VHDL VITAL™

Simulation Guide

Microsemi

Table of Contents

Introduction

This *VHDL Vital Simulation Guide* contains information about using the ModelSim and Cadence NC-VHDL to simulate designs for Microsemi SoC devices. Refer to the online help for additional information about using the SoC software. Refer to the documentation included with your simulator for information about performing simulation.

Document Assumptions

This document assumes the following:

1. You have installed the Libero SoC software. This document is for Libero SoC software v10.0 and above. For previous versions of software, see the Legacy VHDL Vital Simulation Guide.
2. You have installed your VHDL VITAL simulator.
3. You are familiar with UNIX workstations and operating systems or with PCs and Windows operating environments.
4. You are familiar with FPGA architecture and FPGA design software.

Document Conventions

This document uses the following variables:

- FPGA family libraries are shown as <act_fam>. Substitute the desired FPGA family variable with the device family as needed. For example:

  ```
  vcom -work <vhd_fam> <act_fam>.vhd
  ```

- Compiled VHDL libraries are shown as <vhd_fam>. Substitute <vhd_fam> for the desired VHDL family variable as needed. The VHDL language requires that the library names begin with an alpha character.

Online Help

Microsemi SoC software comes with online help. Online help specific to each software tool is available from the Help menu.

1 – Setup

This chapter contains information on setting up the ModelSim or Cadence NC-VHDL simulator to simulate Microsemi SoC designs.

This chapter includes software requirements, steps describing how to compile Microsemi SoC FPGA libraries, and other setup information for the simulation tool you use.

Software Requirements

The information in this guide applies to the Microsemi Libero SoC Software v10.0 and above and IEEE-1076-compliant VHDL simulators. Additionally, this guide contains information about using ModelSim simulators.

For specific information about which versions this release supports, go to the technical support system on the Microsemi web site (**http://www.actel.com/custsup/search.html**) and search the keyword **third party**.

ModelSim

Since the installation path varies for each user and each installation, this document uses $ALSDIR to indicate the location where the software is installed. If you are a Unix user, simply create an environment variable called ALSDIR and set its value to the installation path. If you are a Windows user, replace $ALSDIR with the installation path in the commands.

Use the following procedure to compile libraries for the ModelSim simulators. Type UNIX commands at the UNIX prompt. Type Windows commands on the command line of the ModelSim Transcript window. The commands below are for Windows. To make the commands work for UNIX, use forward slashes instead of back slashes.

This procedure compiles a Microsemi VITAL library in the $**ALSDIR\lib\vtl\95\mti** directory. You must compile the FPGA library models for the VITAL libraries to work properly.

Note: If there is already an MTI directory in the $ALSDIR\lib\vtl\95 directory, compiled libraries may be present, and you may not need to perform the following procedure.

1. Create a library called **mti** in the **$ALSDIR\lib\vtl\95** directory.
2. Invoke the ModelSim simulator (Windows only).
3. Change to the $ALSDIR\lib\vtl\95\mti directory. Enter the following command at the prompt:

   cd $ALSDIR\lib\vtl\95\mti
4. Create a <vhd_fam> family library. Enter the following command at the prompt:

   ```
   vlib <vhd_fam>
   ```
5. Map the VITAL libray to the <vhd_fam> directory. Enter the following command at the prompt:

   ```
   vmap <vhd_fam> $ALSDIR\lib\vtl\95\mti\<vhd_fam>
   ```
6. Compile your VITAL libraries.

   ```
   vcom -work <vhd_fam> ../<act_fam>.vhd
   ```

   For example, to compile the 40MX library for your simulator, type the following command:

   ```
   vcom -work a40mx ../40mx.vhd
   ```
7. (Optional) Compile the migration library. Only perform this step if you need to use the migration library. Type the following command at the prompt:

   ```
   vcom -work <vhd_fam> ../<act_fam>_mig.vhd
   ```

2 – Design Flow

This chapter describes the design flow for simulating designs with a VHDL VITAL-compliant simulation tool.

VHDL VITAL Design Flow

The VHDL VITAL design flow has four main steps:

1. Create Design
2. Implement Design
3. Programming
4. System Verification

The following sections detail these steps.

Create Design

During design creation/verification, a design is captured in an RTL-level (behavioral) VHDL source file. After capturing the design, you can perform a behavioral simulation of the VHDL file to verify that the VHDL code is correct. The code is then synthesized into a gate-level (structural) VHDL netlist. After synthesis, you can perform an optional pre-layout structural simulation of the design. Finally, an EDIF netlist is generated for use in Libero SoC and a VHDL structural post-layout netlist is generated for timing simulation in a VHDL VITAL-compliant simulator.

VHDL Source Entry

Enter your VHDL design source using a text editor or a context-sensitive HDL editor. Your VHDL design source can contain RTL-level constructs, as well as instantiations of structural elements, such as Libero SoC cores.

Behavioral Simulation

Perform a behavioral simulation of your design before synthesis. Behavioral simulation verifies the functionality of your VHDL code. Typically, you use zero delays and a standard VHDL test bench to drive simulation. Refer to the documentation included with your simulation tool for information about performing functional simulation.

Synthesis

After you have created your behavioral VHDL design source, you must synthesize it. Synthesis transforms the behavioral VHDL file into a gate-level netlist and optimizes the design for a target technology. The documentation included with your synthesis tool contains information about performing design synthesis.

EDIF Netlist Generation

After you have created, synthesized, and verified your design, software generates an EDIF netlist for place-and-route in Libero SoC.

This EDIF netlist is also used to generate a structural VHDL netlist for use in structural simulation.

Structural VHDL Netlist Generation

Libero SoC generates a gate-level VHDL netlist from your EDIF netlist for use in post-synthesis pre-layout structural simulation. The file is available in the /synthesis directory if you wish to perform simulation manually.

Structural Simulation

Perform a structural simulation before placing-and-routing. Structural simulation verifies the functionality of your post-synthesis pre-layout structural VHDL netlist. Unit delays included in the compiled Libero

SoC VITAL libraries are used. Refer to the documentation included with your simulation tool for information about performing structural simulation.

Implement Design

During design implementation, you place-and-route a design using Libero SoC. Additionally, you may perform timing analysis. After place-and-route, perform post layout (timing) simulation with a VHDL VITAL-compliant simulator.

Programming

Program a device with programming software and hardware from Microsemi SoC or a supported third-party programming system. Refer to the programmer online help for information about programming a Microsemi SoC device.

System Verification

You can perform system verification on a programmed device using the Silicon Explorer diagnostic tool. Refer to the *Silicon Explorer Quick Start* for information about using the Silicon Explorer.

3 – Generating Netlists

This chapter describes the procedures for generating EDIF and structural VHDL netlists.

Generating an EDIF Netlist

After capturing your schematic or synthesizing your design, generate an EDIF netlist from your schematic capture or synthesis tool. Use the EDIF netlist for place-and-route. Refer to the documentation included with your schematic capture or synthesis tool for information about generating an EDIF netlist.

Generating a Structural VHDL Netlist

Structural VHDL netlist files are generated automatically as part of your Libero SoC project.

You can find your VHDL netlist files in the /synthesis directory of your Libero project. For example, if your project directory is named project1, then your netlist files are in /project1/synthesis.

Some families enable you to export these files manually for use in external tools. If your device supports this feature you can export netlist files from **Tools > Export > Netlist**.

4 – Simulation with ModelSim

This chapter describes steps to perform behavioral, structural and timing simulation using the ModelSim simulator.

The procedures shown are for PC. The same setup procedures work similarly for UNIX. Use forward slashes in place of back slashes. For PC, type commands into the MTI window. For UNIX, type commands into a UNIX window.

Behavioral Simulation

Use the following procedure to perform a behavioral simulation of a design. Refer to the documentation included with your simulation tool for additional information about performing behavioral simulation.

1. Invoke your ModelSim simulator. (PC only)
2. Change directory to your project directory. This directory must include your VHDL design files and testbench. Type:

   ```
   cd <project_dir>
   ```
3. Map to the Library. If any cores are instantiated in your VHDL source, type the following command to map them to the compiled VITAL library:

   vmap <vhd_fam> $ALSDIR\lib\vtl\95\mti\<vhd_fam>

   To reference the family library in your VHDL design files, add the following lines to your VHDL design files:

   ```
   library <vhd_fam>;
   use <vhd_fam>.components.all;
   ```
4. Create a “work” directory. Type:

   ```
   vlib work
   ```
5. Map to the “work” directory. Type the following command:

   ```
   vmap work .\work
   ```
6. Perform a behavioral simulation of your design. To perform a behavioral simulation using your V-System or ModelSim simulator, compile your VHDL design and testbench files and run a simulation. For hierarchical designs, compile the lower-level design blocks before the higher level design blocks.

The following commands demonstrate how to compile VHDL design and testbench files:

```
vcom -93 <behavioral>.vhd
vcom -93 <test_bench>.vhd
```

To simulate the design, type:

```
vsim <configuration_name>
```

For example:

```
vsim test_adder_behave
```

The entity-architecture pair specified by the configuration named test_adder_behave in the testbench will be simulated. If your design contains a PLL core, use a 1ps resolution:

```
vsim -t ps <configuration_name>
```

For example:

```
vsim -t ps test_adder_behave
```

Structural Simulation

Use the following procedure to perform structural simulation.

1. Generate a structural VHDL netlist. If you are using Synopsys Design Compiler, generate a structural VHDL netlist using this tool.

   If you are using other synthesis tools, generate a gate-level VHDL from your EDIF netlist by using the file generated automatically in your project. Some design families enable you to generate the files directly from the **Tools > Export > Netlist** menu.

Note: The generated VHDL uses std_logic for all ports. The bus ports will be in the same bit order as they appear in the EDIF netlist.

2. Map to the VITAL library. Run the following command to map the compiled VITAL library.

   ```
   vmap <vhd_fam> $ALSDIR\lib\vtl\95\mti\<vhd_fam>
   ```

3. Compile the structural netlist. Compile your VHDL design and testbench files. The following commands demonstrate how to compile VHDL design and testbench files:

   ```
   vcom -just e -93 <structural>.vhd
   vcom -just a -93 <structural>.vhd
   vcom <test_bench>.vhd
   ```

Note: First, the application compiles the entities. Then, it compiles the architectures, as required for VHDL netlists written by some tools.

4. Run the structural simulation. To simulate your design, type:

   ```
   vsim <configuration_name>
   ```

   For example:

   ```
   vsim test_adder_structure
   ```

   The entity-architecture pair specified by the configuration named test_adder_structure in the testbench will be simulated.

   If your design contains a PLL core, use a 1ps resolution:

   ```
   vsim -t ps <configuration_name>
   ```

   For example:

   ```
   vsim -t ps test_adder_structure
   ```

Timing Simulation

To perform timing simulation:

1. If you have not done so, back-annotate your design and create your testbench.
2. To perform a timing simulation using your V-System or ModelSim simulator, compile your VHDL design and testbench files, if they have not already been compiled for a structural simulation, and run a simulation. The following commands demonstrate how to compile VHDL design and testbench files:

   ```
   vcom -just e -93 <structural>.vhd
   vcom -just a -93 <structural>.vhd
   vcom <test_bench>.vhd
   ```

Note: Performing the previous steps compiles the entities first and then the architectures, as required for VHDL netlists written by some tools.

3. Run the back-annotation simulation using the timing information in the SDF file. Type:

   ```
   vsim -sdf[max|typ|min] /<region>=<design name>.sdf -c
   <configuration_name>
   ```

   The <region> option specifies the region (or path) to an instance in a design where back annotation begins. You can use it to specify a particular FPGA instance in a larger system design or testbench that you wish to back annotate. For example:

   ```
   vsim -sdfmax /uut=adder.sdf -c test_adder_structural
   ```

   In this example, the entity adder has been instantiated as instance “uut” in the testbench. The entity-architecture pair specified by the configuration named “test_adder_structural” in the testbench will be simulated using the maximum delays specified in the SDF file.

   If your design contains a PLL core, use a 1ps resolution:

   ```
   vsim -t ps -sdf[max|typ|min] /<region>=<design name>.sdf -c
   <configuration_name>
   ```

   For example:

   ```
   vsim -t ps -sdfmax /uut=adder.sdf -c test_adder_structural
   ```

A – 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

Visit the Customer Support website (www.microsemi.com/soc/support/search/default.aspx) for more information and support. Many answers available on the searchable web resource include diagrams, illustrations, and links to other resources on the website.

Website

You can browse a variety of technical and non-technical information on the SoC home page, at www.microsemi.com/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. Sales office listings can be found at www.microsemi.com/soc/company/contact/default.aspx.

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_itar@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 (949) 380-6100
Sales: +1 (949) 380-6136
Fax: +1 (949) 215-4996

Microsemi Corporation (NASDAQ: MSCC) offers a comprehensive portfolio of semiconductor solutions for: aerospace, defense and security; enterprise and communications; and industrial and alternative energy markets. Products include high-performance, high-reliability analog and RF devices, mixed signal and RF integrated circuits, customizable SoCs, FPGAs, and complete subsystems. Microsemi is headquartered in Aliso Viejo, Calif. Learn more at **www.microsemi.com**.

5-57-9006-12/11.12