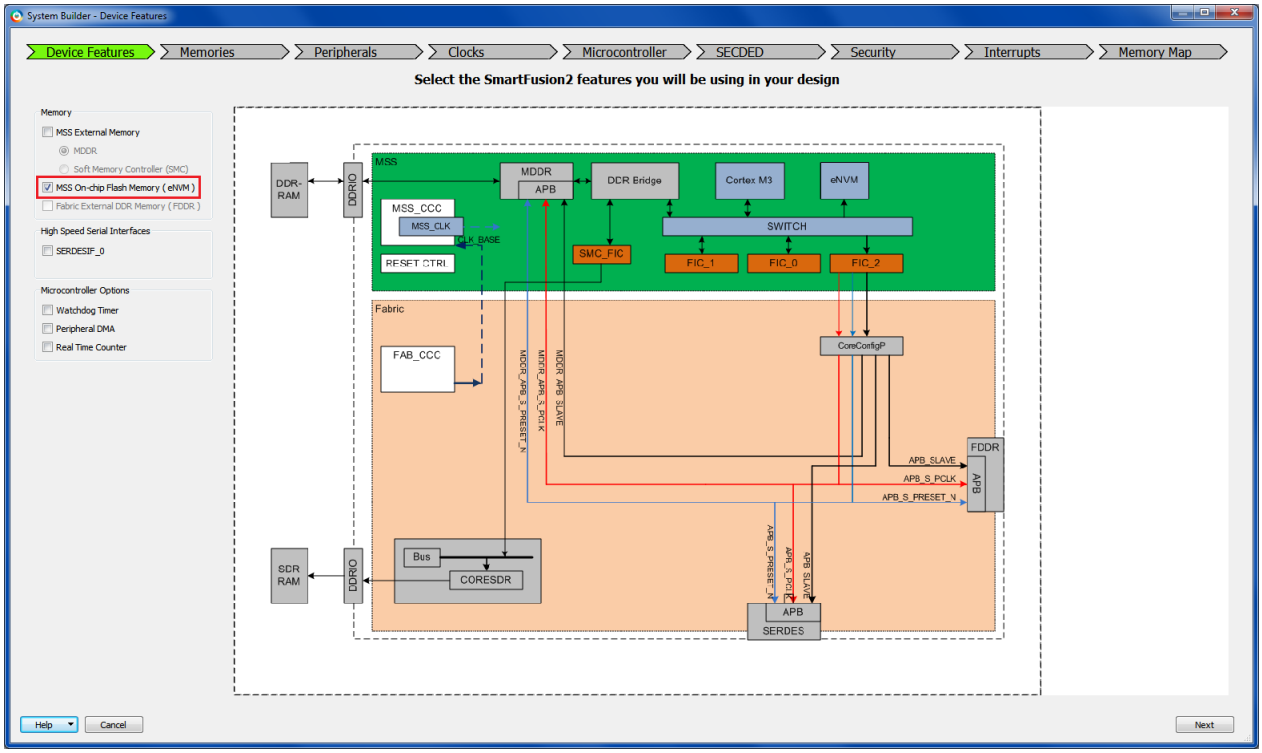
For more information about how to generate a complete System Builder component for IGLOO2, refer to the *IGLOO2 System Builder User Guide*.

2.6.1.2 System Builder Configuration for SmartFusion2

For configuring SmartFusion2 using System Builder, follow these steps:

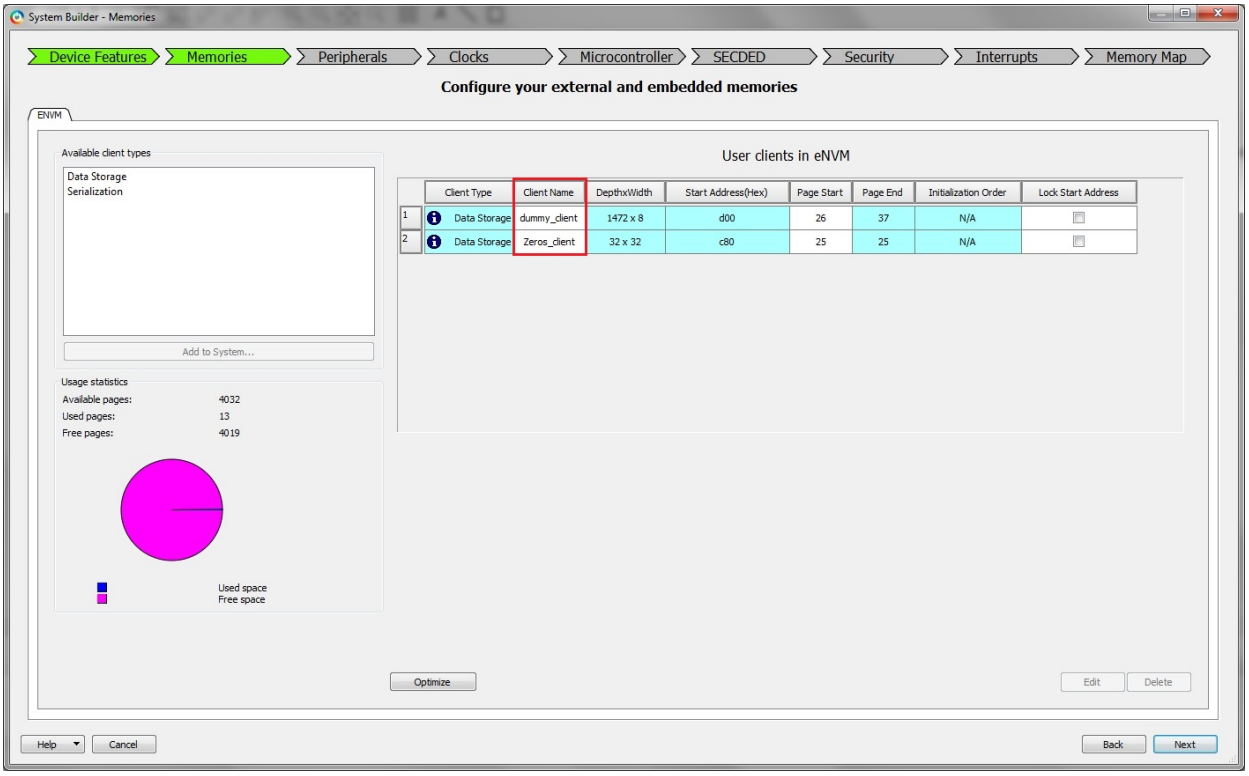
1. In the **Device Features** page, ensure **MSS On-chip Flash Memory (eNVM)** check box is selected, as shown in the following figure.

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



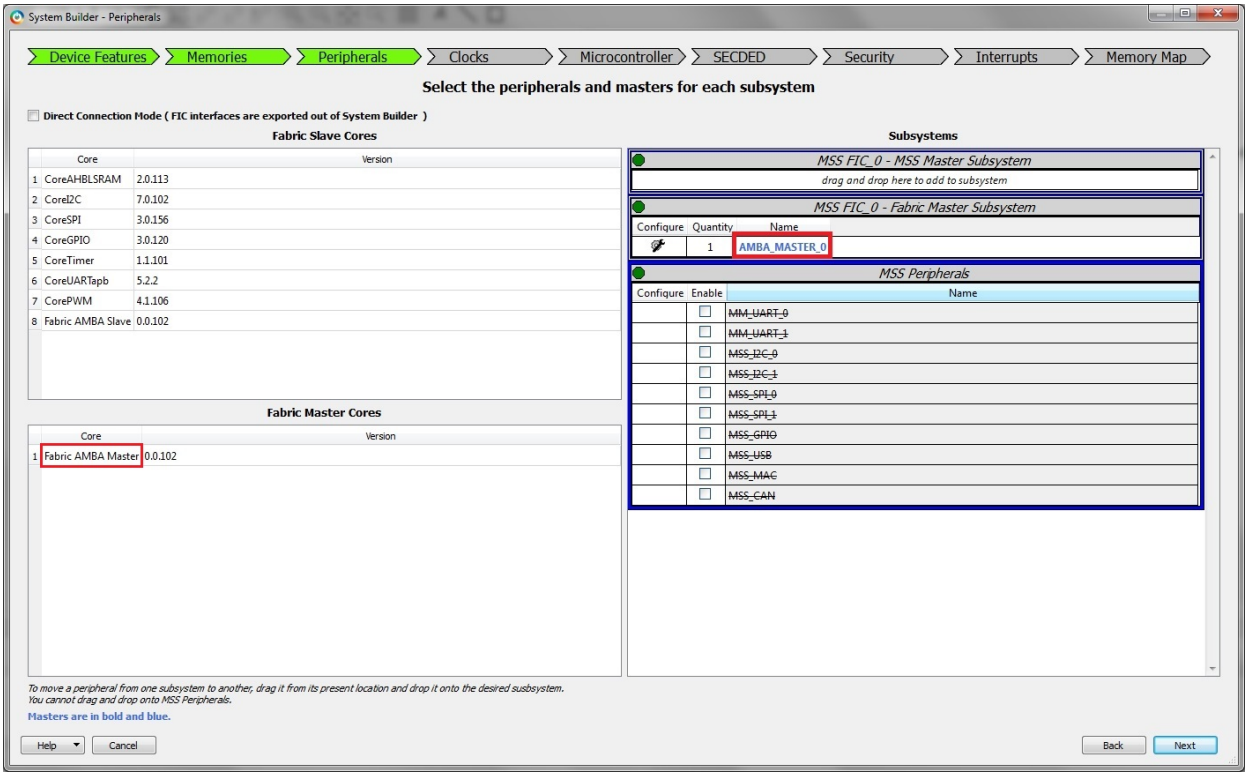
2. In the **Memories** page, add **Zeros_client** and **dummy_client**, as shown in the following figure.

Figure 8 • **SmartFusion2—Memories Page**



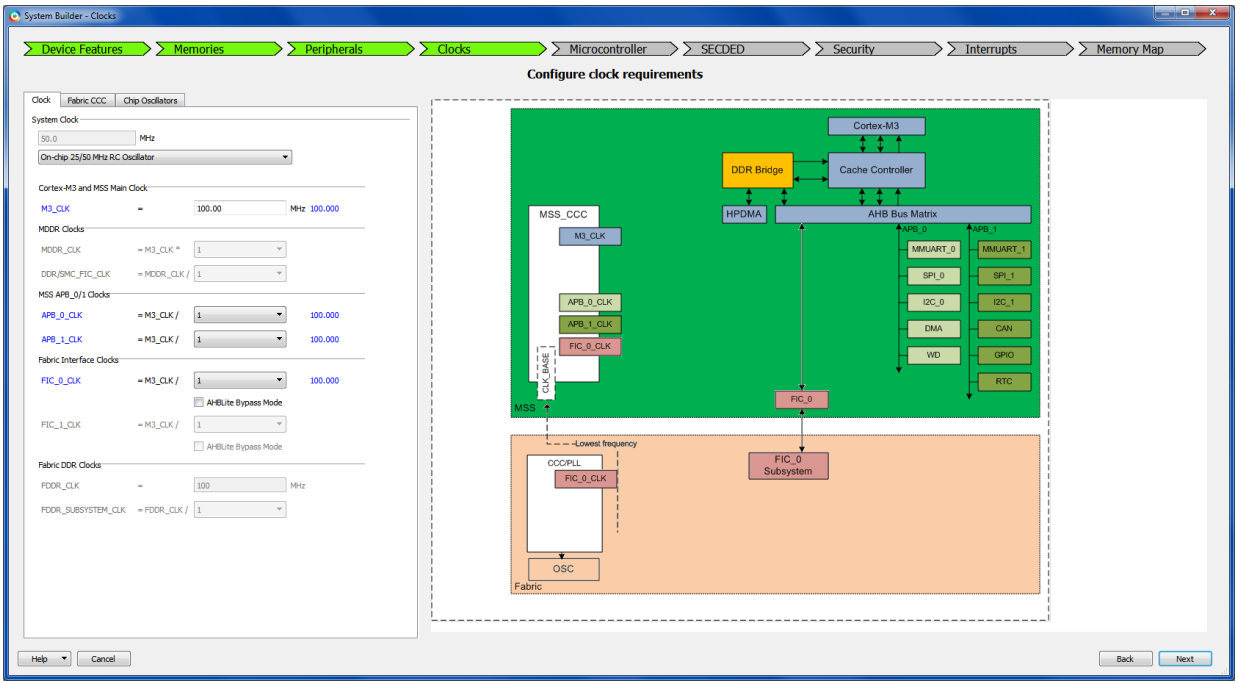
3. In the **Peripherals** page, drag **Fabric AMBA Master** to **MSS FIC_0 - Fabric Master Subsystem.** It provides the AHBL master interface to the user logic, as shown in the following figure.

Figure 9 • **SmartFusion2—Peripherals Page**



4. In the **Clocks** page, select **On-chip 25/50 MHz RC Oscillator** as **System Clock** and **100** MHz as **M3_CLK** frequency, as shown in the following figure.

Figure 10 • **SmartFusion2—Clocks Page**

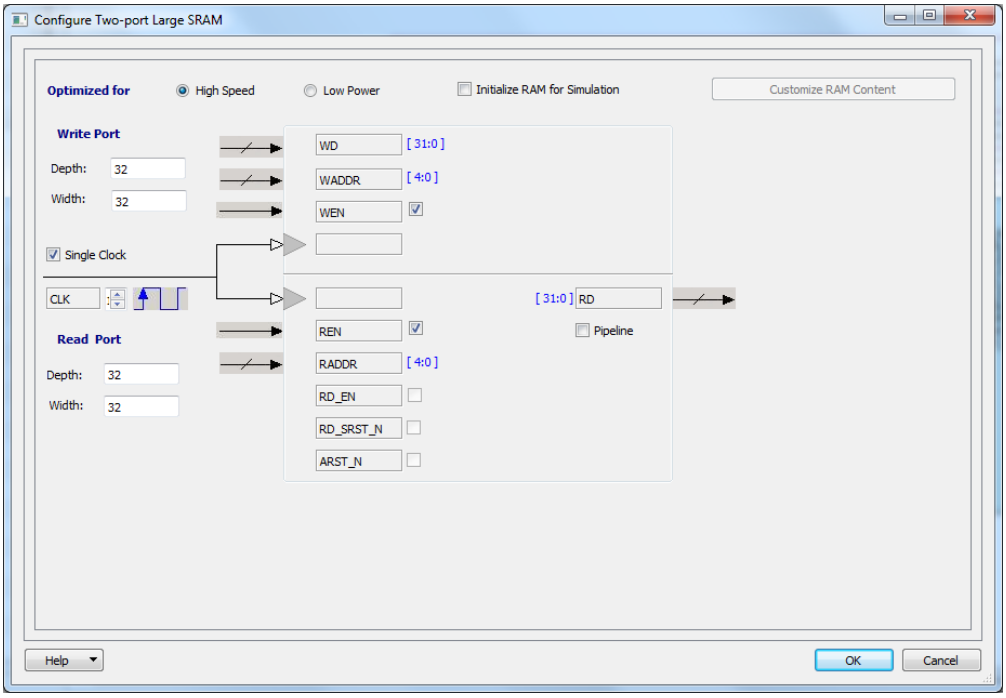


For more information about how to generate a complete System Builder component for SmartFusion2, refer to the *SmartFusion2 System Builder User Guide*.

2.6.1.3 TPSRAM IP Configuration

In SmartDesign, TPSRAM IP is configured as: write port 32 (depth) × 32 (width) and read port 32 (depth) × 32 (width). The following figure shows the TPSRAM IP configuration.

Figure 11 • **TPSRAM Configuration**



2.7 Simulation

To simulate the design, on the **Design Flow** tab, right-click **Simulate** under **Verify Pre-Synthesized Design**, and select **Open Interactively**.

2.7.1 eNVM Simulation

The following figures show the eNVM write command sequence. For more information, refer Appendix 2: eNVM and eSRAM Write/Read Operations, page 22.

Figure 12 • **eNVM Write Command Sequence—1**

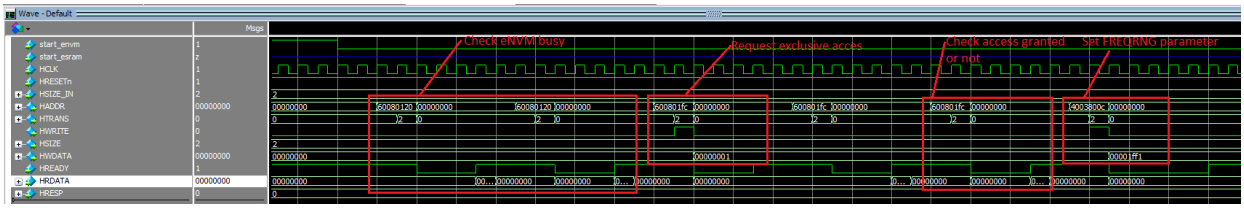


Figure 13 • **eNVM Write Command Sequence—2**

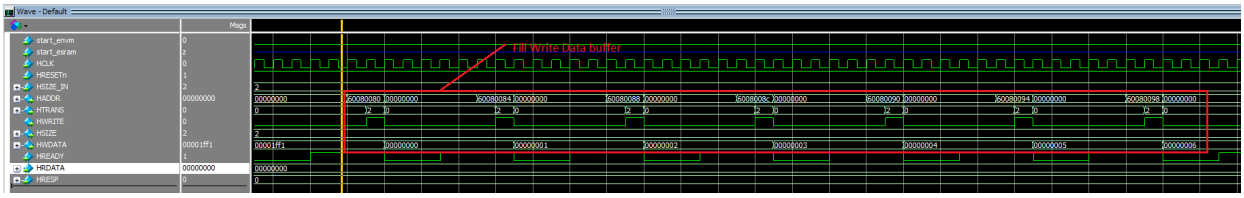
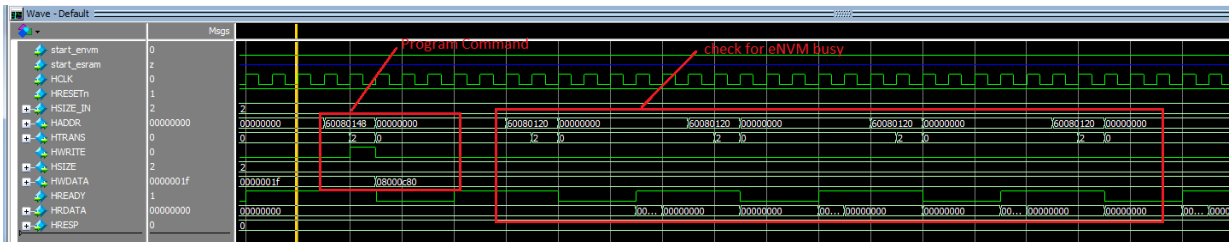
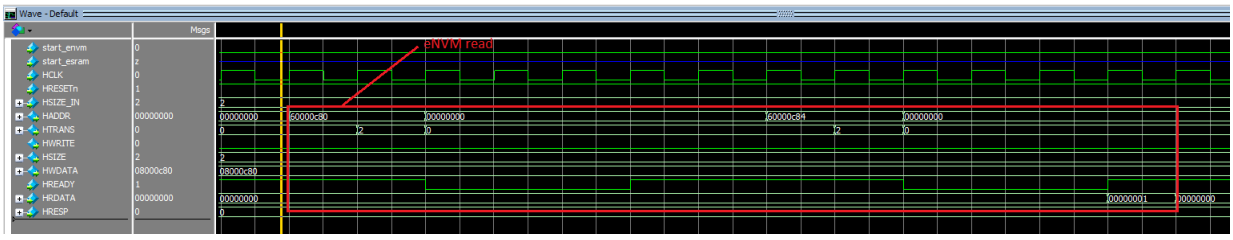


Figure 14 • **eNVM Write Command Sequence—3**



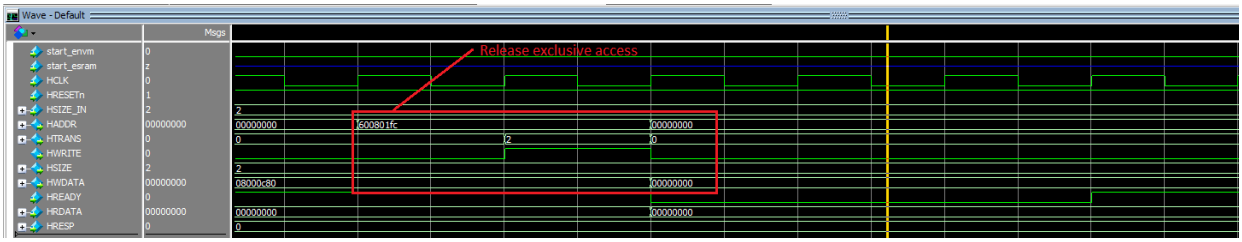
The following figure shows the eNVM read simulation (AHB read operation).

Figure 15 • **eNVM Read Simulation**



The following figure shows the release exclusive access to eNVM.

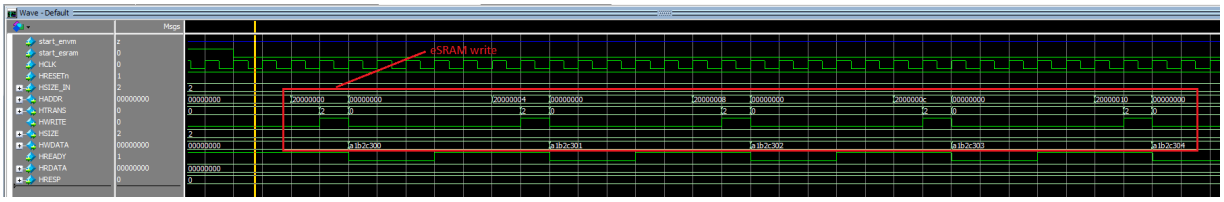
Figure 16 • **Release Exclusive Access to eNVM**



2.7.2 eSRAM Simulation

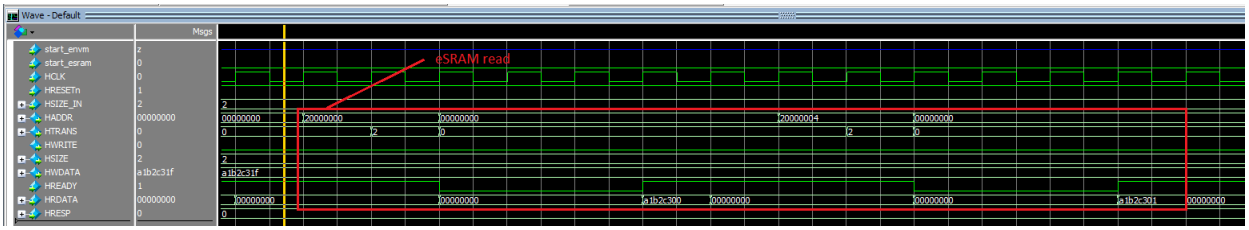
The following figure shows the eSRAM write simulation (AHB write operation).

Figure 17 • **eSRAM Write Simulation**



The following figure shows the eSRAM read simulation (AHB read operation).

Figure 18 • **eSRAM Read Simulation**



2.8 Setting Up the Design

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

1. Connect the jumpers on the IGLOO2 Evaluation Kit board according to the following table.

Table 2 • **Jumper Settings for IGLOO2 Evaluation Kit Board**

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 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 USB cable to the J18 connector.

The following steps describe how to set up the hardware demo for the SmartFusion2 Security Evaluation Kit board:

1. Connect the jumpers on the SmartFusion2 Security Evaluation Kit board according to the following table.

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

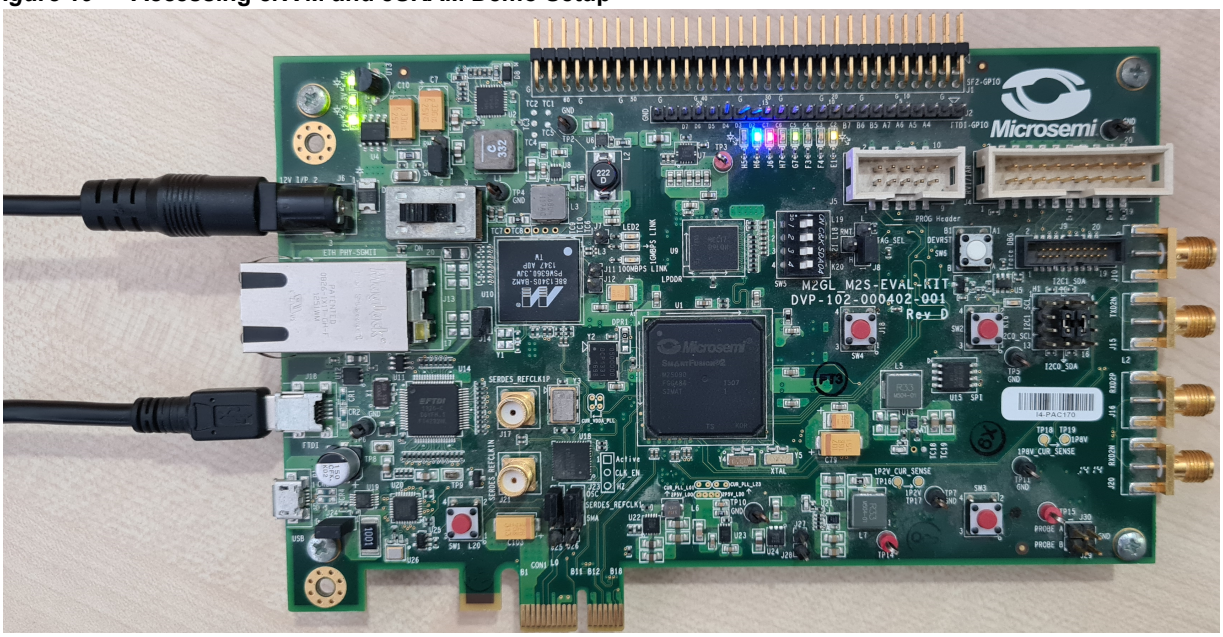
Jumper	Function	Default Settings
J23	Selects switch-side MUX inputs of A or B to the line side	–
	Pin 1-2 (Input A to the line side) that is on board 125 MHz differential clock oscillator output is routed to the line side	Closed
	Pin 2-3 (Input B to the line side) that is external clock required to source 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 the power supply switch, SW7, is switched off while connecting the jumpers on the SmartFusion2 Security Evaluation Kit.

2. Connect the power supply to the J6 connector.
3. Switch on the power supply switch, SW7.
4. Connect the USB cable to the J18 connector.

The following figure shows the demo setup for SmartFusion2 Security Evaluation Kit and IGLOO2 Evaluation Kit.

Figure 19 • **Accessing eNVM and eSRAM Demo Setup**



2.9 Programming the Device

Program the SmartFusion2 and IGLOO2 Evaluation Kit board with the job file provided as part of the design files using FlashPro Express software, refer to *"Appendix 1: Programming the Device Using FlashPro Express" on page 19*.

2.10 Running the Design

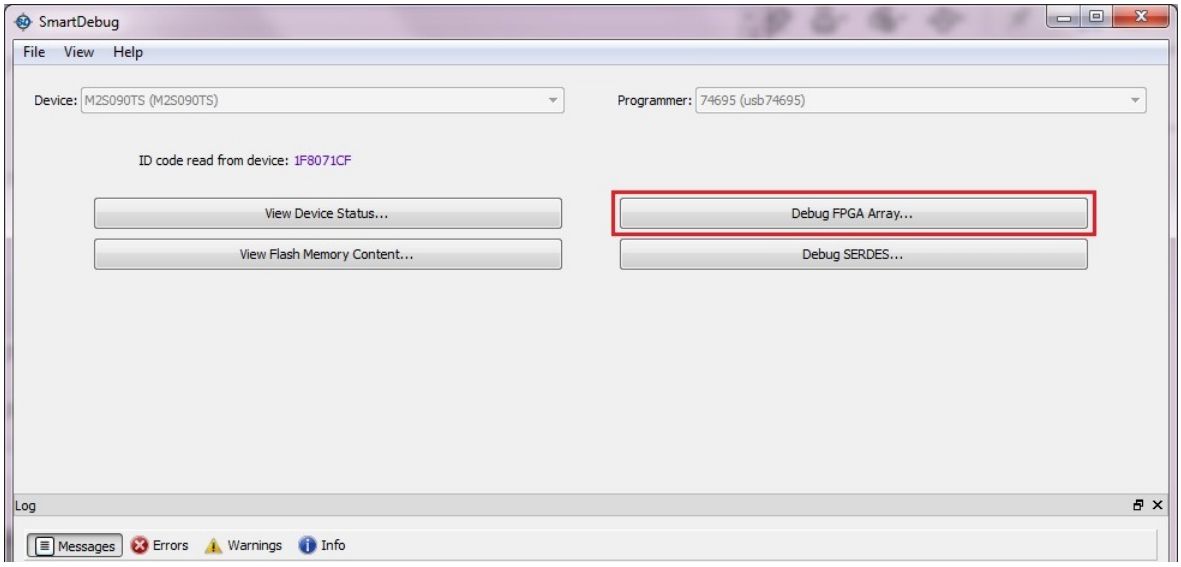
The design can be run to perform the eNVM and eSRAM write/read operations.

2.10.1 eNVM Write and Read Operations

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

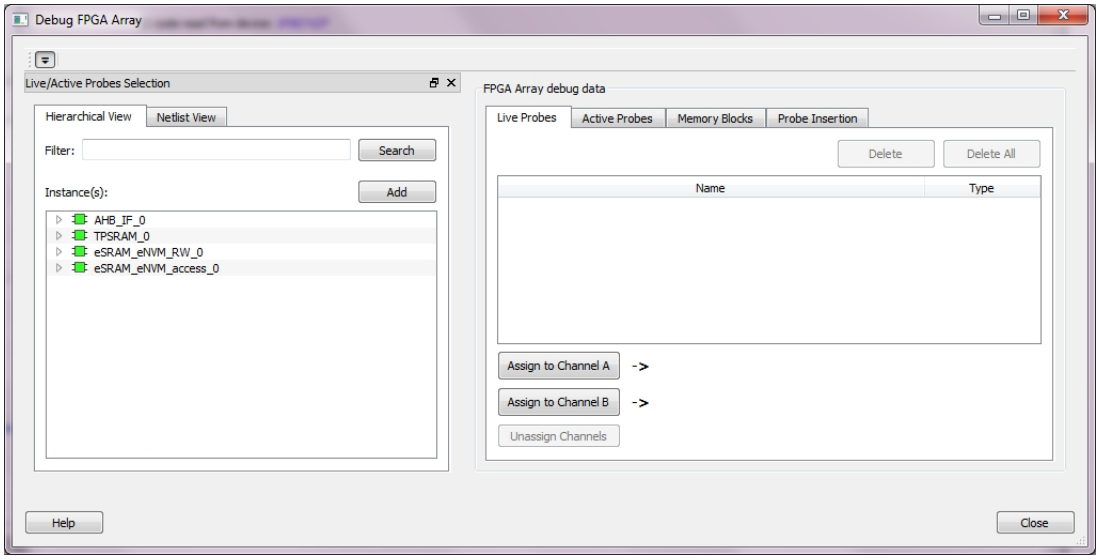
1. Make 1 to 0 transition using the SW5-1 DIP switch (FPGA pin number: L19) to write to and read from the eNVM. An incremental data starting from 0x00000000 to 0x0000001F is written to page 25 of the eNVM, and the data is read back from the eNVM and stored in the fabric SRAM. For more information about eNVM write operation, refer to Appendix 2: eNVM and eSRAM Write/Read Operations, page 22.
   The write and read operations can be verified using the SmartDebug tool in Libero SoC.
2. In Libero SoC, go to **Design Flow** > **Program and Debug Design** > **Debug Design** > **SmartDebug Design**. Right-click and select **Open Interactively**. The **SmartDebug** window opens up, as shown in the following figure.
3. Click **Debug FPGA Array**.

Figure 20 • **SmartDebug Window**



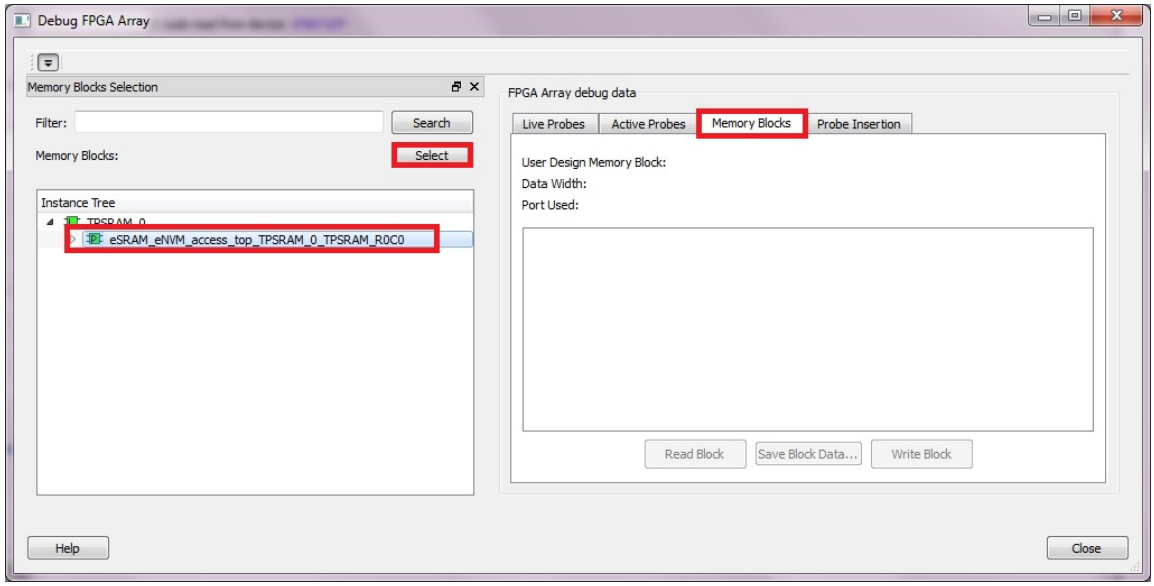
The **Debug FPGA Array** window opens, as shown in the following figure.

Figure 21 • **Debug FPGA Array**



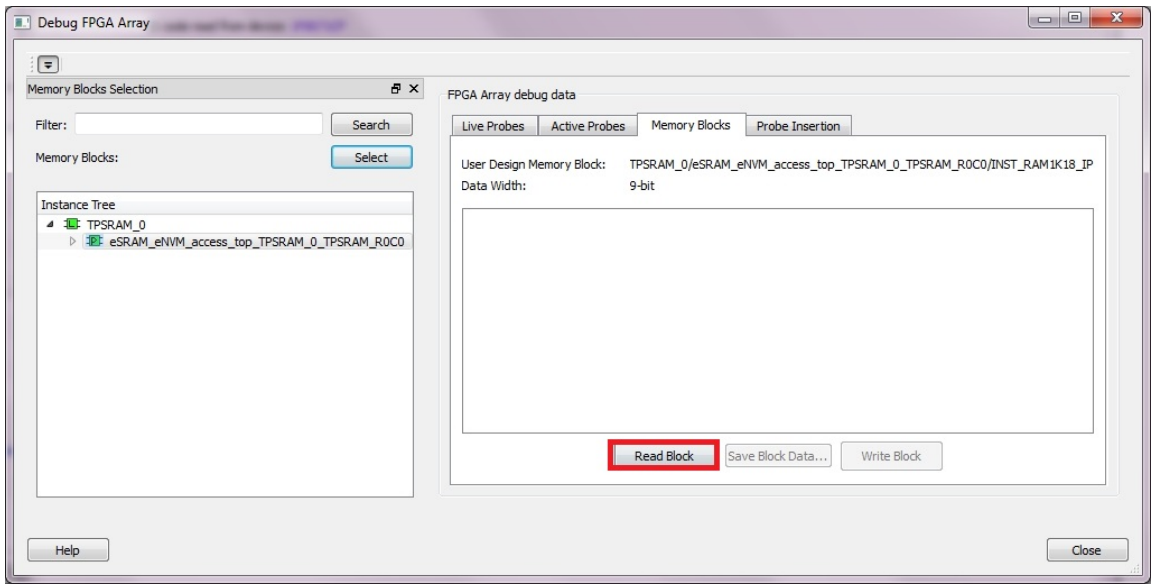
4. Click **Memory Blocks** tab to list the available memory blocks.
5. Select the desired memory block and click **Select**, as shown in the following figure.

Figure 22 • **Select Debug FPGA Array**



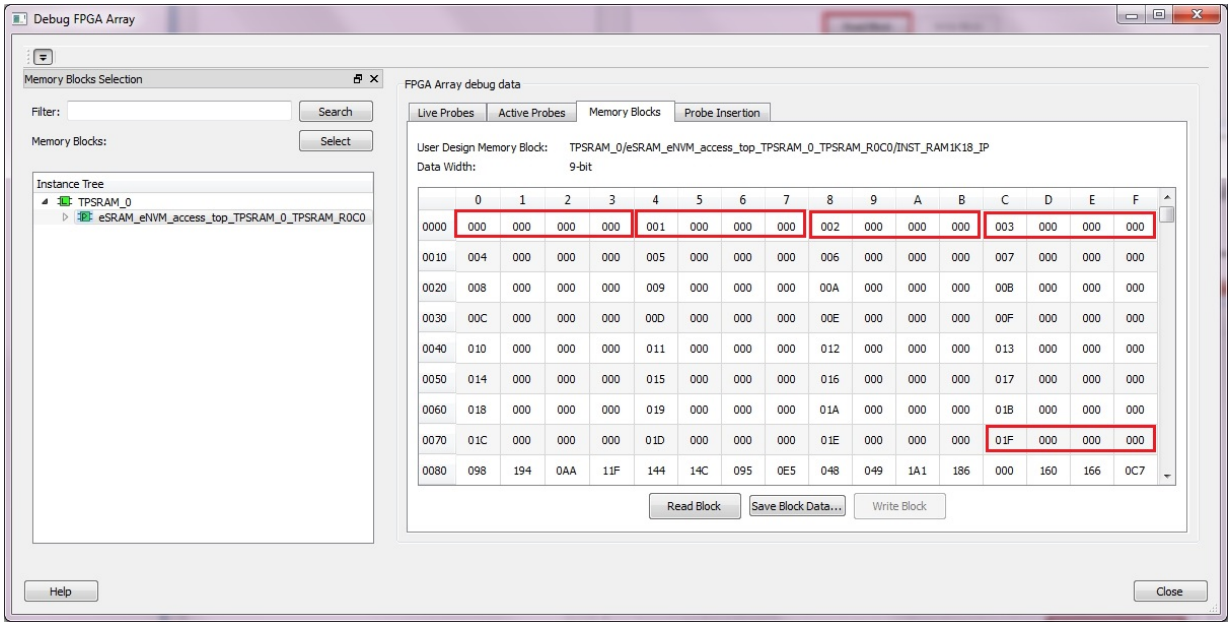
6. Click **Read Block**, as shown in the following figure.

Figure 23 • **Debug FPGA Array Window**



The fabric SRAM (TPSRAM) memory content is displayed, as shown in the following figure. If the initial data written to the eNVM matches the displayed data, eNVM write and read are successful.

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



7. Close the **Debug FPGA Array** window and the **SmartDebug** window.

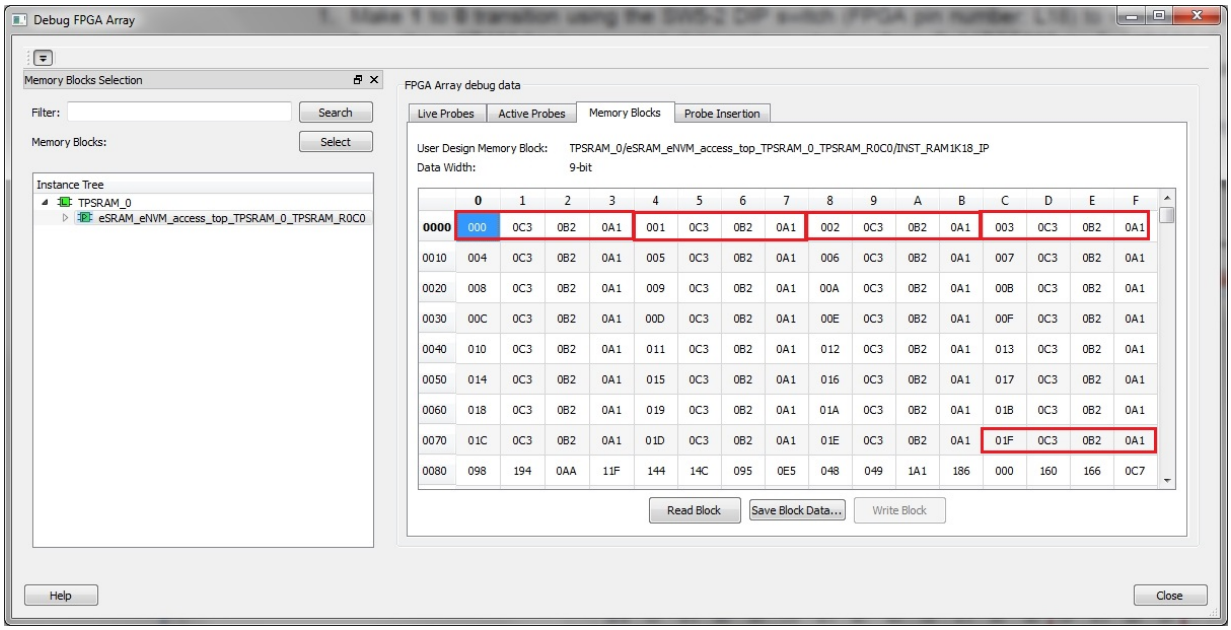
For more information on how to interpret the memory block data, go to **Help** > **Help Topics** > **Debug Design** > **SmartFusion2 and IGLOO2 SmartDebug** > **Debug FPGA Array** > **Memory Blocks**.

2.10.2 eSRAM Write and Read Operation

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

1. Make 1 to 0 transition using the SW5-2 DIP switch (FPGA pin number: L18) to write to 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 the eSRAM and stored in the fabric SRAM.
2. To verify the write and read operations using the SmartDebug tool in Libero SoC, follow steps 2 to 8 in eNVM Write and Read Operations, page 14.
   The fabric SRAM (TPSRAM) memory content is displayed, as shown in the following figure. If the initial data written to the eSRAM matches the displayed data, eSRAM write and read operations are successful.

Figure 25 • **Debug FPGA Array—Memory Blocks**



For more information on how to interpret the memory block data, go to **Help** > **Help Topics** > **Debug Design** > **SmartFusion2 and IGLOO2 SmartDebug** > **Debug FPGA Array** > **Memory Blocks**.

2.11 Conclusion

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

3 Appendix 1: Programming the Device Using FlashPro Express

This section describes how to program the SmartFusion2 and IGLOO2 devices with the programming job file using FlashPro Express.

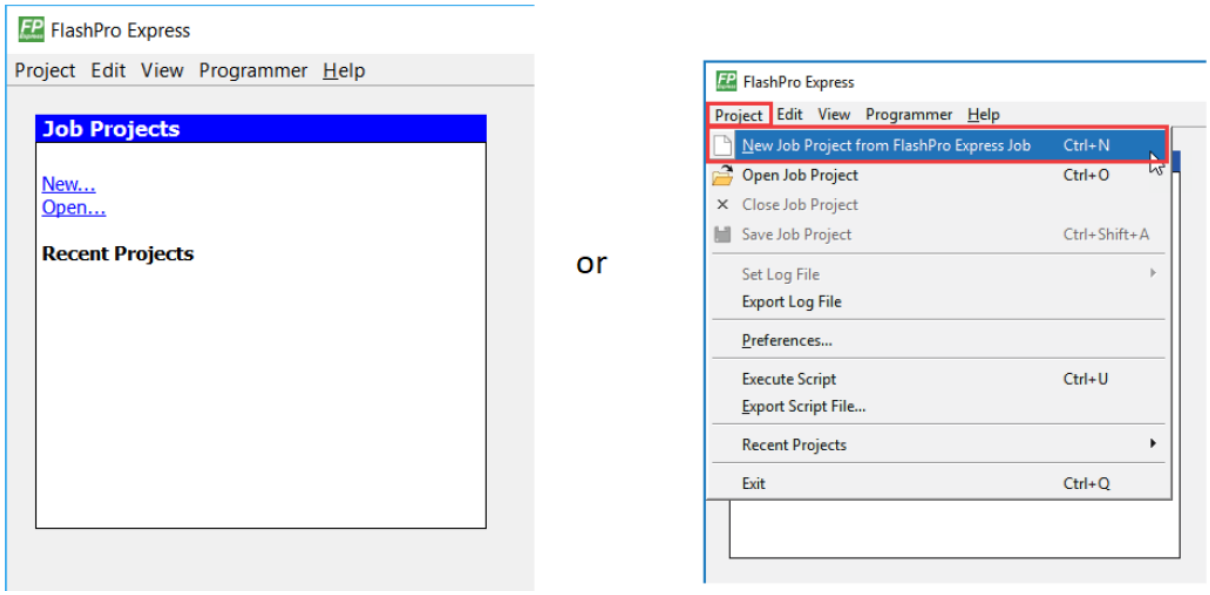
To program the device, perform the following steps:

1. Ensure that the jumper settings on the board are the same as those listed in Table 2, page 12 for IGLOO2 Evaluation Kit and Table 3, page 13 for SmartFusion2 Security Evaluation Kit.

Note: The power supply switch must be switched off while making the jumper connections.

2. Connect the power supply cable to the **J6** connector on the board.
3. Power **ON** the power supply switch **SW7**.
4. On the host PC, launch the **FlashPro Express** software.
5. Click **New** or select **New Job Project from FlashPro Express Job** from **Project** menu to create a new job project, as shown in the following figure.

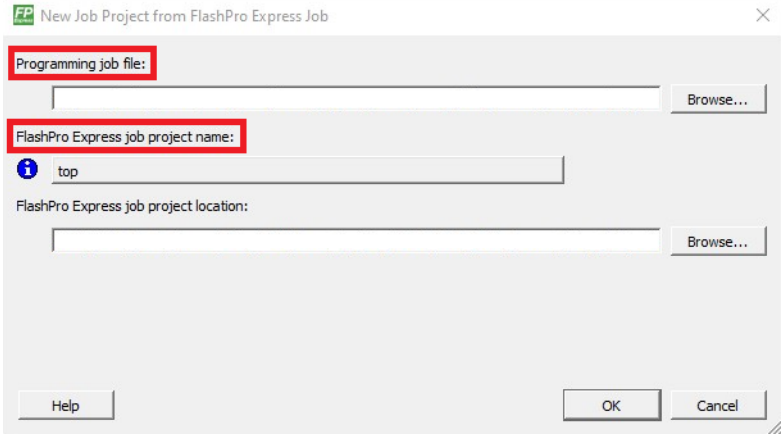
Figure 26 • **FlashPro Express Job Project**



6. Enter the following in the **New Job Project from FlashPro Express Job** dialog box:

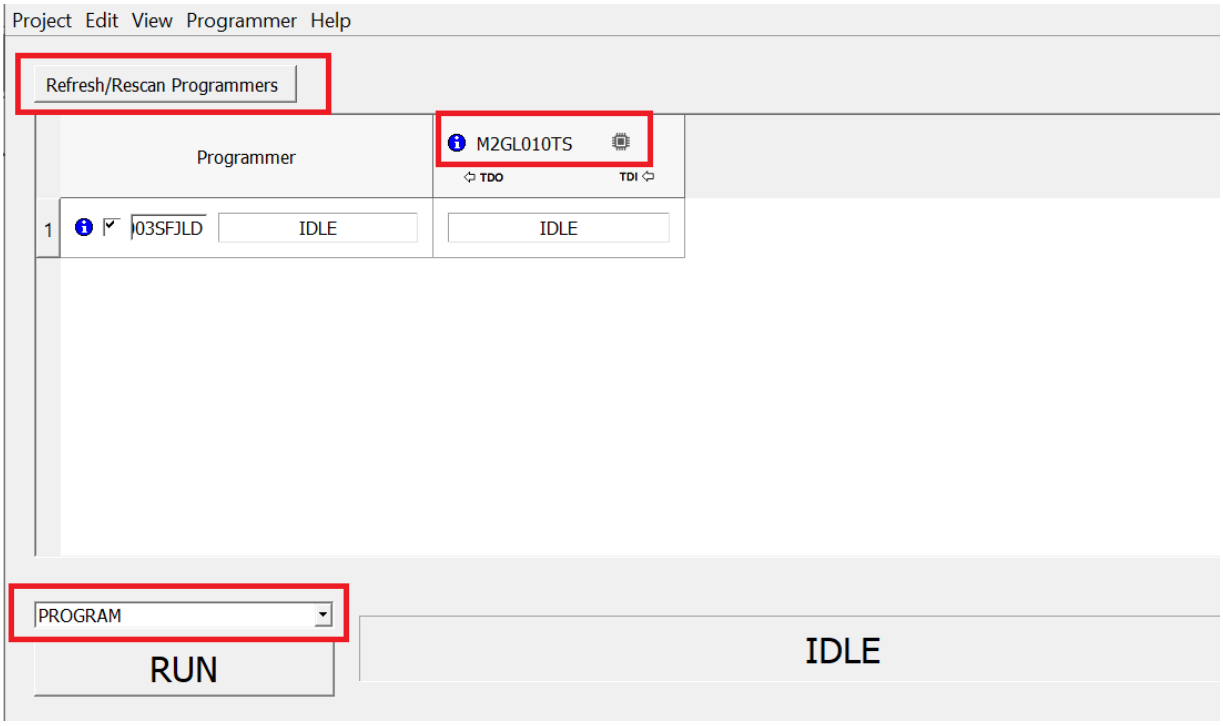
- **Programming job file:** Click **Browse**, and navigate to the location where the .job file is located and select the file. The default location is:
  `<download_folder>m2s_m2gl_ac429_df\Programming_Job`
- **FlashPro Express job project name:** Click **Browse** and navigate to the location where you want to save the project.

Figure 27 • **New Job Project from FlashPro Express Job**



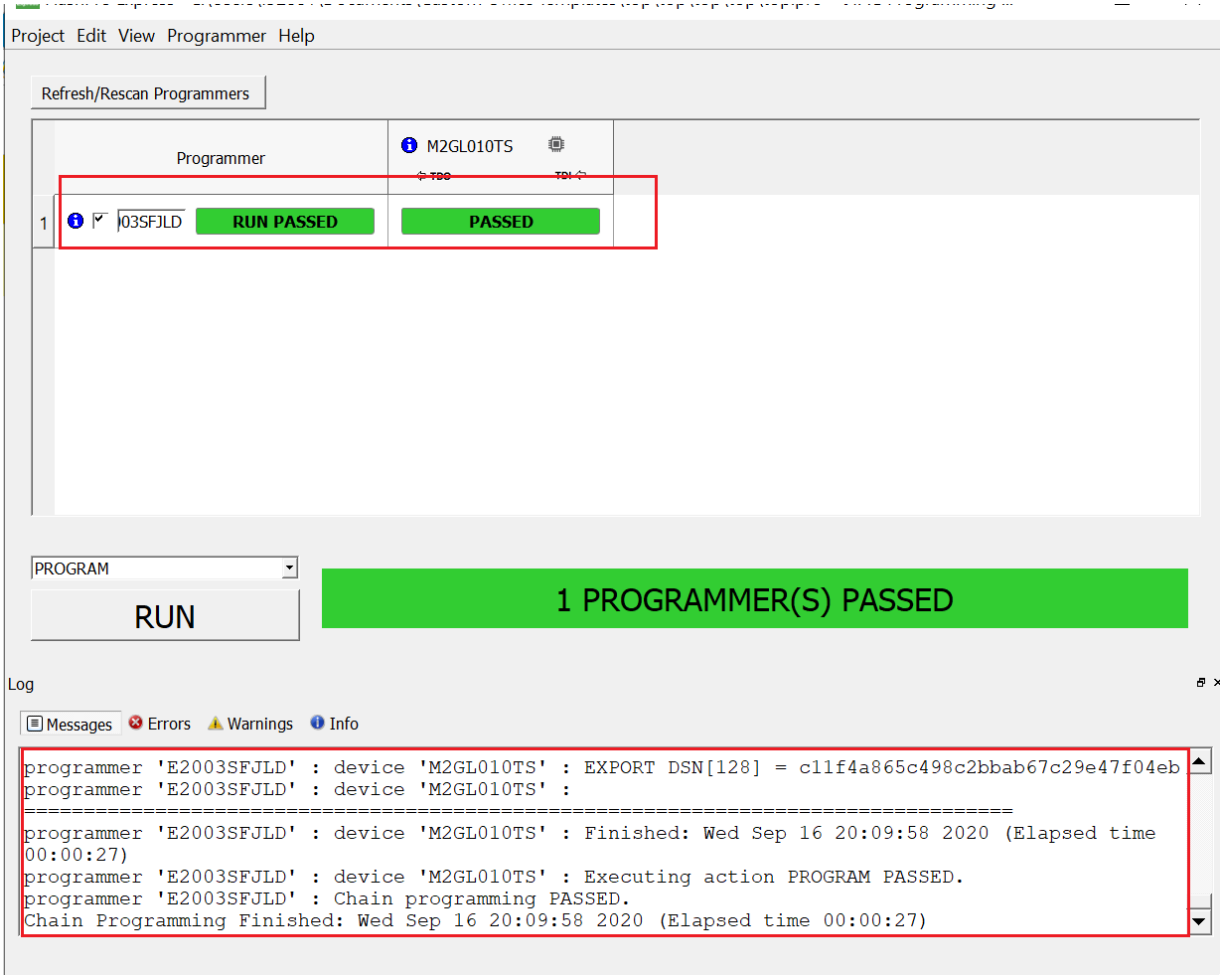
7. Click **OK**. The required programming file is selected and ready to be programmed in the device.
8. The FlashPro Express window appears as shown in the following figure. Confirm that a programmer number appears in the Programmer field. If it does not, confirm the board connections and click **Refresh/Rescan** Programmers.

Figure 28 • **Programming the Device**



9. Click **RUN**. When the device is programmed successfully, a **RUN PASSED** status is displayed as shown in the following figure.

Figure 29 • **FlashPro Express—RUN PASSED**



10. Close **FlashPro Express** or in the Project tab, click **Exit**.

4 Appendix 2: eNVM and eSRAM Write/Read Operations

The following steps describe how eNVM write operation is performed:

1. Wait for Bit 0 of the 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 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 the request is granted, by reading back from the REQACCESS register. On read back, check for data bits [2:0], which must be equal to 6.
4. If data bits [2:0] = 0×6, it implies that the request is granted by the fabric.
5. If bit 2 is 0, the request for exclusive access is denied. The eNVM cannot be written at this time.
6. Write 0x00001FF1 to ENVM_CR register (address: 0x4003800C). This changes the FREQRNG register field to 15 decimals.
7. Writes to the eNVM are buffered. First, write data into the write data buffer (WDB), which is a byte addressable 1024-bit buffer. Its base address is 0x60080080 for eNVM_0 and 0x600C0080 for eNVM_1. Then, use a single command to commit (program) data into one page of the eNVM. Write the data into the WDB.
8. Compute the values of bits that need to be written into the eNVM command register.
    - Bits 31-20 must be 0x080.
    - Bit 19 must be 0x0.
    - Bits 18-7 corresponds to the number of the page to be written (for the 25th page, the bit field is 000 0000 1100 1).
    - Bits 6-0 must be 0x0.
9. Write the eNVM command register (address: 0x60080148) with the data computed in step 8. eNVM does not respond to further commands until the write is complete.
10. Release exclusive access to the eNVM by writing 0x0 to the REQACCESS register (address: 0x600801FC).

Note: Special command sequences are not required for 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 operation.