DSP Flow for SmartFusion2 and IGLOO2 Devices - Libero SoC v11.6

TU0312 Quickstart and Design Tutorial

Microsemi

Table of Contents

Introduction

This tutorial describes the flow for generating the RTL files from the design or higher level algorithm created in the Mathworks MATLAB® Simulink® software. It assumes that the Mathworks MATLAB Simulink software and license are already installed. In addition, the Synopsys® Synphony Model Compiler ME must be installed. The Synphony Model Compiler ME can only be launched from the MATLAB Simulink tool.

For more information about MATLAB Simulink software, visit *www.mathworks.com/products/product_listing/index.html*

For Synphony Model Compiler ME, visit *http://www.microsemi.com/products/fpga-soc/design-resources/design-software/synphony#downloads*.

Tutorial Requirements

Table 1 shows the tutorial requirements.

Table 1 Software Requirements

Software Requirements	Description
Libero® System-on-Chip (SoC)	v11.6
ModelSim	v10.3c
Synphony Model Compiler ME	J-2015.03M
MATLAB Simulink	–

Project Files

The following are the project files associated with this tutorial:

- Source
- Solution
- Readme file

Download the project files from:

- SmartFusion2: *http://soc.microsemi.com/download/rsc/?f=m2s_tu0312_liberov11p6_df*
- IGLOO2: *http://soc.microsemi.com/download/rsc/?f=m2gl_tu0312_liberov11p6_df*

Refer to the `Readme.txt` file for the complete directory structure.

Synphony Model Compiler ME (Microsemi SoC Products Group Edition) Software and License Availability

The Synphony Model Compiler ME software and licenses are available for free to Microsemi SoC Products Group customers at: http://www.microsemi.com/products/fpga-soc/fpga-and-soc.

Using Synphony Model Compiler ME J-2015.03M with Libero SoC

Introduction

The Synphony Model Compiler ME translates a design from a higher level algorithm description in Simulink into RTL code that can be synthesized using Synplify Pro ME. The Synphony Model Compiler ME also creates an HDL testbench for the design by capturing the stimulus used to test the design within the Simulink environment. This facilitates verification and makes the RTL-bit and cycle accurate, when compared to the Simulink model of the DSP design.

Currently MATLAB Simulink and Synphony Model Compiler ME are not integrated in Libero® System-on-Chip (SoC). Also, the current Synphony Model Compiler ME supports SmartFusion®2 SoC field programmable gate array (FPGA) and IGLOO®2 FPGA devices. To infer the multiply-accumulate (MACC) blocks, the SmartFusion2 SoC FPGA or IGLOO2 FPGA device is used for RTL generation in Synphony Model Compiler.

This document gives step-by-step instructions on how to run Synphony Model Compiler ME and import the design files, testbench, and test vector files into Libero SoC. It also describes the options and settings required by Libero SoC for a smooth design flow. This tutorial uses digital down converter (DDC) model design. This design is created in Simulink using Synphony Model Compiler Blockset. For more information about creating design using Simulink, visit *www.mathworks.com*.

Figure 1 shows the overall DSP design flow using MATLAB Simulink and Libero SoC with Synplify Pro ME. The Synphony Model Compiler ME translates the DSP design created in Simulink into register-transfer level (RTL) code.

The RTL code can be imported to Libero SoC to facilitate smooth synthesis, simulation, place-and-route, and programming of the design.

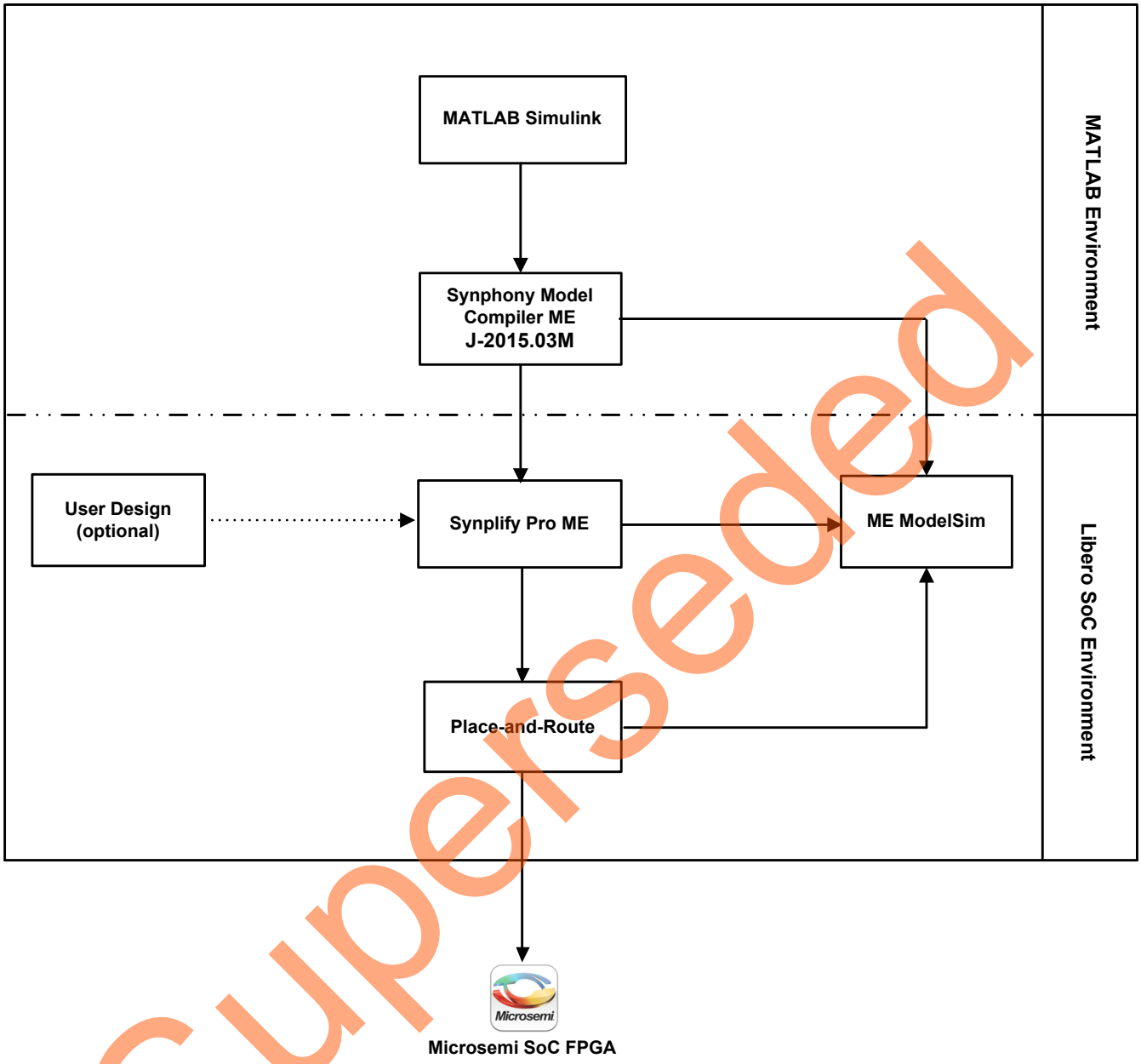

Figure 1. DSP Design Flow

Tutorial Steps

This tutorial is demonstrated by selecting the SmartFusion2 device as the family. For IGLOO2 devices, select IGLOO2 as the family. This tutorial comprises the following steps:

Step 1: Create the RTL from the DSP block in MATLAB

Step 2: Create a New Libero SoC Project

Step 3: Import the RTL, Testbench, and Test Vector Files

Step 4: Set Up the Simulation Environment and Perform Simulation

Step 5: Synthesize the Design with Synplify Pro ME

Step 6: Place and route the Design Using Libero SoC tool

Step 1 is performed in the MATLAB environment and the remaining steps are performed in Libero SoC.

Step 1: Create an RTL from the DSP Block in MATLAB

The following steps describe how to open MATLAB and create RTL code using Synphony Model Compiler ME:

1. Download the *ddc.zip* file from *www.microsemi.com/soc/documents/DDC.zip* to the desktop. Unzip the file, and save all items to **C:\demo**.
2. Open **MATLAB** and set the current folder location as **C:\demo**.
3. Double-click the **ddc.mdl** file to open the design file. This file is located in **C:\demo**.
4. The `ddc.mdl` design is shown in Figure 2. Ignore the messages that pops-up by clicking **OK** and if `ddc_obsolete.mdl` pops-up, close the file.

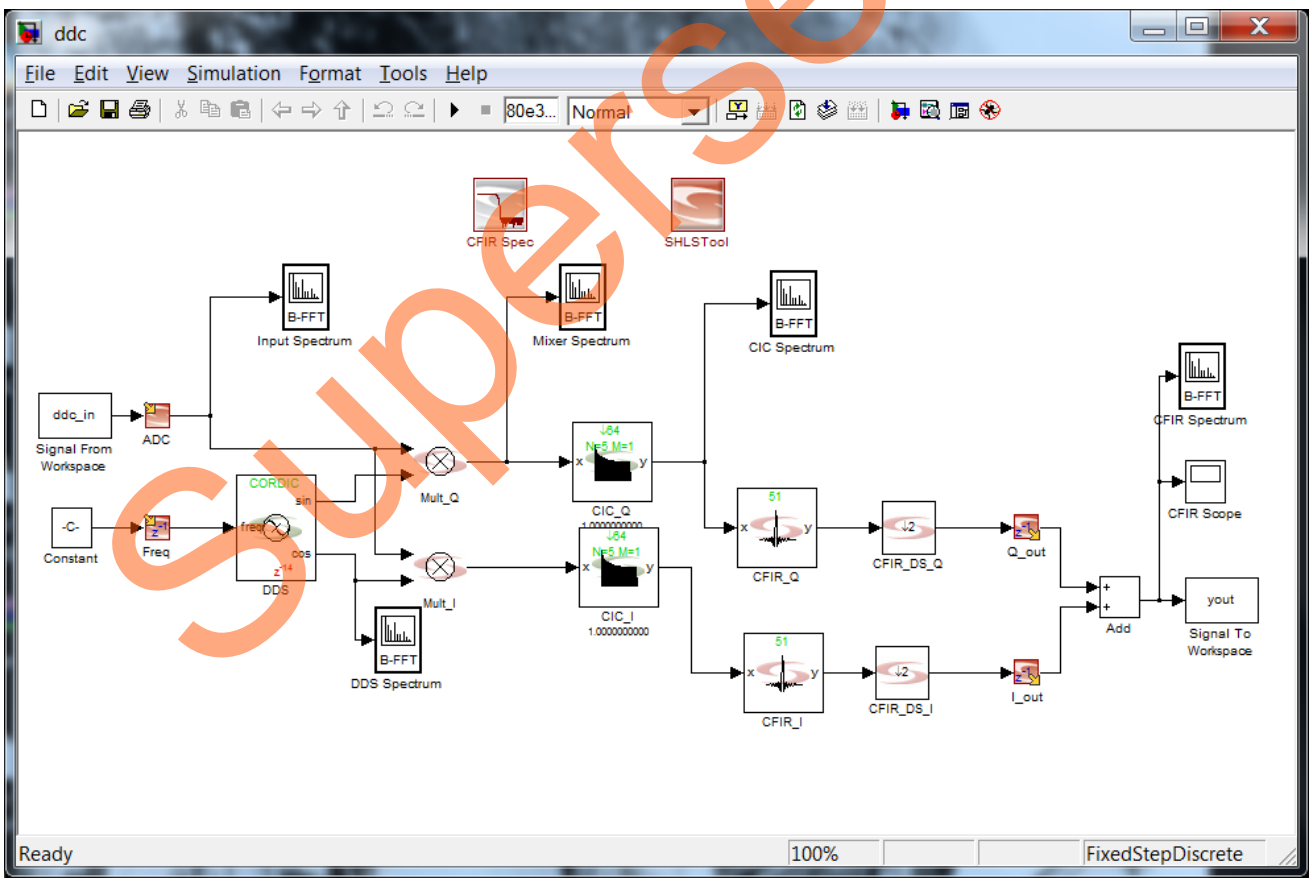

Figure 2. DDC Design

5. Click **Q_out**, and on the **Function Block Parameters** dialog box select the **Capture test vectors for RTL Test bench** and **Register output** check boxes, as shown in Figure 3.

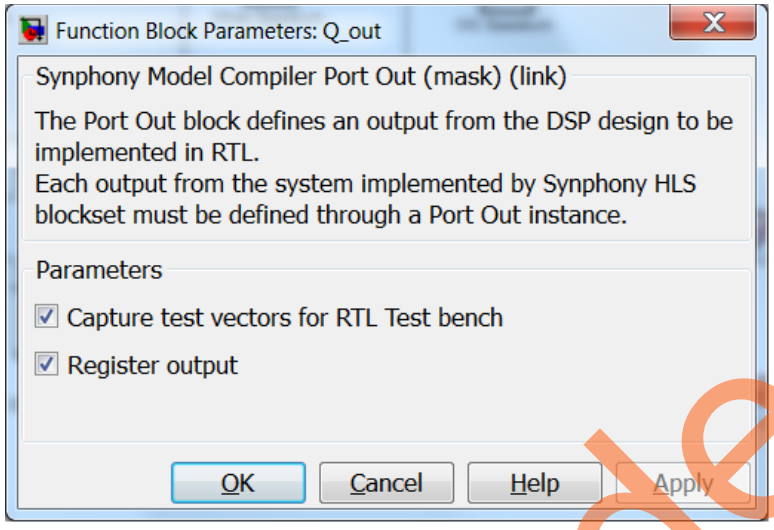

Figure 3. Setting the Output Port Q_out

6. Click **I_out**, and on the Function **Block Parameters** dialog box select the **Capture test vectors for RTL Test bench** and **Register output** check boxes, as shown in Figure 4.

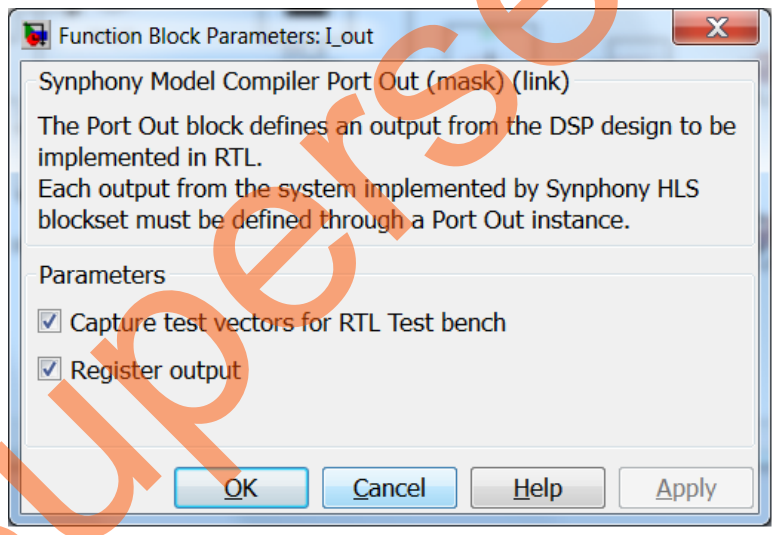

Figure 4. Setting the Output Port I_out

7. Click the arrow button to simulate the design, as shown in Figure 5. The spectrums can be viewed in the scope.

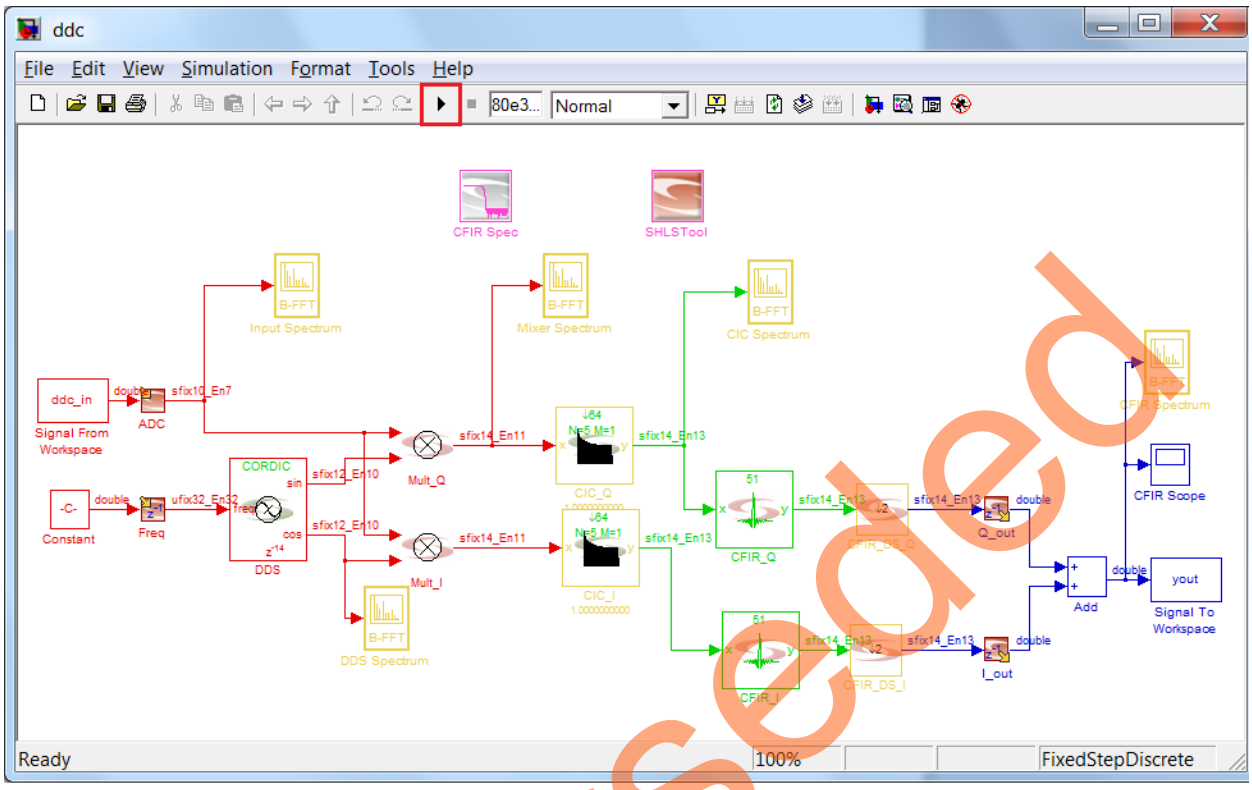

Figure 5. DDC Design After Simulation

8. Double-click the **SHLS Tool** toolbox inside the model file and launch the Synphony Model Compiler ME J-2015.03M, as shown in Figure 6.

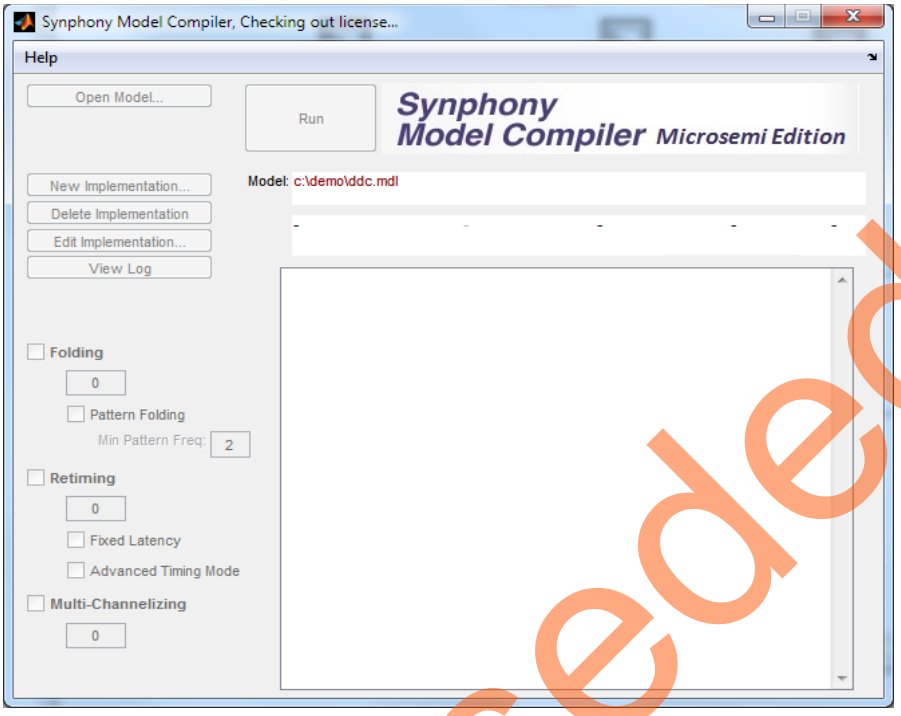

Figure 6. Synphony Model Compiler ME User Interface

9. Click **New Implementation** and in the **Target Options** tab, select SmartFusion2 or IGLOO2 as the **Technology**.

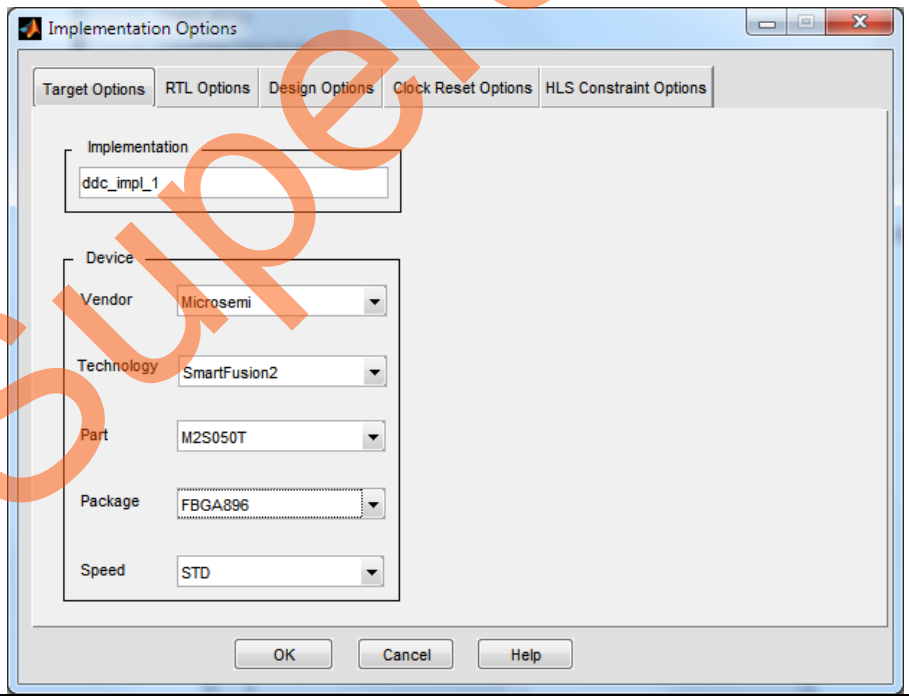

Figure 7. Implementation Options - Target Options tab

10. On the **RTL Options** tab, select the **Generate VHDL**, Generate **Verilog**, and **Generate RTL testbench** check boxes, as shown in Figure 8.

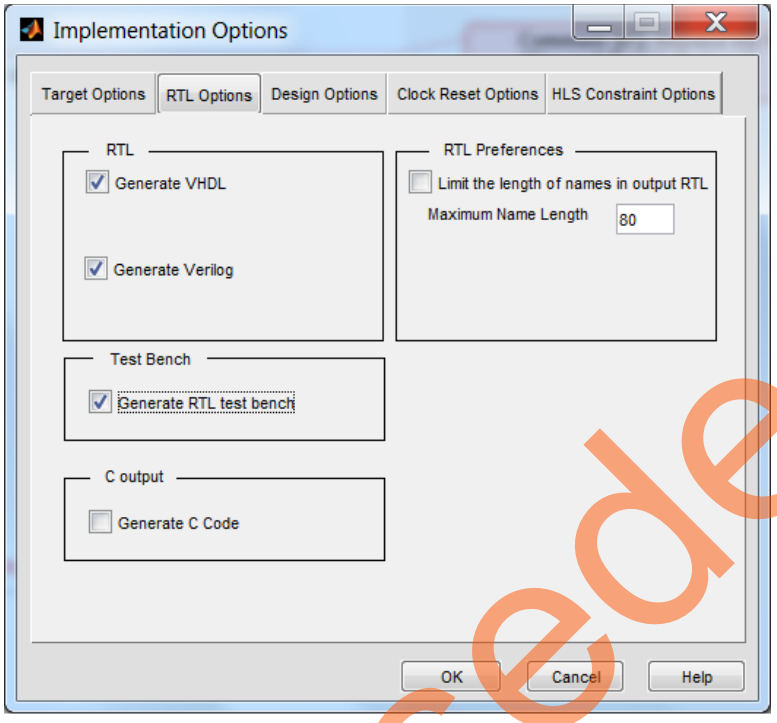

Figure 8. Implementation Options - RTL Options tab

11. Click the **Design Options** tab and set the options, as shown in Figure 9.

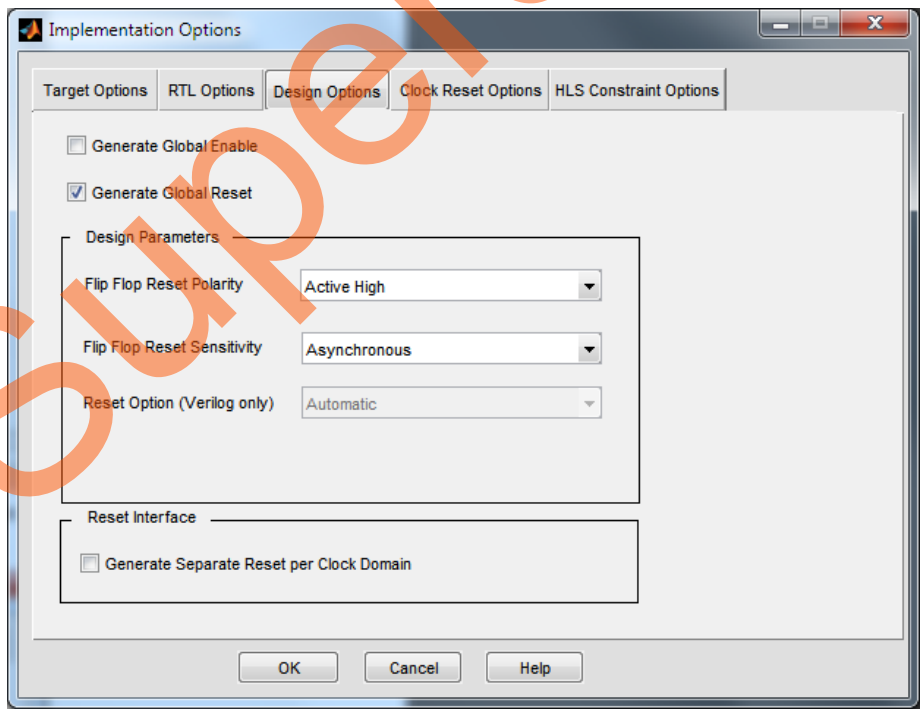

Figure 9. Implementation Options - Design Options tab

The Flip Flop Reset Sensitivity can either be Synchronous or Asynchronous, as shown in Figure 9.

12. Click **OK** to accept all the other implementation settings.
13. Click **Run** to run synthesis with the Synphony Model compiler, as shown in Figure 10.

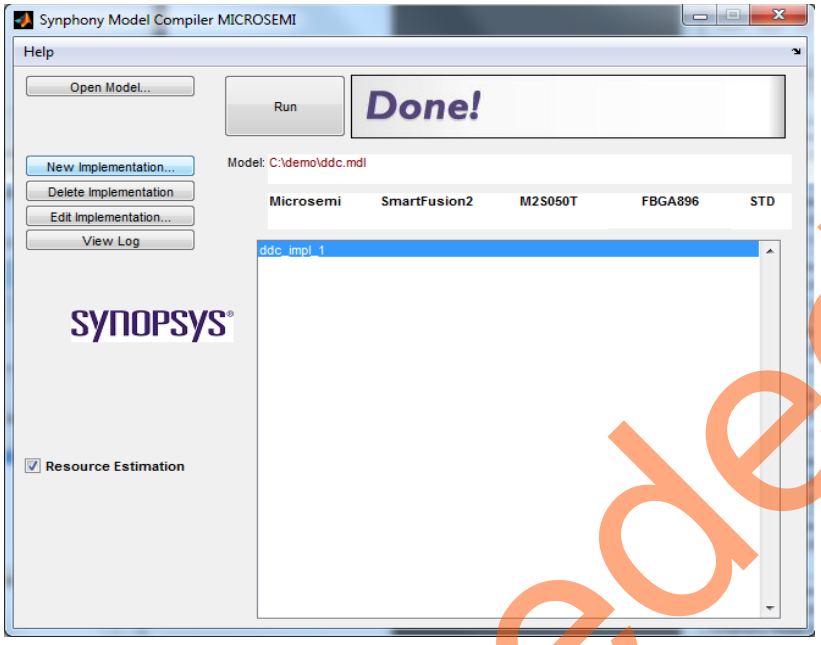

Figure 10. Run Synphony Model Compiler

14. After the synthesis is done, close the Synphony Model compiler. The Synphony Model Compiler ME creates files in the path: *C:\demo\ddc_impl_1\vhdl*, including the RTL, as shown in Figure 11.

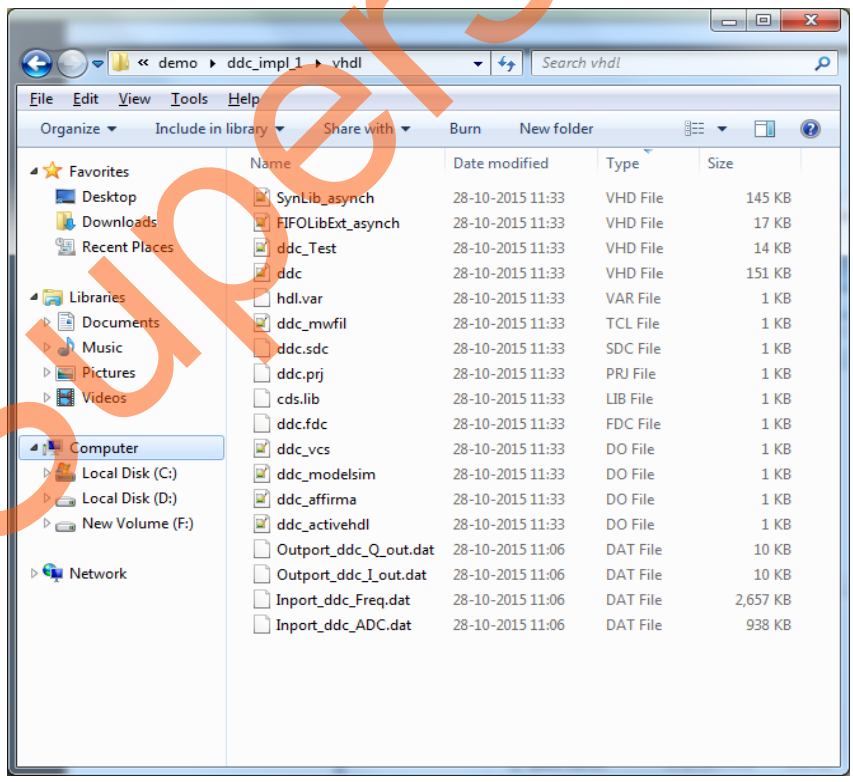

Figure 11. VHDL Files Generated by Synphony Model Compiler ME

For Verilog flow, Synphony Model Compiler ME creates files at: *C:\demo\ddc_impl_1\verilog*, as shown in Figure 12.

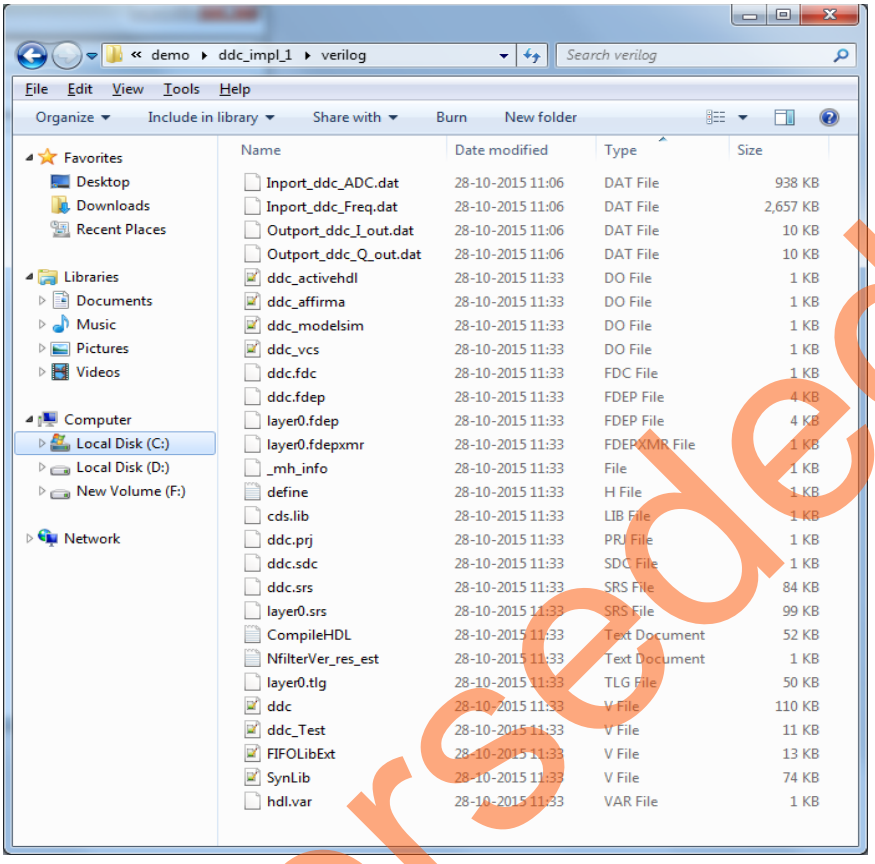

Figure 12. Verilog Files Generated by Synphony Model Compiler ME

15. Close **MATLAB**.

    The RTL files generated from the MATLAB Simulink software using Synphony Model Compiler ME J-2015.03M are ready to be used and evaluated on a hardware platform. The Libero SoC is the comprehensive software suite for designing SmartFusion2 and IGLOO2 devices, managing the entire design flow from design entry, synthesis and simulation through place-and-route, and timing and power analysis with enhanced integration of the embedded design flow.

Step 2: Create a New Libero SoC Project

The following steps use the Libero SoC software to create a project for the tutorial design. A Libero SoC project sets the design name, the HDL flavor (VHDL or Verilog), and the tool locations.

1. Double-click the Libero SoC icon on the desktop to start the Libero SoC Project Manager.
2. In the Project menu, select New Project. This displays the New Project dialog box, as shown in Figure 13.
3. Set the following values in the New Project dialog box:
   - Project name: DDC_top
   - Project location: C:\demo
   - Preferred HDL type: VHDL

   Note: For Verilog flow, preferred HDL type: **Verilog**.

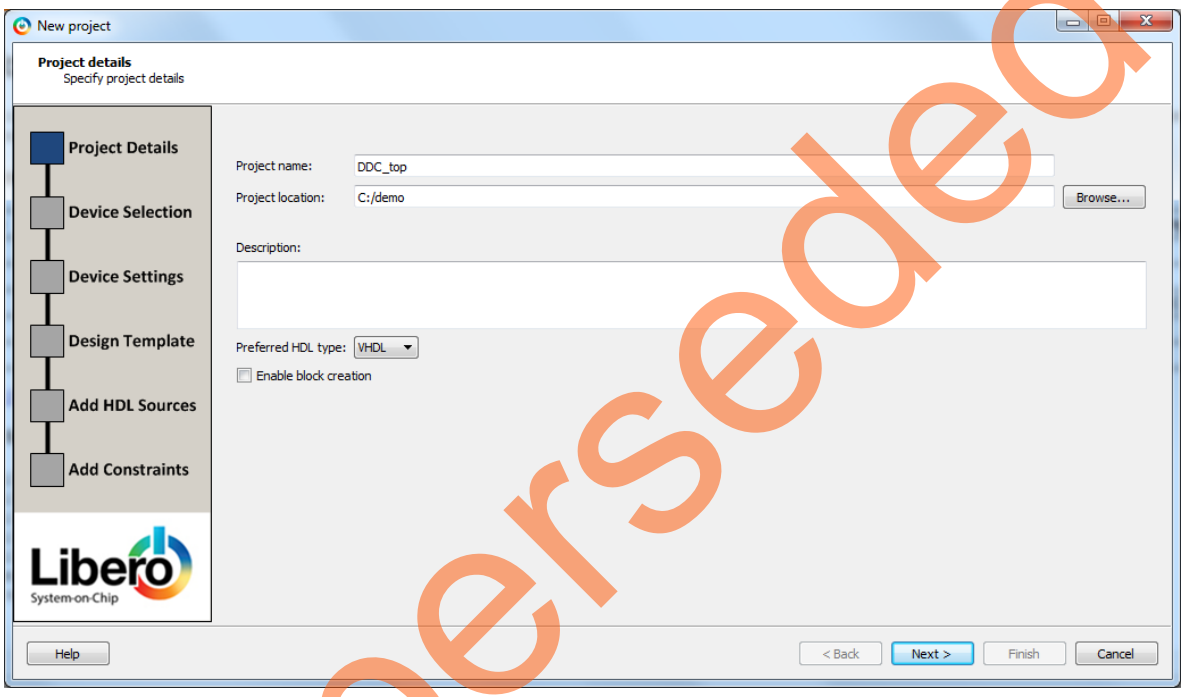

Figure 13. Libero SoC New Project – Project Details

4. Set the following values, as shown in Figure 15:
   - Family: SmartFusion2
   - Die: M2S050T
   - Package: 896 FBGA
   - Speed: STD
   - Core voltage: 1.2 V
   - Range: COM

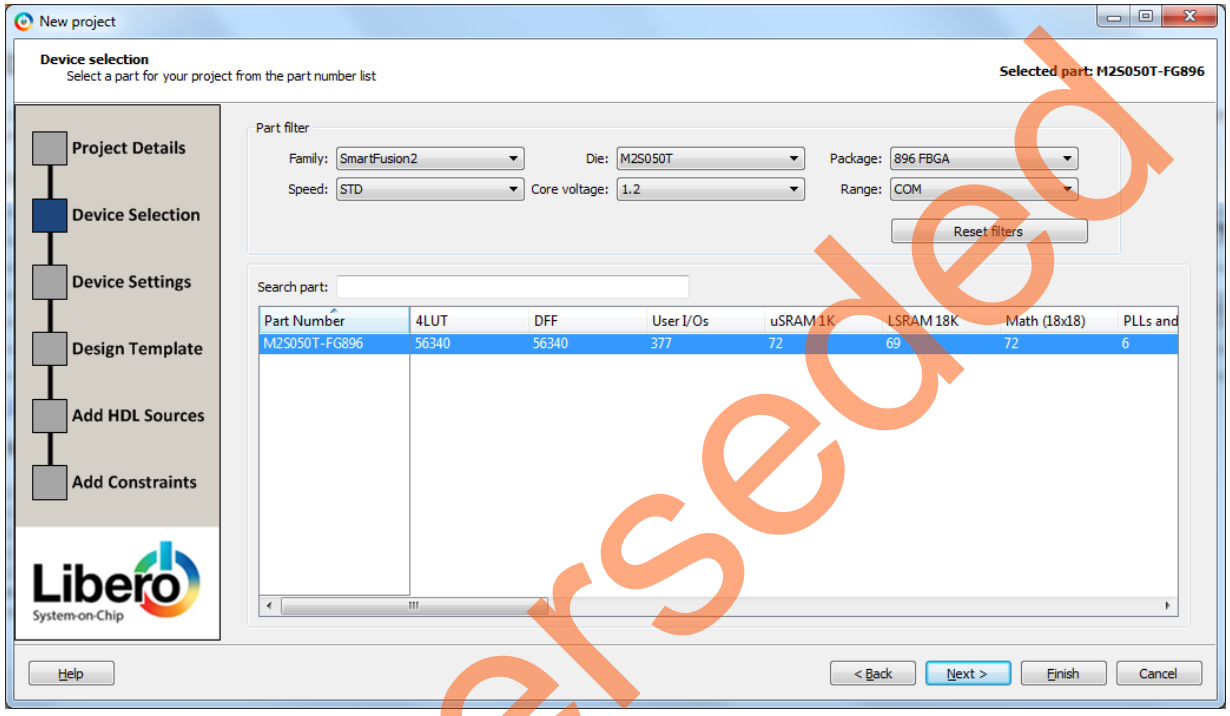

Figure 14. Libero SoC New Project – Device Selection

5. Click **Next** to continue.
6. On the **Design template** window, select **None** for Design templates and creators.
7. Click **Next** to continue and then click **Finish**.

8. Choose the appropriate tools settings, if they are not already selected, as shown in Figure 15. The tools shown in this dialog box depend on the installation. To change the default tool settings, refer to the *Libero SoC Online Help*.

   Note: FPGA programming is not performed as a part of the tutorial. Therefore, the programming tool selection is not important for this tutorial.

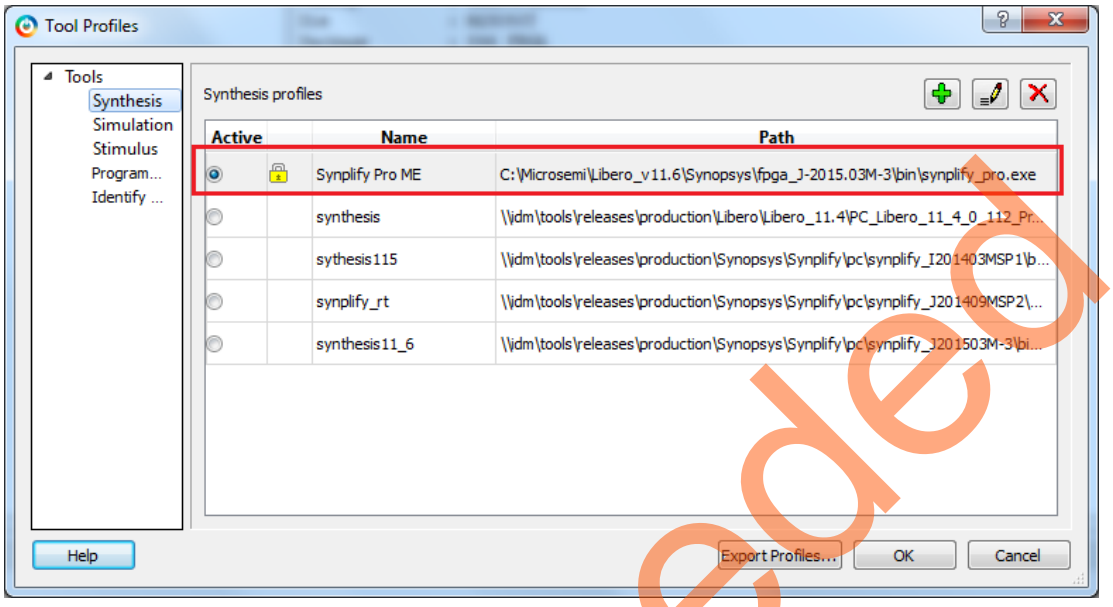

Figure 15.Tool Profiles Dialog Box

Step 3: Import the RTL, Testbench, and Test Vector Files

The following steps describe how to import the RTL, testbench, and test vector into Libero SoC:

1. Navigate to the **File** menu in Libero SoC, and choose **Import Files**. For **Files of type** select **HDL source files** (`*.vhd,*.v,*.h`) and import the `ddc.vhd`, `SynLib_asynch.vhd`, and `Cordic_asynch.vhd` files from the path *c:\demo\ddc_impl1\vhdl*, as shown in Figure 16. In this design tutorial, only the `ddc.vhdl` file and its associated files generated in Step 2: Create a New Libero SoC Project are used.

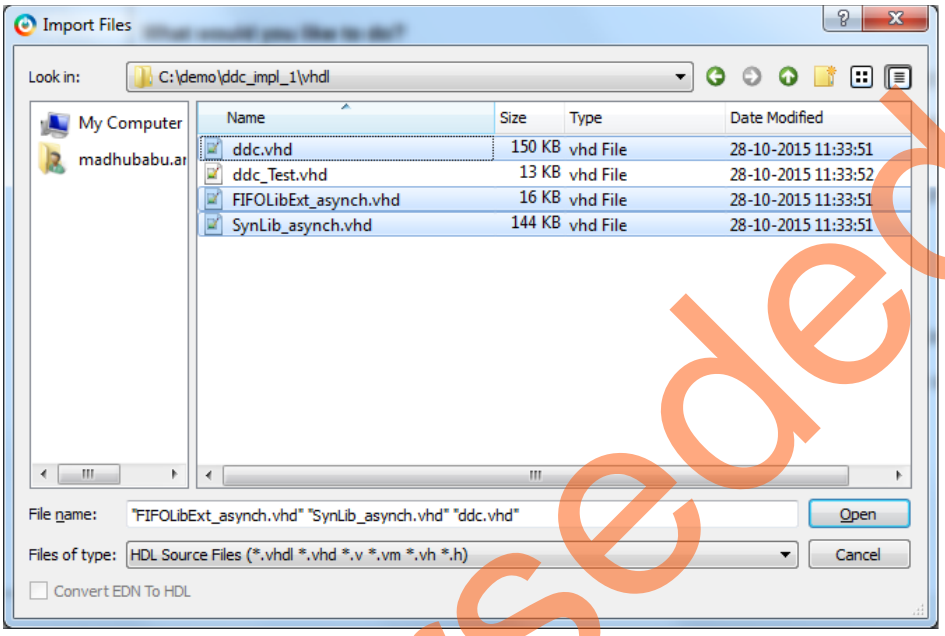

Figure 16. Importing VHDL Source Files

For Verilog flow, import `ddc.v`, `SynLib.v`, and `define.h` files, from the path *c:\demo\ddc_impl1\verilog*, as shown in Figure 17.

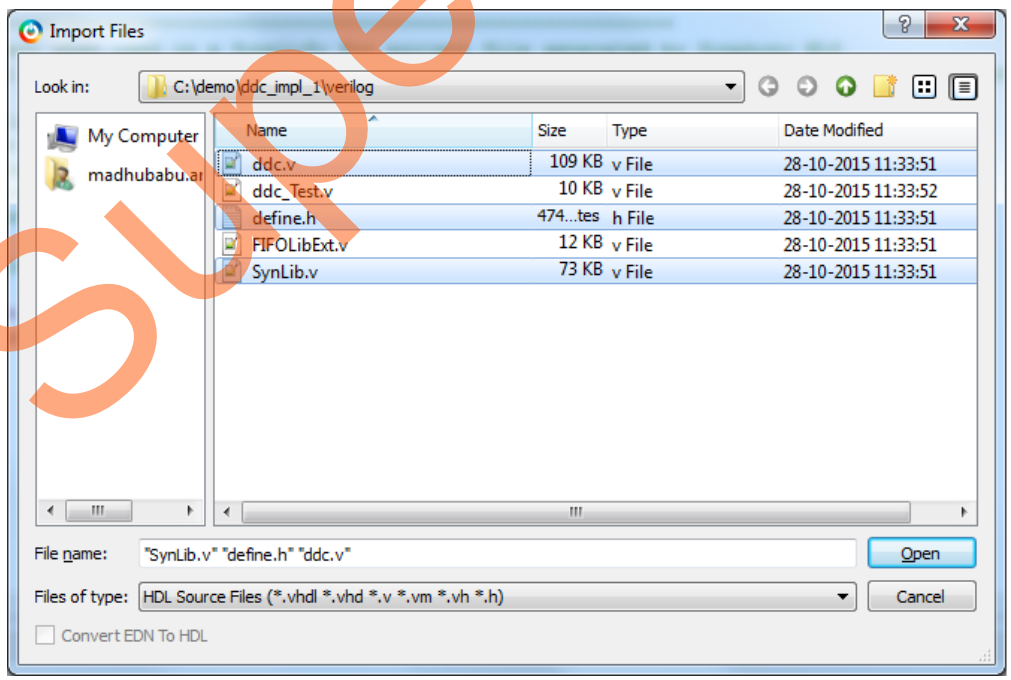

Figure 17. Importing Verilog Source Files

2. In the **File** menu in Libero SoC, select **Import Files**. For **Files of type**, select **HDL Stimulus file** (`*.vhd,*.v`) and import the `ddc_Test.vhd` file. For Verilog flow, import `ddc_Test.v` file.
3. In the **File** menu in Libero SoC, select **Import Files**. For **Files of type**, select **Simulation files** (`*.mem,*.bfm,*.dat,*.txt,*. do`) and import all the `*.dat` files. The required files are copied into the respective folders, as shown in Figure 18. The Verilog files are copied into the respective folders, as shown in Figure 19.

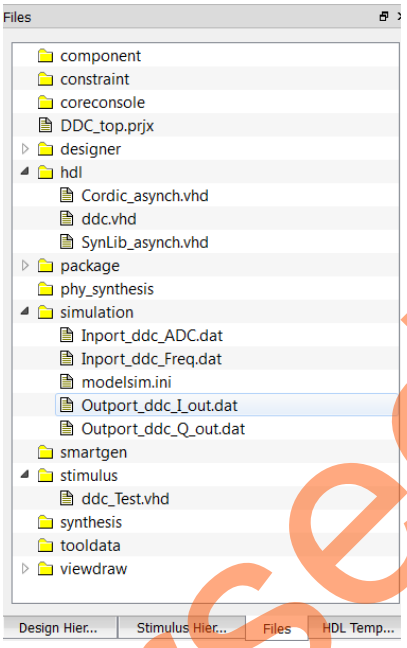

Figure 18. Imported VHDL Source Files into Libero SoC

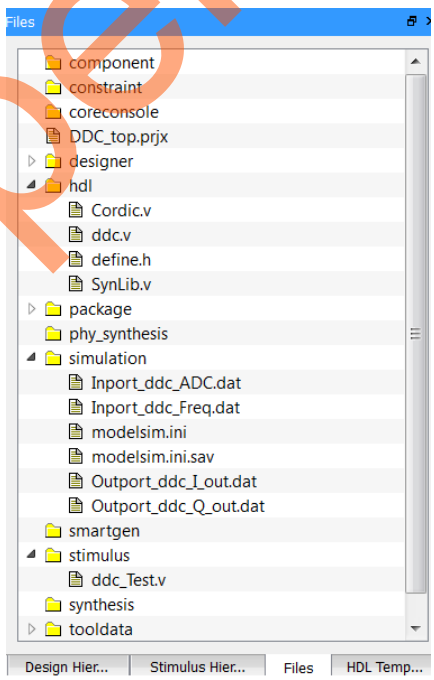

Figure 19. Imported Verilog Source Files into Libero SoC

4. In VHDL flow, open the `ddc.vhd`, `SynLib_asynch.vhd`, and `Cordic_asynch.vhd` files. Use the Libero SoC Find and Replace feature to replace **SHLSLib** with **work** and save the file. This change is required to use work, a default library, so that ModelSim and Synplify can find the package definition from the work library.
5. In Verilog flow, open the `ddc.v` file and add **include "define.h"** before module declaration and save the file.

Step 4: Set Up the Simulation Environment and Perform Simulation

After the RTL, testbench, and test vector files are imported, use ModelSim to perform a pre-synthesis simulation. Associate a stimulus file with the design. This lets the ModelSim tool to know which testbench to be used for the simulation.

To Set Up the Stimulus File:

1. For a VHDL flow, on the **Design Hierarchy** tab, right-click **ddc.vhd** and select **Set As Root**, whereas for a Verilog flow, right-click **ddc.v** and select **Set As Root**, as shown in Figure 20.

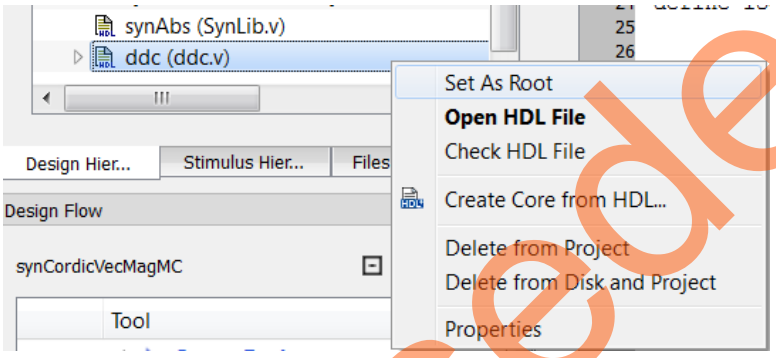

Figure 20. Set as Root

2. Under **Verify Pre-synthesized Design**, right-click **Simulate** and select **Organize Input files > Organize Stimulus Files**, as shown in Figure 21. The **Organize Stimulus** dialog box is displayed, as shown in Figure 22.

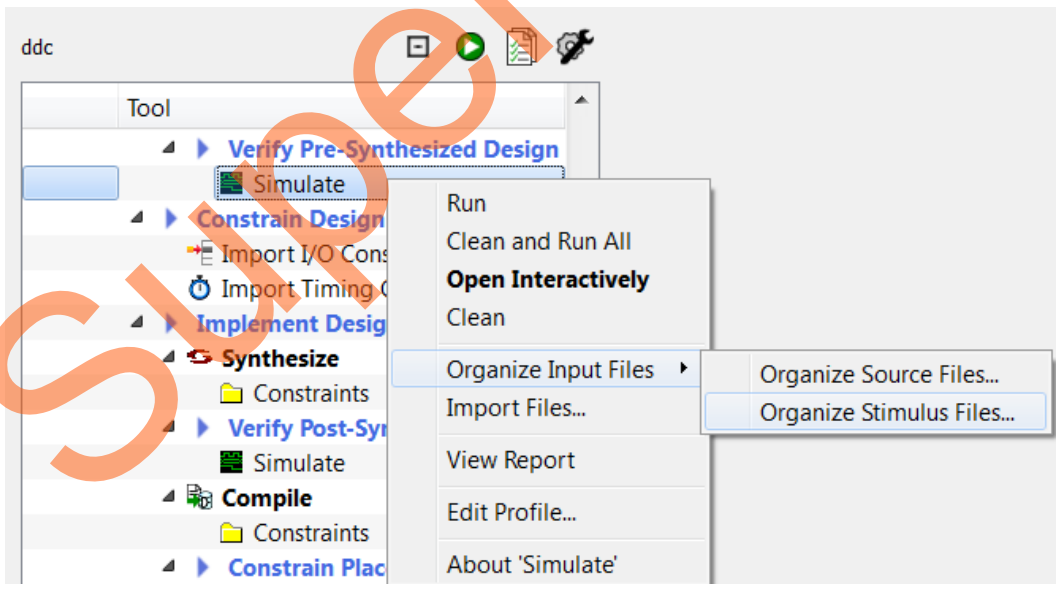

Figure 21. Organize Stimulus File

3. Select **User** option to add the stimulus file under **Associated Stimulus Files**, as shown in Figure 22 for VHDL flow and Figure 23 for Verilog flow.

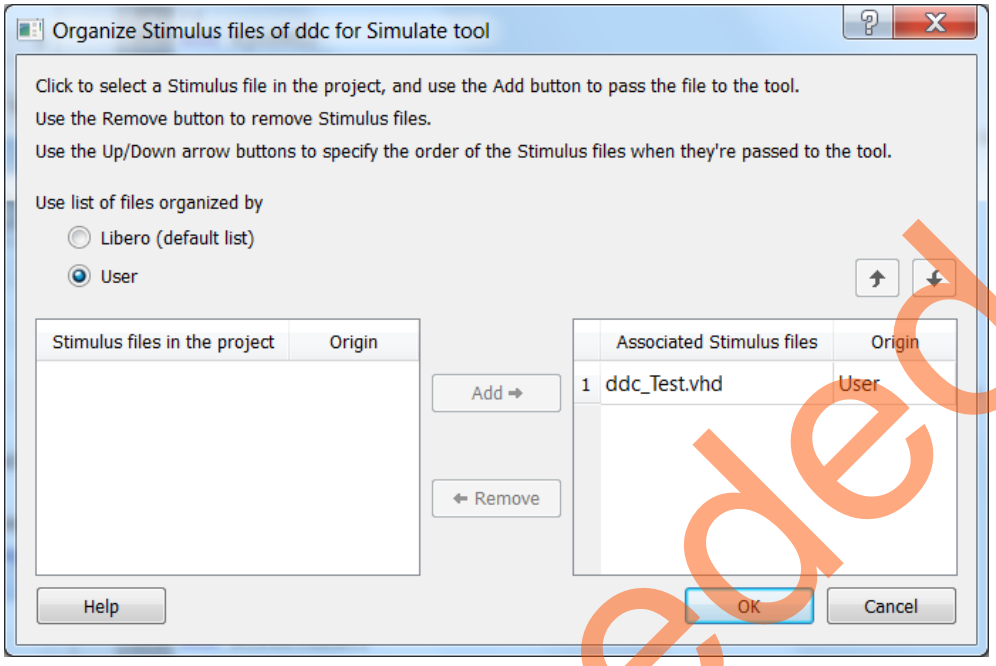

Figure 22. Organize Stimulus Dialog Box for VHDL

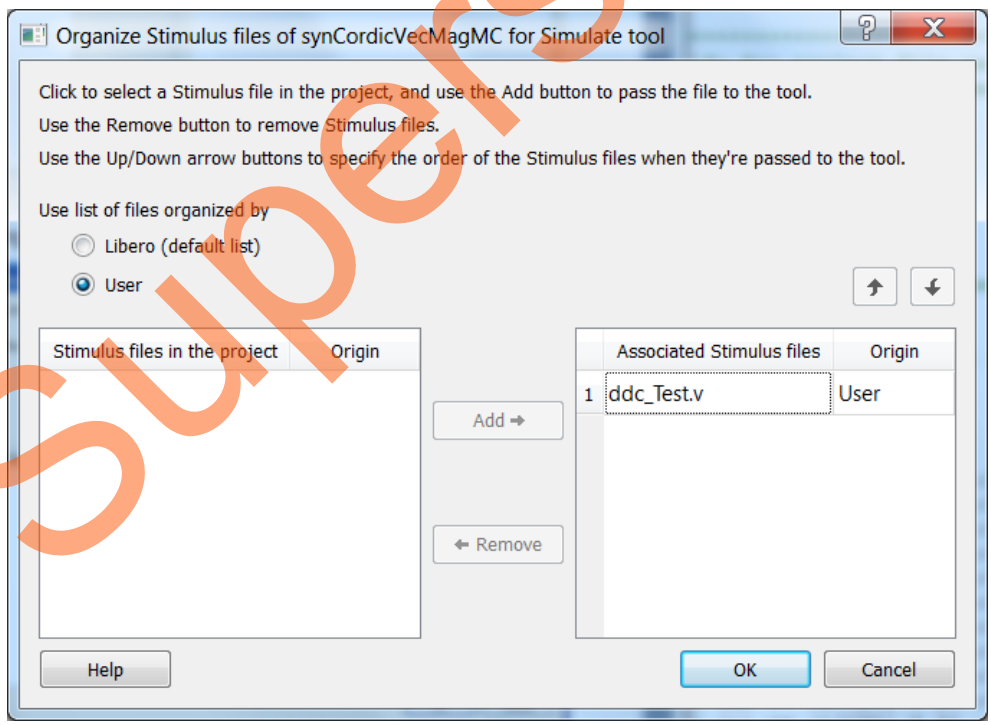

Figure 23. Organize Stimulus Dialog Box for Verilog

After selecting the stimulus file, set the simulation environment.

4. In the **Project** menu, select **Project Settings** to open the **Project Settings** dialog box, as shown in Figure 24.

5. In the **Simulation** options, click the **Do** tab and set the following:
   - Simulation runtime: **-all**
   - Testbench module name: **test_ddc**
   - Top Level instance: **i_ddc**
6. Click **Save**.

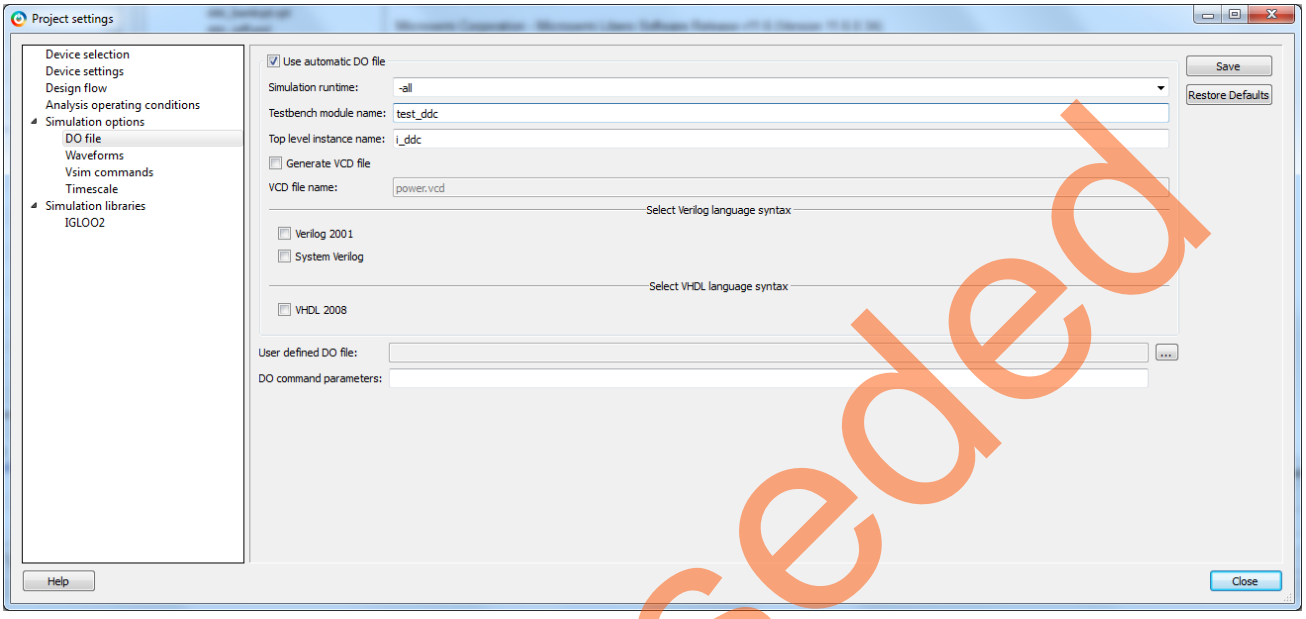

Figure 24. Project Settings – Simulation Options Dialog Box

7. Under **Verify Pre-synthesized Design**, right-click **Simulate** and select **Open interactively**. The ModelSim VHDL simulator opens and runs simulation using the `run.do` file, as shown in Figure 25.

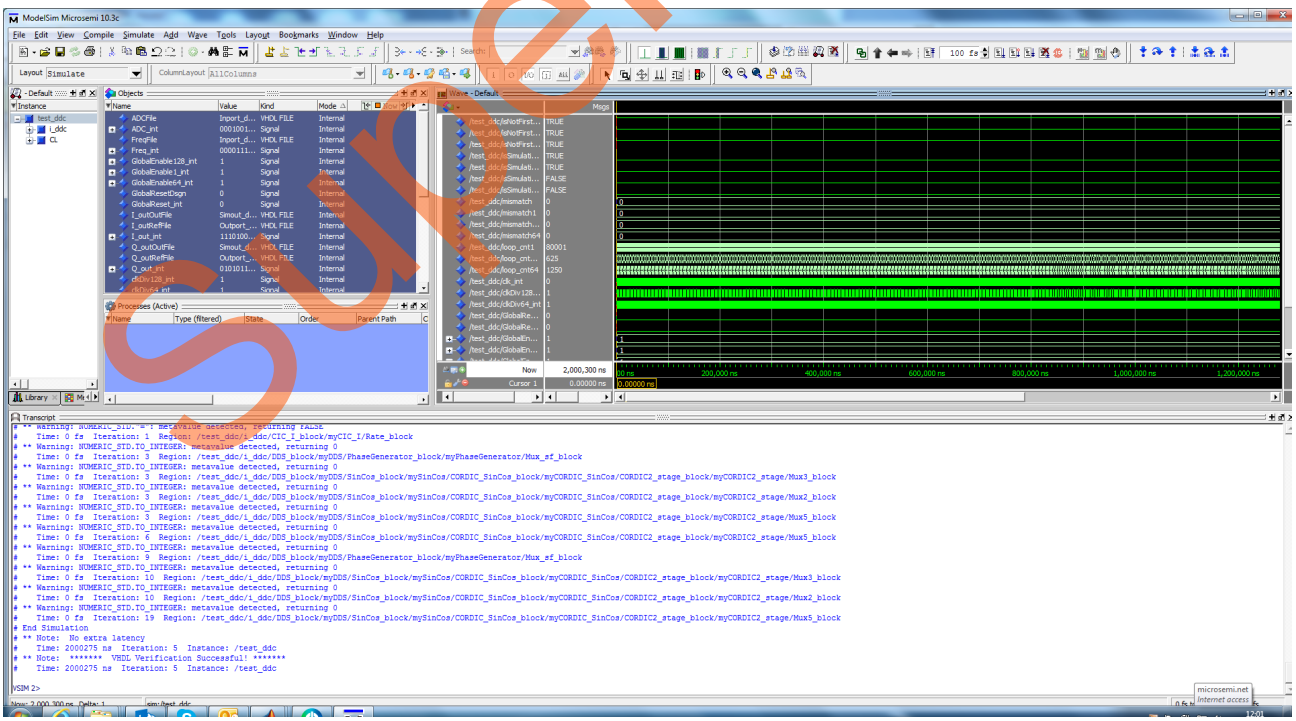

Figure 25. ModelSim User Interface for VHDL

For VHDL, the following message appears after the successful completion of testbench simulation. The testbench simulation takes two seconds to complete.

```
Time: 2000275 ns Iteration: 5 Instance: /test_ddc
# ** Note:  *******  VHDL Verification Successful! *******
```

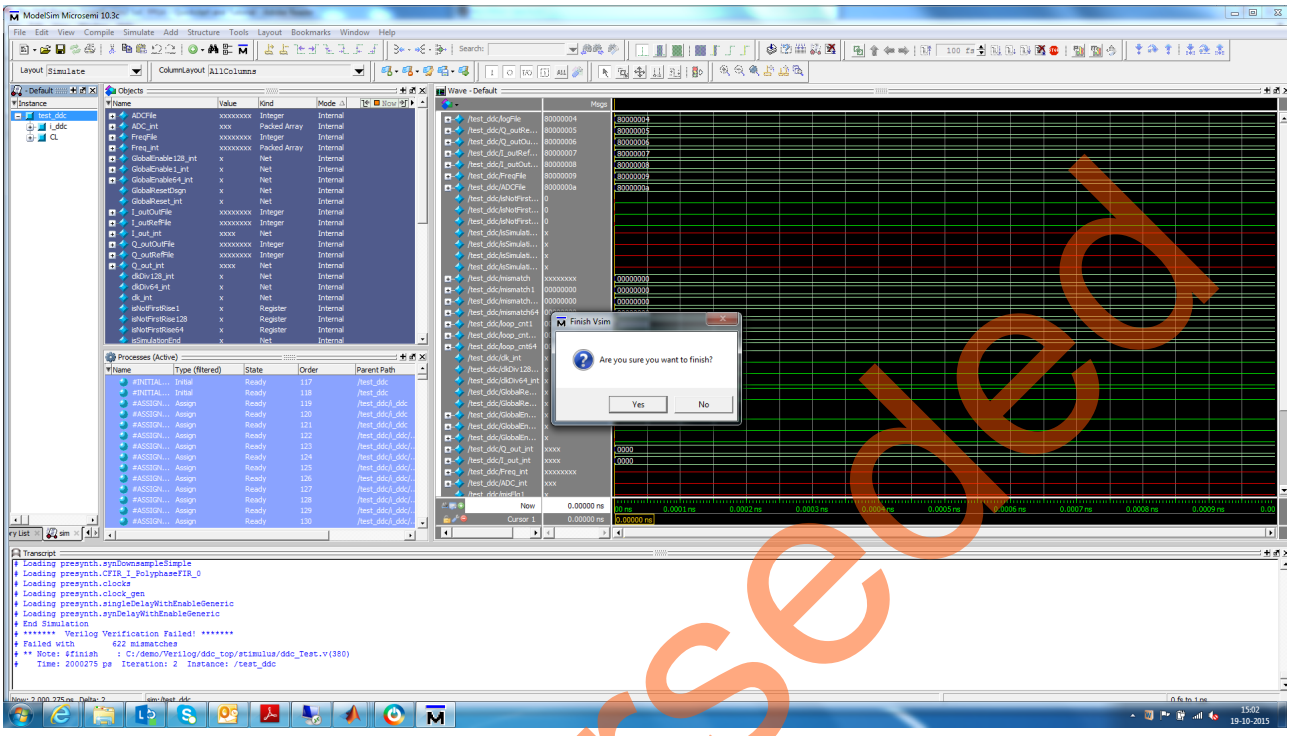

Figure 26. ModelSim User Interface for Verilog

For Verilog, the following message appears after the successful completion of testbench simulation:

```
*******  Verilog Verification Successful! *******
Time: 2000275 ps Iteration: 2 Instance: /test_ddc
```

8. Click **Yes** to finish Verilog simulation.

Step 5: Synthesize the Design with Synplify Pro ME

The following steps describe how to generate an EDIF netlist (**.edn* file) by synthesizing the design with Synplify Pro ME, as shown in Figure 30. Synplify Pro ME instantiates I/O buffers, synthesizes logic for the behavioral blocks in the design, utilizes hardcore mathblocks for multiplication and addition operations, and generates an EDIF netlist for you to place-and-route.

1. Add constraints by adding the SDC file (`ddc.sdc`) to the Synthesis tool from the path *C:\demo\ddc_impl_1\verilog or C:\demo\ddc_impl_1\vhdl*, as shown in Figure 27, Figure 28, and Figure 29.

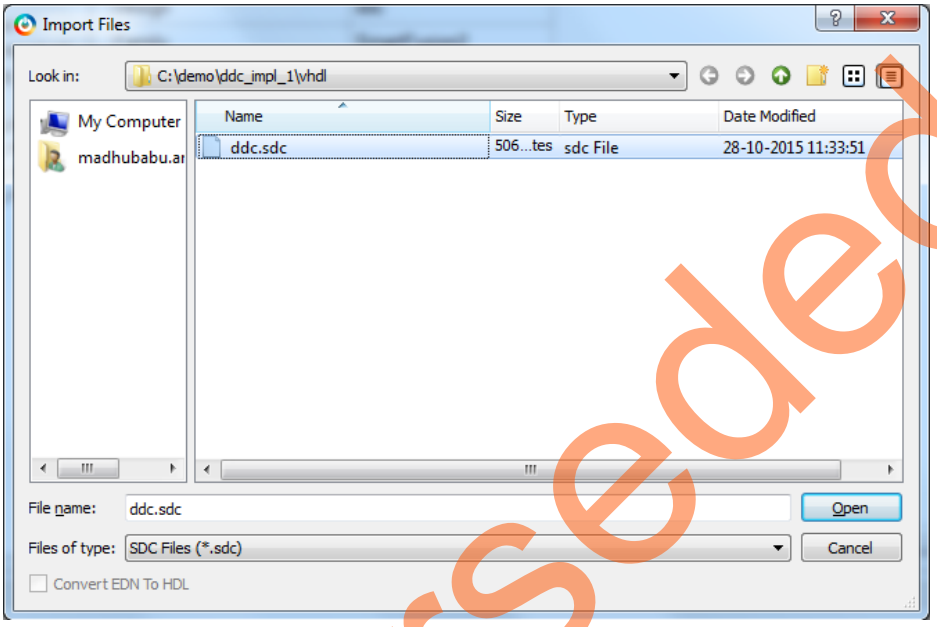

Figure 27. Importing SDC Source File

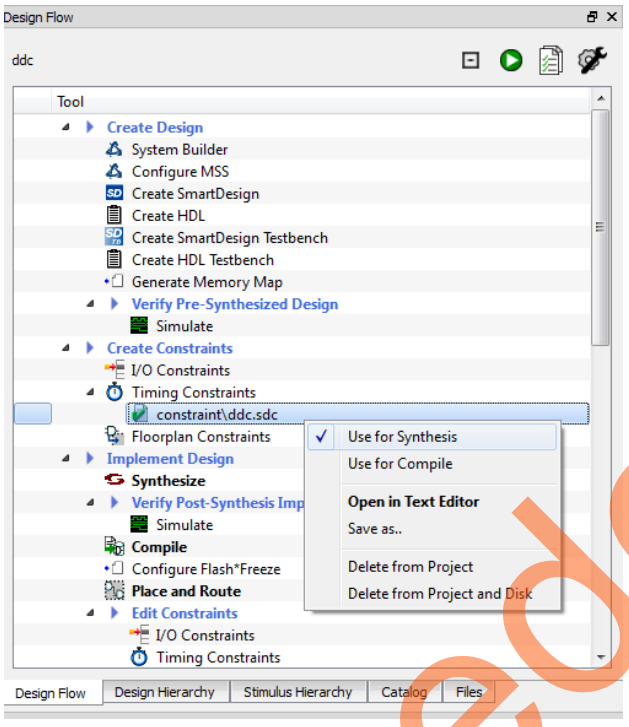

Figure 28. Selecting Constraints to Synthesis Tool

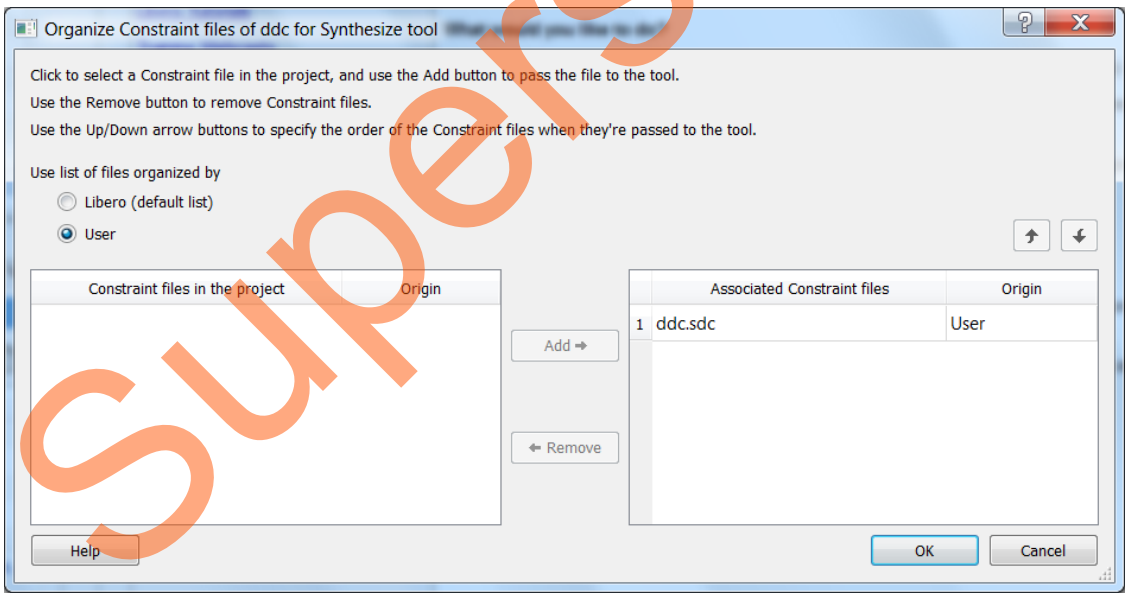

Figure 29. Organize Constraint Files of ddc for Synthesis Tool

2. Under **Implement design**, right-click **Synthesize** and click **Open Interactively**. Ignore the pop-up messages displayed. This launches the Synopsys Synplify Pro ME tool with the appropriate design files, as shown in Figure 30.
3. Set the **Frequency** to 40 MHz (Default).
4. For VHDL flow, select **Run** to synthesize the design.

After successful synthesis, in the Project Status tab, under **Area Summary**, click **Detailed report** for the number of DSP math blocks inferred for multiplier operations, as shown in Figure 30.

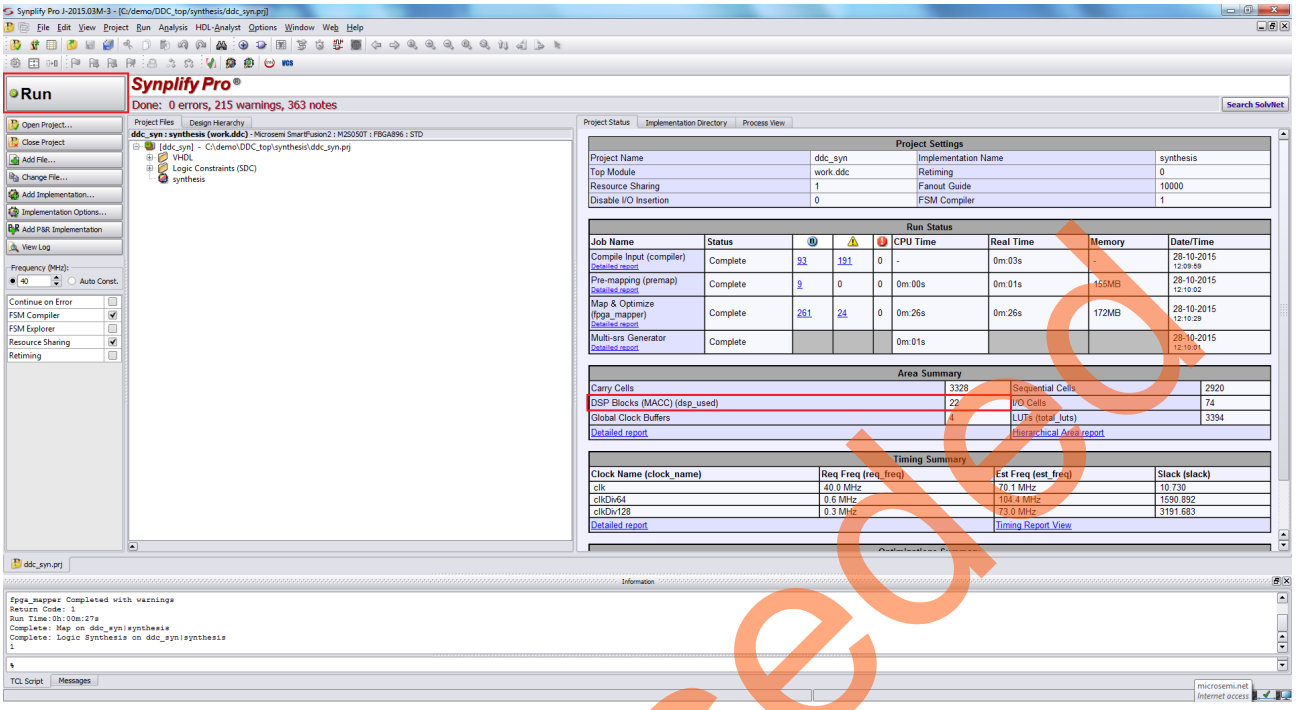

Figure 30. Synplify Pro GUI for VHDL Flow

5. For a Verilog flow, under Synopsys Synplify Pro ME, click **Implementation Options** and select Verilog language as **Verilog2001**, as shown in Figure 31.
6. Select **Run** to synthesize the design, as shown in Figure 30.

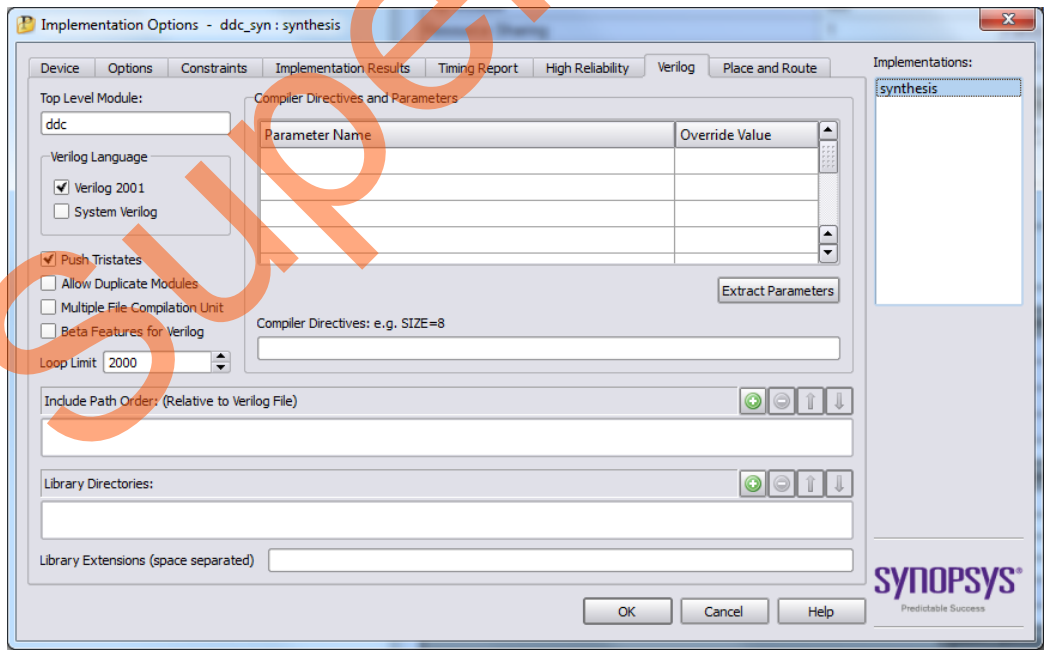

Figure 31. Implementation Options for Verilog Flow

In Figure 32, note the number of DSP mathblocks inferred for multiplier operations in DDC design.

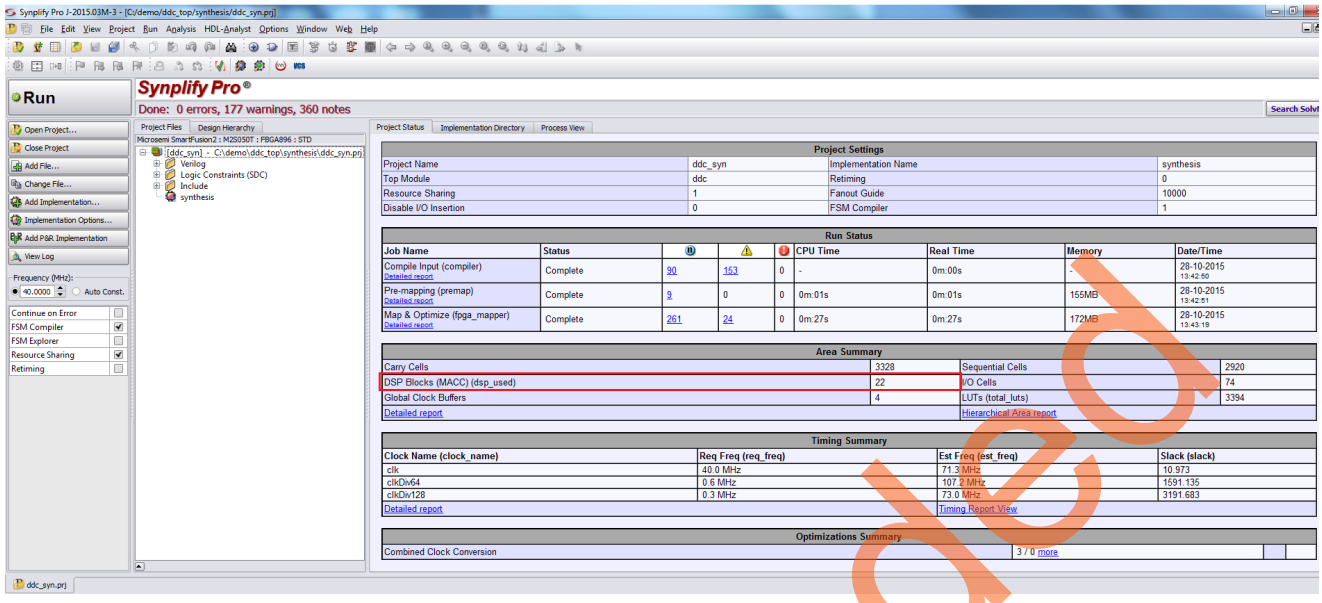

Figure 32. Synplify Pro GUI for Verilog Flow

Step 6: Place-and-Route the Design Using Libero SoC Tool

The following steps describe how to place-and-route the design in a SmartFusion2 M2S050T device using the Microsemi SoC Products Group designer software.

1. Right-click **Compile**, select **Organize Input Files** and click **Organize Constraint Files**, as shown in Figure 33.

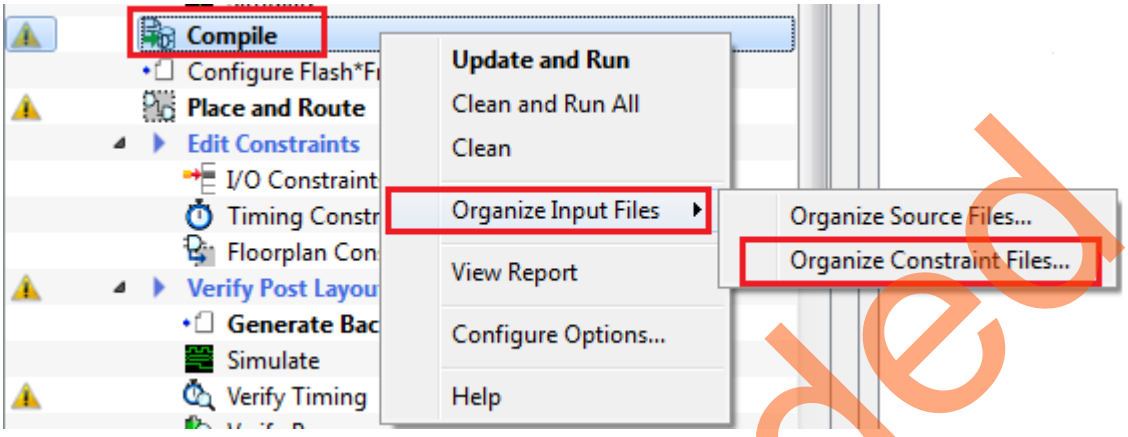

Figure 33. Organize Timing Constraints

2. Select **ddc_sdc.sdc**, click **Add** and click **OK**, as shown in Figure 34.

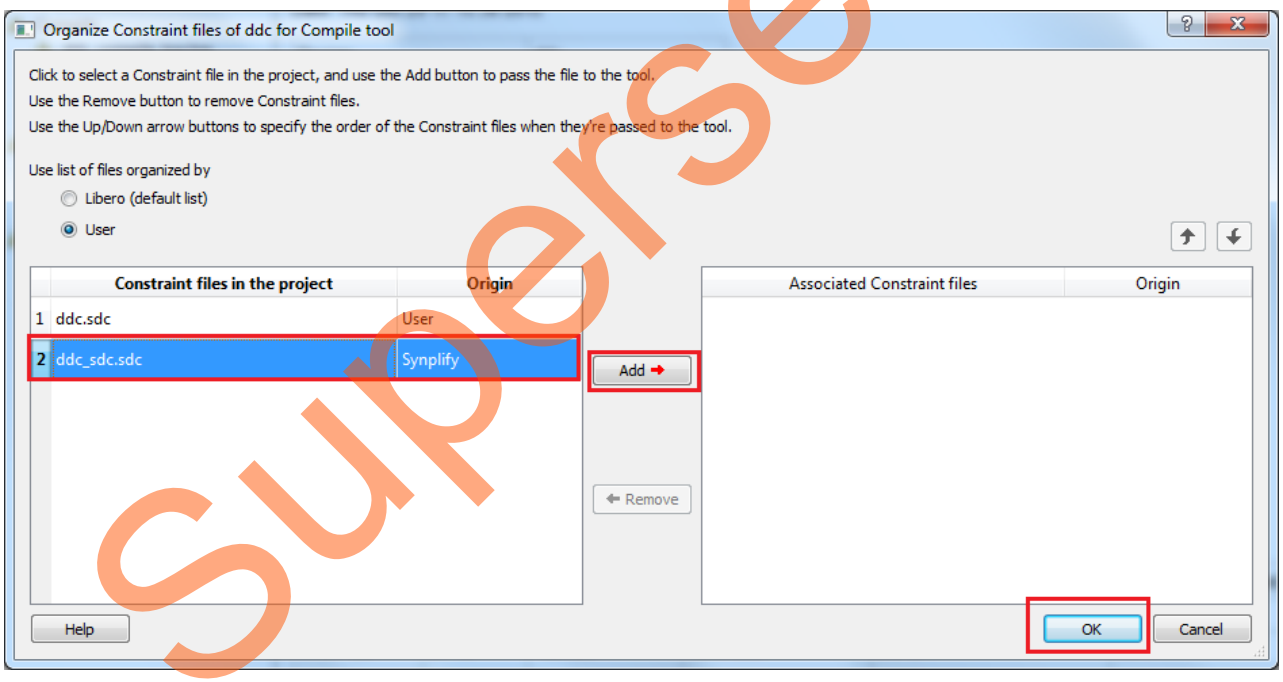

Figure 34. Add Constraint in Compile

3. On the **Design flow** tab, right-click **Verify Timing** and select **Run**. After successful completion, a green tick mark is displayed next to **Place and Route** and **Verify Timing**, as shown in Figure 35.

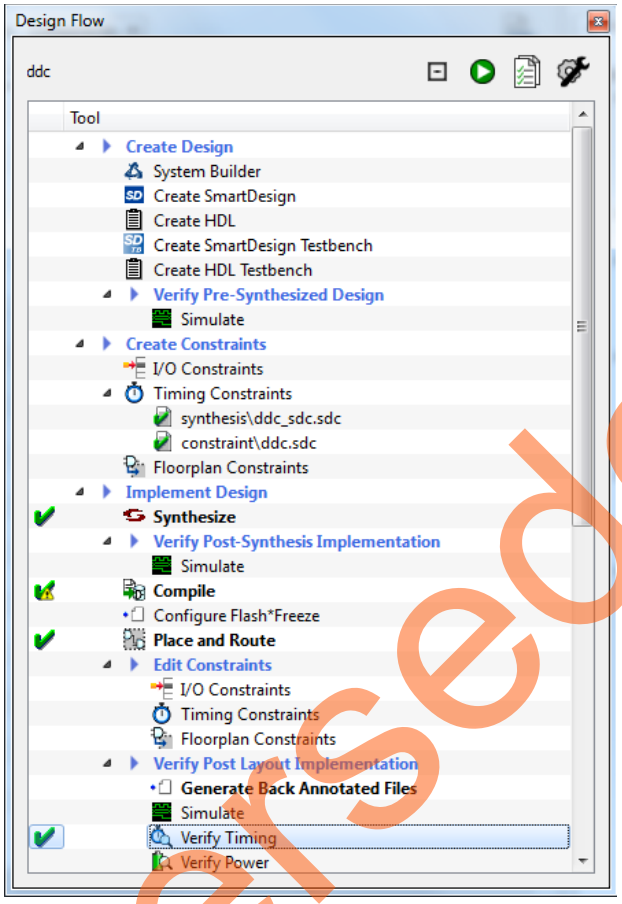

Figure 35. Implementing Place-and-Route

List of Changes

The following table shows the important changes made in this document for each revision.

Revision	Changes	Page
Revision 6 (November 2015)	Updated the document for Libero SoC 11.6 software release (SAR 73097).	NA
Revision 5 (April 2013)	Updated the document for 11.0 production SW release (SAR 47103).	NA
Revision 4 (February 2013)	Updated the document for Libero 11.0 beta SP1 software release (SAR 45203).	NA
Revision 3 (November 2012)	Updated the document for Libero 11.0 beta SPA software release (SAR 42881).	NA
Revision 2 (October 2012)	Updated the document for Libero 11.0 beta launch (SAR 41733).	NA
Revision 1 (May 2012)	Updated the document for LCP2 software release (SAR 38955).	NA

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

.

Product Support

Microsemi SoC Products Group backs its products with various support services, including Customer Service, Customer Technical Support Center, a website, electronic mail, and worldwide sales offices. This appendix contains information about contacting Microsemi SoC Products Group and using these support services.

Customer Service

Contact Customer Service for non-technical product support, such as product pricing, product upgrades, update information, order status, and authorization.

From North America, call **800.262.1060**
From the rest of the world, call **650.318.4460**
Fax, from anywhere in the world **408.643.6913**

Customer Technical Support Center

Microsemi SoC Products Group staffs its Customer Technical Support Center with highly skilled engineers who can help answer your hardware, software, and design questions about Microsemi SoC Products. The Customer Technical Support Center spends a great deal of time creating application notes, answers to common design cycle questions, documentation of known issues and various FAQs. So, before you contact us, please visit our online resources. It is very likely we have already answered your questions.

Technical Support

For Microsemi SoC Products Support, visit

http://www.microsemi.com/products/fpga-soc/design-support/fpga-soc-support

Website

You can browse a variety of technical and non-technical information on the Microsemi SoC Products Group home page, at http://www.microsemi.com/products/fpga-soc/fpga-and-soc.

Contacting the Customer Technical Support Center

Highly skilled engineers staff the Technical Support Center. The Technical Support Center can be contacted by email or through the Microsemi SoC Products Group website.

Email

You can communicate your technical questions to our email address and receive answers back by email, fax, or phone. Also, if you have design problems, you can email your design files to receive assistance. We constantly monitor the email account throughout the day. When sending your request to us, please be sure to include your full name, company name, and your contact information for efficient processing of your request.

The technical support email address is soc_tech@microsemi.com.

My Cases

Microsemi SoC Products Group customers may submit and track technical cases online by going to My Cases.

Outside the U.S.

Customers needing assistance outside the US time zones can either contact technical support via email (soc_tech@microsemi.com) or contact a local sales office. Visit About Us for sales office listings and corporate contacts.

ITAR Technical Support

For technical support on RH and RT FPGAs that are regulated by International Traffic in Arms Regulations (ITAR), contact us via soc_tech@microsemi.com. Alternatively, within My Cases, select **Yes** in the ITAR drop-down list. For a complete list of ITAR-regulated Microsemi FPGAs, visit the ITAR web page.

Microsemi Corporate Headquarters
One Enterprise, Aliso Viejo,
CA 92656 USA

Within the USA: +1 (800) 713-4113
Outside the USA: +1 (949) 380-6100
Sales: +1 (949) 380-6136
Fax: +1 (949) 215-4996

E-mail: sales.support@microsemi.com

Microsemi Corporation (Nasdaq: MSCC) offers a comprehensive portfolio of semiconductor and system solutions for communications, defense & security, aerospace and industrial markets. Products include high-performance and radiation-hardened analog mixed-signal integrated circuits, FPGAs, SoCs and ASICs; power management products; timing and synchronization devices and precise time solutions, setting the world's standard for time; voice processing devices; RF solutions; discrete components; security technologies and scalable anti-tamper products; Ethernet Solutions; Power-over-Ethernet ICs and midspans; as well as custom design capabilities and services. Microsemi is headquartered in Aliso Viejo, Calif., and has approximately 3,600 employees globally. Learn more at **www.microsemi.com**.

50200312-6/11.15