LCMXO3LF-9400C Simple Hardware Management Demo

User Guide

FPGA-UG-02021 Version 1.0

June 2017

Contents

Figures

Tables

1. Introduction

Hardware management capability is one of the main features of the MachXO3-9400 Development Board. The board features the L-ASC10 (L-Analog Sense and Control 10 rails) device, an in-system programmable hardware management expander. It is ideal for system level complex power management, which includes power monitoring, reset control and flexible power sequence control. It is also capable of monitoring and adjusting the environment temperature, and logging faults under control of MachXO3™ or other Lattice Semiconductor FPGAs.

The LCMXO3LF-9400C Simple Hardware Management Demo highlights the hardware resource management features of the MachXO3-9400 development board. The demo covers the following operations:

- Configuring and controlling L-ASC10 through MachXO3 using the Lattice Diamond® Platform Designer tool without HDL coding.
- Setting voltage and temperature monitoring in Platform Designer.
- Designing control sequence without RTL coding using the Logic Builder tool.
- Using a user Verilog file to import design to the interface design environment.
- Using the Platform Manager I^2C interface and operate all registers in L-ASC10 through the I^2C bus.

1.1. Demo Design Overview

The LCMXO3LF-9400C Simple Hardware Management Demo design consists of the following sections:

- Power monitor setting
  - Configuring L-ASC10 power/temperature monitoring functionalities in Platform Designer
- Power On/Off sequencing
  - Simulating power ON/OFF using LEDs
  - Sequencing and delay control using the Logic Builder tool in Platform Designer
- Operating registers in L-ASC10 through I^2C bus using Platform Manager I2C Utility tool
- Protecting L-ASC10 from being written over I^2C bus

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.

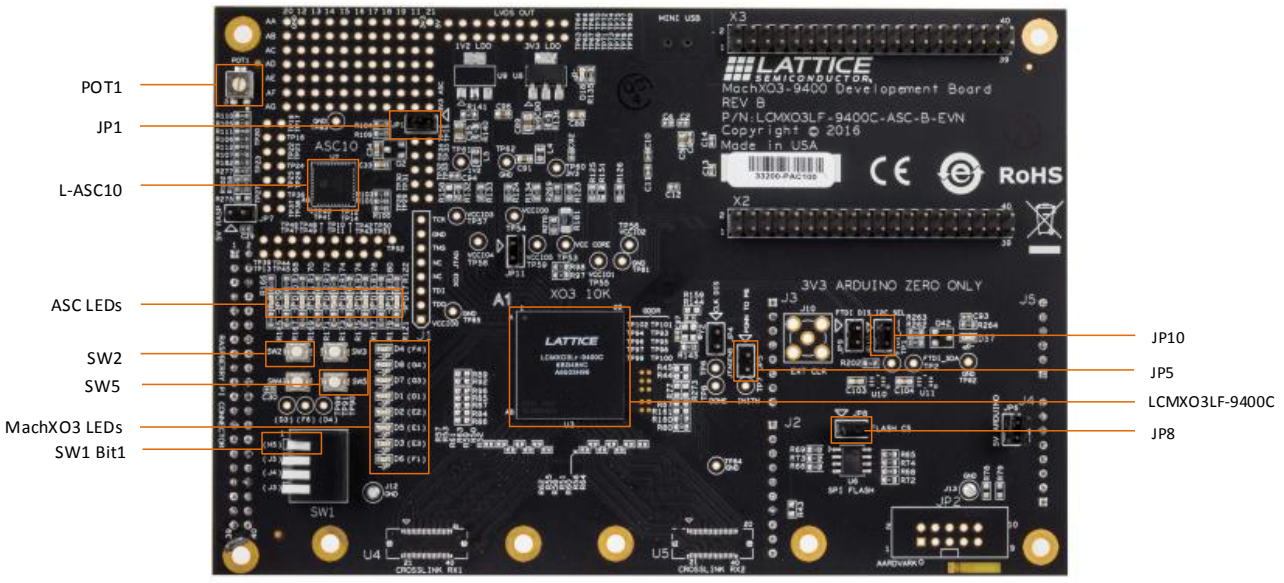


Figure 1.1. MachXO3-9400 Development Board

- JP1 must be set to provide power to the L-ASC10 device.
- JP5 should be set for connecting MachXO3 PROGRAM pin to SW5 so that the MachXO3 can be reconfigured by pressing SW5.
- JP10 is used to enable the LPTM2 I2C Utility tool to access the L-ASC10 device.
- SW1, DIP Switch, SW1 bit1 is used for protecting the L-ASC10 device from being written over the I^2C bus.
- SW2 is used for starting sequencing/reverse-sequencing demo.
- SW5 is used to drive MachXO3 PROGRAMN low for reconfiguration when JP5 is set.
- POT1 is used as part of the power monitoring demo and the L-ASC10 I^2C access demo.
- ASC_LED1 (D9) driven by L-ASC10/MachXO3 is used to indicate whether the L-ASC10 device is protected from being written over the I^2C bus controlled by DIP Switch SW1 bit1. When SW1 bit1 is pushed down, D9 is ON. When SW1 bit1 is pulled up, D9 is OFF.
- ASC_LED2~9(D10~D17), which are driven by L-ASC10, simulate the powers to be enabled (ON) or disabled (OFF).
- MachXO3 LEDs (D4, D8, D7, D1, D2, D5), which are driven by MachXO3, indicate the power status of +1.2 V, VCC1_8FT (1.8 V from FT2232H), +3.3V_RASP (3.3 V from Raspberry Pi), +3.3V_AR (3.3 V from Arduino), VBUS_5V and POT1. After the board is programmed, MachXO3 LED D6 starts to blink. JP8 should be removed during the demo.

2. Functional Description

Figure 2.1 shows the block diagram detailing all available resources for the demo.

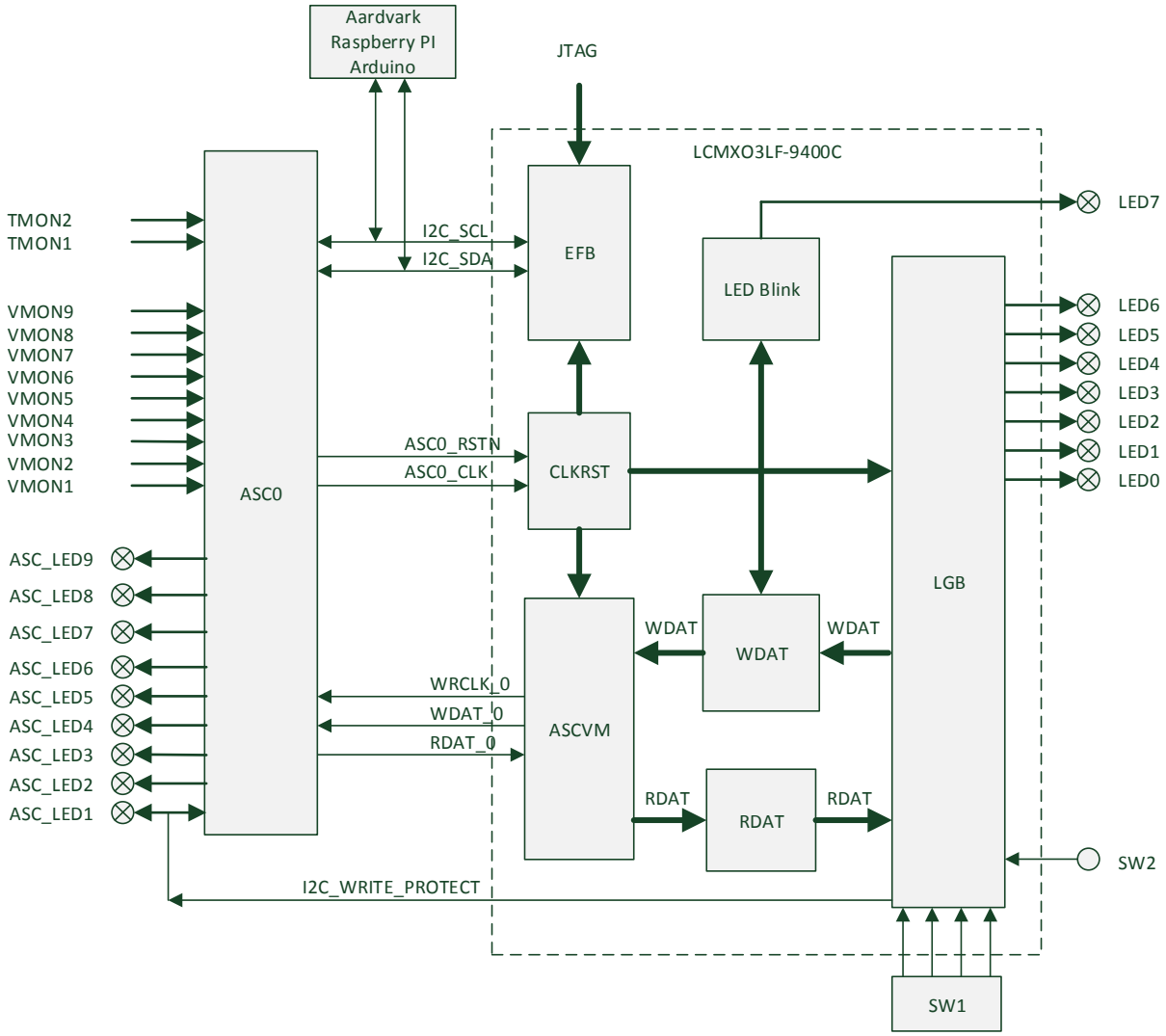


Figure 2.1. Simple Hardware Management Demo Block Diagram

- The L-ASC10 device and the LCMXO3LF-9400C device communicate with each other through a 3-wire bidirectional synchronous serial interface.
- The I^2C link is used to configure the L-ASC10 device or access and control the L-ASC10 device through the PC / Raspberry PI / Arduino or Aardvark cable.
- L-ASC10 provides a reset and an 8 MHz clock to the LCMXO3LF-9400C device.
- L-ASC10 samples and monitors the power input at VMON1~9 and temperature at TMON1~2 and stores them to internal registers.
- Most of the control logic in LCMXO3LF-9400C are automatically generated by the Lattice Diamond design tool based on user configuration.
- The LGB block is generated through the Logic Builder tool integrated in Lattice Diamond. You can easily create the sequence control logic through clicking the mouse without RTL coding.
- The Logic Builder tool also supports importing user Verilog HDL modules. A simple LED blink module is used to demo the process.

3. Demo Package

The demo package includes the following:

- Verilog source code for the demo logic design
- Lattice Diamond project and preference files
- JED file for programming the MachXO3 internal configuration flash
- HEX file for programming L-ASC10 internal configuration EEPROM

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 and L-ASC10 devices

3.2. Software Requirements

To run the demo, the following software are required:

- Lattice Diamond version 3.9 or later
- Lattice Mico System for Lattice Diamond 3.9 or later
- Lattice Diamond Programmer software for bitstream downloading
- Platform ManagerI2C Utility Tool(Under the tool directory of the design package)

Note: The Lattice Diamond and Lattice Mico System software programs are available at www.latticesemi.com/en/Products/DesignSoftwareAndIP.

4. Port Assignments and Descriptions

Table 4.1 shows all ports definitions in the design.

Table 4.1. Design Port Definitions

Port Name	Direction	Description
ASC0_RSTN	Input	External Reset input from L-ASC0, active low
ASC0_CLK	Input	8 MHz clock input from L-ASC10
SW2	Input	Button input
DIPSW1	Input	SW1 bit1. Drive I2C_WRITE_PROTECT inside FPGA
I2C_WRITE_PROTECT	Output	Driven by SW1 bit1(DIPSW1) for protecting L-ASC10 from being written over I^2C bus
WRCLK_0	Output	ASC interface for communicating between MachXO3 and L-ASC10
WDAT_0	Output	
RDAT_0	Input	
I2C_SCL	Inout	I^2C bus clock
I2C_SDA	Inout	I^2C bus data
ASC_LED1(D9)	Output	Driven by DIPSW1 via I2C_WRITE_PROTECT. Indicates that L-ASC10 is protected when it is ON and unprotected when it is OFF
ASC_LED2~9(D10-D17)	Output	Used for demo sequencing/reverse sequencing control. Simulates the power on sequence when they are ON one by one from D10 to D17. Simulates the power off sequence when they are OFF one by one in the reversed order
LED0~5(D4,D8,D7,D1,D2,D5)	Output	Power status of +1.2V, VCC1_8FT, +3.3V_RASP, +3.3V_AR, VBUS_5V and POT1. Indicate the voltage is within the range set when they are ON and out of the range when they are OFF
LED7 (D6)	Output	Blinks after MachXO3/L-ASC10 is programmed

5. Demo Package Directory Structure

Figure 5.1 shows the demo package directory structure.

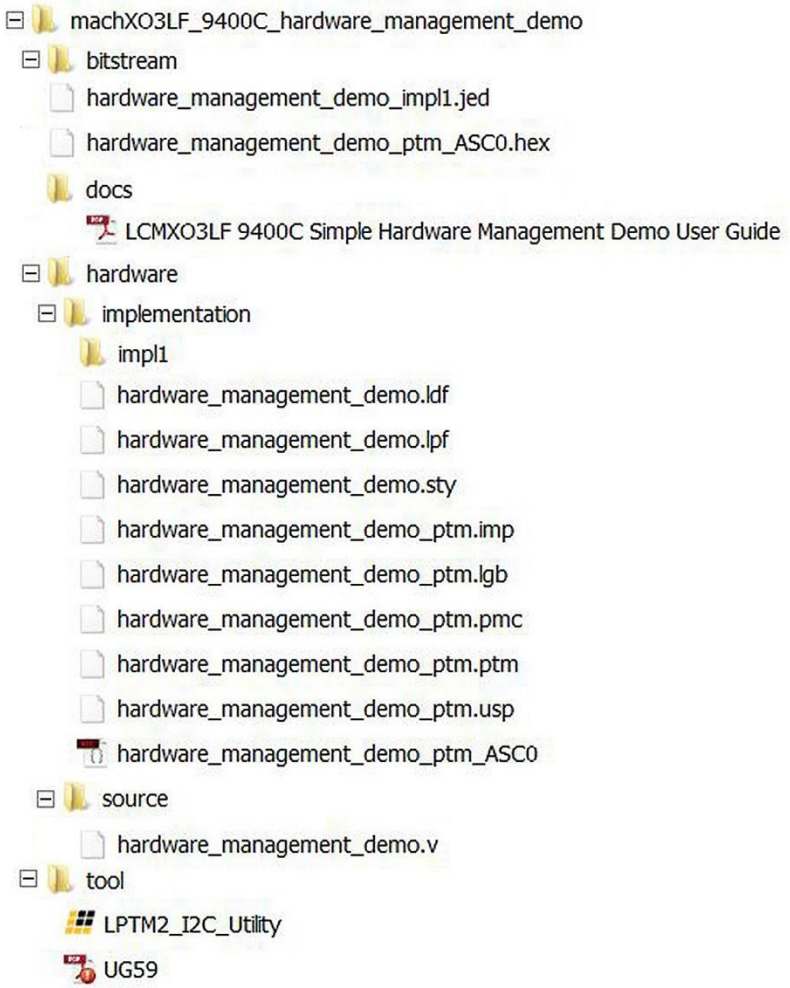


Figure 5.1. Demo Package Directory Structure

The *bitstream* folder includes the programming files for the demo.

The *docs* folder includes this document.

The *implementation* folder contains working directories of Lattice Diamond and Platform Manager. It includes the project files. The *source* sub folder includes a simple Verilog HDL file for demostrating source file importing.

The *tool* folder includes the Platform Manager I2C Utility tool for the demo.

6. Configuring the Project in Platform Designer

This section introduces the basic operations in Platform Designer for L-ASC10 device configuration and control. For more details, please refer to the Lattice Diamond Platform Designer User Guide.

6.1. Global Settings

Figure 6.1 shows the settings for boot mode and ASC write protection.

Boot mode is either set as dual boot or normal. For this demo, leave Boot Mode as **Normal**, which is the default setting. The ASC write protection option allows you to use MachXO3 GPIO to enable or disable ASC I2C write protection. In ASC I2C Write Feature, select **Controlled by ASCx_GPIO1**.

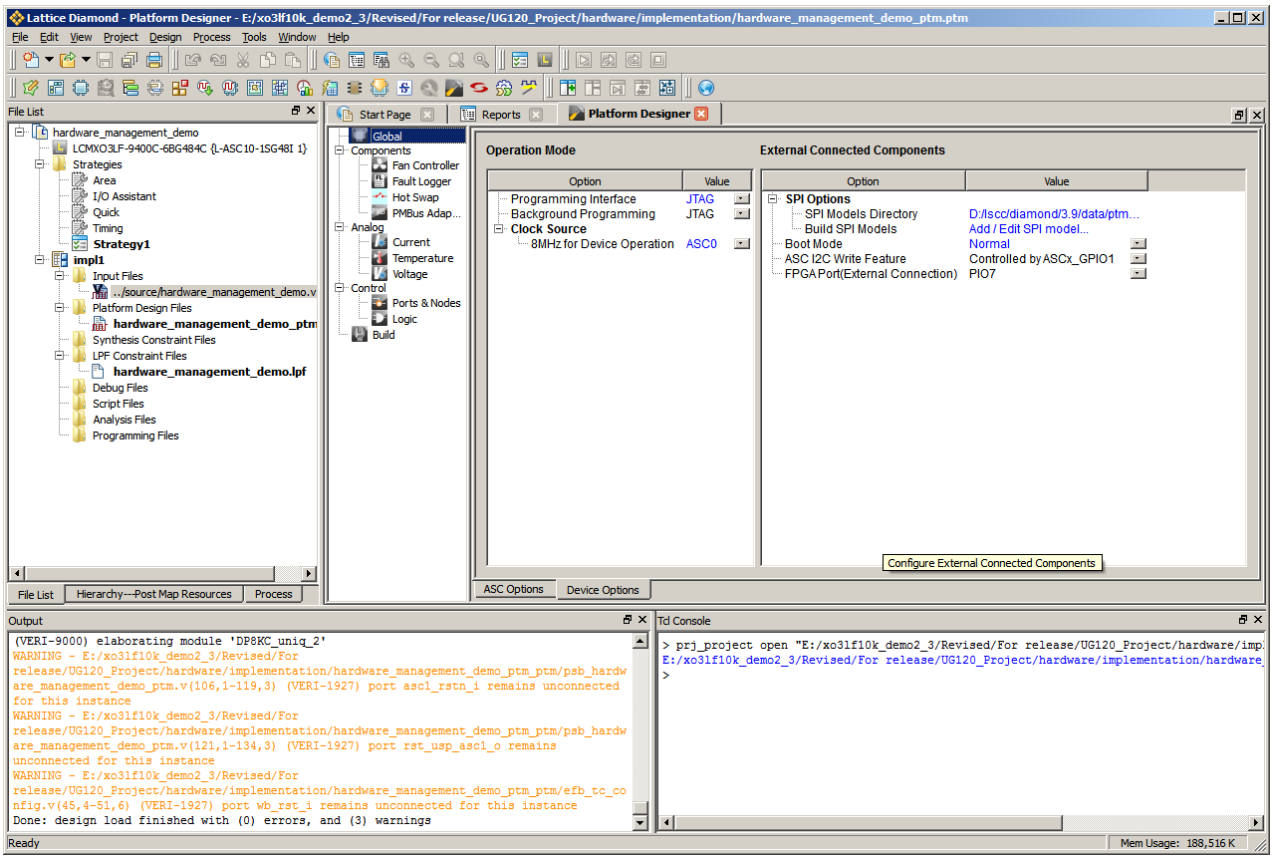


Figure 6.1. Boot Mode and ASC I2C Write Feature Settings

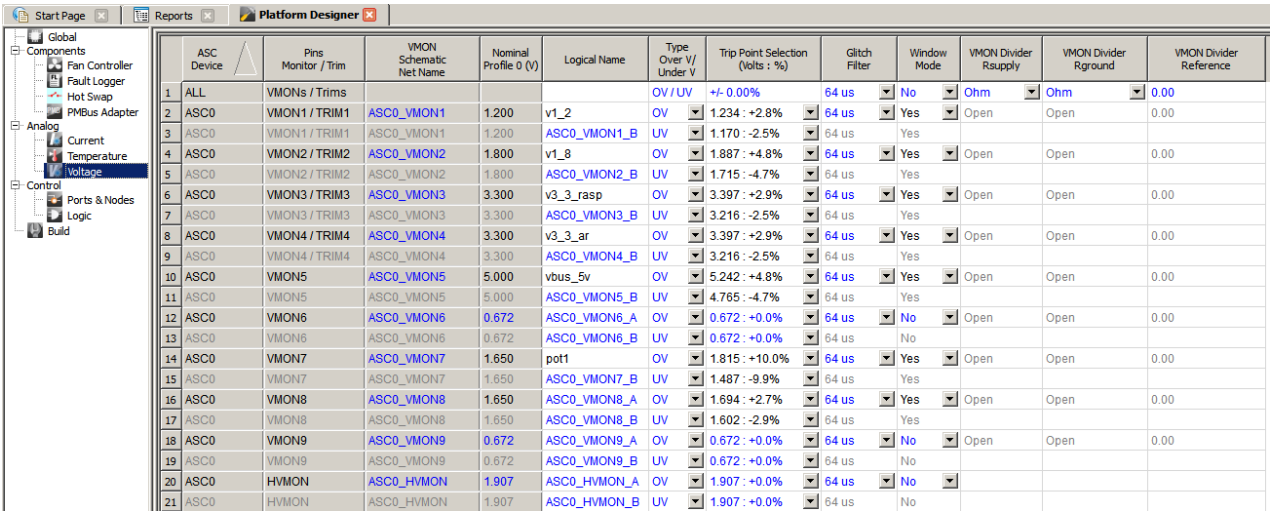
6.2. Voltage Monitor Settings

Figure 6.2 shows the interface for monitoring power voltage connected to the VMON ports of the device in window mode. The monitor outputs “1” when the voltage is within the set window based on the nominal voltage. The monitor outputs “0” when the voltage is out of the set window.

For example:

- The OV voltage for 1.2 V is set to 1.2 +2.8% (1.234 V) and UV voltage is set to 1.2 -2.5% (1.117 V). In this case, ASC0_VMON1 outputs “1” when V_1.2 V input is between 1.117~1.234V. The output becomes “0” when V_1.2 V is greater than 1.234 V or smaller than 1.117 V.
- V1_8, V3_3_rasp, V3_3_ar, Vbus_5v are set based on their nominal value in the same way

- POT1 is pulled to V3.3 on the board, the OV/UV trip points are respectively set to 3.3/2 +10% and 3.3/2 -10%. This means that the monitor outputs “1” when POT1 is adjusted to the window and “0” when POT1 is adjusted to leave the window. The trip points can be set to any value for testing the behavior of the monitor (comparator).



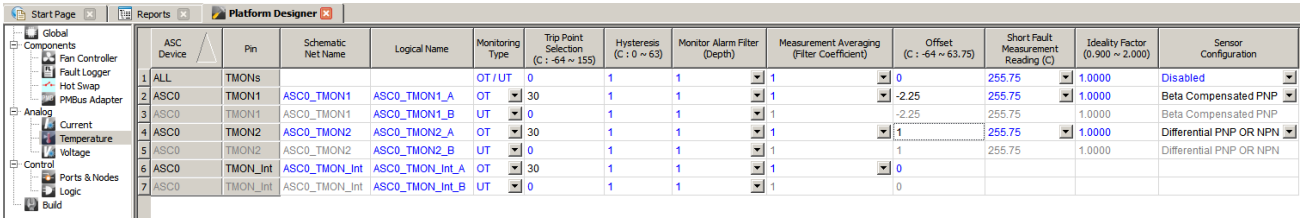
	ASC Device	Pins Monitor / Trim	VMON Schematic Net Name	Nominal Profile 0 (V)	Logical Name	Type Over V/ Under V	Trip Point Selection (Volts : %)	Glitch Filter	Window Mode	VMON Divider Rsupply	VMON Divider Rground	VMON Divider Reference
1	ALL	VMONs / Trims				OV / UV	+/- 0.00%	64 us	No	Ohm	Ohm	0.00
2	ASC0	VMON1 / TRIM1	ASC0_VMON1	1.200	v1_2	OV	1.234 : +2.8%	64 us	Yes	Open	Open	0.00
3	ASC0	VMON1 / TRIM1	ASC0_VMON1	1.200	ASC0_VMON1_B	UV	1.170 : -2.5%	64 us	Yes			
4	ASC0	VMON2 / TRIM2	ASC0_VMON2	1.800	v1_8	OV	1.887 : +4.8%	64 us	Yes	Open	Open	0.00
5	ASC0	VMON2 / TRIM2	ASC0_VMON2	1.800	ASC0_VMON2_B	UV	1.715 : -4.7%	64 us	Yes			
6	ASC0	VMON3 / TRIM3	ASC0_VMON3	3.300	v3_3_rasp	OV	3.397 : +2.9%	64 us	Yes	Open	Open	0.00
7	ASC0	VMON3 / TRIM3	ASC0_VMON3	3.300	ASC0_VMON3_B	UV	3.216 : -2.5%	64 us	Yes			
8	ASC0	VMON4 / TRIM4	ASC0_VMON4	3.300	v3_3_ar	OV	3.397 : +2.9%	64 us	Yes	Open	Open	0.00
9	ASC0	VMON4 / TRIM4	ASC0_VMON4	3.300	ASC0_VMON4_B	UV	3.216 : -2.5%	64 us	Yes			
10	ASC0	VMON5	ASC0_VMON5	5.000	vbus_5v	OV	5.242 : +4.8%	64 us	Yes	Open	Open	0.00
11	ASC0	VMON5	ASC0_VMON5	5.000	ASC0_VMON5_B	UV	4.765 : -4.7%	64 us	Yes			
12	ASC0	VMON6	ASC0_VMON6	0.672	ASC0_VMON6_A	OV	0.672 : +0.0%	64 us	No	Open	Open	0.00
13	ASC0	VMON6	ASC0_VMON6	0.672	ASC0_VMON6_B	UV	0.672 : +0.0%	64 us	No			
14	ASC0	VMON7	ASC0_VMON7	1.650	pot1	OV	1.815 : +10.0%	64 us	Yes	Open	Open	0.00
15	ASC0	VMON7	ASC0_VMON7	1.650	ASC0_VMON7_B	UV	1.487 : -9.9%	64 us	Yes			
16	ASC0	VMON8	ASC0_VMON8	1.650	ASC0_VMON8_A	OV	1.694 : +2.7%	64 us	Yes	Open	Open	0.00
17	ASC0	VMON8	ASC0_VMON8	1.650	ASC0_VMON8_B	UV	1.602 : -2.9%	64 us	Yes			
18	ASC0	VMON9	ASC0_VMON9	0.672	ASC0_VMON9_A	OV	0.672 : +0.0%	64 us	No	Open	Open	0.00
19	ASC0	VMON9	ASC0_VMON9	0.672	ASC0_VMON9_B	UV	0.672 : +0.0%	64 us	No			
20	ASC0	HVMON	ASC0_HVMON	1.907	ASC0_HVMON_A	OV	1.907 : +0.0%	64 us	No			
21	ASC0	HVMON	ASC0_HVMON	1.907	ASC0_HVMON_B	UV	1.907 : +0.0%	64 us	No			

Figure 6.2. Voltage Monitor Settings

6.3. Temperature Monitor Settings

Figure 6.3 shows the interface for temperature monitoring. Two transistors are connected to the L-ASC10 TMON input as temperature sensors for measuring external temperature.

- TMON1 should be configured as Beta Compensated PNP and TMON2 should be configured Differential PNP or NPN.
- Offset options are used as calibration.
- Trip point setting impacts the output of the comparators of each channel.



	ASC Device	Pin	Schematic Net Name	Logical Name	Monitoring Type	Trip Point Selection (C : -64 ~ 155)	Hysteresis (C : 0 ~ 63)	Monitor Alarm Filter (Depth)	Measurement Averaging (Filter Coefficient)	Offset (C : -64 ~ 63.75)	Short Fault Measurement Reading (C)	Ideality Factor (0.900 ~ 2.000)	Sensor Configuration
1	ALL	TMONs			OT / UT	0	1	1	1	0	255.75	1.0000	Disabled
2	ASC0	TMON1	ASC0_TMON1	ASC0_TMON1_A	OT	30	1	1	1	-2.25	255.75	1.0000	Beta Compensated PNP
3	ASC0	TMON1	ASC0_TMON1	ASC0_TMON1_B	UT	0	1	1	1	-2.25	255.75	1.0000	Beta Compensated PNP
4	ASC0	TMON2	ASC0_TMON2	ASC0_TMON2_A	OT	30	1	1	1	1	255.75	1.0000	Differential PNP OR NPN
5	ASC0	TMON2	ASC0_TMON2	ASC0_TMON2_B	UT	0	1	1	1	1	255.75	1.0000	Differential PNP OR NPN
6	ASC0	TMON_Int	ASC0_TMON_Int	ASC0_TMON_Int_A	OT	30	1	1	1	0			
7	ASC0	TMON_Int	ASC0_TMON_Int	ASC0_TMON_Int_B	UT	0	1	1	1	0			

Figure 6.3. Temperature Monitor Settings

6.4. Ports and Nodes Configuration

Ports and node can be configured in Platform Designer to provide each signal in the design a specific name. The Port signals defined here are used for control logic building or for connecting ports of imported user HDL files. The Node signals are used for internal flags or status recording during programming in the Logic editor. Figure 6.4 and Figure 6.5 show the interface for configuring ports and nodes.

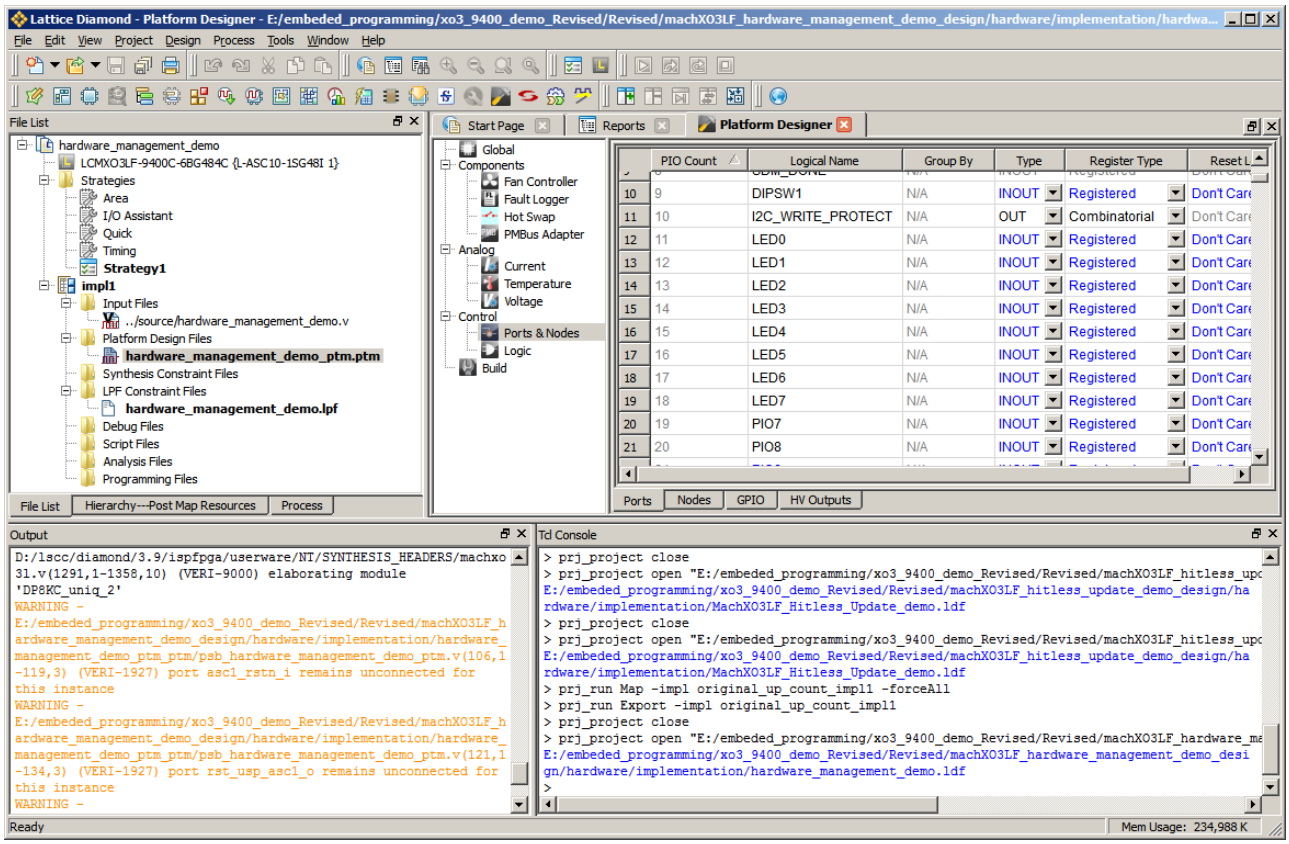


Figure 6.4. Ports Definition

In this project, a node POWER_OFF is defined as shown in Figure 6.5.

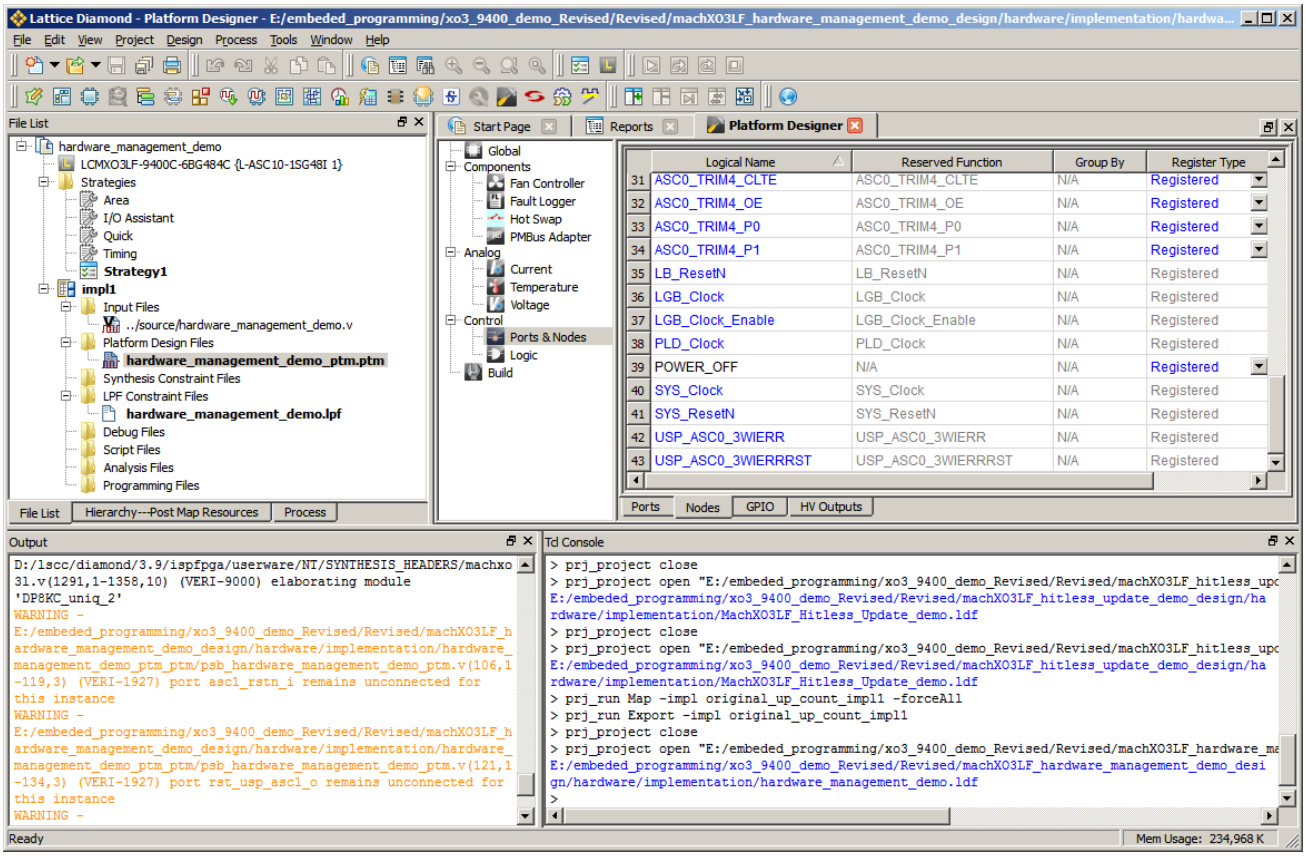


Figure 6.5. Nodes Definition

6.5. Controlling Logic in Platform Designer Logic Editor

The Logic Editor tool in Platform Designer allows you to create control logic without RTL coding. You can open the project in Diamond and use the design to learn how to create control logic in Logic Editor.

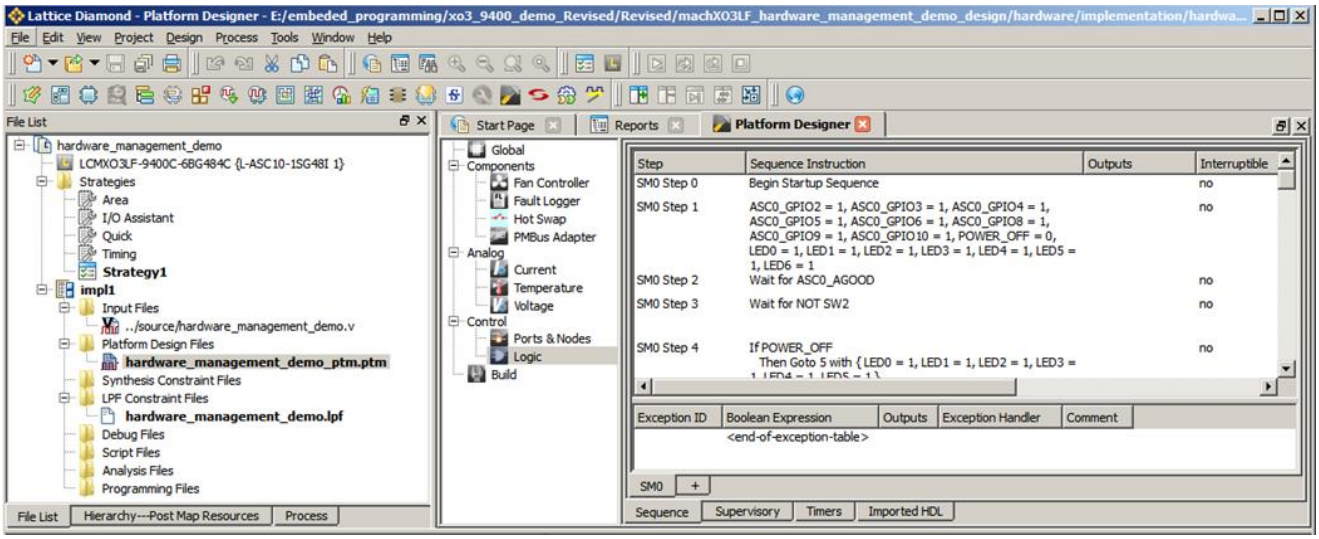


Figure 6.6. Building Control Logic

6.6. Importing User Verilog File to Platform Designer

Platform Designer allows you to import an HDL module to the design.

To import a HDL module:

1. Add the HDL file to Diamond as shown in Figure 6.7.
2. Check **Enable Imported HDL**.
3. Enter **Module Name** and **Instance Name**.
4. Connect the defined port signals with ports in the HDL module

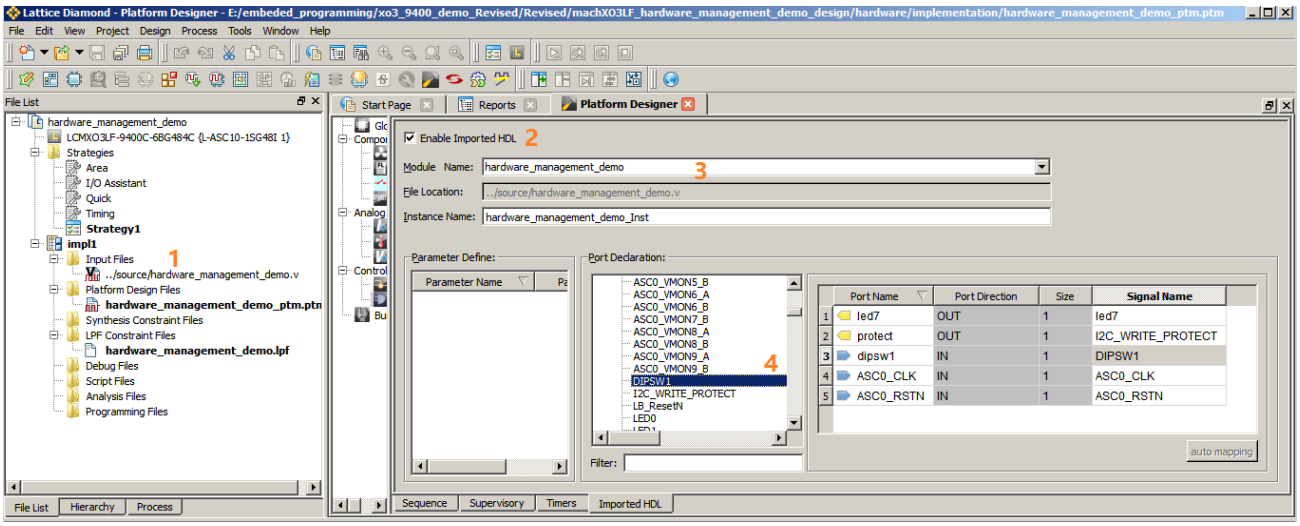


Figure 6.7. Importing HDL Module to Platform Designer

7. Building the Demo

After completing all configurations and building the control logic, click the **DRC**, **Compile**, **Pin Assignment**, and **Generate Jedec** buttons (in that order) to generate the necessary files for programming MachXO3 and L-ASC10. The Build interface is shown in Figure 7.1.

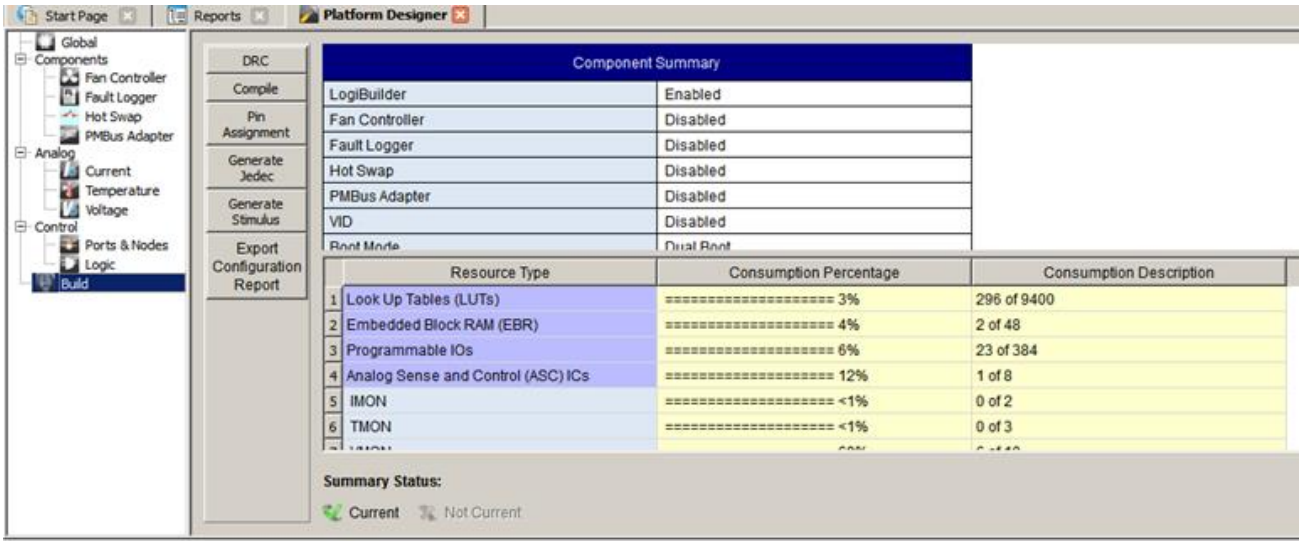


Figure 7.1. Building the Project

8. Running the Demo

8.1. Programing L-ASC10

To program L-ASC10:

1. Before powering on the board, make sure that JP1 is set for providing L-ASC10 power. JP5 should also be set and JP8 should be removed.
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**.

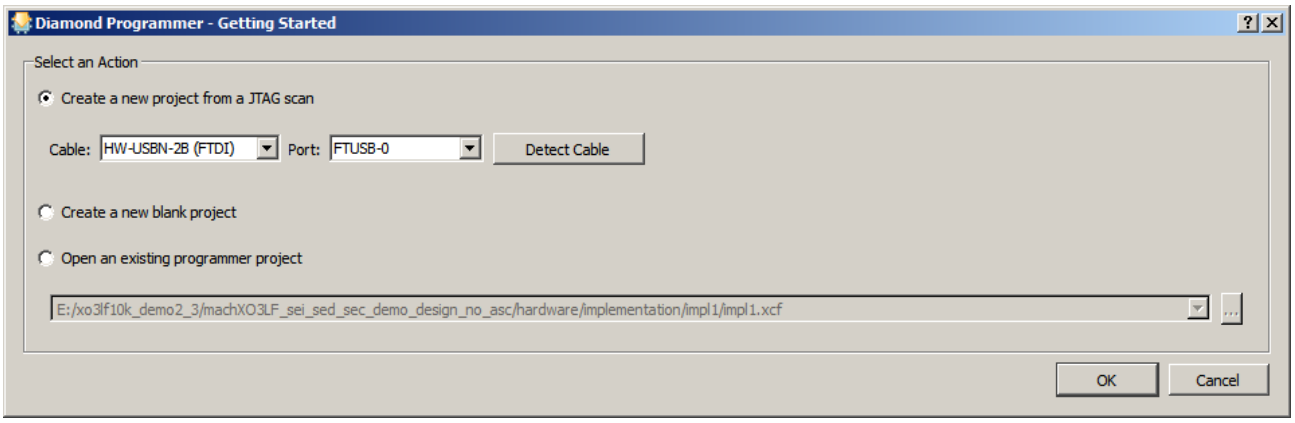


Figure 8.1. Creating a New Project

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.

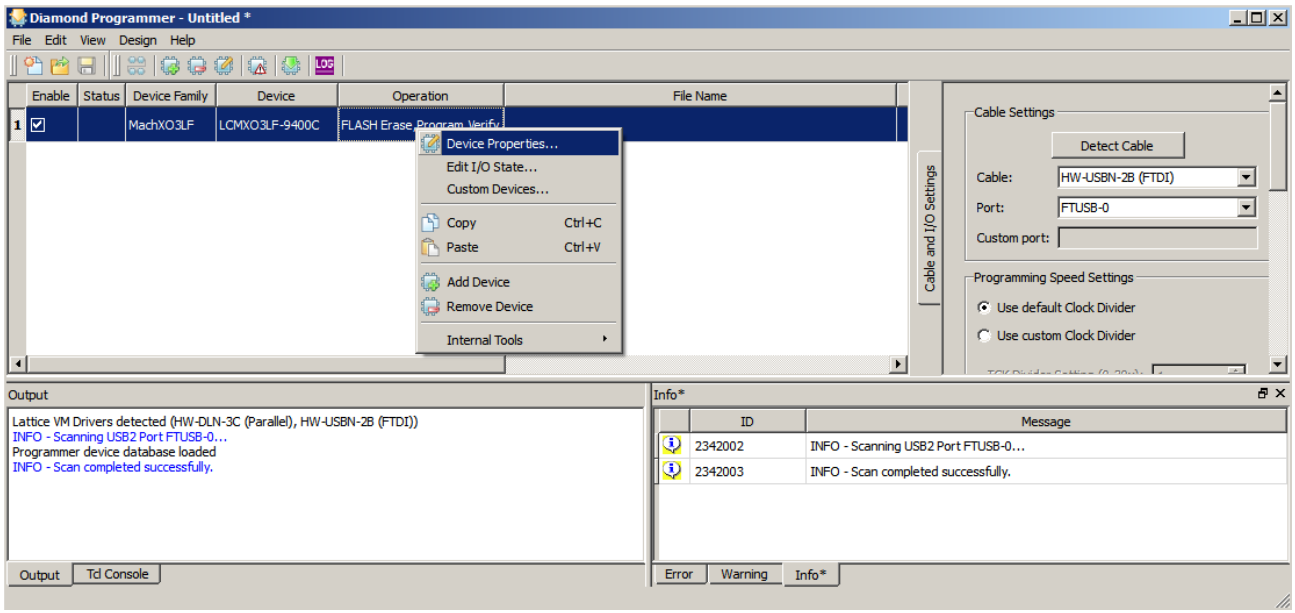


Figure 8.2. Opening Device Properties

5. Set device properties as shown in Figure 8.2.
   Open the Device Properties dialog box as shown in Figure 8.3. Under Device Operation, select **PTM Programming** in Access mode and **PTM Bypass** in Operation.

Under External ASC Options, add an external ASC device by selecting **External ASC Device #1**. Select the EEPROM programming file \bitstream\hardware_management_demo_ptm_ASC0.hex in File option. Select **ASC Erase, Program, Verify** in Operation option.
Click **OK**.

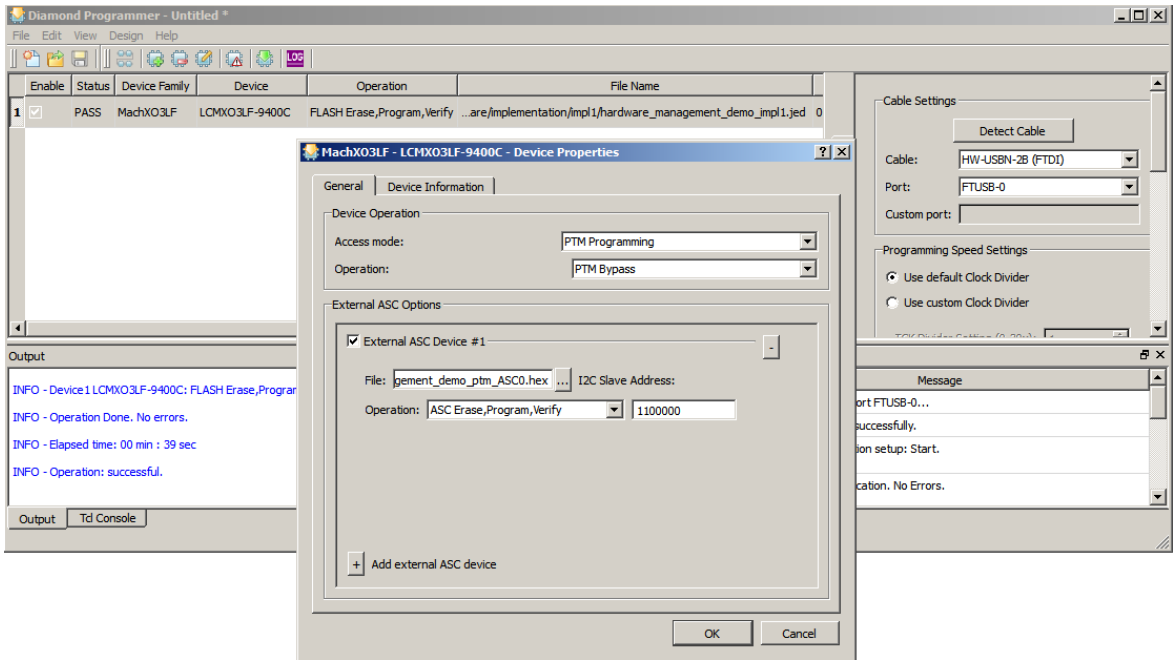


Figure 8.3. Setting L-ASC10 Device Properties

6. Click the **Program** button to start programming the L-ASC10 device as shown in Figure 8.4.

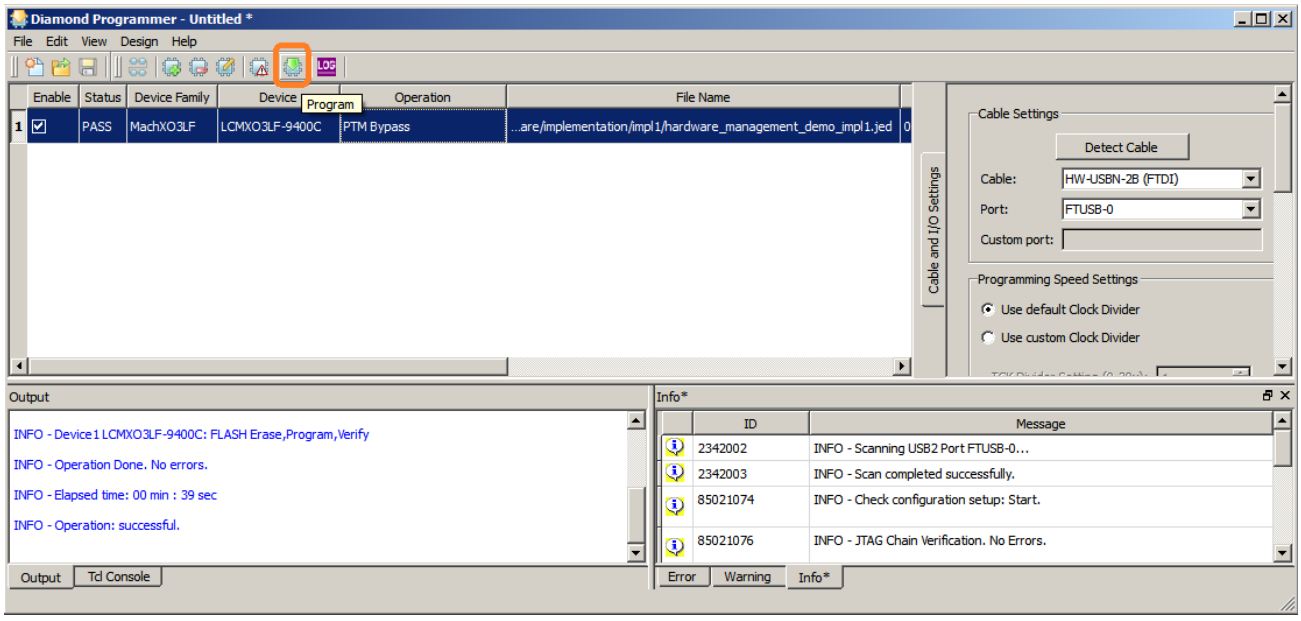


Figure 8.4. Starting Programming Operation

7. If programming is successful, “Operation: successful” is reported under Output.

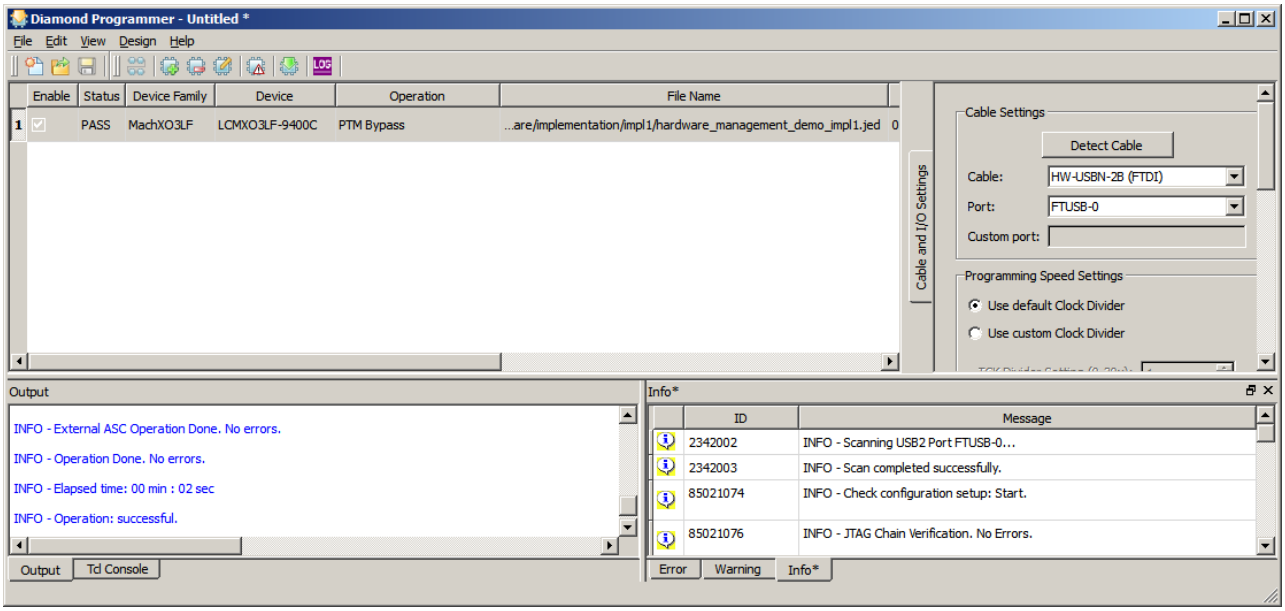


Figure 8.5. Programming Successfully

If the software reports “Failed to read the Device’s IDCODE…” reset the MachXO3 device feature row by erasing the MachXO3 flash and then retry the operation.

To erase the MachXO3 Flash, Open the Device Properties dialog box as shown in Figure 8.3. Under Device Operation , select **Flash Programming Mode** in Access mode and **Flash Erase Only** in Operation. Click **OK**. Then click the **Program** button as shown in Figure 8.4.

8.2. Programing MachXO3 Internal Configuration Flash

To program the MachXO3 device:

1. Right-click on the device and select **Device Properties** in the context menu. Then set device properties as shown in Figure 8.6.
   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\hardware_management_demo_impl1.jed** in Programming file. Click **OK**.

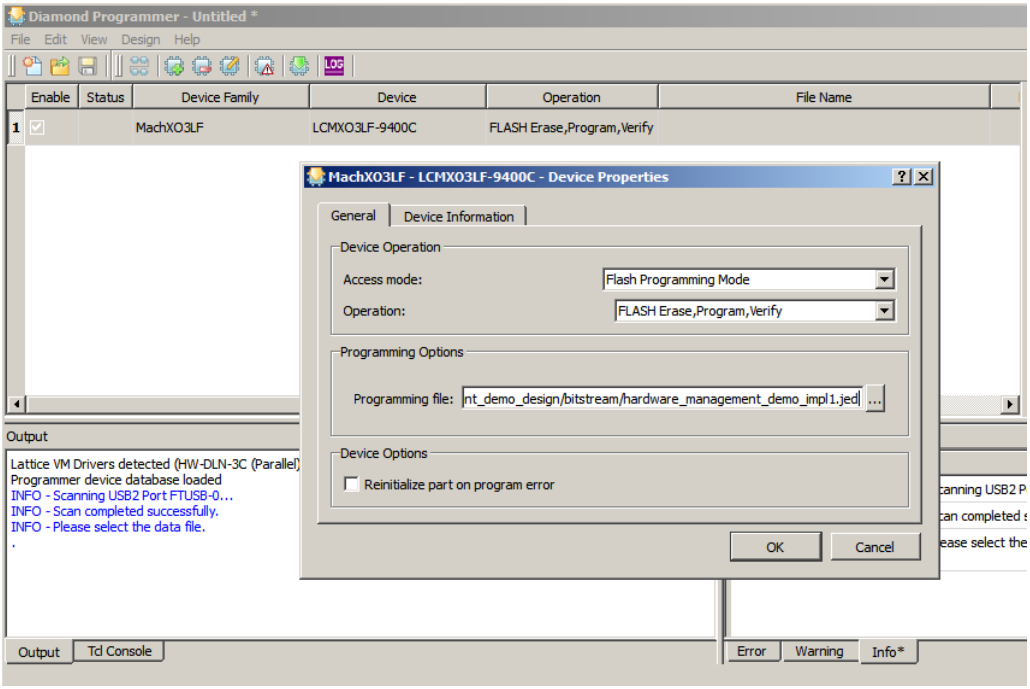


Figure 8.6. Setting MachXO3 Device Properties

2. Click the **Program** button as shown in Figure 8.7.

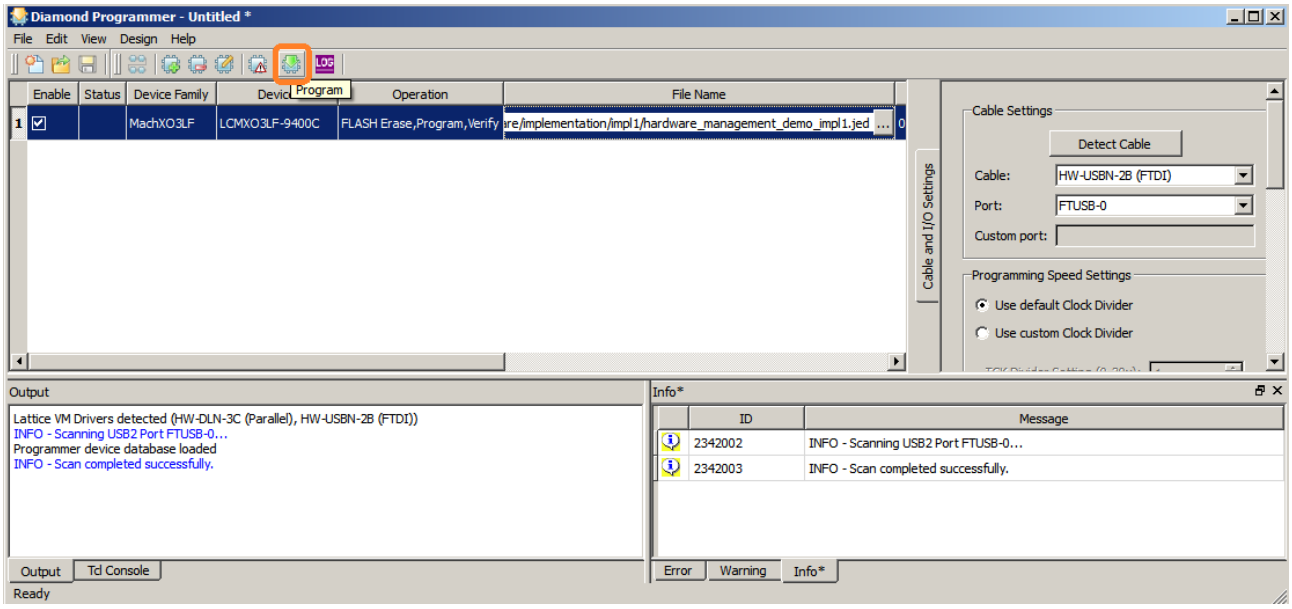


Figure 8.7. Starting Programming Operation

3. If programming is successful, “Operation: successful” is reported under Output. LED D6 starts to blink.

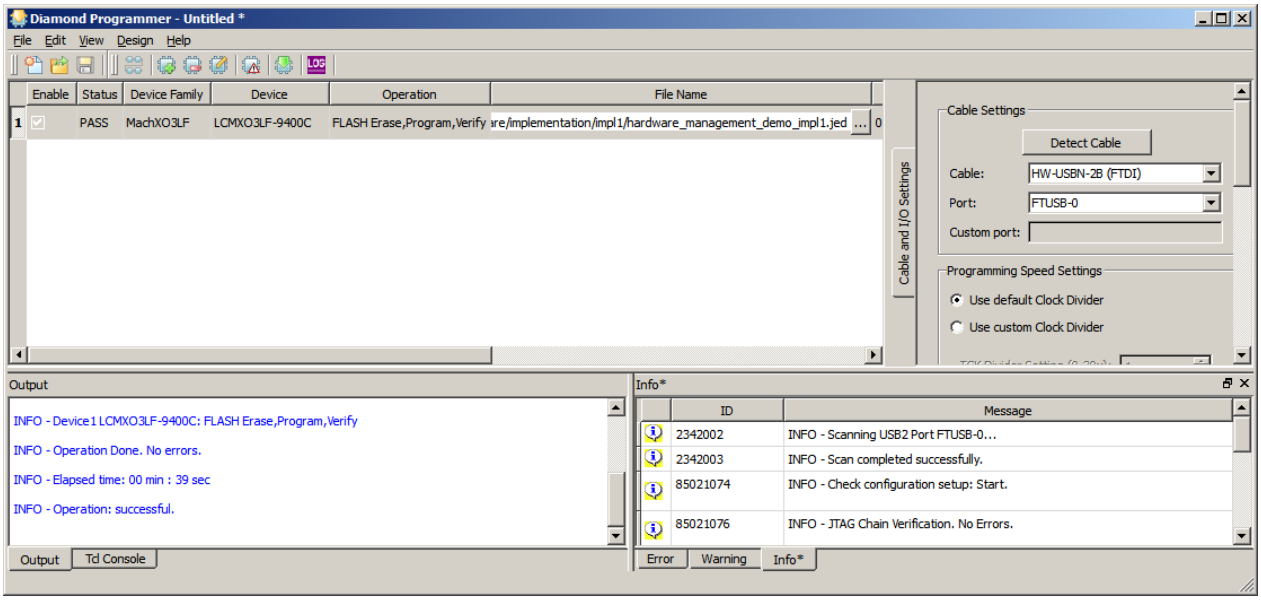


Figure 8.8. Programming Operation is Successful

8.3. Basic Demos

- When programming is done, LED D6 starts to blink, indicating that the board is working.
- When SW2 is pressed, LEDs D10~D17, driven by ASC GPIO, are turned ON one by one under the control of MachXO3. This demonstrates power up sequencing control.
- LEDs D4, D8, D7, D1, D2 and D5 are driven ON or OFF directly by MachXO3 based on the power monitor status of the L-ASC10. These are turned ON when the corresponding input VMONx is within the trip point settings window shown in Figure 6.2. These are turned OFF if the corresponding VMONx is out of the range.
- D5 represents the monitor status of the voltage from POT1. Adjusting POT1 causes the LED to turn ON or OFF when the voltage is within or out of the set range.
- Set JP10 and start the Lattice Platform Manager I2C Utility tool from the \machXO3LF_9400C_hardware_management_demo\tool folder. The LPTM I2C Power, Thermal & Control Plane Management dialog box opens, as shown in Figure 8.9. Click the **VMON** button. Please check JP10 if the software report **Required ASC device is not found**

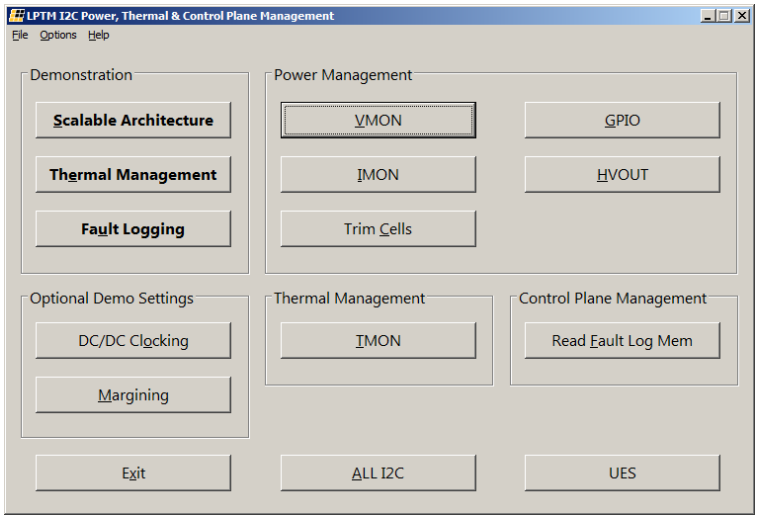


Figure 8.9. LPTM I2C Power, Thermal & Control Plane Management Dialog Box

The LPTM I2C VMON Voltages, Comparator Status and Threshold dialog box opens, as shown in Figure 8.10. All monitored voltages are displayed in the leftmost column. Click the **Read** button while POT1 is adjusted and VMON7 changes at the same time. When VMON7 is adjusted between 1.481 and 1.788, LED D5 is turned ON.

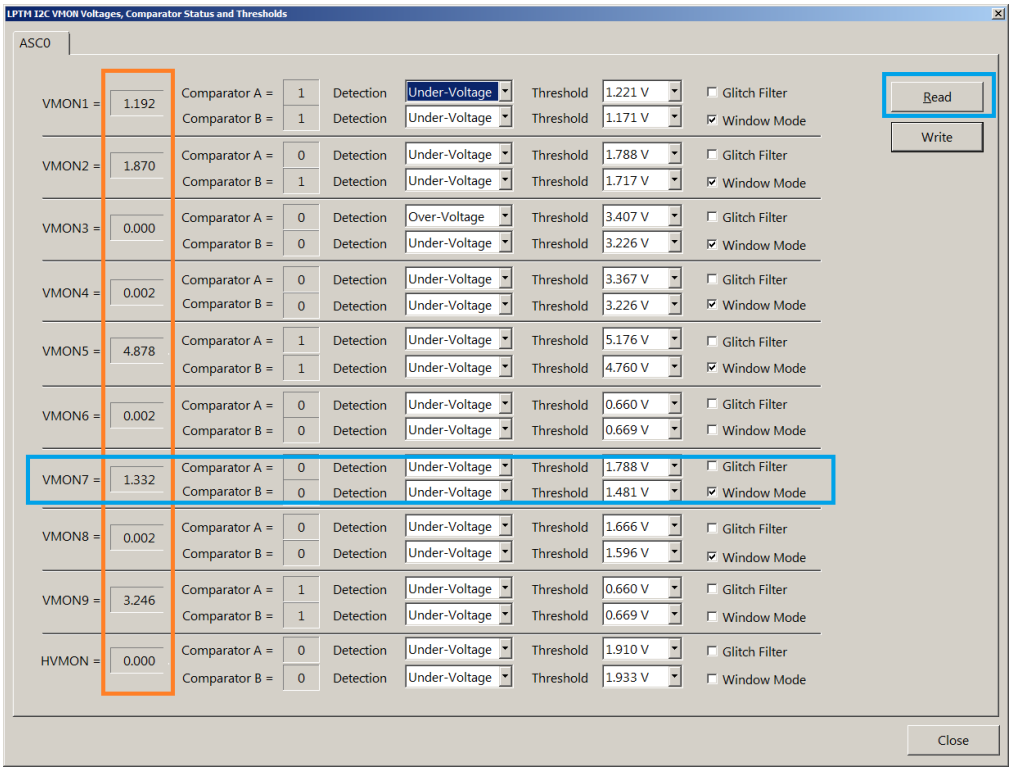


Figure 8.10. LPTM I2C VMON Voltages, Comparator Status and Threshold Dialog Box

- Push down SW1 bit1 to demonstrate I²C write protect. When SW1 bit1 is pushed down, LED D9 is turned ON.
- Adjust POT1 to turn OFF LED D5, assuming that VMON7 is **2.006** V as shown in Figure 8.11. Then set VMON Threshold of comparator A to **2.105** and click the **Write** button.
  There is no change to LED D5. This indicates that the I²C Write is protected.

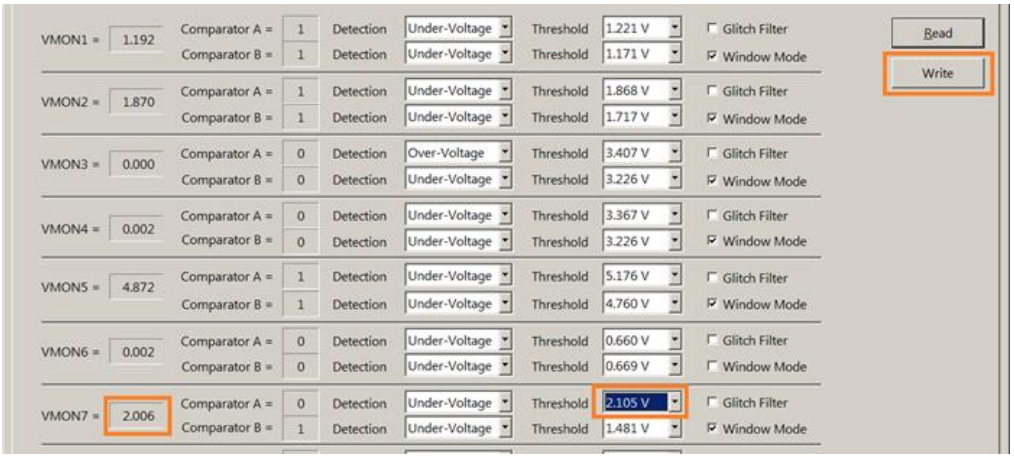


Figure 8.11. I2C Write Protect Test

- Pull up SW1 bit1 to release I²C Write protection and click the **Write** button again. LED D5 is turned ON again.
- Press SW2 again. LEDs D4, D8, D7, D1, D2, and D5 are driven OFF and LEDs D10~D17 are turned OFF one by one in reversed order as they are driven ON. This demonstrates power off sequencing control.

9. 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.

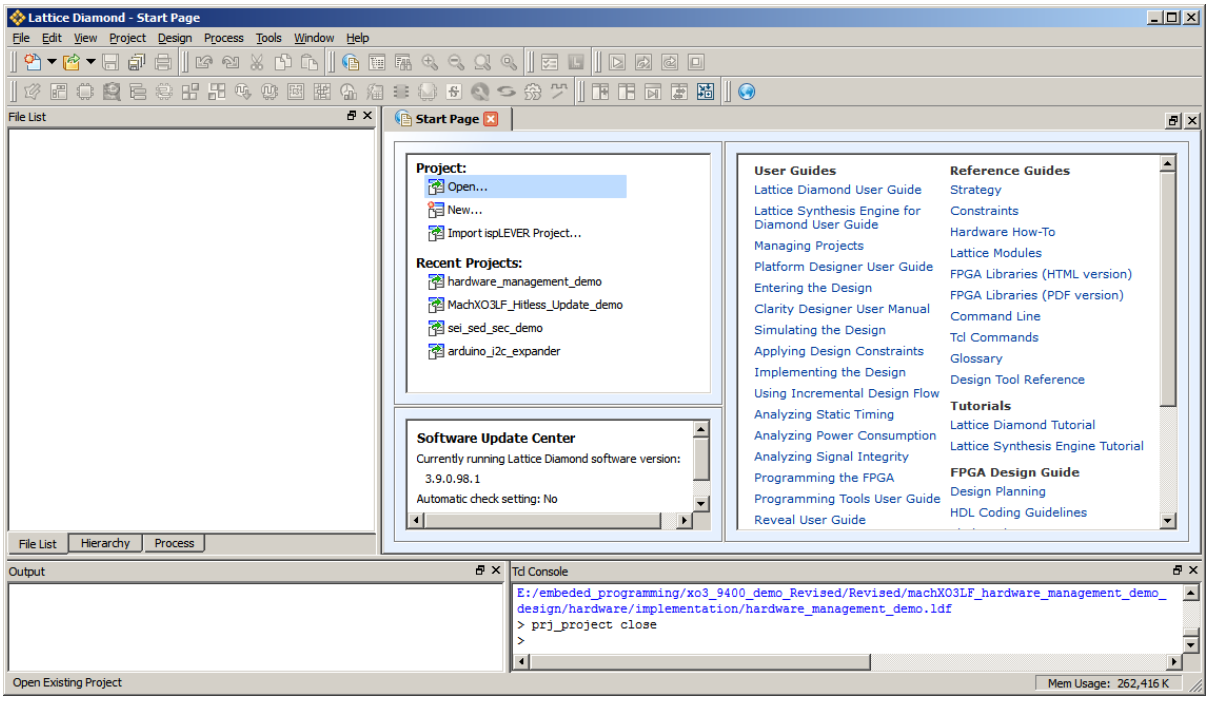


Figure 9.1. Opening Design File

2. After the project is opened, double-click "**hardware_management_demo_ptm.ptm**" as shown in Figure 9.2. This opens Platform Designer.

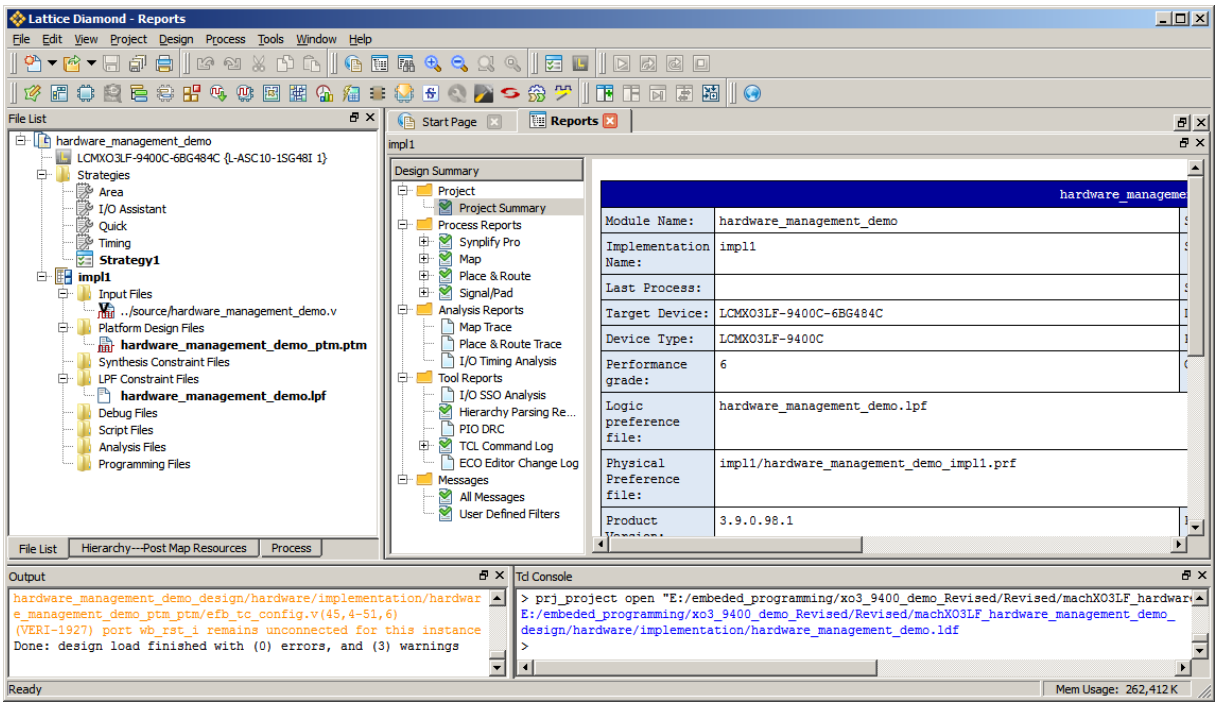


Figure 9.2. Opened Project in Lattice Diamond

3. In Platform Designer, make sure **Boot Mode** option is selected as **Normal** as Figure 13.3 below

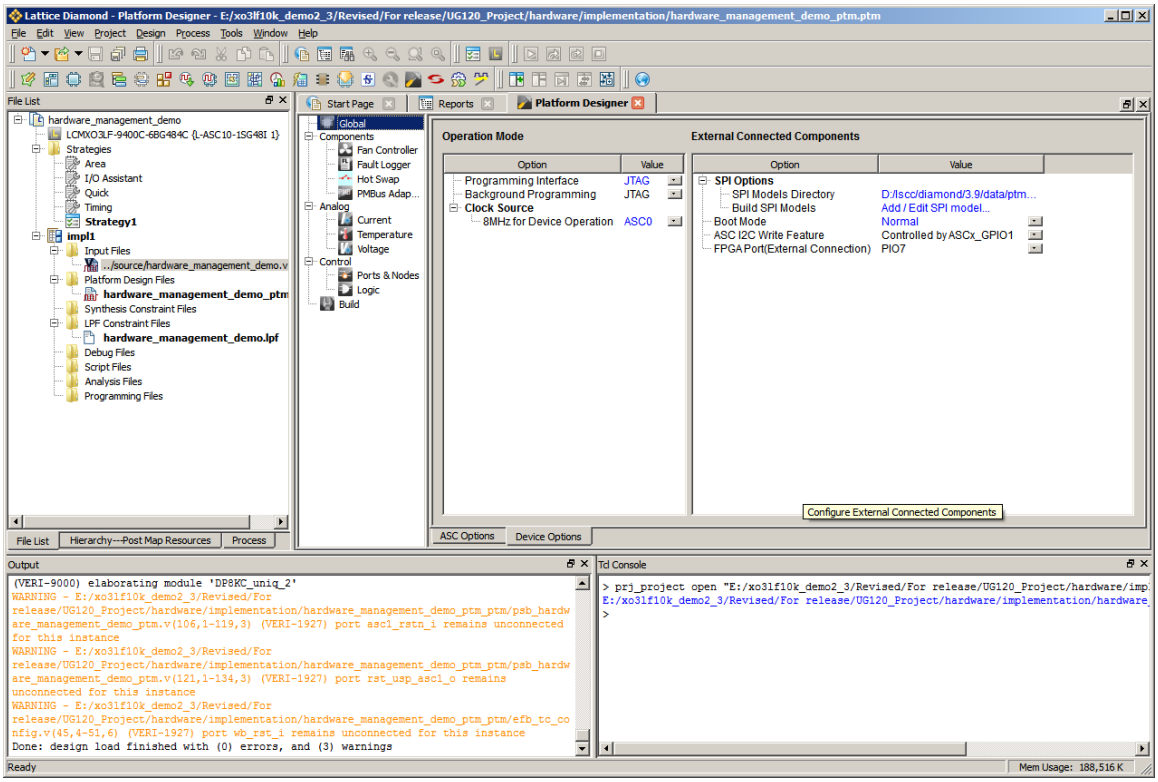


Figure 9.3. Setting Boot Mode Option in Platform Designer

4. In the Platform Designer main window, click the **Build** button. Click also the **DRC** and **Compile** buttons as shown in Figure 9.4.

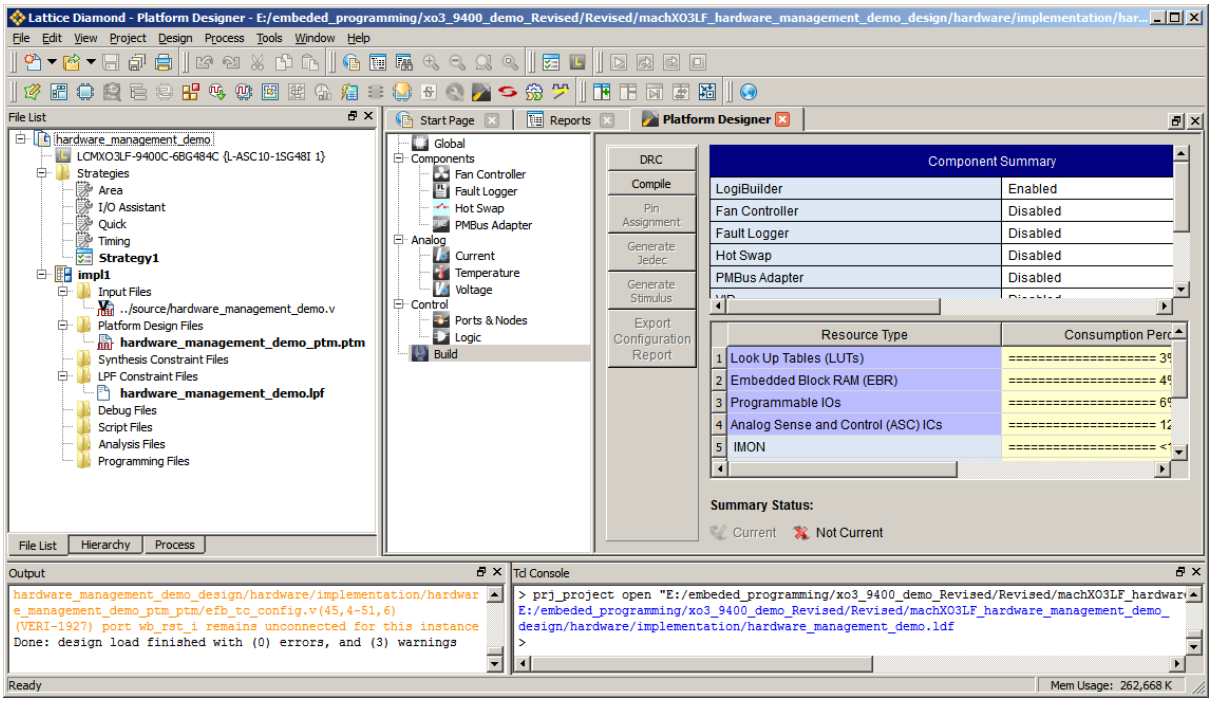


Figure 9.4. Building the Design

5. Then click **Pin Assignment** and open Spreadsheet View to set SDM_PORT to PROGRAM, set DUALBOOTGOLDEN to EXTERNAL as shown in Figure 9.5.

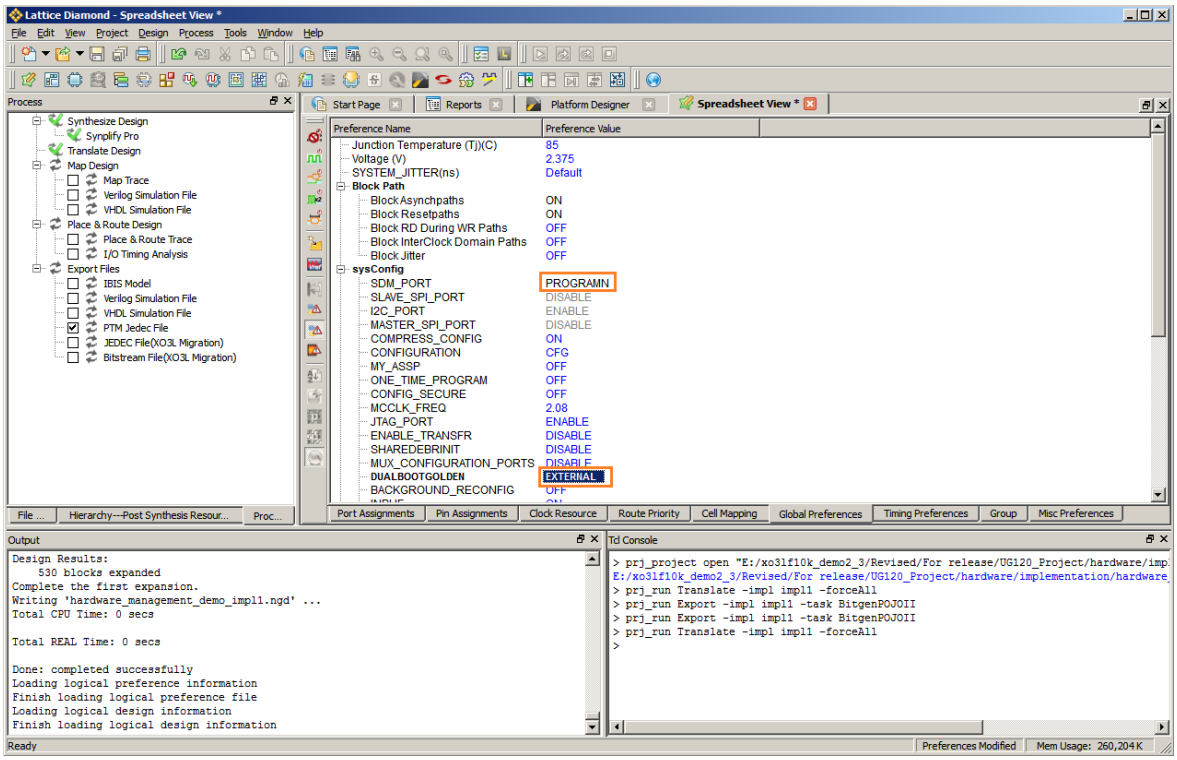


Figure 9.5. Setting Global Preferences for Pin Assignment

6. Close Spreadsheet View and click **Generate Jedec** as shown in Figure 9.6. The software reports “Done: completed successfully”.

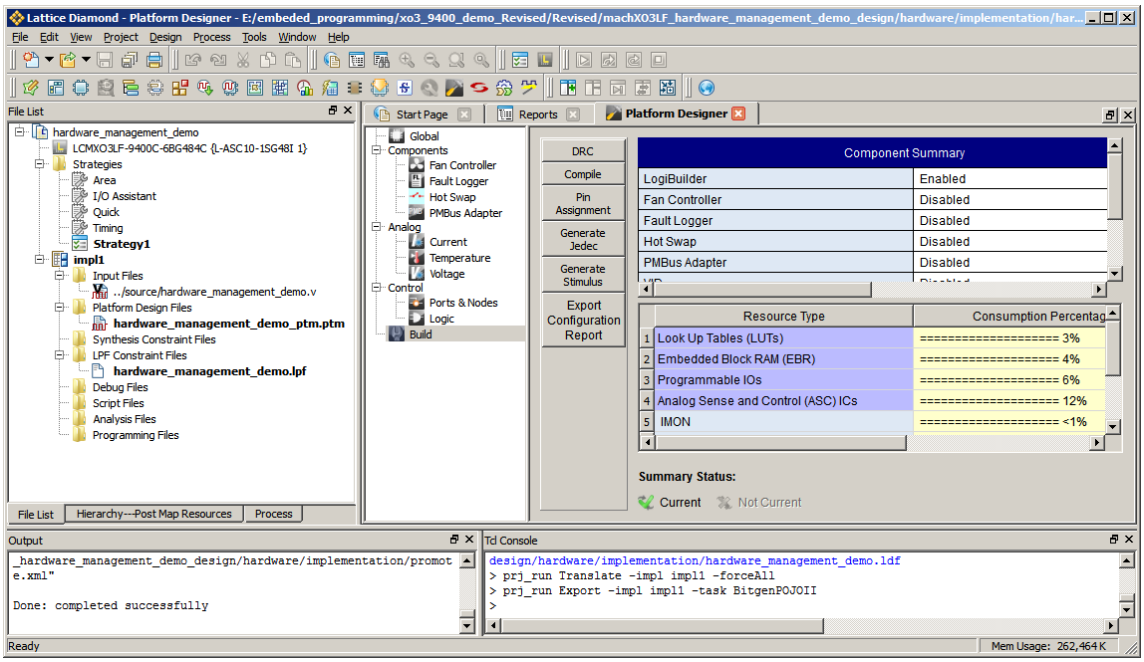


Figure 9.6. Generating the Bitstream

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

References

- DS1047 – MachXO3 Family Data Sheet
- DS1042 – L-ASC10 Data Sheet
- FPGA-EB-02004 – MachXO3LF 9400 Development Board User 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