LCMXO3LF-9400C SED/SEC Demo

User Guide

FPGA-UG-02023 Version 1.0

June 2017

Contents

Figures

Tables

1. Introduction

An SRAM-based Programmable Logic Device (PLD) stores logic configuration data in SRAM cells. As the number and density of SRAM cells in a PLD increase, the probability that a memory error will alter the programmed logical behavior of the system increases.

The Lattice Semiconductor MachXO3L™ and MachXO3LF devices have a hardware implemented circuit called Soft Error Detection (SED) which is used to detect SRAM errors and allow them to be corrected.

The MachXO3LF Soft Error Correction (SEC) is performed with background reconfiguration. All bits are rewritten during reconfiguration and the erroneous bit is replaced with the correct data. During background reconfiguration, writing the same value into configuration SRAM cells does not affect the SRAM cell output.

Soft Error Injection (SEI) is a feature of the Lattice Diamond® Programmer software. It is used to inject an error in the MachXO3 configuration SRAM.

This demo shows you all the SED, SEC, and SEI features of the MachXO3 and how the development tool is used.

1.1. Demo Design Overview

This demo design consists of two major parts:

- SED Hard Module – Performs read and error detection of SRAM content.
- Control Logic – A state machine generating control signals for driving SED module. This includes soft error indication and a function block that rotates the onboard LEDs.

The status of the demo is indicated by the onboard LEDs. The SEI bitstream is used to induce a single SRAM error into the running demo. The SEI bitstream is generated using the Soft Error Injection Editor available in the Tools tab of the Lattice Diamond® software.

This demo covers the following operations:

- Programming the board with the design
- Enabling SED
- Loading the SEI bitstream in the background and detecting the inserted error
- Background refreshing of the image after the error is detected by SED

1.2. MachXO3-9400 Development Board and Resources

Figure 1.1 shows the top side of the MachXO3-9400 Development Board and resources used for the demo.

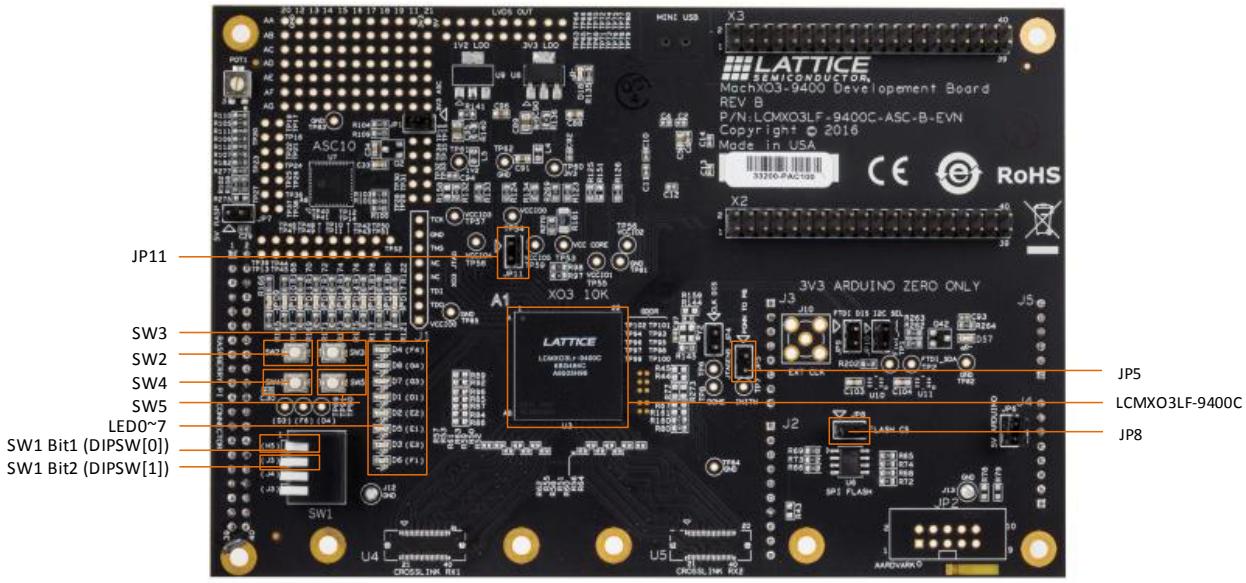

Figure 1.1. MachXO3-9400 Development Board

- JP11 must be set to provide an external 12 MHz clock for the design
- JP5 is for auto SEC when set
- SW1: 4bit DIP Switch: Up - 1; Down - 0; “110x” to enable SED
- SW2: Active low reset input for the design
- SW3: Starts SED for error detection
- SW4: Forces SED to generate an error output for testing when SW1 (Bit4-1) is set to “1101”
- SW5: PROGRAMN when JP5 is set
- JP8 should be removed
- LED0~7: LED0~5 start to shift when the MachXO3 device is programmed with the design; LED6 (D3) is lighted when an error is detected; LED7 (D6) blinks to indicate that the SED is enabled when SW1(Bit4~1) is set to “1101”

2. Functional Description

Figure 2.1 shows the block diagram of the demo design.

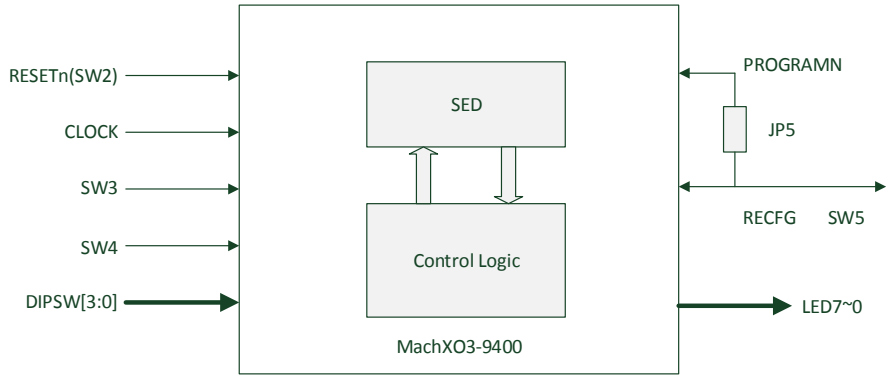

Figure 2.1. MachXO3-9400 Development Board SED/SEC Demo Block Diagram

3. Demo Package

3.1. Hardware Requirements

To run the demo, the following hardware are required:

- PC running Windows 7 Operating System
- MachXO3-9400 Development Board
- Mini USB cable for programming the MachXO3 device

3.2. Software Requirements

To run the demo, the following software are required:

- Lattice Diamond version 3.9 or later
- Lattice Diamond Programmer software for bitstream downloading

Note: The software programs are available at www.latticesemi.com/en/Products/DesignSoftwareAndIP.

4. Port Assignments and Descriptions

Table 4.1 provides the design's port assignments and definitions.

Table 4.1. Design Port Definitions

Port Name	I/O Type	Description
RESETn	Input	Asynchronous reset, active low (SW2)
CLK	Input	External 12 MHz clock input
DIPSW[3:0]	Input	Up-1; Down-0. DIPSW[3]-Bottom; DIPSW[0]-Top; DIPSW[3:0]='1101' enables SED detection; LED7 (D6) starts blinking
SW3	Input	Starts SED for a detection cycle
SW4	Input	Forces SED to generate an error output and turns on LED6 (D3) when DIPSW[3] is Up(1)
RECFG	Output	For background SEC through reconfiguring MachXO3 after an error is detected; Connected with PROGRAMn and SW5, Hiz after configuration
LED[5:0]	Output	Rotating LED0~5 (D4, D8, D7, D1, D2, D5) indicating user logic is running
LED6	Output	D3, indicates that an error is detected by SED
LED7	Output	D6, indicates that the SED is enabled (when blinking)

5. Demo Package Directory Structure

Figure 5.1 shows the demo package directory structure.

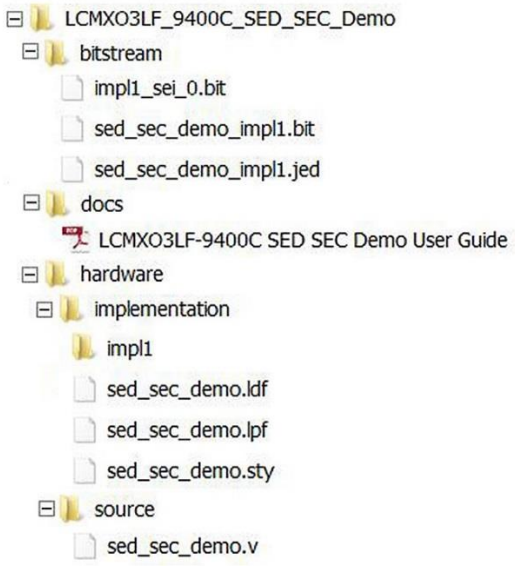

Figure 5.1. Demo Package Directory Structure

6. Running the Demo

6.1. Programming the Board with the Design

To load and run the SED/SEC demo design:

1. Before powering on the board, make sure JP5 is set.
2. Power on the board by connecting it to the PC through the USB cable.
3. Launch the Diamond Programmer software (version 3.9 or above). In the Getting Started dialog box, select **Create a new project file from JTAG scan** and click **OK**.

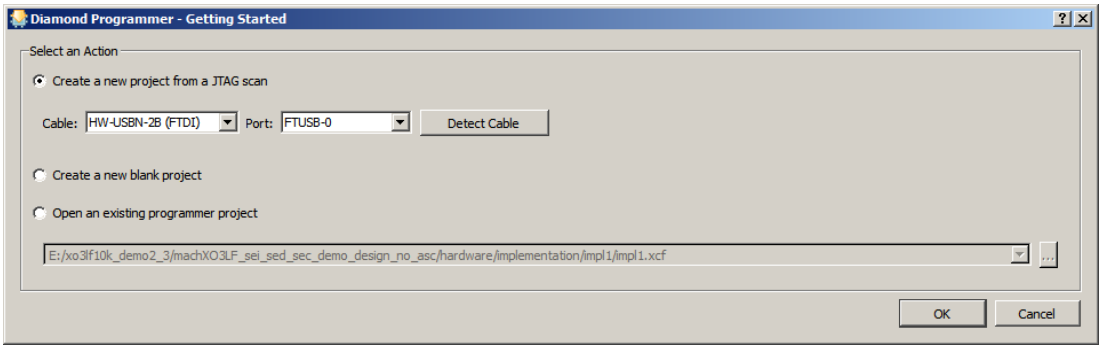

Figure 6.1. Launching Diamond Programmer

4. The LCMXO3LF-9400C device is detected and displayed in the main interface. Right-click on the device and select **Device Properties** in the context menu.

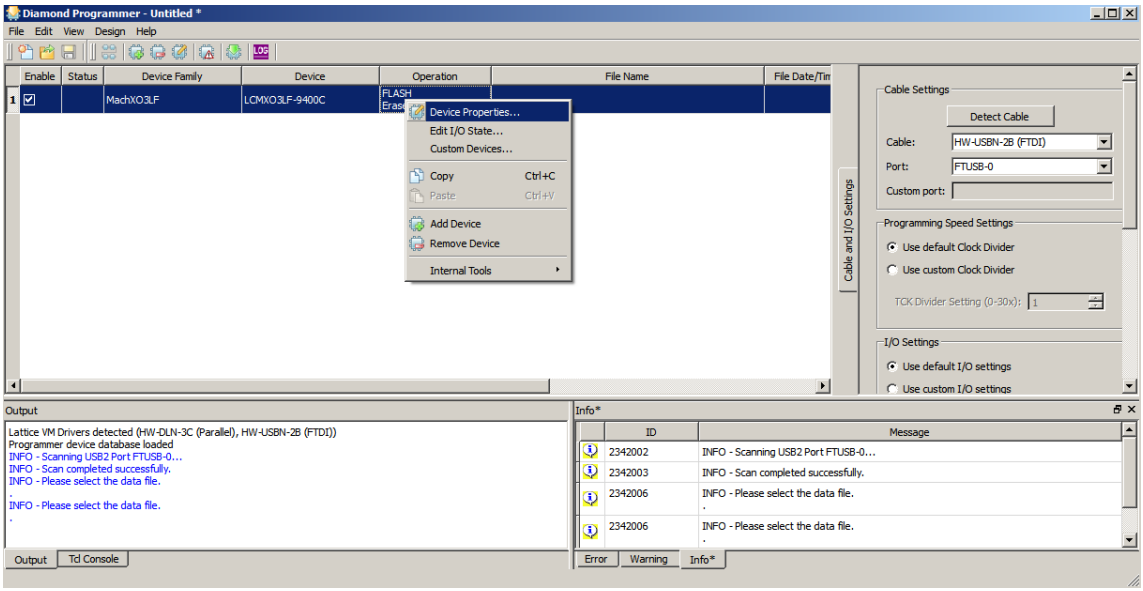

Figure 6.2. Opening Device Properties

5. Set device properties as shown in Figure 6.3.
   In the Device Properties dialog box, select **Flash Programming Mode** in Access mode and **Flash Erase, Program, Verify** in Operation. Select the bitstream file **\bitstream\sed_sec_demo_impl1.jed** in Programming file. Click **OK**.

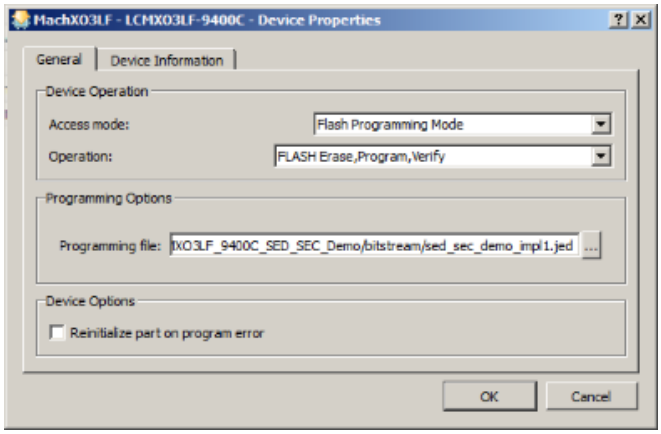

Figure 6.3. Setting Device Properties

6. Click the **Program** button as shown in Figure 6.4.

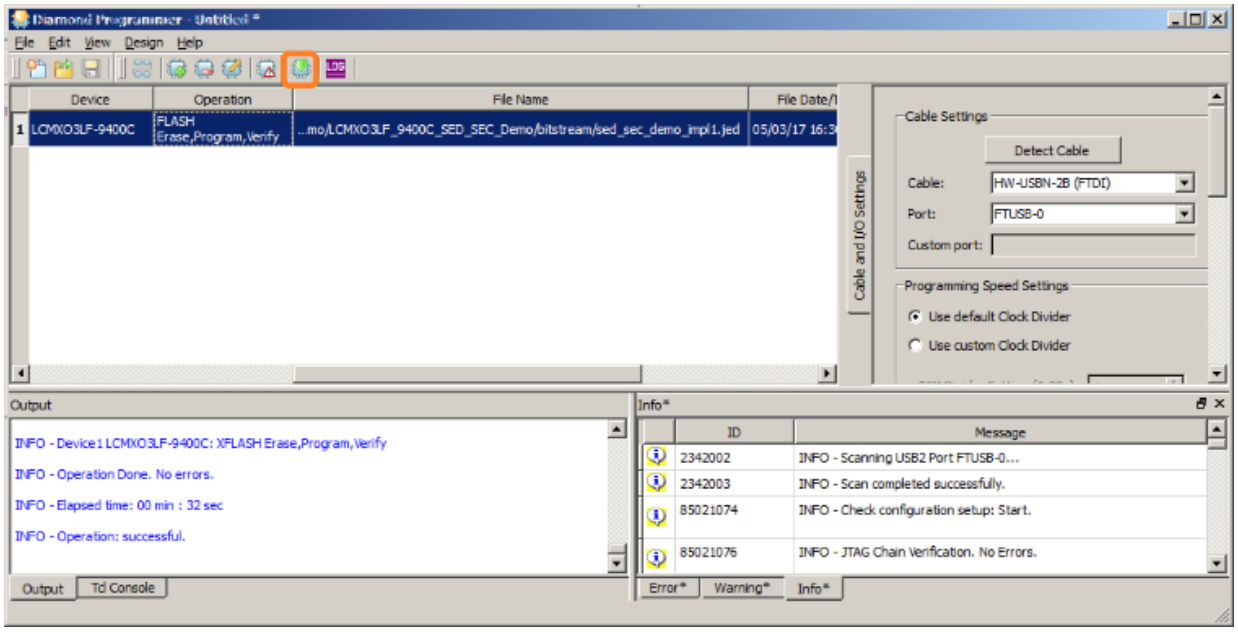

Figure 6.4. Starting Programming Operation

If programming is successful, “Operation: successful” is reported under Output. Rotating LEDs from D4 to D5 are also seen on the board.

6.2. Enabling SED

To enable SED:

1. Make sure JP11 is set to provide the external 12 MHz clock for the design
2. Set SW1 (DIPSW[3:0]) all bits up.
3. Push down SW1 BIT2 (DIPSW[1]) to enable SED. LED7 (D6) starts to blink. This indicates that SED is enabled.
4. Force SED to generate an error.
   a. Set SW1-BIT1 DIPSW[0] up.
   b. Press SW4 to generate the error.

   LED6 (D3) turns on. This indicates that an error is generated.

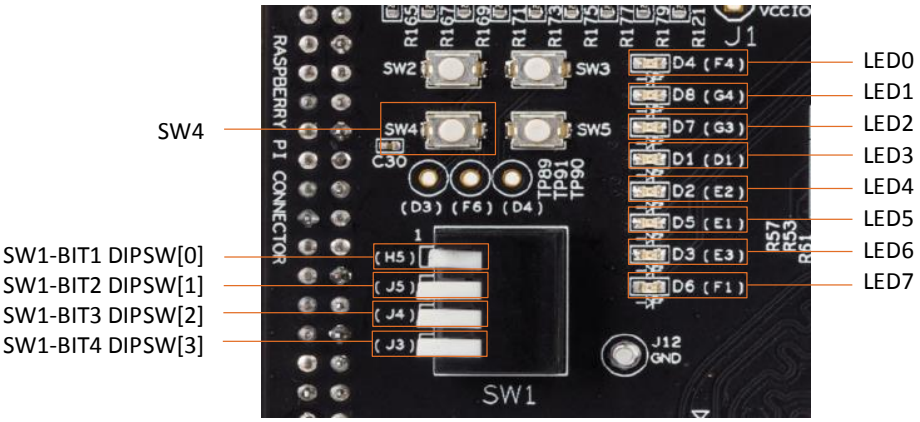

Figure 6.5. Pressing SW4 to Force SED to Generate Error

5. Press SW3 to start the SED detection cycle. LED6 (D3) turns off. This indicates that no error is injected/detected yet. See the Loading SEI Bitstream in Background and Detecting Inserted Error section for error injection and detection.

6.3. Loading SEI Bitstream in Background and Detecting Inserted Error

To insert an error in the SRAM array:

1. Launch the Diamond Programmer software (version 3.9 or above). In the Getting Started dialog box, select **Create a new project file from JTAG scan** and click **OK**.

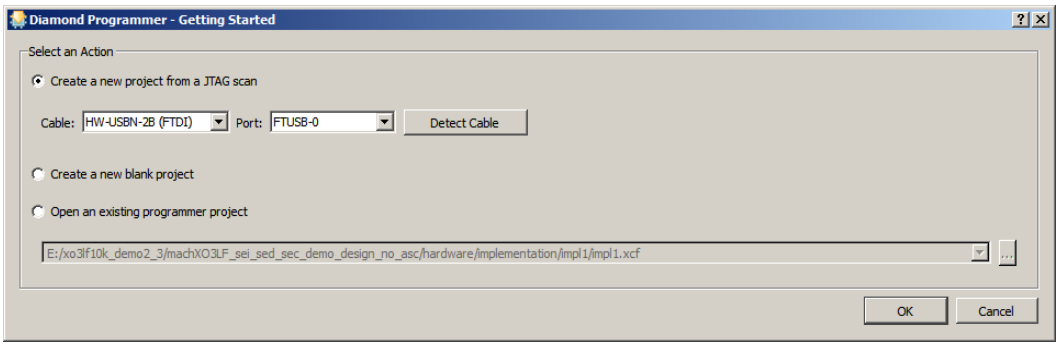

Figure 6.6. Launching Diamond Programmer

2. The LCMXO3LF-9400C device is detected and displayed in the main interface. Right-click on the device and select **Device Properties** in the context menu.

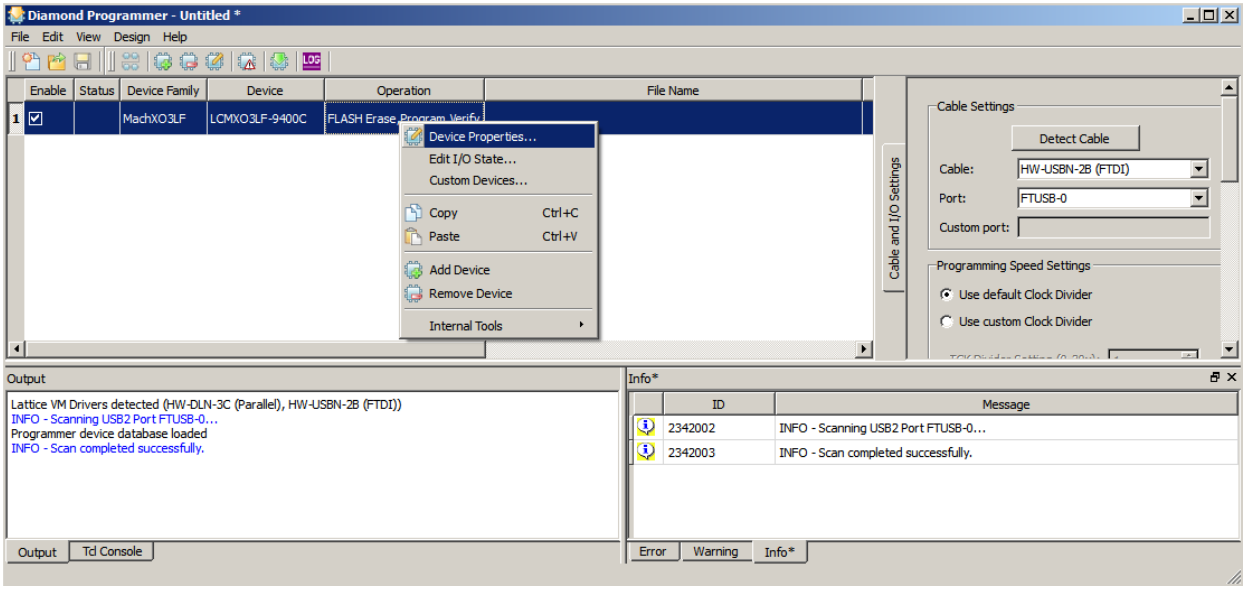

Figure 6.7. Selecting the Device

3. Set device properties as shown in Figure 6.8.
In the Device Properties dialog box, select **Static RAM Cell Background Mode** in Access mode and **XSRAM SEI Fast Program** in Operation. Select the SEI bitstream file **\bitstream\impl1_sei_0.bit** in Programming file. Click **OK**.

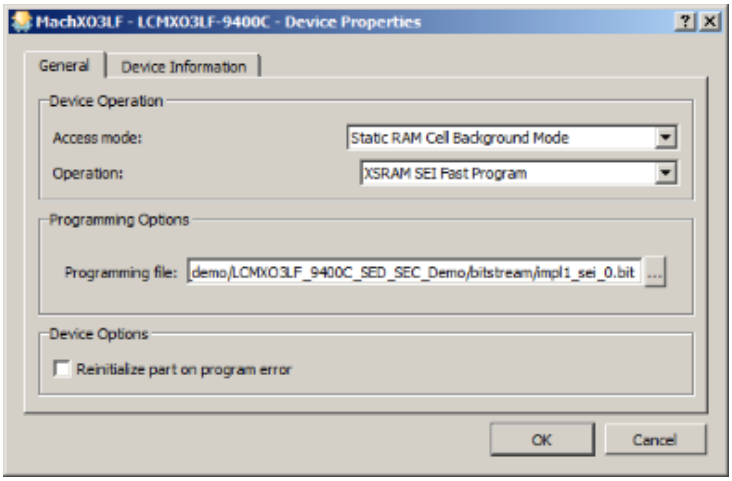

Figure 6.8. Setting MachXO3 Device Properties

4. Click the **Program** button as shown in Figure 6.9 to initiate background programming.

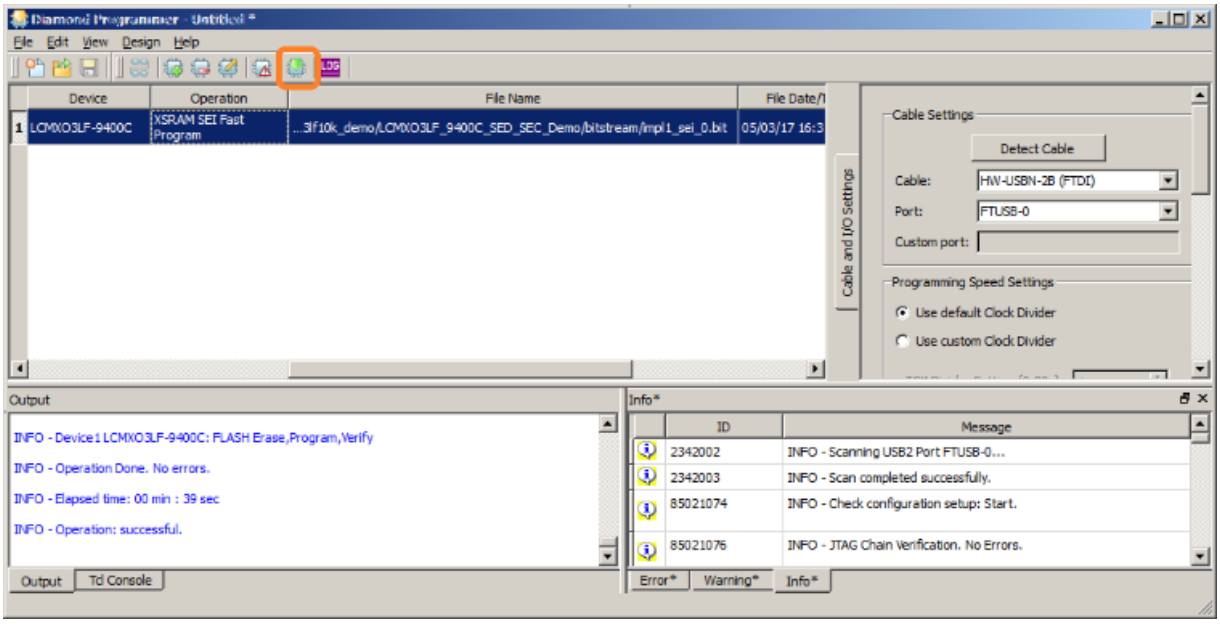

Figure 6.9. Starting XSRAM SEI Fast Program Operation

5. Remove JP5.
6. With JP5 removed, press SW3 again. LED6 (D3) turns off each time SW3 is pressed and on when SW3 is released. This indicates that the error exists and SEC is not working.
7. Set JP5 and press SW3 again. LED6 (D3) does not turn on after pressing SW3 the second time. This indicates that SEC is already working in the background by driving PROGRAMn low based on the design.

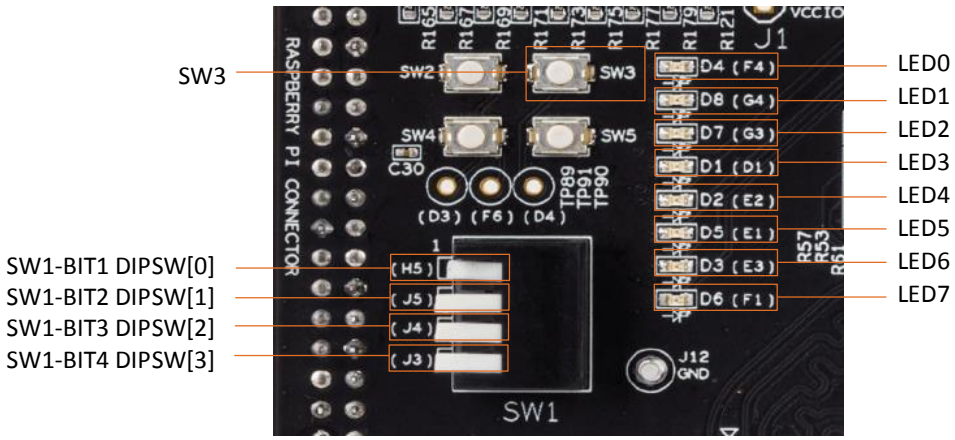

Figure 6.10. Pressing SW3 to Start SED Detection of Injected Error

6.4. Background Refreshing the Image after Soft Error Detection

There are several ways to reload the design bitstream in background. The MachXO3 device supports the following methods of refreshing the SRAM array and correcting detected soft errors.

- Providing the original bitstream in the background through an external sysConfig slave port. For example, this can be done through JTAG or by using I^2C/SPI embedded programming technique.
- Transferring the original bitstream from internal or external Flash memory, initiated through sysConfig REFRESH command.
- Transferring the original bitstream from internal or external Flash memory, initiated through external PROGRAMN pin assertion. This is demonstrated in steps 6, 7 and 8 of the Loading SEI Bitstream in Background and Detecting Inserted Error section.

6.4.1. Providing the Original Bitstream in the Background

Refreshing the SRAM array with the original bitstream through the JTAG port is similar to loading the SEI bit file in the Loading SEI Bitstream in Background and Detecting Inserted Error section. The only difference is the bitstream to be used for loading.

To provide the original bitstream in the background:

1. Make sure JP5 is removed and repeat the appropriate steps in the Loading SEI Bitstream in Background and Detecting Inserted Error section to inject error again.
2. In the Device Properties dialog box, select the design bitstream file **\bitstream\ sed_sec_demo_impl1.bit** in Programming file. Click **OK**.

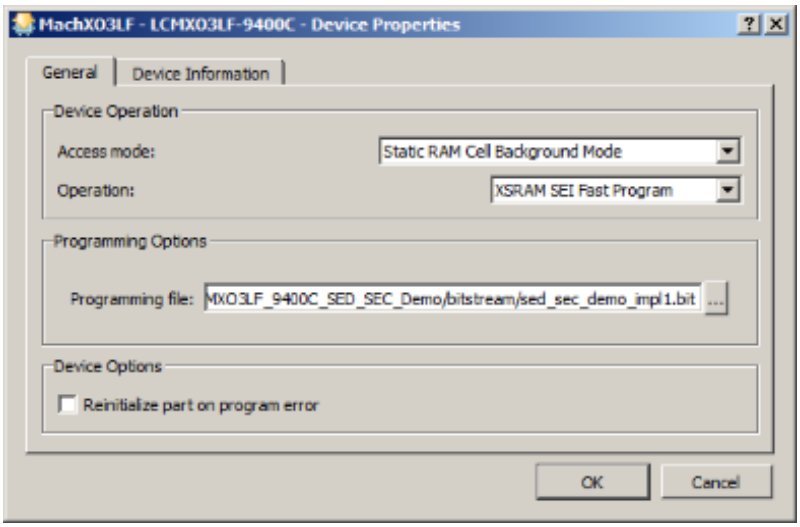

Figure 6.11. Reloading Design Bitstream in Background through JTAG

3. Click the **Program** button as shown in Figure 6.12 to initiate background programming of the device. The original design bitstream is loaded after the software reports “Operation: successful”.

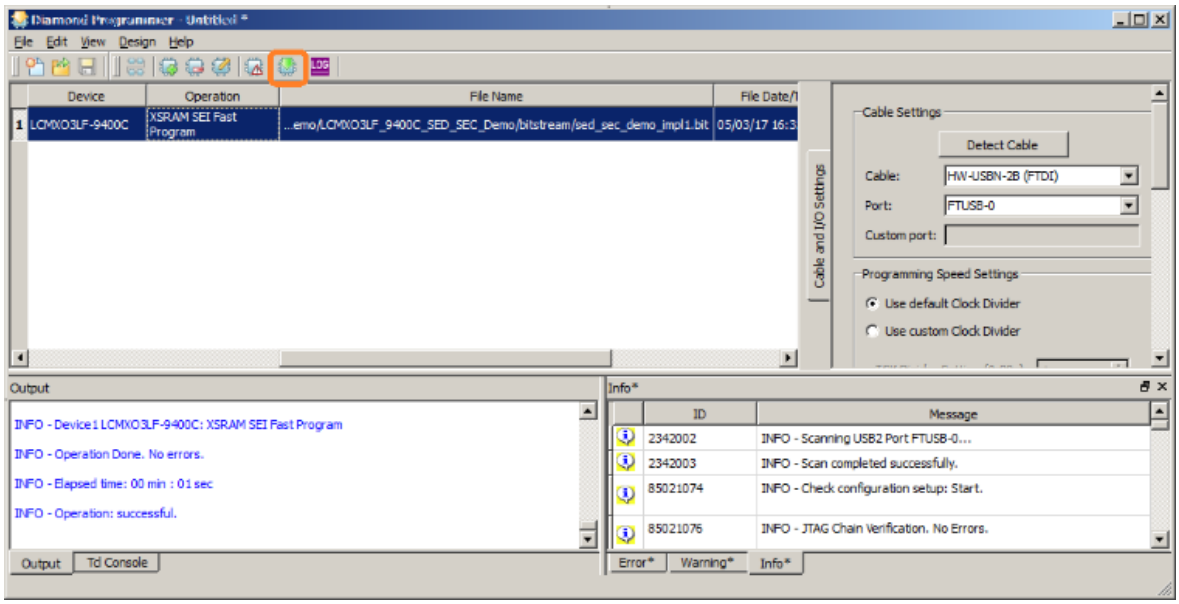

Figure 6.12. Reloading Design Bitstream in Background through JTAG

4. Press SW3 to start the SED and detect no error. LED6 (D3) is not turned on. That indicates that the injected error is corrected by background reconfiguration.

6.4.2. Transferring Original Bitstream Using REFRESH

Reloading the design bitstream in the background refreshes the SRAM array from the internal flash using the REFRESH command.

To load the original design bitstream from internal flash:

1. Make sure JP5 is removed and repeat the appropriate steps in the Loading SEI Bitstream in Background and Detecting Inserted Error section to inject the error again.
2. In the Device Properties dialog box, select **Static RAM Cell Background Mode** in Access mode and **XSRAM Refresh** in Operation as shown in Figure 6.13. Click **OK** after the command is set.

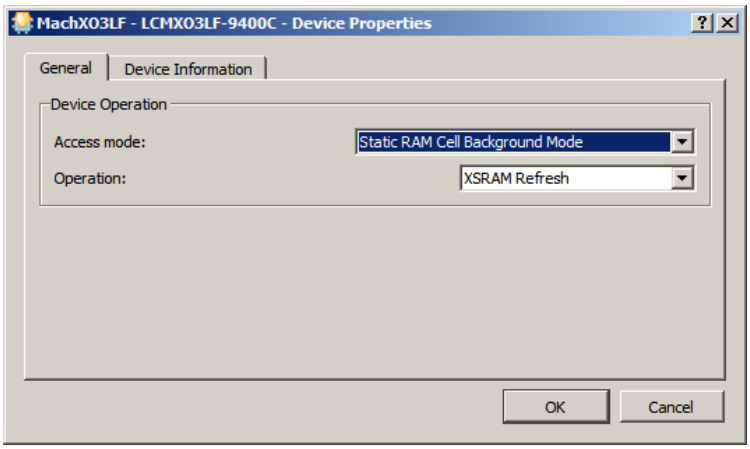

Figure 6.13. Using XSRAM Refresh

3. Click the **Program** button as shown in Figure 6.14 to initiate background programming of the device. The original design bitstream is loaded after the software reports “Operation: successful” while the logic is working.

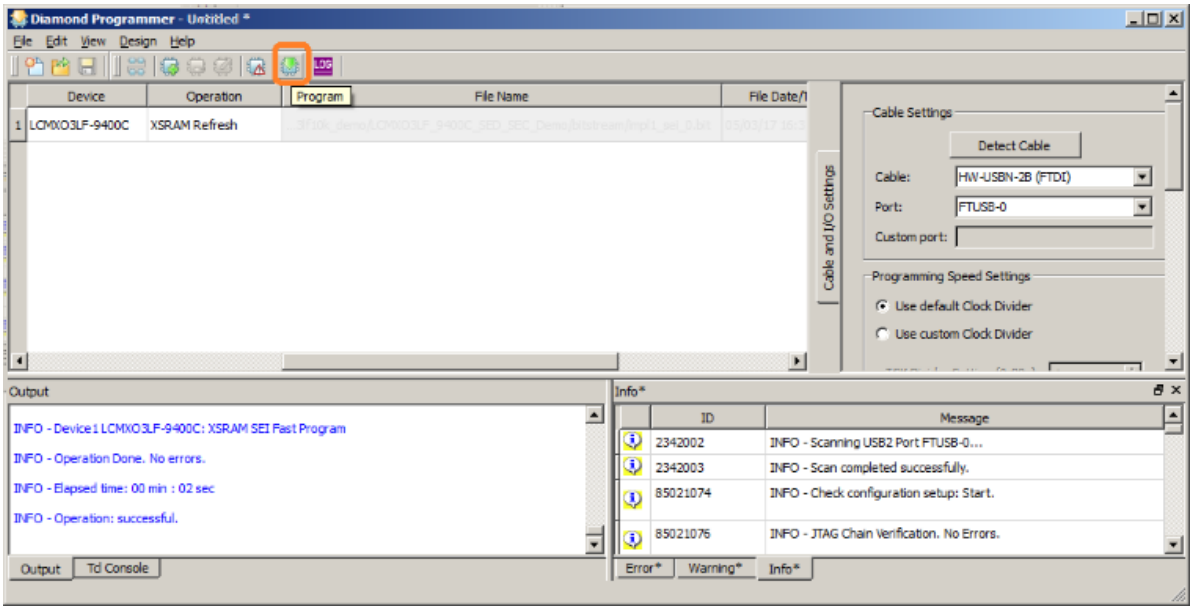

Figure 6.14. Starting XSRAM Refresh Programming Operation

4. Press SW3 to start the SED and detect no error. LED6 (D3) is not turned on. That means the injected error is corrected by background refresh.

6.4.3. Transferring Original Bitstream Using External PROGRAMN Pin Assertion

To transfer original bitstream from internal or external Flash memory by asserting external PROGRAMN pin:

1. Set JP5 as shown in Figure 6.15.

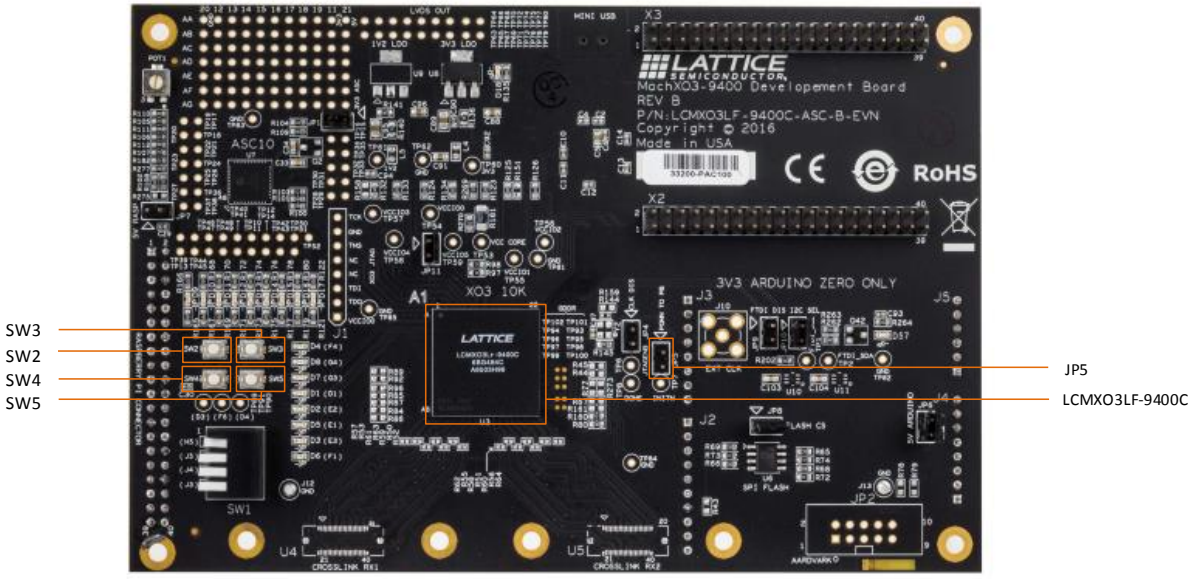

Figure 6.15. Auto Background Reconfiguration

2. After SEI operation is completed (as detailed in the Loading SEI Bitstream in Background and Detecting Inserted Error section), press SW3 to start SED. LED6 (D3) turns on. This indicates that the error is injected. At the same time, the design drives PROGRAMN low if JP5 is set to reconfigure the device in the background while the logic is still working.
3. Press SW3 again to restart SED and detect no error. The LED6 (D3) is not turned on. This indicates that the injected error is corrected by the background reconfiguration.

7. Rebuilding the Design

To rebuild the design:

1. Start the Diamond software. In the Navigator window, click **Open** under Project. Open the Lattice design file (*.ldf) located in the /hardware/implementation folder.

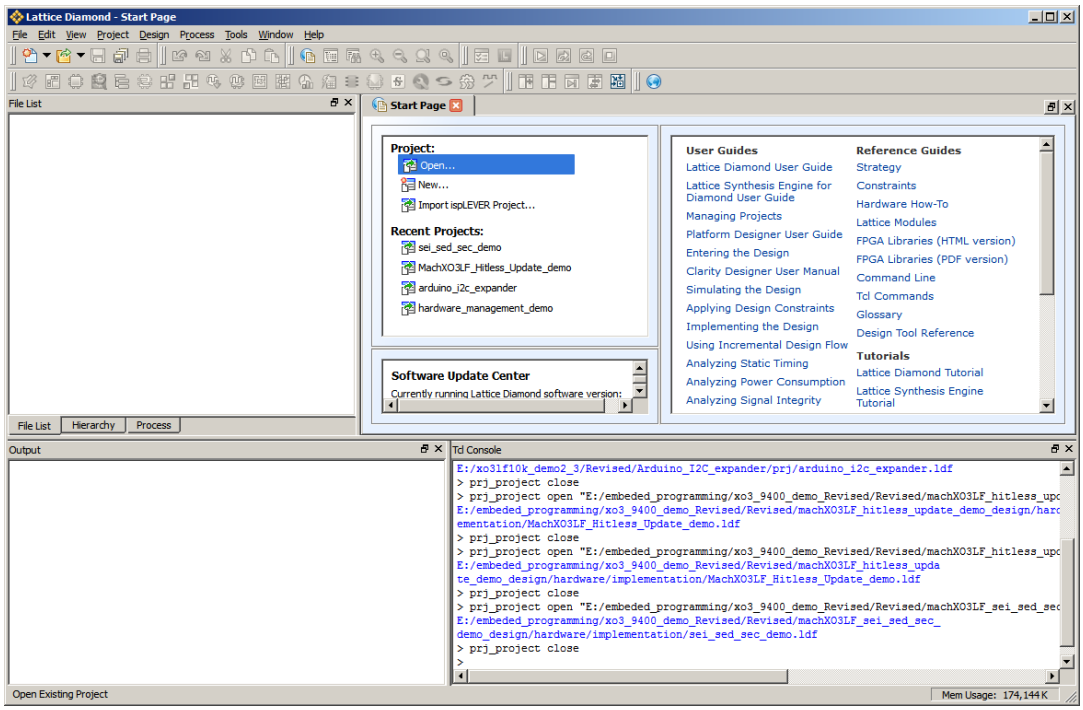

Figure 7.1. Opening Design File

2. The opened project is shown in Figure 7.2.

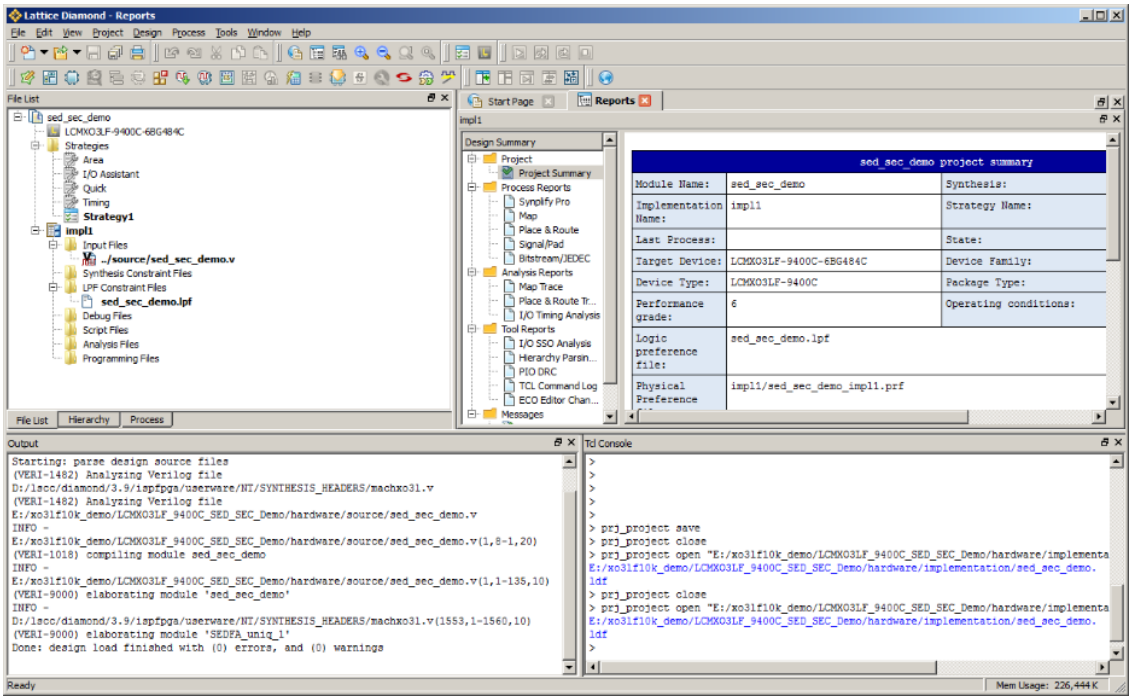

Figure 7.2. Opened Project

3. Synthesize the design to make sure there are no errors. Refer to the user guide of your Lattice Diamond software for details on how to perform this process.
4. When you see a green check mark on the **Synthesize Design** checkbox, click **Spreadsheet View**. In the Global Preferences tab, set SDM_PORT to **PROGRAMN** and BACKGROUND_RECONFIG to **ON** as shown in Figure 7.3.

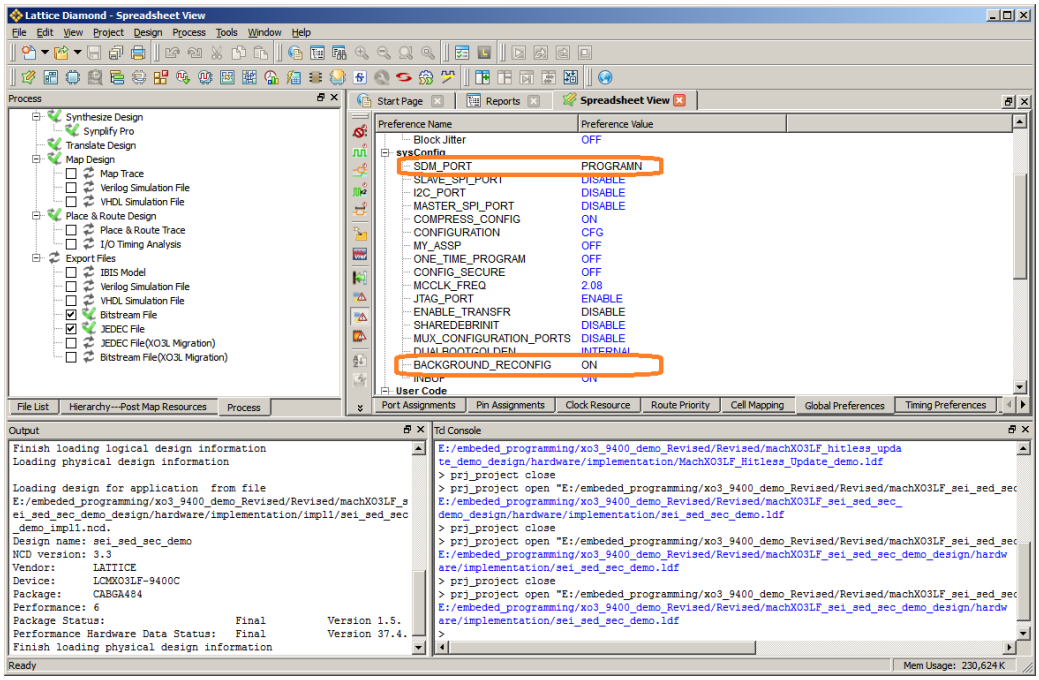

Figure 7.3. Setting Global Preferences

5. Click **File > Save** to save your settings.
6. In the Process view, under the Export Files process select the **Bitstream File** and **JEDEC File** checkboxes as shown in Figure 7.4.
7. Double click the **Export Files** process to run the export feature.
8. From the menu bar, select **Process > Rerun All** to run the entire process. This generates the bitstream file and the JED file.

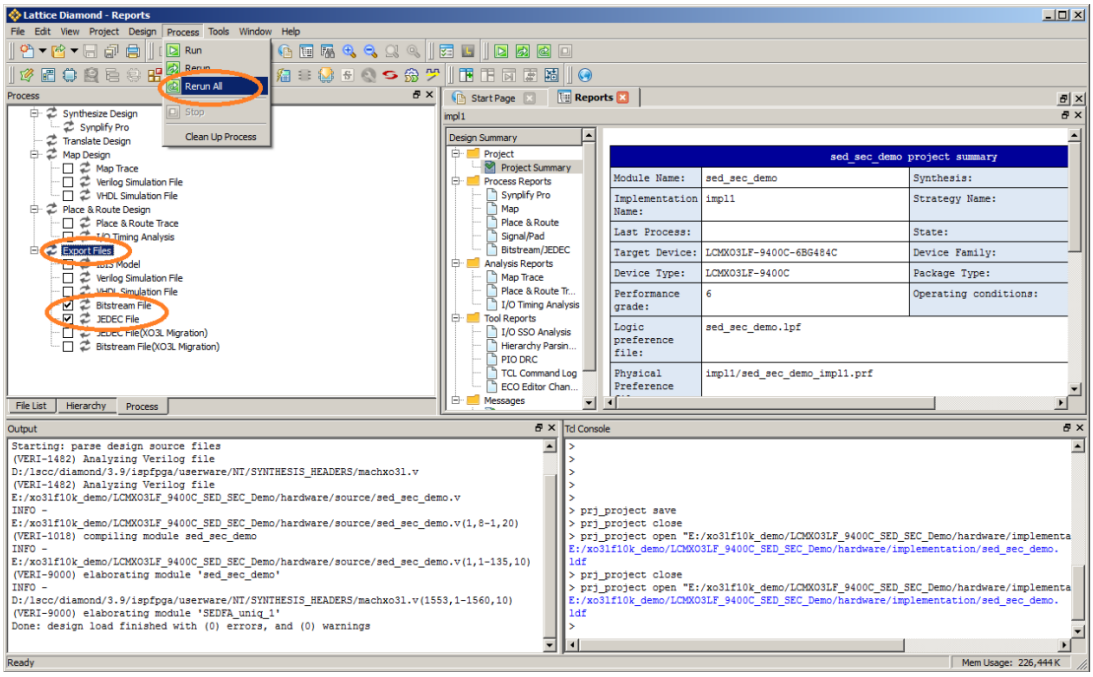

Figure 7.4. Generating the Bitstream

The SEI bitstream for error injection can be generated using the SEI Editor tool as shown in Figure 7.5.

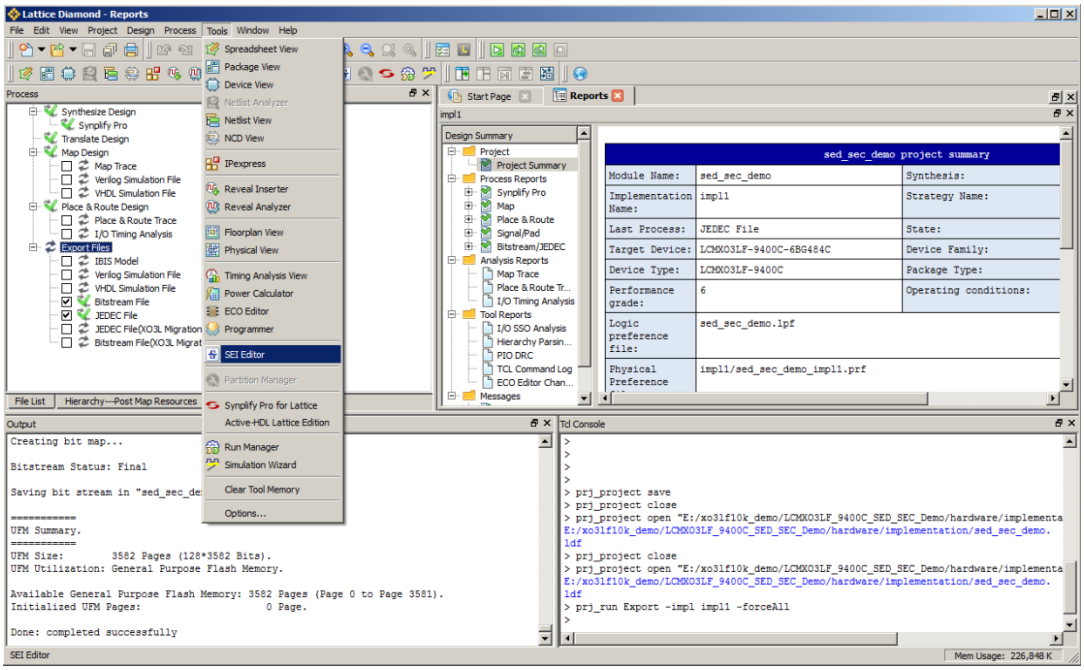

Figure 7.5. Generating the SEI Bitstream

The rebuilt design can be programmed in the FPGA as described in the Running the Demo section.

References

- DS1047 – MachXO3 Family Data Sheet
- FPGA-EB-02004 – MachXO3LF 9400 Development Board User Guide
- TN1292 – MachXO3 SED Usage Guide
- TN1279 – MachXO3 Programming and Configuration Usage Guide

Technical Support

For assistance, submit a technical support case at www.latticesemi.com/techsupport.

Revision History

Date	Version	Change Summary
June 2017	1.0	Initial release.

7th Floor, 111 SW 5th Avenue
Portland, OR 97204, USA
T 503.268.8000
www.latticesemi.com