UG0648
User Guide
Motor Control Libero Project

Microsemi
Power Matters.

Microsemi Corporate Headquarters
One Enterprise, Aliso Viejo,
CA 92656 USA
Within the USA: +1 (800) 713-4113
Outside the USA: +1 (949) 380-6100
Fax: +1 (949) 215-4996
Email: sales.support@microsemi.com
www.microsemi.com

About Microsemi

Microsemi Corporation (Nasdaq: MSCC) 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. Microsemi is headquartered in Aliso Viejo, California, and has approximately 4,800 employees globally. Learn more at www.microsemi.com.

50200648. 3.0 11/16

Contents

Figures

1 Revision History

1.1 Revision 3.0

There were no changes to the technical content in revision 3.0 of this document.

1.2 Revision 2.0

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

- Updated the Opening the Libero Project, page 3 section for the updated screen shots (SAR 80524).
- Added the Downloading Cores from Libero, page 4 section (SAR 80524).
- Updated the Importing Cores, page 4 section for download feature (SAR 80524).
- Added BLDC Sensorless and Stepper Motor Project, page 12, BLDC Encoder Project, page 13, and BLDC with Hall Project, page 14 (SAR 80524).

1.3 Revision 1.0

Revision 1.0 was the first publication of this document.

2 About this Guide

2.1 Purpose

This document familiarizes users with the motor control design in the Libero® System-on-Chip software user guide. It describes the steps for importing cores, running the design, editing I/O constraints, editing timing, and describes the motor control project.

2.2 Intended Audience

This document is not a complete Libero software user guide. It is intended for users who are unfamiliar with the Libero SoC tool to get started with motor control design.

2.3 Reference

Ensure that the latest version of Libero is downloaded from http://www.microsemi.com/products/fpga-soc/design-resources/design-software/libero-soc and installed. Download the Motor Control project.

For more information on Microsemi's Motor Control Solutions, visit http://www.microsemi.com/applications/motor-control#solutions.

Note: The downloaded project is based on encrypted IP blocks. It causes the design flow process to run up to 5x slower than an unencrypted (plaintext) IP.

3 Running the Libero Project

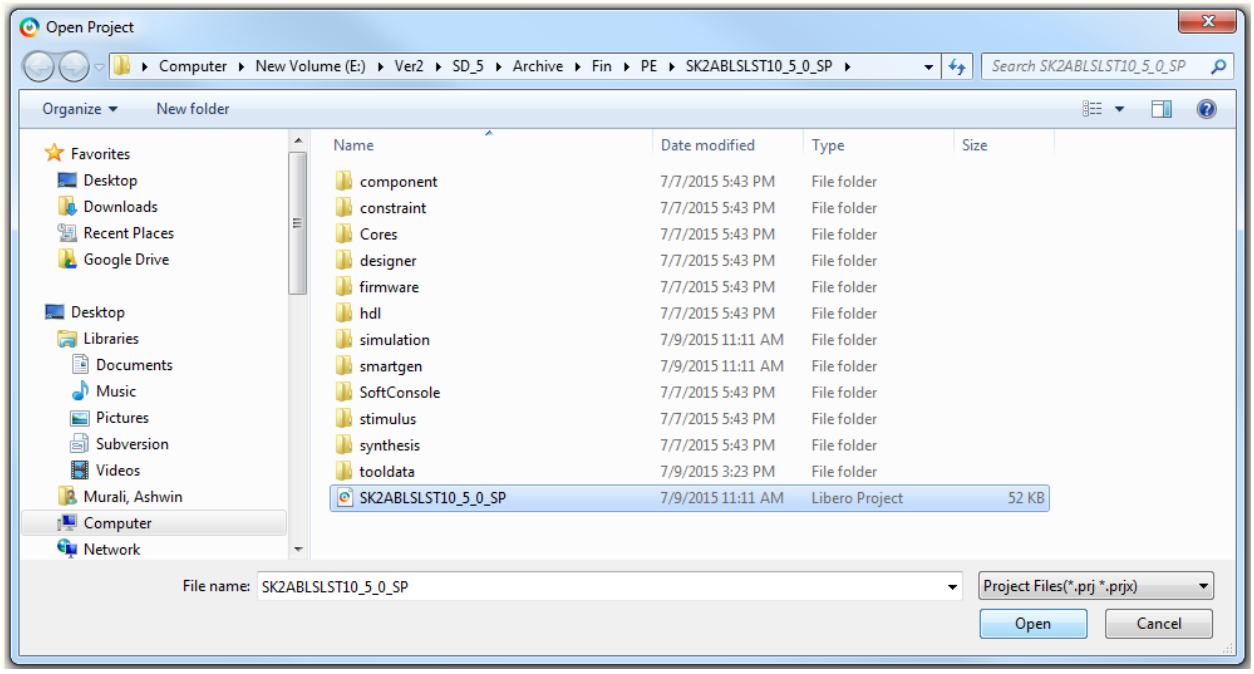
This section describes how to run the motor control project in Libero.

3.1 Opening the Libero Project

The following steps describe the procedure for opening the libero project:

1. Launch the installed Libero software.
2. To open the downloaded project, click **Project** -> **Open Project**; browse to the location and select the project file as shown in the following figure and click **Open**.

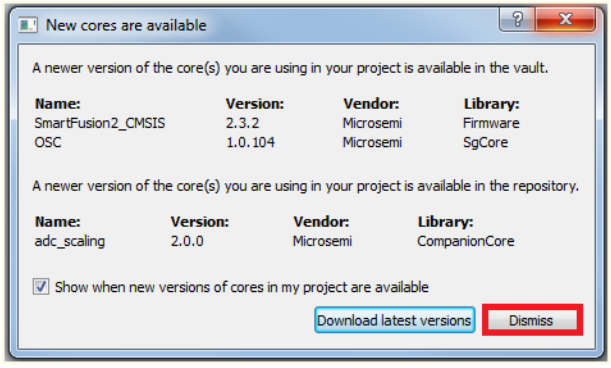
Figure 1 • **Opening the Downloaded Project**



A pop-up appears, as shown in the following figure.

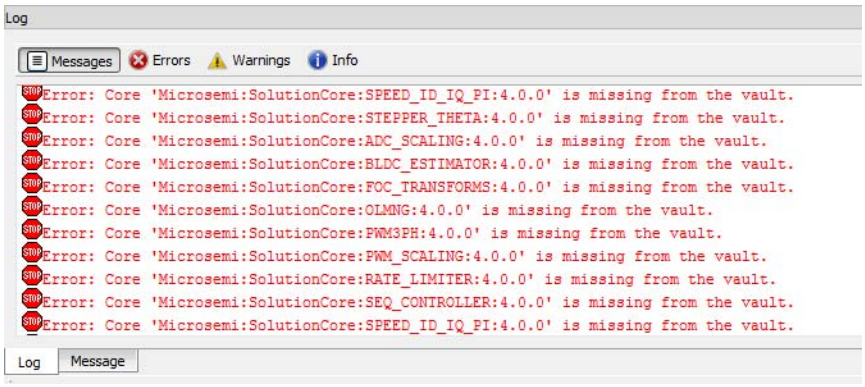
3. Click **Dismiss**, the Libero window is ready for use.

Figure 2 • **Pop-Up Window after Opening the Project File**



The Log window displays error messages of missing cores as shown in the following figure. These cores can be directly downloaded from Libero or can be imported from the downloaded contents.

Figure 3 • **Core Error - Log Window**

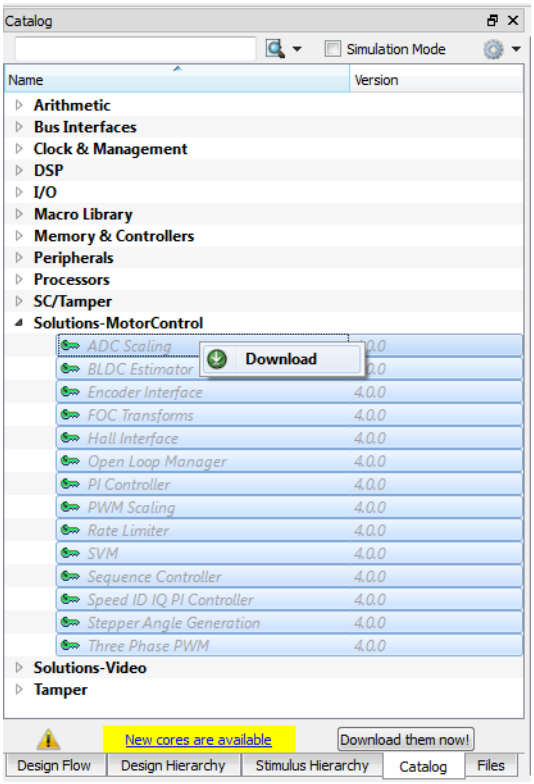


3.2 Downloading Cores from Libero

The following steps describe how to download IP cores from Libero SoC:

1. Click the **Catalog** tab, as shown in the following figure. If the IP cores are not already available in the Libero SoC vault, they are displayed in grey.
2. Select all the Motor Control IP cores by clicking on each of them holding the **Ctrl** key. Right-click the selected IP cores and click **Download** to download the cores.

Figure 4 • **Catalog – IP Cores Download**



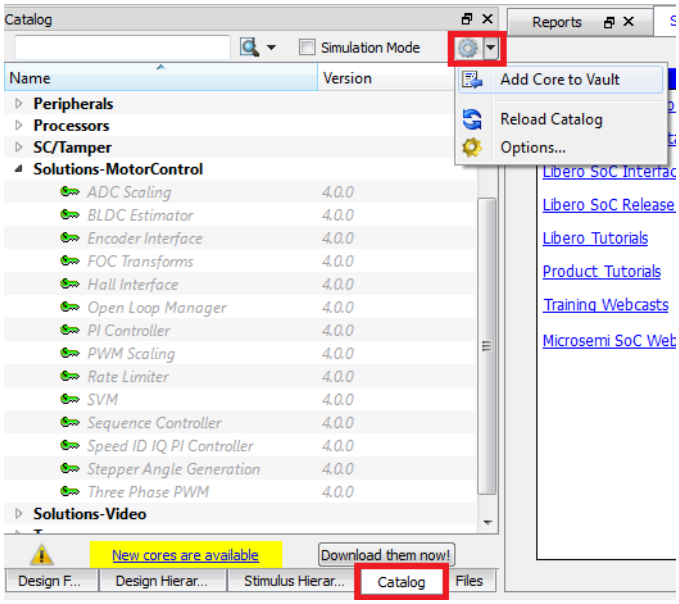
3.2.1 Importing Cores

The motor control IP cores can also be imported from the downloaded folder.

Note: It is recommended to download the IP cores directly from Libero SoC as it has the latest version of cores.

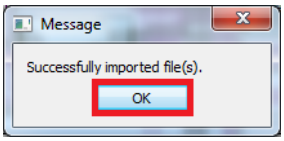
1. Click the **Catalog** tab as shown in the following figure.

Figure 5 • **Catalog Tab in the Libero Project**



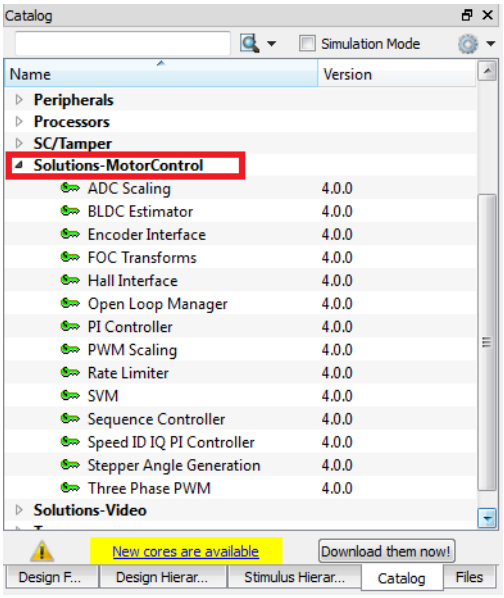
2. On the top right corner of the screen, click **Settings**. In the drop down list, click **Add core to vault**.
3. In the following screen, browse to the cores folder (unzipped from the downloaded file). Select all the *.cpz files, and then click **Open**. After importing the files successfully, a message appears as shown in the following figure.

Figure 6 • **Message after Importing Cores Successfully**



4. After the cores are imported, the Catalog tab must resemble the following figure. Expand the **Solutions-MotorControl** tab to view the cores.

Figure 7 • **Catalog Tab after Importing Cores**

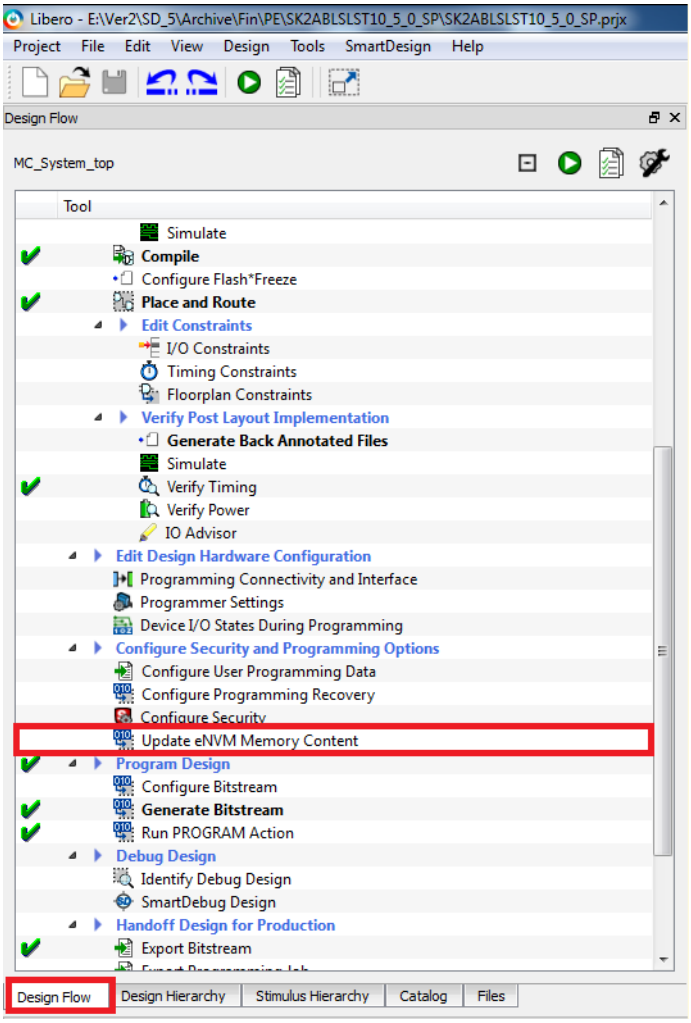


3.3 Running the Motor Control Design

The following steps describe how to verify the source directory for the eNVM content (.hex file).

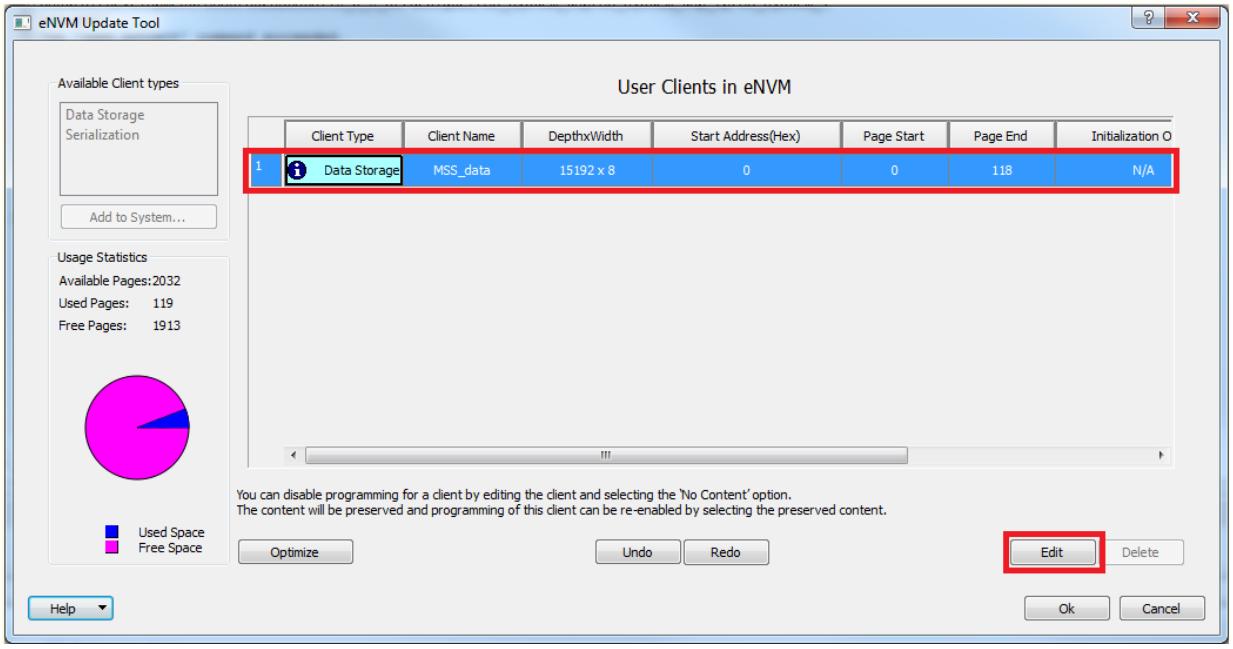
1. In the **Design Flow** tab, double-click **Update eNVM Memory Content** as shown in the following figure.

Figure 8 • **Updating eNVM Content – Design Flow Tab**



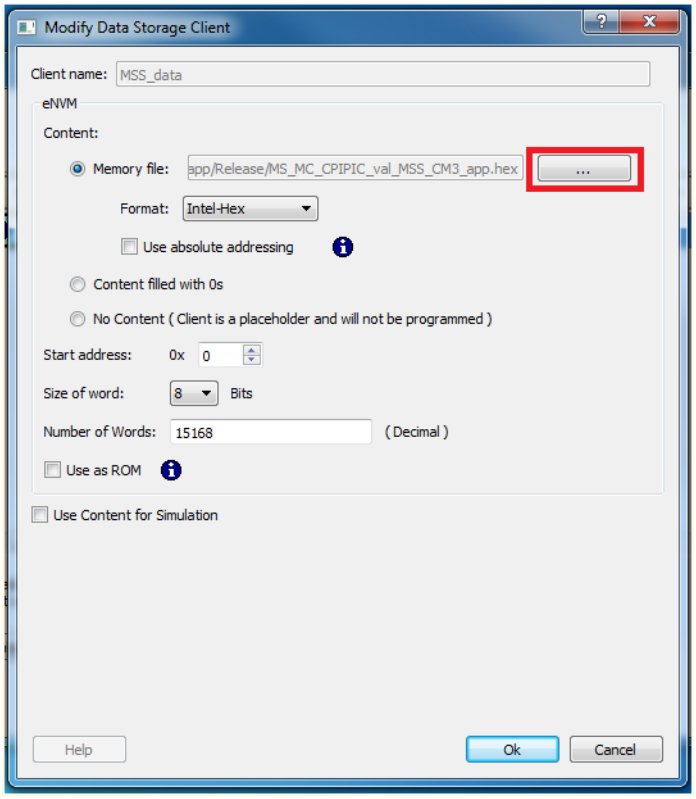
This opens the **eNVM Update Tool** pop-up window as shown in the following figure.

Figure 9 • **eNVM Update Tool Window to Update .hex File**



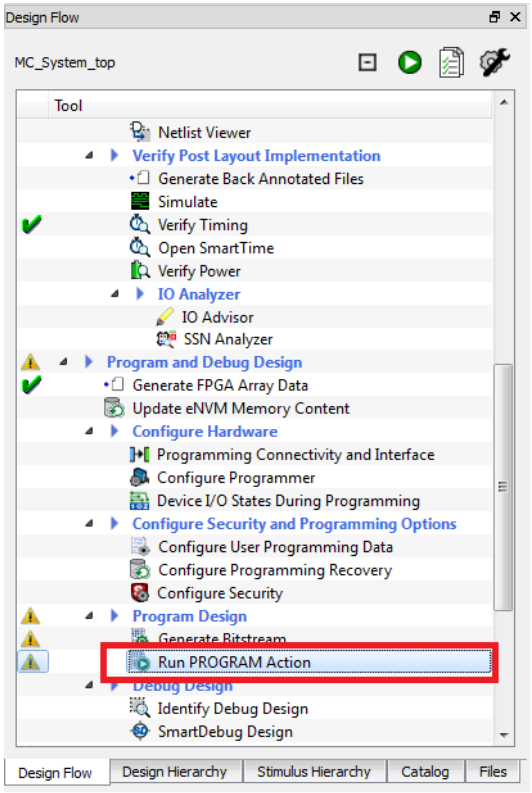
2. In the **eNVM Update Tool** window, select the **User Client (MSS_data)** by clicking on it. Then click **Edit**.
   This opens the "**Modify Data Storage Client**" pop-up window as shown in the following figure.
3. In the “Modify Data Storage Client” window. Click **Browse**, select .hex file from: **project folder -> SoftConsole -> Motor_Control_5_1_0 -> Motor_Control_sb_MSS_CM3_app -> Release** (Motor_Control_sb_MSS_CM3_app.hex). Click **Ok** to close the "Modify Data Storage Client", and click **Ok** again to close the eNVM Update Tool window.

Figure 10 • **Modify Data Storage Client Window to Replace .hex File**



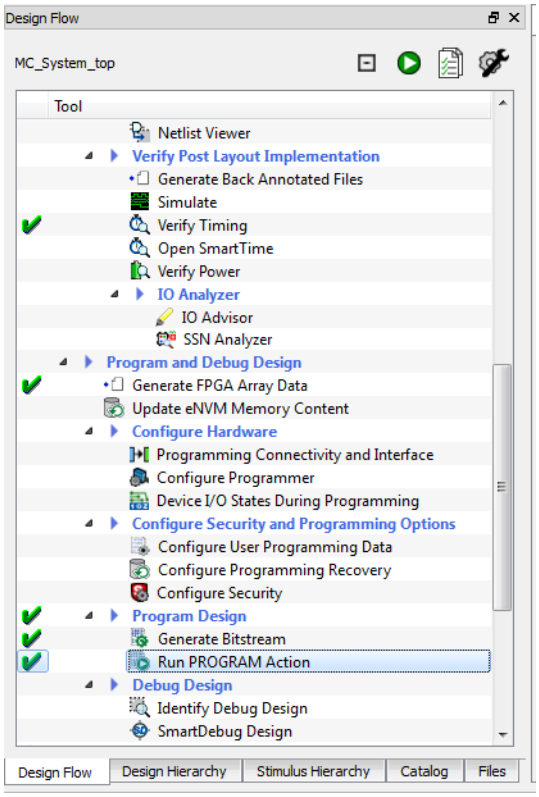
4. Connect FlashPro4 to the board and switch ON the switch SW2.
5. In the **Design Hierarchy** tab, double-click **Run PROGRAM Action** to program the device as shown in the following figure.

Figure 11 • **Run Program Action in the Design Flow Tab**



6. If the programming is successful, it is indicated by a green mark as shown in the following figure. The motors can be run using the GUI as described in the *DG0598: SmartFusion2 Dual-Axis Motor Control Starter Kit Demo Guide*.

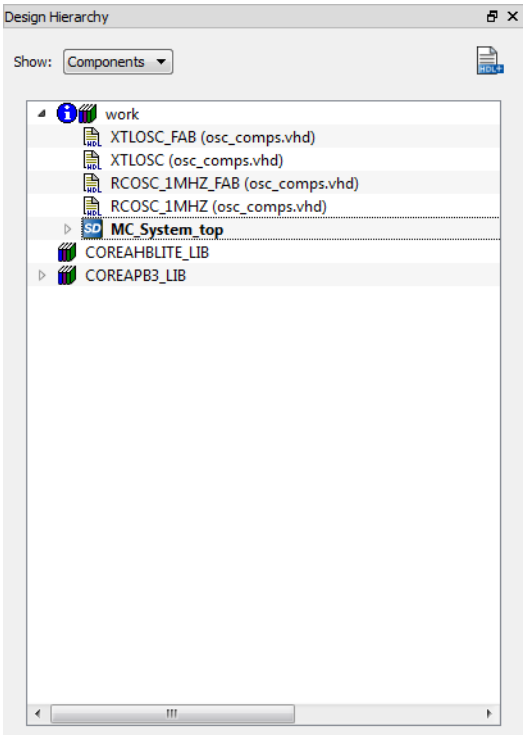
Figure 12 • **Design flow tab after device has been programmed**



4 Project Description

Select the **Design Hierarchy** tab as shown in the following figure. This window shows various modules in the design. The module in bold is the root as shown in the following figure. Generally, the root module and other modules listed under root module are synthesized.

Figure 13 • **Design Hierarchy Tab with Root Module – MC_System_top**

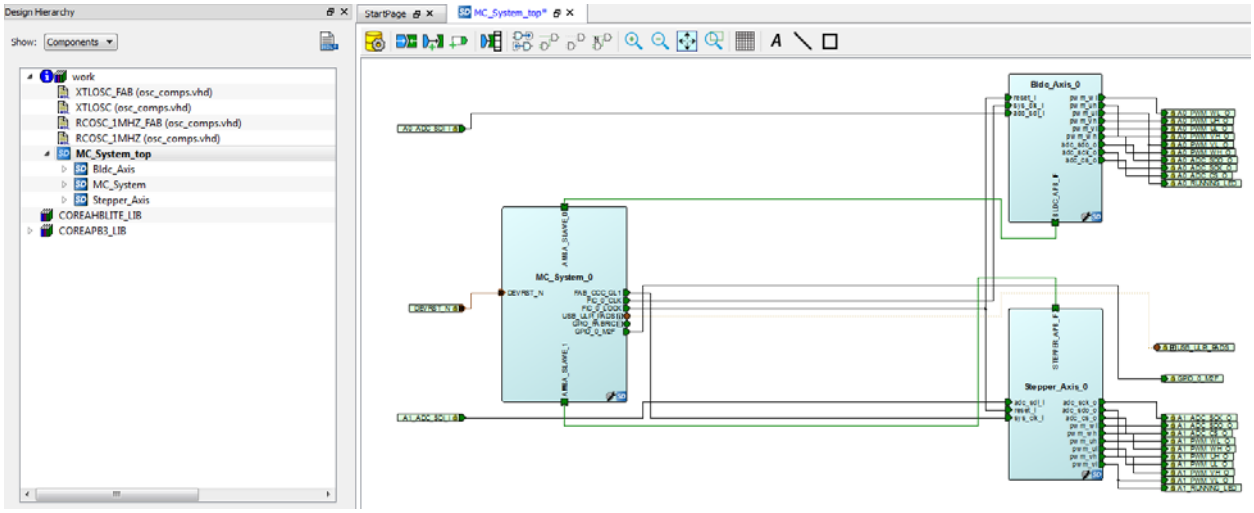


4.1 BLDC Sensorless and Stepper Motor Project

The following steps describe how to create a BLDC sensorless and stepper motor project:

1. Double-click the **MC_System_top**. The SmartDesign tab opens with **MC_System_top**, which instantiates the BLDC Axis, Stepper Axis and MC_System block as shown in the following figure. The MC_System block contains the microcontroller subsystem (MSS).

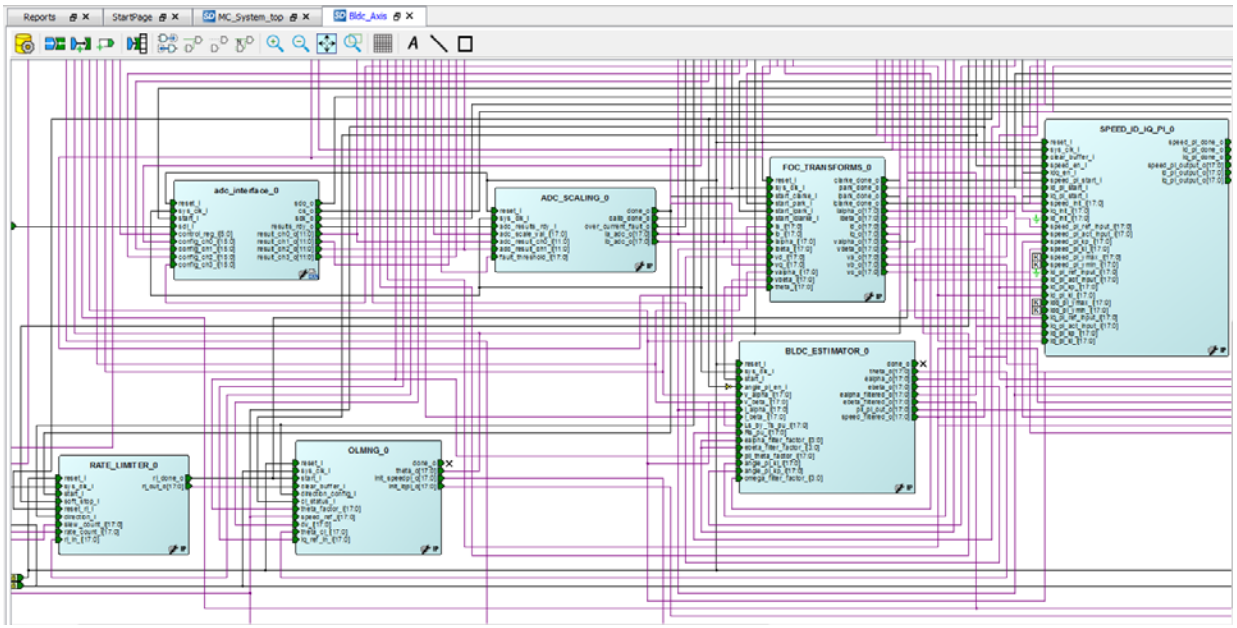
Figure 14 • **MC_System_top Design**



2. In the **Design Hierarchy** tab, expand the **MC_System_top** to show the modules instantiated under the module and double-click **Bldc_Axis** to open the Bldc_Axis module, as shown in the following figure. Alternatively, double-click **Bldc axis** in the SmartDesign window to open the **Bldc_Axis** module.

The Bldc_Axis tab is displayed with the visual connections of the blocks instantiated within the Bldc_Axis.

Figure 15 • **Bldc_Axis Tab**



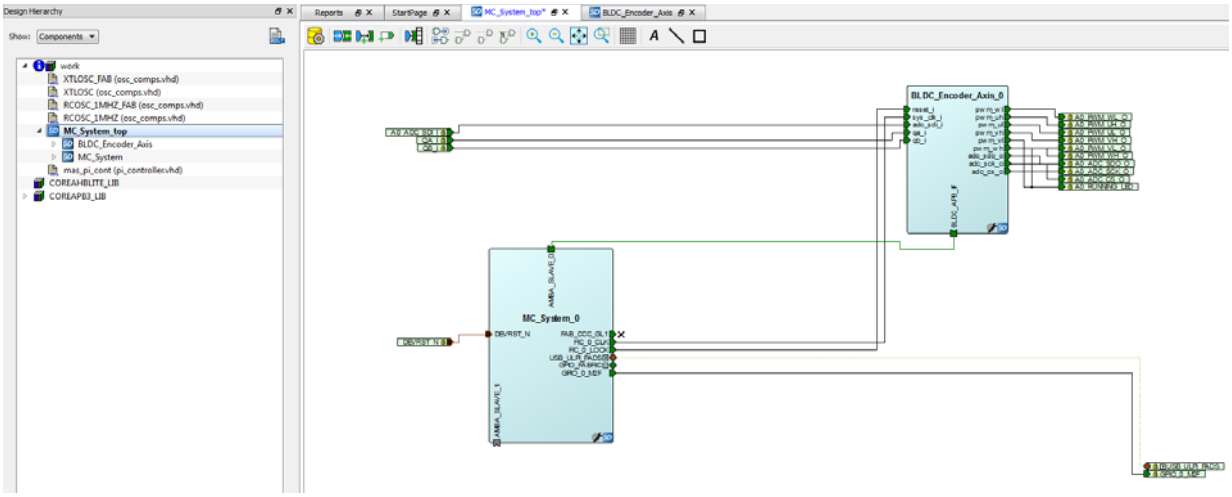
Refer to *AC445: Motor Control Design using SmartFusion2/IGLOO2 Devices Application Note* for more information on BLDC Sensorless and Stepper Motor Project.

4.2 BLDC Encoder Project

The following steps explain how to create a BLDC encoder project:

1. Double-click **MC_System_top** on the left. The **MC_System_top** tab is displayed on the right with the design, which instantiates the BLDC Encoder Axis and MC_System blocks, as shown in the following figure. The MC_System block contains the microcontroller subsystem (MSS).

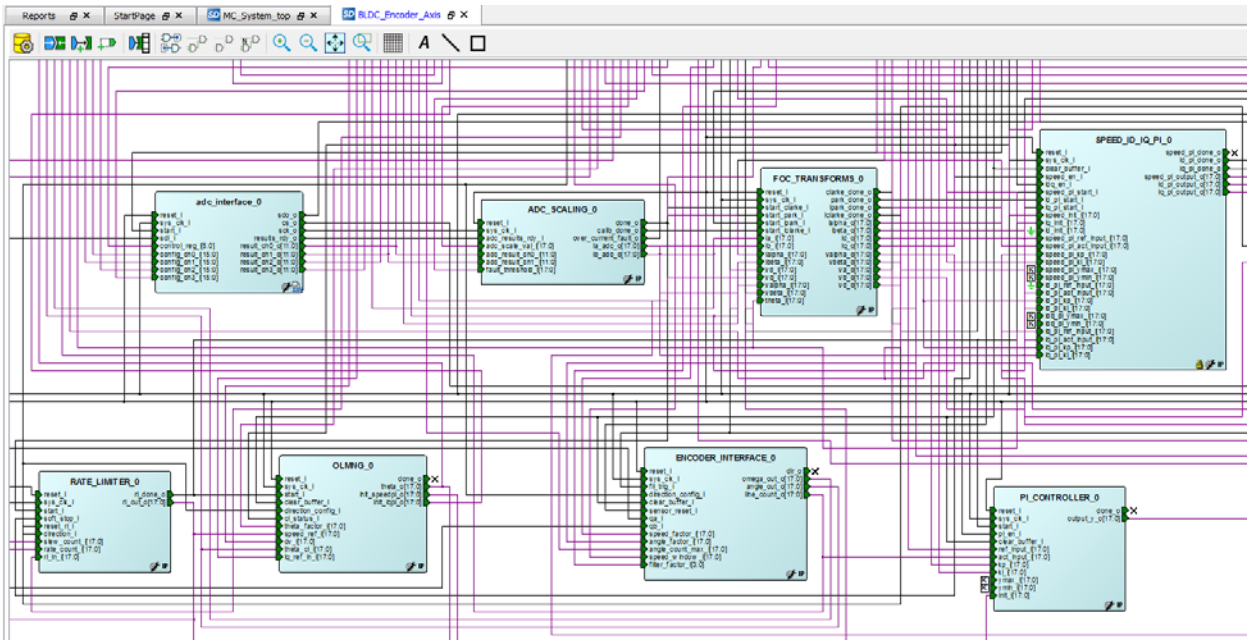
Figure 16 • **MC_System_top Design**



2. In the **Design Hierarchy** tab, expand the **MC_System_top** to show the modules instantiated under the module and double-click **BLDC_Encoder_Axis**, as shown in Figure 18. Alternatively, double-click **BLDC_Encoder_Axis** in the SmartDesign window to open the BLDC_Encoder_Axis module.

The **BLDC_Encoder_Axis** is displayed with the visual connections of the blocks instantiated within Bldc_Encoder_Axis.

Figure 17 • **BLDC_Encoder_Axis Tab**

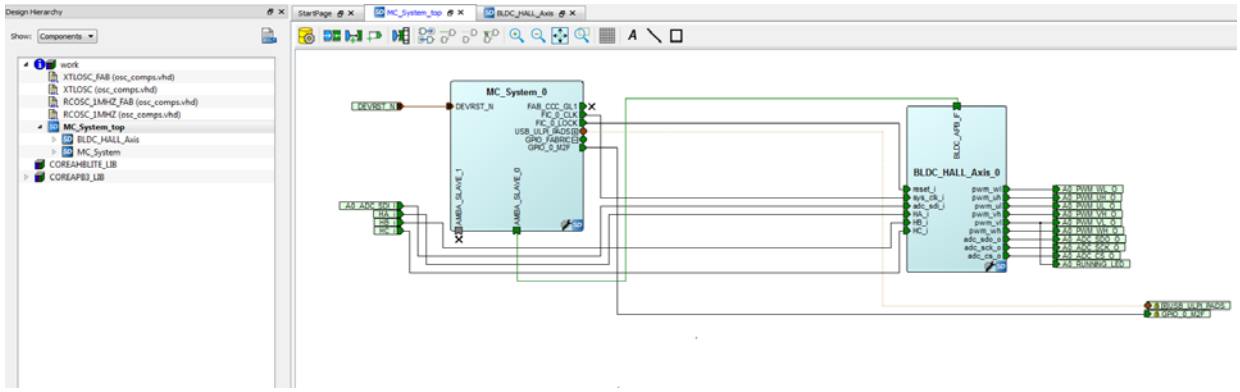


4.3 BLDC with Hall Project

The following steps enable to create a BLDC with hall project:

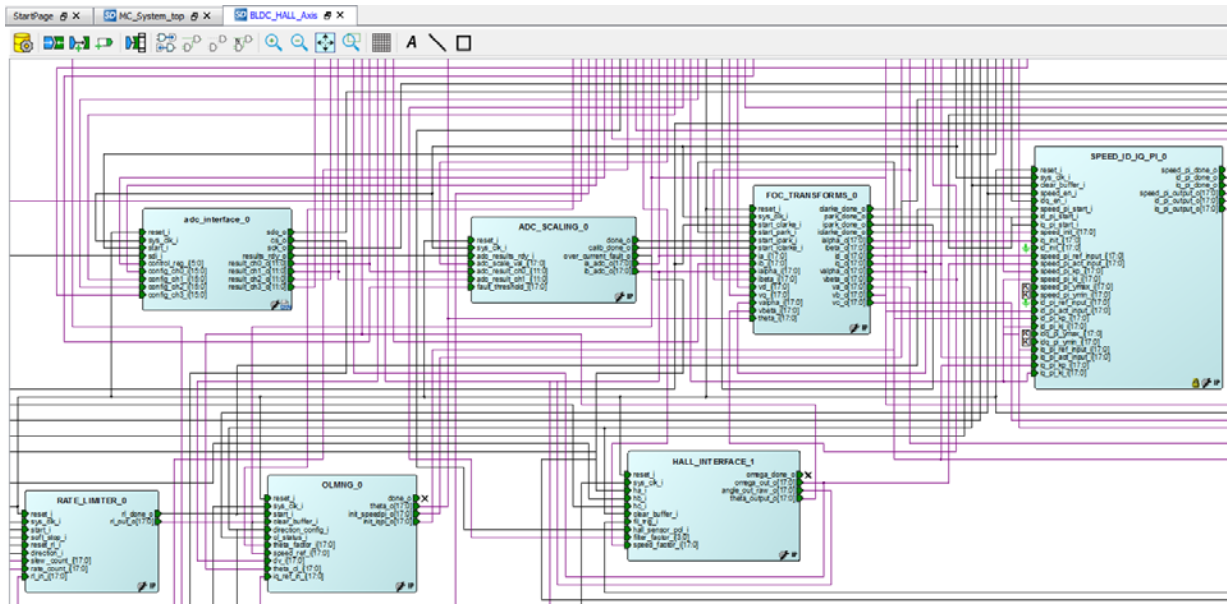
1. Double-click **MC_System_top** on the left. The SmartDesign tab opens with MC_System_top on the right, which instantiates the **BLDC_HALL_Axis and MC_System** blocks, as shown in the following figure. The MC_System block contains the microcontroller subsystem (MSS).

Figure 18 • **MC_System_top Design**



2. In the **Design Hierarchy** tab, expand the **MC_System_top** to show the modules instantiated under the module and double-click **BLDC_HALL_Axis** to open, as shown in the following figure. Alternatively, double-click BLDC_HALL_Axis in the SmartDesign window to open the BLDC_HALL_Axis module.
   The **BLDC_HALL_Axis** tab is displayed with connections of the blocks instantiated within BLDC_HALL_Axis.

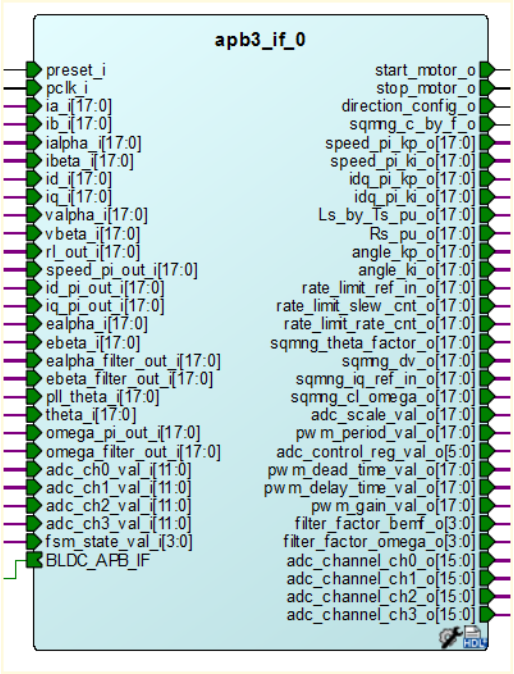
Figure 19 • **BLDC_HALL_Axis Tab**



4.4 Description of Interfacing IPs

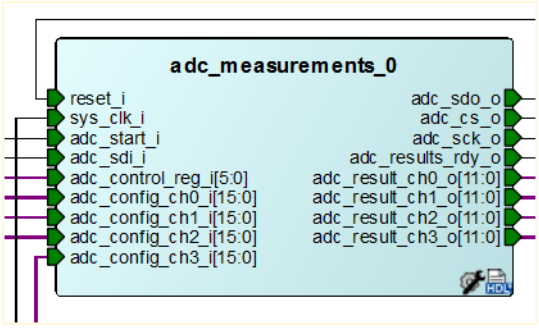
The APB3_IF block acts as a bridge between the MSS and the FPGA fabric blocks. Several parameters are computed in the MSS based on GUI inputs and passed to the fabric. This feature allows the designer to modify parameters without rerunning the FPGA flow. The fabric sends internal variables to the MSS for visualizing them in a GUI and debugging the design. The APB3 interface block is shown in the following figure.

Figure 20 • **The apb3_if Block**



The ADC interface block provides multi-bit data from an ADC device SPI port. It acts as an SPI master, while the ADC device works as a slave, providing motor current information to other blocks in the control algorithm. It also allows the designer to select the number of channels and the clock divider to provide the SPI clock (SCK). The ADC measurements block is shown in the following figure.

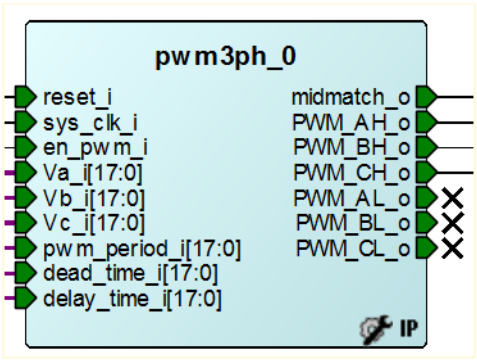
Figure 21 • **ADC Measurements Block**



The three phase Pulse width modulation (PWM) generation block generates three independent channels of complementary PWM, with features like configurable PWM period, dead time, and delay time.

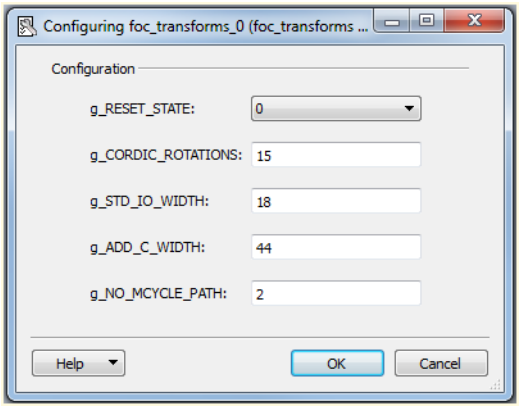
These parameters are configured from the MSS through the APB3 interface. Depending on the driver device, complementary PWMs can be used. The three phase PWM block is shown in the following figure.

Figure 22 • **Three Phase PWM Block**



1. In the **Design Hierarchy** window, double-click the **Stepper Axis** tab to open the **Stepper Motor SmartDesign** tab. It shows the interconnection between the blocks used in the stepper design. The Stepper design uses the ADC interface and uses two phases of the three phase PWM block.
2. To configure an IP block, double-click **IP Block**. A configuration window appears, which allows the designer to modify port widths and other constants. For example, if the FOC_Transforms block is to be configured, double click the foc_transforms_0 block. A configurator window opens, as shown in the following figure. For further information, refer individual IP user guides at http://www.microsemi.com/applications/motor-control#resources.

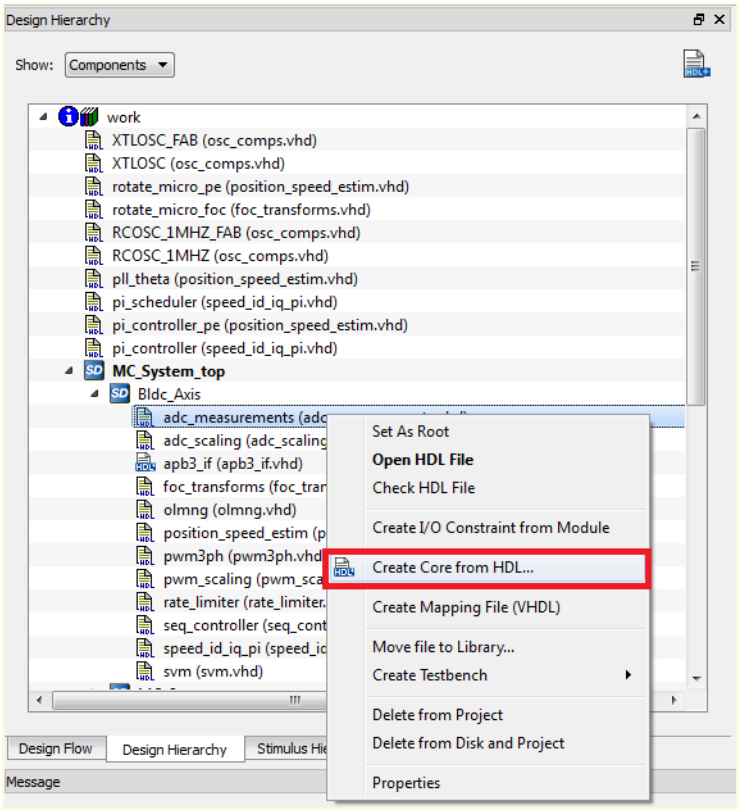
Figure 23 • **Configuration Window for IP Blocks**



3. IP blocks are available as modules in the Project Catalogue for use in SmartDesigns. Select an IP in the catalog and drag it into a SmartDesign to instantiate the IP.
4. Drag SmartDesign blocks into other SmartDesign modules to re-instantiate them.
5. To re-use other modules (from other HDL files), identify the module in the **Design Hierarchy** tab and drag it into the required SmartDesign tab. In case the module must be re-used with different configuration settings in each instantiation, the module needs to be converted to SmartHDL. To convert a module into SmartHDL, right-click the module in the **Design Hierarchy** tab and click **Create Core from HDL** as shown in the following figure, and then drag the block into the desired places.

Note: SmartHDL files are represented with a symbol.

Figure 24 • **Creating a SmartHDL File from HDL Files**



5 Modifying the Design

This section describes editing I/Os to run the design on a different board, editing timing constraints, and viewing reports generated by Libero SoC.

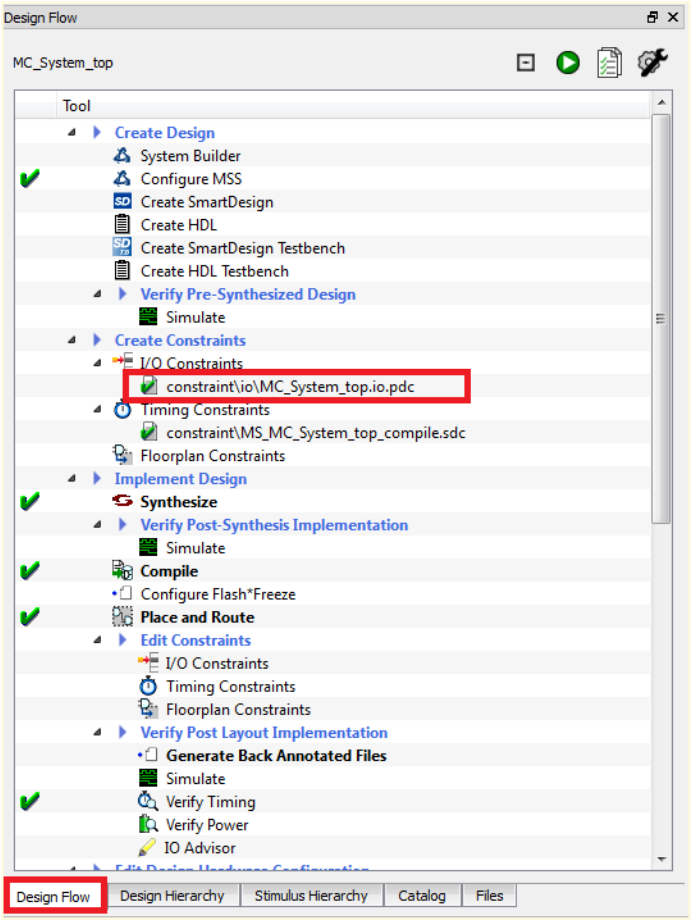
5.1 Editing I/O Configuration

The following steps describe the process of editing I/O configuration:

1. Click the **Design Flow** tab.
2. Use one of the following methods to edit I/O constraints:

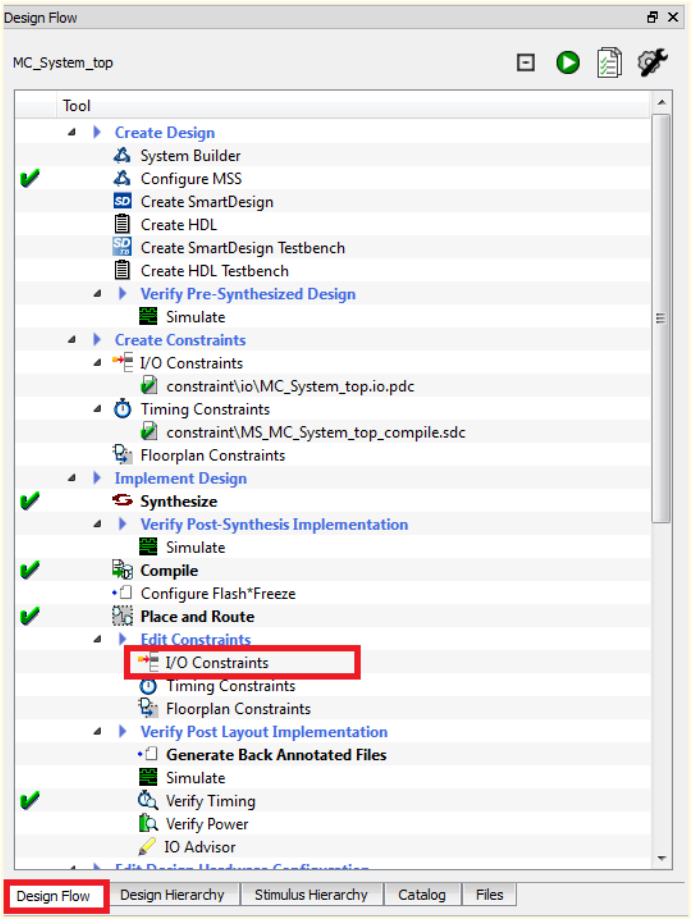
- Edit the I/O constraints .pdc file to edit the pin map, by clicking the constraint file, as shown in the following figure.

Figure 25 • **Editing I/O Constraints**



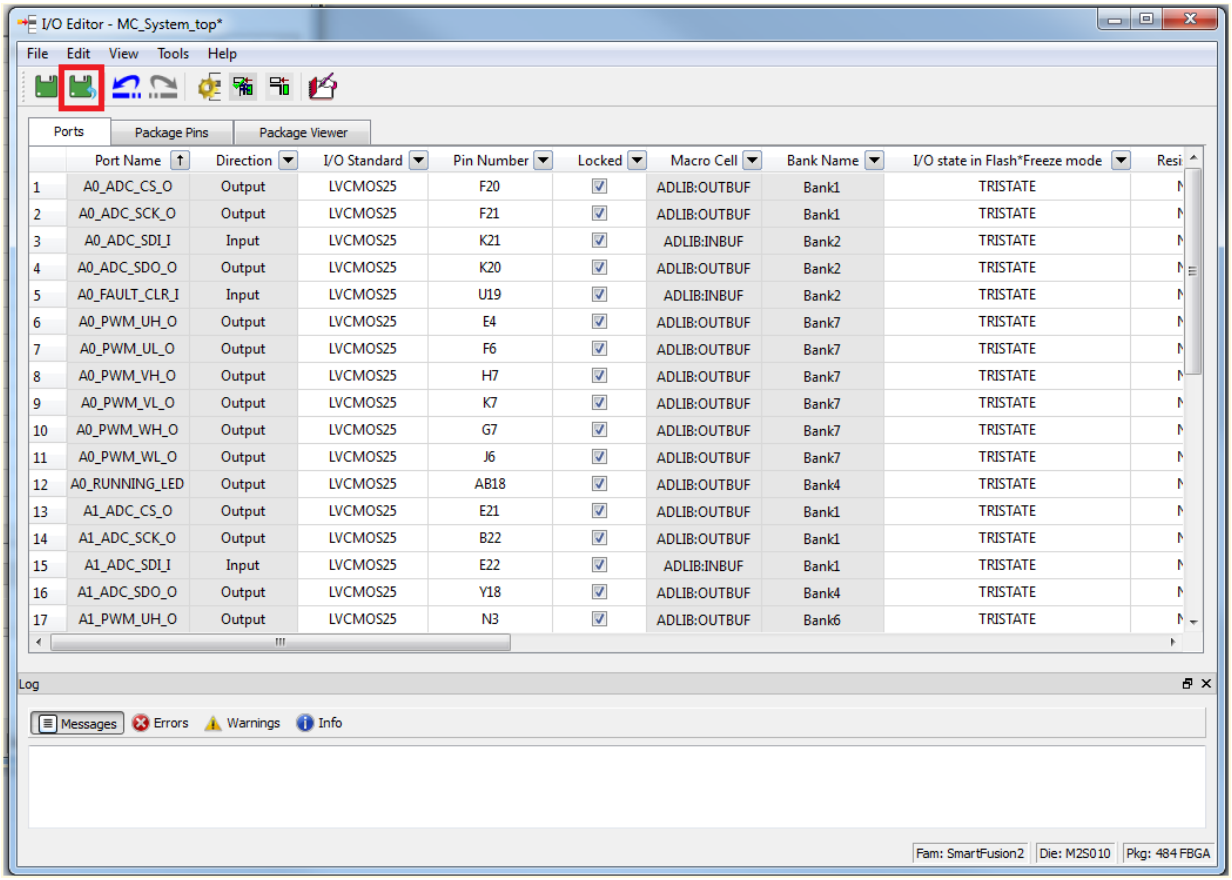
- Double-click **I/O Constraints** under **Edit Constraints**, as shown in the following figure.

Figure 26 • **Opening the I/O Editor**



The **I/O Editor** window is displayed, as shown in the following figure. You can assign or re-assign device pins and modify the drive strength in a GUI window. After changing the configuration, click **Commit and Check**, as shown in the following figure.

Figure 27 • **I/O Editor Window**



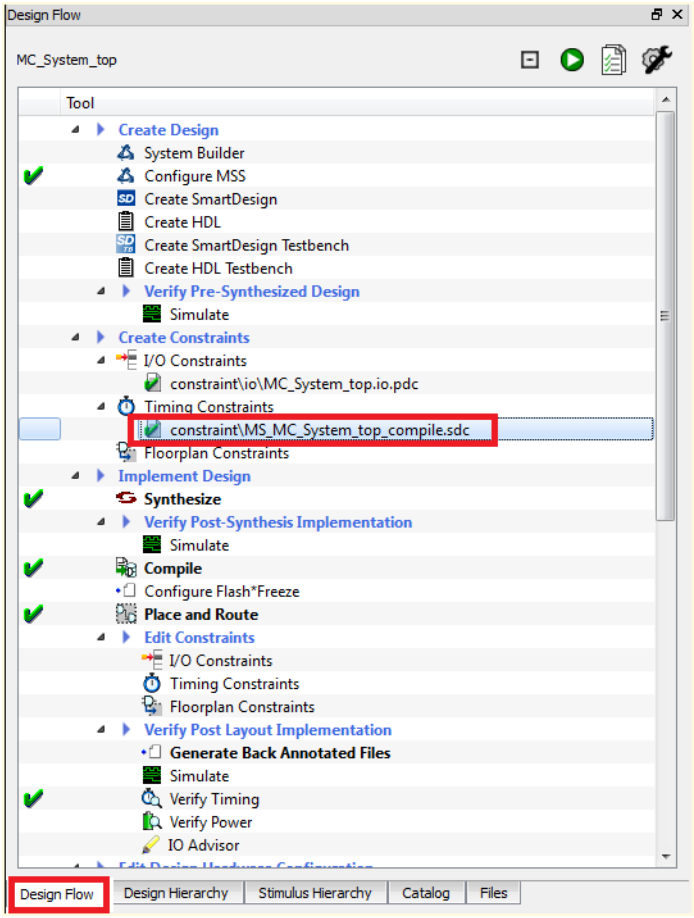
	Port Name	Direction	I/O Standard	Pin Number	Locked	Macro Cell	Bank Name	I/O state in Flash*Freeze mode	Resi
1	A0_ADC_CS_O	Output	LVCMOS25	F20	☑	ADLIB:OUTBUF	Bank1	TRISTATE	N
2	A0_ADC_SCK_O	Output	LVCMOS25	F21	☑	ADLIB:OUTBUF	Bank1	TRISTATE	N
3	A0_ADC_SDI_I	Input	LVCMOS25	K21	☑	ADLIB:INBUF	Bank2	TRISTATE	N
4	A0_ADC_SDO_O	Output	LVCMOS25	K20	☑	ADLIB:OUTBUF	Bank2	TRISTATE	N
5	A0_FAULT_CLR_I	Input	LVCMOS25	U19	☑	ADLIB:INBUF	Bank2	TRISTATE	N
6	A0_PWM_UH_O	Output	LVCMOS25	E4	☑	ADLIB:OUTBUF	Bank7	TRISTATE	N
7	A0_PWM_UL_O	Output	LVCMOS25	F6	☑	ADLIB:OUTBUF	Bank7	TRISTATE	N
8	A0_PWM_VH_O	Output	LVCMOS25	H7	☑	ADLIB:OUTBUF	Bank7	TRISTATE	N
9	A0_PWM_VL_O	Output	LVCMOS25	K7	☑	ADLIB:OUTBUF	Bank7	TRISTATE	N
10	A0_PWM_WH_O	Output	LVCMOS25	G7	☑	ADLIB:OUTBUF	Bank7	TRISTATE	N
11	A0_PWM_WL_O	Output	LVCMOS25	J6	☑	ADLIB:OUTBUF	Bank7	TRISTATE	N
12	A0_RUNNING_LED	Output	LVCMOS25	AB18	☑	ADLIB:OUTBUF	Bank4	TRISTATE	N
13	A1_ADC_CS_O	Output	LVCMOS25	E21	☑	ADLIB:OUTBUF	Bank1	TRISTATE	N
14	A1_ADC_SCK_O	Output	LVCMOS25	B22	☑	ADLIB:OUTBUF	Bank1	TRISTATE	N
15	A1_ADC_SDI_I	Input	LVCMOS25	E22	☑	ADLIB:INBUF	Bank1	TRISTATE	N
16	A1_ADC_SDO_O	Output	LVCMOS25	Y18	☑	ADLIB:OUTBUF	Bank4	TRISTATE	N
17	A1_PWM_UH_O	Output	LVCMOS25	N3	☑	ADLIB:OUTBUF	Bank6	TRISTATE	N

5.2 Editing Timing Constraints

The following steps describe how to edit timing constraints:

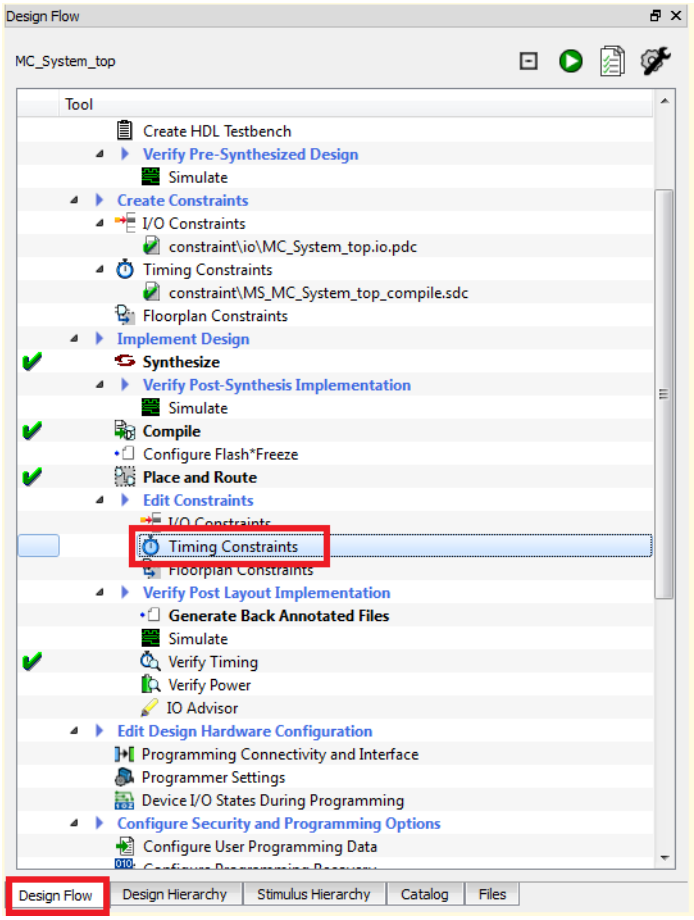
1. Use one of the following methods to edit I/O constraints:
   - Edit the Timing constraints file (.sdc) shown in the following figure.

Figure 28 • **Editing Timing Constraints**



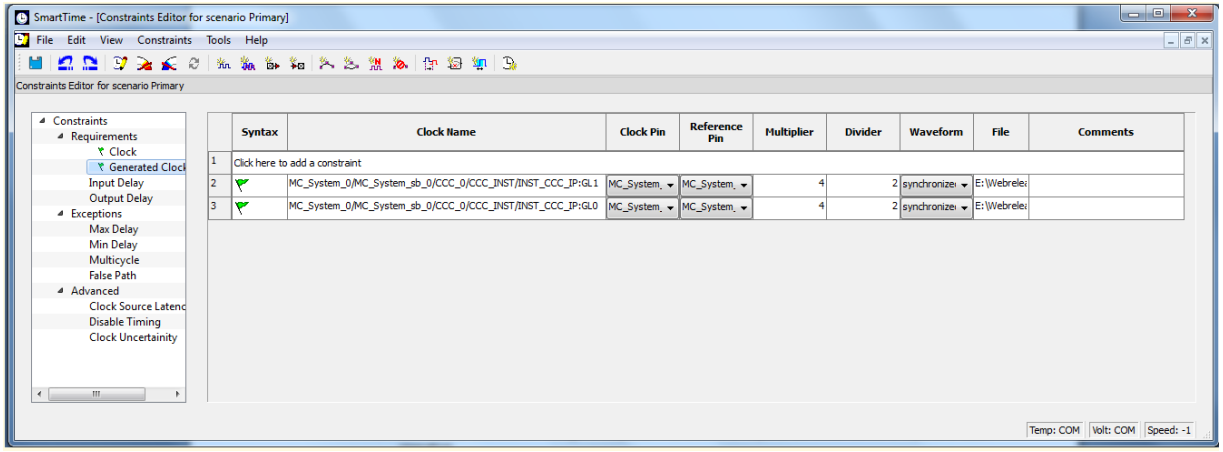
   - Double-click **Timing Constraints** as shown in the following figure.

Figure 29 • **Editing Timing Constraints Through SmartTime**



This opens a GUI (SmartTime) to edit timing Constraints as shown in the following figure.

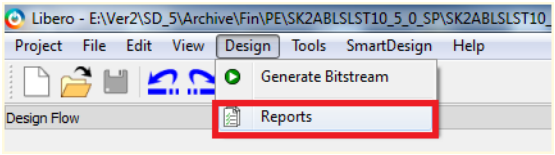
Figure 30 • **SmartTime Window to Edit Timing Constraints**



5.3 Generating and Viewing Reports

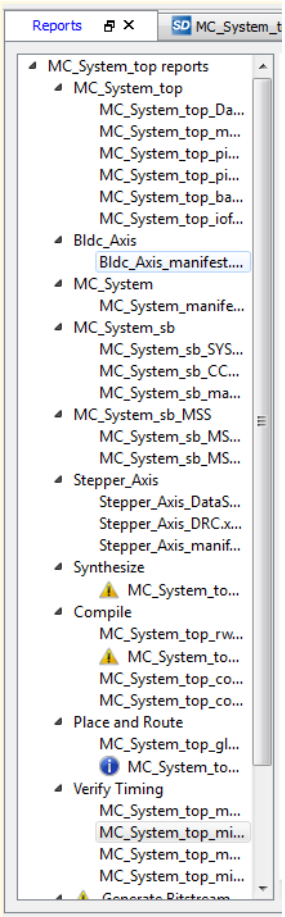
1. In the Menu bar, click **Design -> Reports** as shown in the following figure.

Figure 31 • **Generating Reports**



2. This opens the **Reports** tab, which contains all the reports generated while running the design, as shown in the following figure.

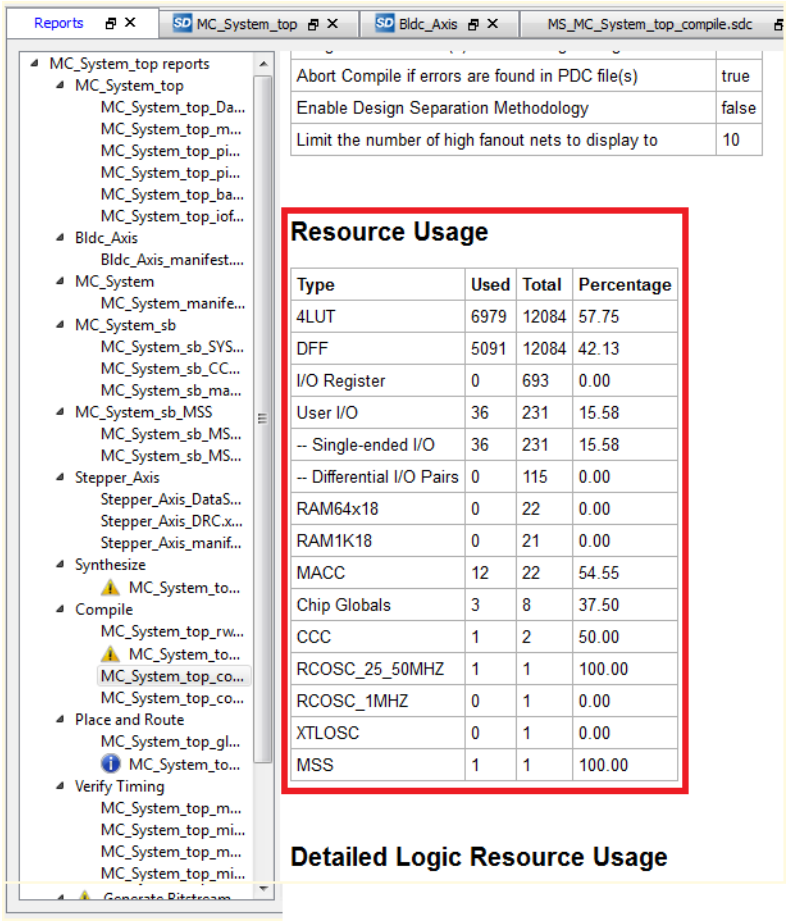
Figure 32 • **List of Reports in the Reports Tab**



3. The resource utilization can be viewed in the compiled report. It helps analyze and decide:

- Whether or not the design fits the device.
- If any additional logic can be added to the existing design.
- Whether or not the design can fit into a smaller device.

Figure 33 • **Resource Utilization from Compile Report**



4. The timing report (resulting from Verify Timing) shows the minimum and maximum delay analysis. This delay analysis helps determine if the design will work without errors for the given clock constraint.

Figure 34 • **Timing Analysis from the Timing Report**