ProASIC

Interface Guide

PRE-PRODUCTION
For the latest version of this document, check the User area on Actel's website at:

http://www.actel.com/user

Windows and UNIX Environments

Actel Corporation, Sunnyvale, CA 94086

Printed in the United States of America

Part Number: 5579014-0

Release: June 1999

Trademarks

This Interface Guide replaces all earlier editions of similar product series.

Table of Contents

Introduction

Actel's ASICmaster™, along with a logic synthesis tool with simulation/timing analysis capability, provides a complete top-down design environment for using Actel's A500K ProASIC™ devices.

This guide provides information about targeting Actel ProASIC technology in your preferred CAE Environment. All Actel ProASIC supported CAE environments are explained in this guide.

Document Organization

The *ProASIC Interface Guide* is divided into the following chapters:

Chapter 1 - Design Flow® contains information and procedures regarding a general design flow integrating Actel OEM tools with third-party software.

Chapter 2 - Simulation Using Cadence VerilogXL® contains a list of related publications, requirements for using the design kit, details about the location and content of the ProASIC A500K Verilog library, and other information and procedures

Chapter 3 - Verilog Simulation Using Model Technology contains information and procedures about setting up and performing functional and timing simulation with the Model Technology V-System or ModelSim simulator in conjunction with the Actel A500K Verilog library in both PC and UNIX environments.

Chapter 4 - VHDL Simulation Using Model Technology contains information and procedures about setting up and performing functional (behavioral and structural) and timing simulation with the Model Technology V-System or ModelSim simulator in conjunction with the Actel A500K VHDL VITAL library in both PC and UNIX environments.

Chapter 5 - VHDL Simulation Using Synopsys VSS® contains information and procedures setting up and performing functional (behavioral and structural) and timing simulation with the Synopsys VSS simulator in conjunction with the Actel A500K VHDL VITAL libraries.

Chapter 6 - Synthesis Using Exemplar Spectrum® contains information and procedures for using the Exemplar Spectrum® system and the Actel-Exemplar libraries to produce optimal results on Actel ProASIC devices.

Chapter 7 - Synthesis Using Synopsys Design Compiler® contains information and procedures for using Synopsys Design Compiler® and the Actel-Synopsys library to produce optimal results with Actel's ASICmaster.

Document Assumptions

This documentation makes the following assumptions:

1. **You have installed ASICmaster and MEMORYmaster.** This design toolset generates netlists for memories to be used with the A500K family devices. It also accepts netlists and constraint files and maps the design onto one of Actel's ProASIC devices.
2. **You have installed the synthesis/simulation software with appropriate licenses.**
3. **You are familiar with the principles of ASIC or FPGA design.**
4. **You have experience with high-level design methodologies and synthesis as well as manual (text entry) design definition.**
5. **You are experienced in digital simulation.**
6. **You are experienced in static timing analysis.**

Document Conventions

The following conventions are used throughout this document.

Table 1: Document Conventions

Convention	Use
Italic type	Is used in text, syntax, and examples to identify placeholders that you need to replace with specific words or values. It is also used selectively for emphasis.
`Courier` typeface	Is used in examples to represent text displayed on your screen.
`Courier bold` typeface	Is used in examples to show data that you should enter. It is also used selectively for emphasis.
< > (angle brackets)	Identify a function key or a key combination that you press, such as <Return> or <Ctrl+C>. (Ctrl refers to the Control key.)
{ } (braces)	Encloses lists of required items.
[ ] (square brackets)	Encloses lists of optional items.
| (vertical bar)	Separates items in a list from which one item must be selected, if the list is enclosed in { } or [ ].

Actel ProASIC Manuals

Actel ProASIC publications support Actel ProASIC software and devices. All manuals are listed below by device series. You can find these publications on-line or on the ASICmaster CD.

ASICmaster Installation and Licencing Guide provides information and procedures for installing and Licencing the ASICmaster software.

ASICmaster User's Guide provides information about the design flow for creating designs for ProASIC device. It inlcudes information and

procedures for placing and routing designs and also information on using timing constraints.

MEMORYmaster User's Guide provides information and procedures for generating embedded and distributed memories and instantiating then into a design.

ProASIC Macro Library Guide provides descriptions of library elements for Actel ProASIC device families. Functional descriptions, truth tables and symbols are included.

ProASIC Interface Guide provides information and procedures for designing Actel ProASIC devices in Exemplar synthesis, Synopsys synthesis, Verilog simulation, and VHDL simulation environments.

1

Design Flow®

This chapter illustrates and describes the design flow for creating ProASIC designs using the ASICmaster software.

ProASIC Design Flow Illustrated

Figure 3-2 shows an example of a design flow for creating an ProASIC device using the ASICmaster software.

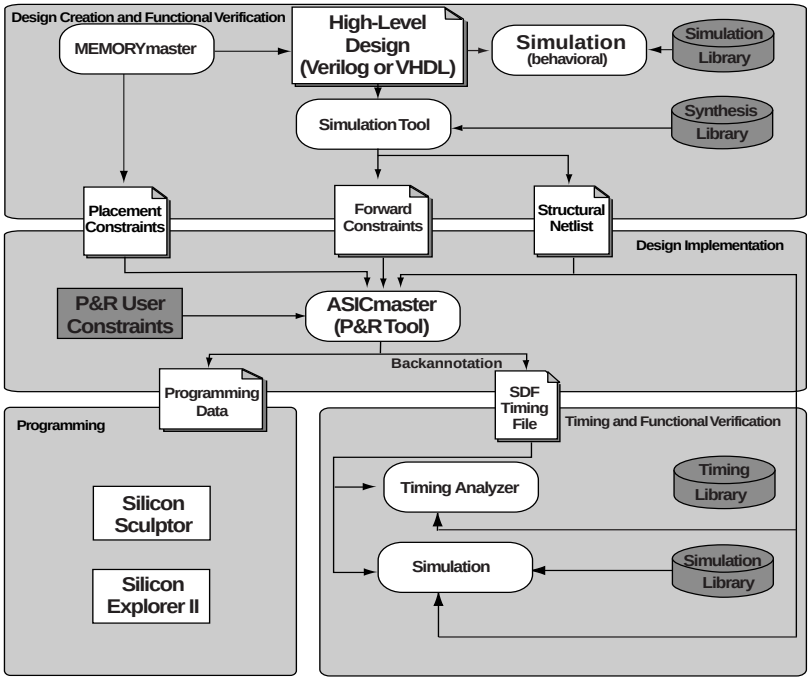

Figure 1-1. ProASIC Design Flow

ProASIC Design Flow Overview

The Actel ProASIC design flow has four main steps; design creation/ verification, design implementation, programming, and system verification. These steps are described in the following sections.

Design Creation/ Verification

During design creation/verification, a design is captured in an RTL-level (behavioral) HDL source file. After capturing the design, a behavioral simulation of the HDL file can be performed with VHDL or Verilog simulator software to verify that the HDL code is correct. The code is then synthesized into a netlist by the synthesis software. After synthesis, a post synthesis simulation of the design can be performed. Finally, the netlist is imported in ASICmaster and an SDF back annotation timing file is generated for the netlist for timing simulation in the VHDL or Verilog simulator.

HDL Design Source Entry

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

Behavioral Simulation

Perform a behavioral simulation of your design before synthesis. Behavioral simulation verifies the functionality of your HDL code. Typically, unit delays are used and a standard HDL test bench can be used to drive simulation.

Synthesis

After you have created your behavioral HDL source file, you must synthesize it before placing and routing it in ASICmaster. Synthesis transforms the behavioral HDL file into a gate-level (structural) netlist and optimizes the design for a target technology.

Structural Simulation

Perform a structural simulation of your design before placing and routing it. Structural simulation verifies the functionality of your post-synthesis structural HDL netlist. Unit delays included in the compiled Actel HDL libraries are used for every gate.

Design Implementation

During design implementation, a design is placed and routed using ASICmaster. After place and route, post-layout (timing) simulation is performed with a VHDL or Verilog simulator software tool.

Place and Route

Use ASICmaster to place and route your design. Define the netlist and constraints.

Static Timing Analysis

Use ASICmaster to generate an SDF back annotation timing file. This file can be read into a static timing analyser after the netlist has been read and will impose the correct timing on the design.

Timing Simulation

Perform a timing simulation of your design after placing and routing it. in ASICmaster. Timing simulation requires information extracted by ASICmaster and made available to the simulation tool in the form of an SDF back annotation timing file.

Programming

Program a device with programming software and hardware from Actel.

System Verification

You can perform system using the Actel Silicon Explorer diagnostic tool. Refer to the *Activator and APS Programming System Installation and User's Guide* or *Silicon Explorer QuickStart* for information about using the Silicon Explorer.

2 Simulation Using Cadence VerilogXL®

The ProASIC™ design kit for a Verilog Simulation Environment consists of a library of Verilog simulation models which describes the ProASIC A500K Family primitive elements and associated timing data. The library allows the simulation of a design, described as a structural Verilog netlist, using VerilogXL simulator.

To facilitate the use of the A500K verilog library with the VerilogXL simulator on UNIX, this chapter presents a list of related publications, requirements for using the design kit, and details about the location and content of the ProASIC A500K Verilog library.

Related Publications

To supplement the information in this document, refer to these additional documents:

- OVI; Standard Delay Format Specification, V2.1
- OVI; VerilogHDL Language Reference Manual, V2.0
- Cadence OpenBook Online Manual
- ASICmaster User's Guide
- MEMORYmaster User's Guide
- ProASIC A500K Data Sheet

Requirements

The following software is required for the design environment that includes the VerilogXL simulator and the Actel ASICmaster:

- Current licenses and software for the VerilogXL (version 2.6 or later) system that complies with OVI V1.0.
- Current ProASIC Verilog simulation library.

Location of the A500K Verilog Library

Files and subdirectories specific to VerilogXL are located as follows:

UNIX

```
$AMHOME/etc/deskits/verilog
```

Where $AMHOME is the UNIX environment variable set to the full pathname of the installation directory of Actel's ASICmaster software and design kits. For details on UNIX environment variables, refer to the *ASICmaster User's Guide*.

PC

```
%AMHOME%\etc\deskits\verilog
```

Where %AMHOME% is the Windows NT environment variable set to the full pathname of the installation directory of the ASICmaster software and design kits. For details on Windows NT environment variables see the *ASICmaster User's Guide*.

CAUTION: Because the library file is a text file, it can be edited by users who have appropriate access privileges. Actel strongly cautions against editing the library files provided. Changes will affect the accuracy and correlation of the model and its timing with the actual performance of Actel ProASIC devices.:

Table 2-1. Verilog Library Directory

Name	Description
lib	Subdirectory that contains the verilog library, named A500K.v

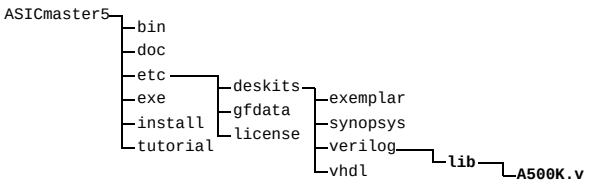

Figure 2-1. A500K Verilog Libraries

Behavioral Simulation

The VerilogXL simulator is started by using the "verilog" command. There are numerous options for this powerful simulator. A normal invocation includes the files to be simulated. To perform a behavioral simulation of an RTL design use the following command:

```
verilog source/adder16.v source/fsm.v \
source/core.v source/test_bench.v
```

Note: If any Actel A500K macros (memories) are instantiated in your source, run the simulator with the "+notimingchecks" and "+neg_tchk" options. The latter must be used with the A500K libraries, because of negative setup times in the memory modules.

Structural Simulation

If a netlist is to be simulated, the cell library has to be added to the simulator call. To do so, the -v option is used, as show below:

```
verilog netlist/core_and_pads.v \
-v /tools/ProASIC/etc/deskits/verilog/lib/A500K \
+neg_tchk
```

Note: This option must be used with the A500K libraries, because of negative setup times in the memory modules.

Timing Simulation

Simulation of a netlist can be performed with the timing back annotation SDF file generated by ASICmaster. The inclusion of this file can be done on the command line or in a test bench file. If done on the command line, invoke the simulator as follows:

```
verilog netlist/core_and_pads.v +sdf_filecore_and_pads.sdf \
-v /tools/ProASIC/etc/deskits/verilog/lib/A500K \
+neg_tchk
```

Note: There is no space between the option +sdf_file and the file name.

If the test bench option is chosen, add the following line to the test bench and run the simulation with the same options as the structural simulation:

```
initial $sdf_annotate("core.sdf",U1,,,"MAXIMUM");
```

VerilogXL Simulation Options

For useful comman specifications and directives, please refer to the VerilogXL online help.

Results

Check the output of VerilogXL in the shell or the log file for any timing violations or other warnings and errors. Make sure that warnings are not applicable to the design's functionality. Functional results can be reviewed by using the waveform viewer SimWave or by using strobe files created by the $fmonitor system task. Strobe files are very useful to compare results from RTL level simulation, timing simulation, and timing back annotation simulation. To identify differences, use the "diff" command to compare files.

3
Verilog Simulation Using Model Technology

This chapter contains information about setting up and performing functional and timing simulation with the Model Technology V-System or ModelSim simulator in conjunction with the Actel A500K Verilog library in both PC and UNIX environments.

The ProASIC design kit for the Verilog simulation consists of three library files that describe the ProASIC primitive elements in Verilog with the associated timing data. The library allows the simulation of a design, described as a structural Verilog netlist, using the Model Technology Verilog Simulator.

Related Publications

To supplement the information in this document, refer to these additional documents:

- *OVI Standard Delay Format Specification, V2.1*
- *OVI VerilogHDL Language Reference Manual, V2.0*
- *ModelSim Reference Manual*
- *ASICmaster User's Guide*
- *MEMORYmaster User's Guide*

Requirements

The following hardware, software, and licenses are required for the design environment that includes the Model Technology V-System or ModelSim simulator and Actel's A500K ProASIC family.

- Current licenses and software for the Model Technology V-System (version 5.1 or later) or ModelSim (version 5.1 or later) simulator.
- Current ProASIC Verilog Simulation library.

Location of the A500K Verilog library

The following Verilog-specific files and subdirectory are located as follows:

UNIX

```
$AMHOME/etc/deskits/verilog
```

Where $AMHOME is the UNIX environment variable set to the full pathname of the installation directory of the ASICmaster software and design kits. For details on UNIX environment variables see the *ASICmaster User's Guide*.

PC

```
%AMHOME%\etc\deskits\verilog
```

Where %AMHOME% is the Windows NT environment variable set to the full pathname of the installation directory of Actel's ASICmaster software and design kits. For details on Windows NT environment variables see the *ASICmaster User's Guide*.

CAUTION: **Because the library files are text files, they can be edited by users who have appropriate access privileges. Actel strongly cautions against editing the library files provided. Changes will affect the accuracy and correlation of the model and its timing with the actual performance of Actel ProASIC devices.**

On either PC or UNIX, the lib subdirectory is available.

Table 3-1. Verilog Libraries

Name	Description
lib	Subdirectory that contains: • the verilog library, named A500K.v

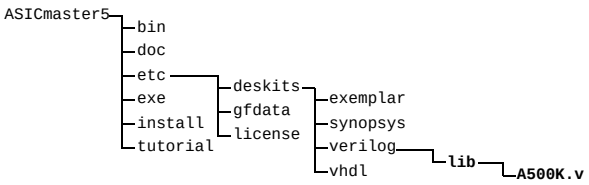

Figure 3-1. Where to find the A500K Verilog libraries

Setup

Before using the Actel A500K Verilog models with the Verilog simulator, compile the Actel A500K libraries. The following procedures describe how to compile libraries for the Model Technology V-System or ModelSim simulator (UNIX and PC).

Note: The procedure shown is for PC. The same setup procedures work similarly for UNIX. Use forward slashes in place of back slashes. For PC, commands are typed into the MTI window. On UNIX, commands are typed at the UNIX command line.

To compile the Actel A500K VITAL libraries:

1. **Create a directory called “mti”.**
2. **Invoke the V-System simulator (PC only).**
3. **Change to the newly created “mti” directory.**
4. **Create an A500K family library directory for the simulator.**
   Type the following command at the prompt:

   ```
   vlib A500K
   ```

5. **Map the Actel A500K Verilog library to the A500K directory.**
   Type the following command at the prompt:

```
vmap A500K <your_path>\mti\A500K
```

6. **Compile the Actel A500K Verilog library with the command below.**

```
vlog -work A500K %AMHOME%\etc\verilog\lib\A500K.v
```

Behavioral Simulation

Once the Verilog descriptions of the logic blocks are created, test and debug the design using the MTI simulator. For information regarding MTI usage, refer to the *Model Technology V-System* or *ModelSim User's Manual.*

To perform behavioral simulation.

1. **(PC only) Invoke the V-System or ModelSim simulator.**
2. **Change directory to the project directory.** This directory must include the Verilog design files and testbench. Type:

```
cd <project_dir>
```

3. **Create a "work" directory.** Type:

```
vlib work
```

4. **Compile the design.** Compile the Verilog design and testbench files. For hierarchical designs, compile the lower level design blocks before the higher level design blocks. The following commands demonstrate how to compile Verilog design and testbench files:

```
vlog <behavioral>.v
vlog <test_bench>.v
```

5. **Simulate the design.** To perform a behavioral simulation using the V-System or ModelSim Simulator, type:

```
vsim <topmost_module_name>
```

For example:

```
vsim test_adder_behave
```

The module test_adder_behave in the testbench will be simulated.

If any Actel A500K macros (memories) are instantiated in the Verilog source, use the following command to simulate the design with the pre-compiled Actel Verilog library.

```
vsim -L <your_path>\mti\A500K <topmost_module_name>
```

Structural Simulation

Use the following procedure to perform structural simulation. After generating a structural Verilog netlist with the synthesis tool,

1. **Compile the structural netlist.** Compile the Verilog netlist and testbench files. The following commands compile Verilog netlist and the associated testbench files:

   ```
   vlog <structural netlist>.v
   vlog <test_bench>.v
   ```

2. **Run the structural simulation.** To simulate the design, type:

   ```
   vsim -L <your_path>\mti\A500K <topmost_module_name>
   ```

   For example:

   ```
   vsim -L am5p1\mti\A500K test_adder_structural
   ```

   The module test_adder_structure in the testbench will be simulated using the pre-compiled A500K Verilog library.

Timing Simulation

Use the following procedure to perform timing simulation.

1. **Place and route the design with ASICmaster.** Refer to the *ASICmaster User's Guide* for information about using ASICmaster.

2. **Extract timing information for the design by specifying a timing file name in the Timing Annotaion tab in the ASICmaster Design Executive window.**

3. **Compile the netlist.** Compile the Verilog netlist and testbench files, if they have not already been compiled for a structural simulation. The following commands compile Verilog netlist and associated testbench files:

```
vlog <structural>.v
vlog <test_bench>.v
```

4. **Simulate the design using the timing information in the SDF file.** Type:

```
vsim -L am5p1\mti\A500K -sdf[max|typ|min] /<region>=<design
name>.sdf -c <topmost_module_name>
```

The <region> option specifies the region (or path) to an instance in a design where sdf backannotation timing data was created. For example:

```
vsim -L am5p1\mti\A500K -sdfmax /uut=adder.sdf -c
test_adder_structural
```

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

Note: If embedded memories are used within the design, the meminit.dat file from the library directory has to be copied to the directory the simulator is run from. This file contains the information of the memory cells at start-up. It is a simple text file so the user can change the contents to what ever start value is needed. The default file sets all the memory cells to ‘0’.

4

VHDL Simulation Using Model Technology

This chapter contains information about setting up and performing functional (behavioral and structural) and timing simulation with the Model Technology V-System or ModelSim simulator in conjunction with the Actel A500K VHDL VITAL library in both PC and UNIX environments.

The ProASIC design kit for VHDL VITAL simulation consists of three library files that describes the ProASIC primitive elements in terms of VHDL VITAL simulation models. as well as associated timing data. The library allows the simulation of a design, described as a structural VHDL netlist, using a VITAL compliant VHDL simulator.

Related Publications

To supplement the information in this document, refer to these additional documents:

- *IEEE Standard VHDL Language Reference Manual*, IEEE Std 1076-1987
- *IEEE Standard Multivalue Logic System for VHDL Model Interoperability*, IEEE Std 1164
- *IEEE VITAL Standard ASIC Modeling Specification*, IEEE Std 1076.4
- *ModelSim Reference Manual*
- *ASICmaster User's Guide*
- *MEMORYmaster User's Guide*

Requirements

The following hardware, software, and licenses are required for the design environment that includes the Model Technology V-System or ModelSim simulator and Actel's A500K ProASIC family.

- Current licenses and software for the Model Technology V-System (version 5.1 or later) or ModelSim (version 5.1 or later) simulator.
- Current ProASIC VHDL VITAL simlation libraries.

Location of the A500K VHDL VITAL library

The following VHDL VITAL-specific files and subdirectory are located as follows:

UNIX

```
$AMHOME/etc/deskits/vhdl
```

Where $AMHOME is the UNIX environment variable set to the full pathname of the installation directory of the ASICmaster software and design kits. For details on UNIX environment variables see the *ASICmaster User's Guide*.

PC

```
%AMHOME%\etc\deskits\vhdl
```

Where %AMHOME% is the Windows NT environment variable set to the full pathname of the installation directory of Actel's ASICmaster software and design kits. For details on Windows NT environment variables see the *ASICmaster User's Guide*.

CAUTION: **Because the library files are text files, they can be edited by users who have appropriate access privileges. Actel strongly cautions against editing the library files provided. Changes will affect the accuracy and correlation of the model and its timing with the actual performance of Actel ProASIC devices.**

On either PC or UNIX the following subdirectories are available:

Table 4-1: Model Technology Library Directory and Subdirectories

Name	Description
lib	Subdirectories containing: • The VDHL library file, named A500K_Vtable.vhd • The VDHL library file, named A500K_Vcomponents.vhd • The VDHL library file, named A500K_VITAL.vhd • The memory configuration fil, named meminit.dat

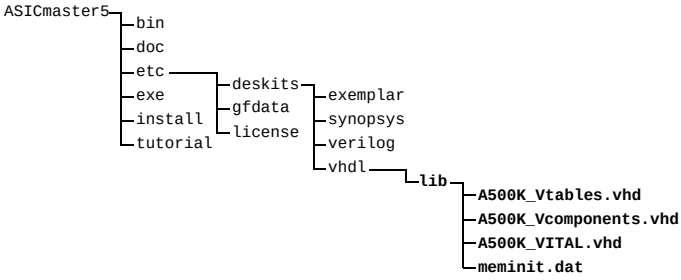

Figure 4-1: A500K Library Location

Setup

Before using the Actel A500K VHDL VITAL models with the VHDL simulator, you must compile the Actel A500K libraries. The following procedures describe how to compile libraries for the Model Technology V-System or ModelSim simulator (UNIX and PC).

Note: The procedure shown is for PC. The same setup procedures work similarly for UNIX. Use forward slashes in place of back slashes. For PC, commands are typed into the MTI window. On UNIX, commands are typed at the UNIX command line.

To compile the Actel A500K VITAL libraries:

1. **Create a directory called “mti”.**
2. **Invoke the V-System simulator (PC only).**
3. **Change to the newly created “mti” directory.**
4. **Create a A500K family library directory for the simulator.**
   Type the following command at the prompt:

   ```
   vlib A500K
   ```

5. **Map the Actel A500K VITAL library to the A500K directory.**
   Type the following command at the prompt:

```
vmap A500K <your_path>\mti\A500K
```

6. **The library should be compiled in the order listed in the commands below.**

```
vcom -work A500K -explicit
       %AMHOME%\etc\vhdl\lib\A500K_Vtables.vhd

vcom -work A500K -explicit
       %AMHOME%\etc\vhdl\lib\A500K_Vcomponents.vhd

vcom -work A500K -explicit
       %AMHOME%\etc\vhdl\lib\A500K_VITAL.vhd
```

Behavioral Simulation

Once the VHDL descriptions of the logic blocks are created, test and debug the design using the MTI simulator. For information regarding MTI usage, refer to the *Model Technology V-System* or *ModelSim User's Manual*.

To perform behavioral simulation.

1. **(PC only) Invoke the V-System or ModelSim simulator.**
2. **Change directory to the project directory.** This directory must include the VHDL design files and testbench. Type:

```
cd <project_dir>
```

3. **Map to the Actel Library.** If any Actel A500K macros (memories) are instantiated in the VHDL source, run the following command to map them to the compiled Actel VITAL library.

```
vmap A500K <your_path>\mti\A500K
```

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

```
library A500K;
```

```
use A500K.all;
```

4. **Create a "work" directory.** Type:

   ```
   vlib work
   ```

5. **Map to the "work" directory.** Type:

   ```
   vmap work .\work
   ```

6. **Compile the design.** Compile the VHDL design and testbench files. 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 <behavioral>.vhd
   vcom <test_bench>.vhd
   ```

7. **Simulate the design.** To perform a behavioral simulation using the V-System or ModelSim Simulator, 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.

Structural Simulation

Use the following procedure to perform structural simulation.

1. **Generate a structural VHDL netlist with the synthesis tool.**
2. **Map to the Actel VITAL library.** Type:

   ```
   vmap A500K <your_path>\mti\A500K
   ```

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

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

   Note: The previous steps compile the entities first (-just e), and then the architectures (-just a), as required for VHDL netlists written by some tools.

4. **Run the structural simulation.** To simulate the 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.

Timing Simulation

Use the following procedure to perform timing simulation.

1. **Place and route the design with ASICmaster.** Refer to the *ASICmaster User's Guide* for information about using ASICmaster.
2. **Extract timing information for the design by specifying a timing file name in the Timing Annotaion tab in the Design Executive window and by checking the Enable box over the input field.**
3. **Compile the netlist.** Compile the VHDL design and testbench files, if they have not already been compiled for a structural simulation. The following commands demonstrate how to compile VHDL design and testbench files:

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

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

4. **Simulate the design 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 sdf backannotation timing data was created. 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.

Note: If embedded memories are used within the design, the meminit.dat file from the library directory has to be copied to the directory the simulator is run from. This file contains the information of the memory cells at start-up. It is a simple text file so the user can change the contents to what ever start value is needed. The default file sets all the memory cells to ‘0’.

5

VHDL Simulation Using Synopsys VSS®

The ProASIC design kit for VHDL VITAL simulation consists of three library files that describe the ProASIC primitive elements in terms of VHDL VITAL simulation models as well as associated timing data. The library allows simulation of the design, described as a structural VHDL netlist, using a VITAL compliant VHDL simulator.

This chapter contains information on setting up and performing functional (behavioral and structural) and timing simulation with the Synopsys VSS simulator in conjunction with the Actel A500K VHDL VITAL libraries. To facilitate the use of the A500K VHDL library with Synopsys VSS simulator on UNIX, this chapter also presents information regarding the titles of related publications, requirements for using the design kit, and details about the location and content of the ProASIC A500K VHDL VITAL library.

Related Publications

To supplement the information in this document, refer to: these:

- *IEEE Standard VHDL Language Reference Manual, IEEE Std 1076-1987*
- *IEEE Standard Multivalue Logic System for VHDL Model Interoperability, IEEE Std 1164*
- *IEEE VITAL Standard ASIC Modelling Specification, IEEE Std 1076.4*
- *Synopsys Online Documentation SOLD*
- *ASICmaster User's Guide*
- *MEMORYmaster User's Guide*

Requirements

The following hardware, software, and licenses are required for the design environment that includes the Synopsys VSS simulator and Actel's A500K ProASIC family.

- Current licenses and software for the Synopsys VSS (version 1999.05 or later) simulator.
- Current ProASIC VHDL VITAL simulation libraries.

Location of the A500K VHDL VITAL Library

The following VHDL VITAL-specific files and subdirectories are located as follows:

UNIX

```
$AMHOME/etc/deskits/vhdl
```

Where $AMHOME is the UNIX environment variable set to the full pathname of the installation directory of the ASICmaster software and design kits. For details on UNIX environment variables see the *Actel ASICmaster User's Guide*.

PC

```
%AMHOME%\etc\deskits\vhdl
```

Where %AMHOME% is the Windows NT environment variable set to the full pathname of the installation directory of Actel's ASICmaster software and design kits. For details on Windows NT environment variables see the *ASICmaster User's Guide*.

CAUTION: Because the library files are text files, which can be edited by users who have appropriate access privileges. Actel strongly cautions against editing the library files provided. Changes will affect the accuracy and correlation

of the model and its timing with the actual performance of Actel ProASIC devices.

Table 5-1: VHDL Library Directory

Name	Description
lib	Subdirectory that contains: • the VDHL library file, named A500K_Vtable.vhd • the VDHL library file, named A500K_Vcomponents.vhd • the VDHL library file, named A500K_VITAL.vhd

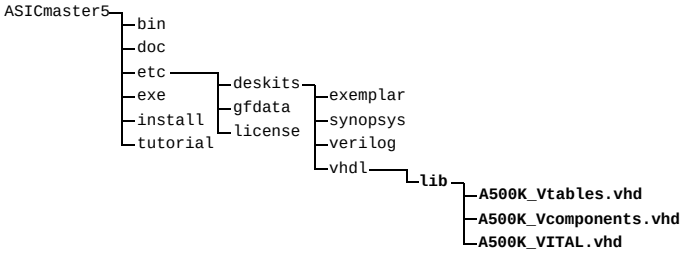

Figure 5-1: A500K VHDL Libraries

Setup

Before using the Actel A500K VHDL VITAL models with a VHDL simulator, compile the Actel A500K libraries. The following procedure describes how to compile libraries for the Synopsys VSS simulator (UNIX only):

1. **Create a directory for the compiled ProASIC libraries.**

   ```
   mkdir A500K
   ```

2. **Map the created directory to the VHDL library statement.** Add the following line to the .synopsys_vss.setup file:

   ```
   A500K : <path_to_the_A500K_directory>/A500K
   ```

3. **Compile the library in the order listed in the commands below:**

```
vhdlan -i -w A500K A500K_Vtables.vhd

vhdlan -i -w A500K A500K_Vcomponents.vhd

vhdlan -i -w A500K A500K_VITAL.vhd
```

Behavioral Simulation

Once the VHDL descriptions of the logic blocks are created, test and debug the design using the VSS simulator. Use the following procedure to perform the behavioral simulation.

1. **Change directory to the project directory.** This directory must include the VHDL design files and testbench. Type:

```
cd <project_dir>
```

2. **Map to the Actel library.** If any macros are instantiated in the VHDL source, look for the following line in the .synopsys_vss.setup file:

```
A500K : <path_to_the_A500K_directory>/A500K
```

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

```
library A500K;
use A500K.all;
```

3. **Create a "work" directory.** Type:

```
mkdir work
```

4. **Map to the "work" directory.** Add the following files to the .synopsys_vss.setup file if they have not already been added:

```
WORK > default

default : ./work
```

5. **Analyze the design.** Analyze the VHDL design and testbench files before 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:

```
vhdlan -i <behavioral>.vhd
vhdlan -i <test_bench>.vhd
```

6. **Simulate the design.** Type:

```
vhdlsim <configuration_name>
```

To simulate in interactive mode and a graphical user interface, type:

```
vhdldbx <configuration_name> &
```

For example:

```
vhdlsim test_adder_behave
```

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

Structural Simulation

Use the following procedure to perform structural simulation.

1. **Generate a structural VHDL netlist with the synthesis tool.**
2. **Map to the Actel Library.** Add the following line in the .synopsys_vss.setup file if it has not already been added:

   ```
   A500K : <path_to_the_A500K_directory>/A500K
   ```

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

   ```
   vhdlan -i <structural>.vhd
   vhdlan -i <test_bench>.vhd
   ```

   Make sure that the Actel family library is referenced in the VHDL netlist. If not, add the following lines to the VHDL netlist files:

   ```
   library A500K;
   use A500K.all;
   ```

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

   ```
   vhdlsim <configuration_name>
   ```

   For example:

   ```
   vhdlsim test_adder_structure
   ```

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

Timing Simulation

Use the following procedure to perform timing simulation.

1. **Place and route the design with ASICmaster.** Refer to the *ASICmaster User's Guide* for information about using ASICmaster.

2. **Extract timing information for the design.** Activate the timing extraction by specifying a timing file name in the Timing Annotaion tab in the Design Executive window and by checking the enable box over the input field.

3. **Analyze the structural netlist.** Analyze the VHDL design and testbench files, if they have not already been analyzed for a structural simulation. The following commands demonstrate how to analyze VHDL design and testbench files:

```
vhdlan -i <structural>.vhd
vhdlan -i <test_bench>.vhd
```

4. **Simulate the design using the timing information in the SDF file.** Type:

```
vhdlsim <configuration_name> -sdf_[max|typ|min] -sdf_top
<hier_region> -sdf <design name>.sdf
```

The <hier_region> option specifies the region (or path) to an instance in a design for which the sdf back-annotation file was generated for. For example:

```
vhdlsim alu_conf -sdf_max -sdf_top tb_alu/alu_inst -sdf
alu.sdf
```

In this example, the entity "alu" has been instantiated as instance "alu_inst" in the testbench. The entity-architecture pair specified by the configuration named "alu_conf" in the testbench will be simulated using the maximum delays specified in the SDF file.

6

Synthesis Using Exemplar Spectrum®

This chapter describes how to use the Exemplar Spectrum® system and the Actel A500K libraries to produce optimal results on Actel ProASIC devices. Additionally, it contains requirements for using the Actel's A500K libraries for Exemplar Spectrum, procedures for installation and encapsulation of the libraries and the setting of environment variables, and details about the location and content of Actel's A500K libraries for Exemplar Spectrum.

The ProASIC design kit for the Exemplar Spectrum Synthesis Environment consists of libraries that describe Actel's ProASIC primitive elements for synthesis and associated timing data for the A500K family of ProASIC devices.

Related Publications

To supplement the information in this chapter, refer to the:

- Spectrum User's Guide.
- Spectrum Command Reference Manual.
- *Spectrum Synthesis and Technology Guide.*
- Spectrum HDL Synthesis Guide.
- *OVI; Standard Delay Format Specification, V 2.1.*
- *OVI; Verilog HDL Language Reference Manual, V 2.0.*

Requirements

The following software is needed for Exemplar Spectrum synthesis system and Actel's ASICmaster design environment:

- Exemplar Spectrum (version 4.2.2 - level 3 or later).
- Current ProASIC Exemplar synthesis libraries.

Location of the A500K Exemplar Libraries

Files and subdirectories specific to Exemplar Spectrum are located as follows:

UNIX

```
$AMHOME/etc/deskits/exemplar
```

Where $AMHOME is the UNIX environment variable set to the full pathname of the installation directory of the ASICmaster software and design kits. To set this variable, type the following line on the command line or include it in your personal setup file:

```
setenv AMHOME <install_dir>
```

To check the setting of this variable, type:

```
echo $AMHOME
```

For details about UNIX environment variables see the *ASICmaster User's Guide*.

PC

```
%AMHOME%\etc\deskits\exemplar
```

Where %AMHOME% is the Windows NT environment variable set to the full pathname of the installation directory of Actel's ASICmaster software and design kits. To set this variable, open the "System Properties" window on your PC and click the "Environment" tab. In the "User Variables" subwindow you should find an entry for AMHOME. Click this variable to change the entry. For details on Windows NT environment variables see the *Actel ASICmaster User's Guide*.

Table 6-1: Exemplar Library Directory

Name	Description
lib	Subdirectory that contains the Actel synthesis library for Exemplar Spectrum, named A500K.syn.
actel.ini edit_actel_ini.csh encapsulate_proasic_spectrum_unix	Files used in automatic encapsulation of the A500K libraries into Spectrum. Please see "Automatic Encapsulation on UNIX" on page 34.

Table 6-1: Exemplar Library Directory (Continued)

Name	Description
actel_422.ini edit_ini_422.csh encapsulate_proasic_422_unix	Files used in automatic encapsulation of the A500K libraries into Leonardo 4.2.2. Please see "Automatic Encapsulation on UNIX" on page 34.

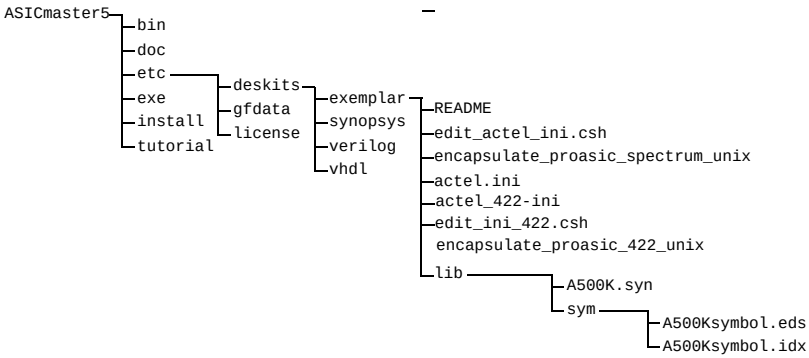

Figure 6-1: A500K Library Location

Setup

This section presents general setup guidelines for using Exemplar Spectrum with Actel's ProASIC devices.

Library Setup

This section describes how to perform the encapsulation operation. Exemplar's Spectrum (release version v1999.1) distribution CD contains the Actel ProASIC synthesis libraries. The interactive selection of this library is similar to the selection of other supported libraries. In a script based session the library can be selected using the "load_library A500K" command.

The A500K library for Exemplar Spectrum, found on the ASICmaster CD, must be encapsulated into the Exemplar Spectrum environment for

Exemplar Spectrum software versions earlier than v1999.1. For additional information, refer to the *Exemplar Spectrum Command Reference Manual*.

Library Encapsulation on UNIX

The $AMHOME/etc/deskit directory contains files to automate the process of encapsulating the Actel libraries into Exemplar Spectrum. After encapsulation, the Actel libraries appear in the Exemplar Spectrum menu system. Note that these files also function for Leonardo release 4.1.3.

Automatic Encapsulation on UNIX

The following three files are used in encapsulation and are located in the "$AMHOME/etc/deskits/exemplar/" directory:

```
actel.ini
edit_actel_ini.csh
encapsulate_proasic_spectrum _unix
actel_422.ini
edit_ini_422.csh
encapsulate_proasic_422_unix
```

To encapsulate the ProASIC libraries into the Exemplar Spectrum system, run the script from the Exemplar directory. Then type the command below:

```
encapsulate_proasic_spectrum_unix
```

Note: It is mandatory to run the script in the Exemplar directory in order to make the script work properly.

To encapsulate the ProASIC libraries into the Exemplar Leonardo 4.2.2 system, run the script from the Exemplar directory. Then type the command below:

```
encapsulate_proasic_422_unix
```

Manual Encapsulation on UNIX

If one of the tasks in the script is not permitted, the script displays instructions. Performing these changes requires permission to edit the Exemplar Spectrum files and to link files into the lib directory.

To perform manual encapsulation on UNIX:

1. **Link the A500K libraries to the library directories under the Exemplar Spectrum installation.** Type the following commands:

```
cd $EXEMPLAR/lib

ln -s $AMHOME/etc/deskits/exemplar/lib/A500K.syn

cd $EXEMPLAR/lib/sym

ln -s $AMHOME/etc/deskits/exemplar/lib/sym/A500Ksymbol.eds

ln -s $AMHOME/etc/deskits/exemplar/lib/sym/A500Ksymbol.idx
```

   Where $EXEMPLAR is the top level of the Exemplar Spectrum release and $AMHOME is the top level directory containing the Actel release.

2. **Add the following line to the devices.ini file in the lib directory:**

```
{"Actel", "A500K", "A500K", "none", "FPGA", "BOTH", "",
"Exemplar Logic", "http://www.actel.com"}
```

Library Encapsulation on Windows NT

Library encapsulation on Windows NT must be performed manually.

To perform manual encapsulation on Windows NT:

1. **Copy the A500K libraries to the library directories under the Exemplar Spectrum installation.** Type the following commands:

```
cd %EXEMPLAR%\lib

copy %AMHOME%\etc\deskits\exemplar\lib\A500K.syn

cd %EXEMPLAR%\lib\sym

copy %AMHOME%\etc\deskits\exemplar\lib\sym\A500Ksymbol.eds

copy %AMHOME%\etc\deskits\exemplar\lib\sym\A500Ksymbol.idx
```

Where %EXEMPLAR% is the top level of the Exemplar Spectrum release and %AMHOME% is the top level directory containing the Actel release.

2. **Add the following line to the devices.ini file in the lib directory:**

```
{"Actel", "A500K", "A500K", "none", "FPGA", "BOTH", "",
"Exemplar Logic", "http://www.actel.com"
```

Translating to ProASIC Libraries

The Exemplar synthesis system will:

- Translate designs cell-by-cell from the original technology, while attempting to preserve the functional structure of the design.
- Attempt to re-map any structures to the ProASIC library that are not functionally identical.

If a design has technology-dependent instances that are not specific to ProASIC, use the following procedure to translate the design to the Actel ProASIC library:

1. **Load the library used by the current netlist.**
2. **Read in the netlist.**
3. **Load the A500K library for Exemplar Spectrum.**
4. **Re-optimize the design using the ProASIC library.**

Synthesis Guidelines

The following sections present general guidelines for Exemplar Spectrum synthesis scripts for Actel ProASIC devices.

Clean Database

Before you work on a new design the database should be reset and the library and design should be read.

Note: If memories generated with MEMORYmaster are used, read these files as normal RTL description. Link the design together at the top

level. Before compiling set a “NOOPT” attribute on all these memories.

Design Environment

The wire tree model, process, and temperature for a design must be specified in synthesis to be coherent with the post-layout results.

Wire Tree Model

Actel’s A500K libraries for Exemplar Spectrum contain general numbers for a balanced wire tree model. Therefore, the settings for the default wire tree model should be “balanced”, as shown below:

```
set wire_tree balanced
```

Note: If you do not set the wire tree to balanced, the interconnect delay will be calculated wrong, unless you have previously back annotated timing into the design.

Process

No default process has been set in the current library The choices are “Best”, “Typical”, and “Worst”. To ensure that the timing numbers used are optimal for all devices with all the possible different process characteristics, set the process to “Worst”, as shown below:

```
set process worst
```

Temperature

Define the temperature in the synthesis tool to optimize timing:

```
set temp 70

Worst case commercial temperature is 70 degree Celsius.
```

Compilation Setup

During optimization Spectrum can include many tasks that otherwise would have to be done manually.

Pad Insertion

Insert pads by setting the chip variable to true:

```
set chip TRUE
```

To specify the pad type to be used on a certain input or output use the following attribute:

```
set_attribute .work.alu.arch reset -name PAD -value IB33 -port
```

Note: A netlist must include pads before you import it into ASICmaster. For descriptions of available I/O types, see the *ProASIC Macro Library Guide*.

Optimization Goal

Synthesis optimization can be done with emphasis on different criteria, such as area or timing. To make sure the optimization generates the needed results set these criteria with the following commands:

```
set area FALSE

set delay TRUE
```

Don't Optimize Memory Blocks

If the design uses memory blocks generated by MEMORYmaster, the "NOOPT" attribute must be set on these blocks to ensure that the automatic memory placement functions properly:

```
set_attribute TOP/RAM_INST1 -name NOOPT -value TRUE
```

Timing Constraints

When setting timing constraints, Actel recommends the following steps:

1. **Create the clocks for the design.**
2. **Before adding any other constraints, optimize once to get an initial idea of the design's performance.**
3. **To set timing or area constraints, type the following commands:**

```
set_max_fanout_load 8
set_attribute required_time 5 <all outputs>
set_attribute arrival_time 5 <all inputs>
set_multicycle_path ...
```

It is important to set rational constraints for the design.

4. **Re-run the optimization.**
5. **If the results are worse than in the initial compilation, relax the constraints and recompile the design.**

Design Optimization

Two optimization steps are usually performed during synthesis. One to map the design and one for timing optimization.

optimize

The optimize command is used to map the design to the target library. For example:

```
optimize -ta A500K .work.alu.INTERFACE -effort standard -chip
```

optimize_timing

The optimize_timing command is used to meet the given timing constraints. For example:

```
optimize_timing .work.alu.INTERFACE -force
```

Note: All memories generated by MEMORYmaster must have the "NOOPT" attribute set to true to ensure that the design flow with automatic memory placement is working properly.

Reports Generation

To become familiar with the gatelevel design, the report options of Exemplar Spectrum should be use extensively. For additional information, refer to the *Exemplar Spectrum Command Reference Manual*.

Forward Constraints

Exemplar Spectrum currently cannot write out timing constraints written in a format that is readable for ASICmaster.

Example Script

The following provides an example of Exemplar synthesis script.

```
################################################
```

```
##                                                ##
##   Setup Library and Work Directory             ##
##                                                ##
####################################################
clean_all
set_working_dir "./"
load_library A500K

####################################################
##                                                ##
##   Set Operating Conditions and Wire-Load       ##
##                                                ##
####################################################
set temp 70
set process worst
set wire_tree balanced

####################################################
##                                                ##
## Set Netlist Format, Analyze and Elaborate      ##
##                                                ##
####################################################
exemplar_format Verilog

set module alu
analyze -format verilog source/alu.v

elaborate alu

####################################################
##                                                ##
## Setup Compilation                              ##
##                                                ##
####################################################

present_design .work.alu.INTERFACE

set chip TRUE
set_attribute .work.alu.INTERFACE data_in* -name PAD
             -value IB33 -port

set area FALSE
set delay TRUE
set target A500K
set hierarchy_preserve TRUE

####################################################
##                                                ##
```

```
## Timing Constraints                          ##
##                                             ##
#################################################

set_clock -port -name .work.alu.INTERFACE.clk -clock_cycle
          33ns
set_attribute -port acc required_time 5
set_attribute -port data_in arrival_time 3
set_max_fanout_load 8

#################################################
##                                             ##
## Optimizations                               ##
##                                             ##
#################################################

optimize -ta A500K .work.alu.INTERFACE -effort standard -chip
         -delay
optimize_timing .work.alu.INTERFACE -force

#################################################
##                                             ##
## Write Reports and Netlist                   ##
##                                             ##
#################################################

set output_file “netlist/alu.v”

report_delay reports/alu.timing.rpt -num_paths 1 -longest_path
 -clock_frequency
report_area reports/alu.area.rpt

auto_write -format Verilog netlist/alu.v

#################################################
##                                             ##
## End of Script                               ##
##                                             ##
#################################################
```

Back Annotation Scripting

The following example script illustrates general guidelines for Exemplar Spectrum back annotation scripts for Actel's ProASIC devices. The basic

steps are similar to a synthesis script, except that the design is not compiled and the timing file is read before reporting the timing. The following steps should be performed:

1. **Setup Libraries.**
2. **Set operation conditions.**
3. **Read netlsit.**
4. **Read SDF timing.**
5. **Set timing constraints.**
6. **Write timing report.**

Example Script

The following provides an example of Exemplar back annotation script.

```
################################################
##                                            ##
##   Setup Library and Work Directory         ##
##                                            ##
################################################
clean_all
set_working_dir "./"
load_library A500K

################################################
##                                            ##
##   Set Operating Conditions and Wire-Load   ##
##                                            ##
################################################
set temp 70
set process worst
set wire_tree balanced

################################################
##                                            ##
## SEt Netlist Format and Read Netlist        ##
##                                            ##
################################################
set input-format Verilog
read -format Verilog alu.v.
unfold

################################################
##                                            ##
```

```
## SEtup Sdf Back Annotation and Read File   ##
##                                           ##
###############################################
set effort.back_annotate
set sdf_hierarchical_names TRUE
set sdf_type maximum
read -format SDF alu.sdf

###############################################
##                                           ##
## Timing Constraints                        ##
##                                           ##
###############################################
present_design .work.alu.INTERFACE

set_clock -port -name .work.alu.INTERFACE.clk \
          -clock_cycle 33ns
set_attribute -port acc required_time 5
set_attribute -port data_in arrival_time 3

###############################################
##                                           ##
## Write Reports and Netlist                 ##
##                                           ##
###############################################
report_delay reports/alu.backanno.timing.rpt \
             -num_paths 1 -longest_path -clock_frequncy

###############################################
##                                           ##
## End of Script                             ##
##                                           ##
###############################################
```

7
Synthesis Using Synopsys Design Compiler®

Actel's ASICmaster™ software, along with a logic synthesis tool and simulation/timing analysis capability, provides a complete top-down design environment for using the Actel A500K ProASIC™ devices.

This chapter describes how to use the Synopsys Design Compiler® and the Actel library to produce optimal results with Actel's ASICmaster. Additionally, it contains information about: requirements for using the Actel design kit for the Synopsys Synthesis Environment, the location and content of the Actel-Synopsys library, setting up the Synopsys environment, and related publications

Related Publications

To supplement the information in this document, refer to these additional documents:

- *Synopsys Design Compiler Reference Manual*
- *Synopsys HDL Compiler for Verilog Manual*
- *Synopsis VHDL Compiler Manual*

Requirements

The following software is required for Synopsys synthesis and Actel's ASICmaster design environments:

- Synopsys Design Compiler (version 1998.02-02 or later) or FPGA Compiler (version 1998.02-02 or later) synthesis tool.

  These tools generate an optimized netlist file for Actel devices.

- Current ProASIC Design Compiler synthesis libraries.

Optional Software:

- Design Analyzer.

  This software tool allows the user to view the results of the Synthesis in a schematic viewer.

- DesignWare Foundation License

This enables the use of the most advanced synthesis architectures and will therefore give the most optimal synthesis results.

Location of the A500K Design Compiler Libraries

Files and subdirectories specific to Design Compiler are located in the following directory:

```
$AMHOME/etc/deskits/synopsys
```

Where $AMHOME is the UNIX environment variable set to the full pathname of the installation directory of the ASICmaster software and design kits. For details on UNIX environment variables see the *ASICmaster User's Guide*.

Within this directory you will find the following subdirectory:

Table 7-1: Synopsys Libraries and Examples

Name	Description
lib	Subdirectory that contains the synthesis library, named A500K.db, and the symbol library, named A500K.sdb

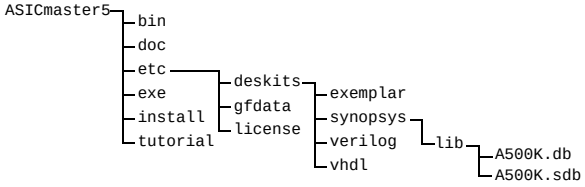

Figure 7-1: A500K Library Location

Setup

This section presents general setup guidelines for using Synopsys Design Compiler with Actel's ProASIC devices. For additional information, refer to the *Synopsys Design Compiler Family Reference Manual*.

Library Setup

In the project specific setup file for Synopsys Design Compiler, the following should be specified:

search_path

The search_path variable should be set to cover the current directory, Synopsys DesignWare directory, and the A500K family specific synthesis library directory. For example:

```
AMHOME= "/export/home/ASICmaster5/"
AMLIB = AMHOME + "/etc/deskits/synopsys/lib/"
search_path = {. synopsys_root + "/libraries/syn " AMLIB };
```

target_library

The target library specifies the library used during mapping. It must point to the Actel ProASIC library.

```
target_library = { "A500K.db"}
```

link_library

The link library specifies the library used to resolve instances in netlist read. It must point to the contents of the memory and the Actel ProASIC library.

```
link_library = {"*" "A500K.db"}
```

symbol_library

The symbol library specifies the library used to link the netlist to a visual representation in Design Analyzer. It must be set for the use of Design Analyzer to the Actel ProASIC symbol library.

```
symbol_library = "A500K.sdb"
```

synthetic_library

The synthetic library specifies the architecture library that Design Compiler uses during optimization.

Note: Actel recommends using DesignWare Foundation with the ProASIC family.

Synthetic library example with DesignWare Foundation:

```
synthetic_library = { standard.sldb dw01.sldb, dw02.sldb, \
                      dw03.sldb dw04.sldb dw05.sldb}
```

Synthetic library example without DesignWare Foundation:

```
synthetic_library = {standard.sldb }
```

Translating to Actel Libraries

If the design has technology-dependent instances that are not Actel-specific, use the following procedure to translate the design to the Actel library:

1. **Set link_library to the original technology library.**
2. **Set target_library to A500K.db.**
3. **UsetheDesignCompilertranslatecommand.**TheDesignCompiler translates designs cell-by-cell from the original technology, while attempting to preserve the functional structure of the design. It also attempts to re-map any structures that are not functionally identical.

Verilog Specific Setup

If Verilog HDL is used as the output netlist format the following variables have to be set:

verilogout_no_tri = true

If this variable is set to true, it prevents Design Compiler from using behavioral style assignments in the netlist.

verilogout_single_bit = false

If this variable is set to false, Design Compiler does not flatten busses to single bits in the netlist.

VHDL Specific Setup

If VHDL is used as the netlist format the following variables have to be set correctly:

```
bus_naming_style = "%s_%d"

bus_dimension_separator_style = "_"
```

These bus naming conventions match the VHDL naming conventions and makes Design Compiler use this naming style from the start.

SDF Specific Setup

In general, cell and net names can not contain a slash "/" if SDF files will be generated or read with Design Compiler. In SDF the slash is used as a hierarchy delimiter. Therefore the following name rule has to be in the Synopsys Design Compiler setup entitled "setup (synopsys_dc.setup)":

```
define_name_rules NO_SLASH -restricted "/" -replacement_char "_"

default_name_rules = NO_SLASH
```

Synthesis Guidelines

The following sections present general guidelines for Synopsys Design Compiler synthesis scripts for Actel's ProASIC devices.

Analyze and Elaborate

Actel recommends using the analyze and elaborate commands to read in the contents of a RTL level HDL description.

Note: If memories generated with MEMORYmaster are used, read these files as normal RTL description. Link the design together at the top level. Before compiling set a "NOOPT" attribute on all these memories.

Inserting Pads

The set_port_is_pad command and the insert_pads command are used to automatically select and inserts input, output, and bidirectional pads in the

netlist. You can control the pad selection for each port by using the set_pad_type -exact command.

Note: A netlist must include pads before importing it into ASICmaster. For descriptions of available I/O types, see the *ProASIC Macro Library Guide.*

If you use the automatic pad selection method, you might get more clock pads than available on the chip, particularly if the clocks are gated. By default, Design Compiler inserts clock buffers on all ports that drive the clock input pin for synchronous cells. The following command removes an extra clock pad from a specific input signal:

```
set_pad_type -no_clock Input_signal_name
```

Using Global Signals

The four dedicated routing resources can be used to implement global signals (such as clocks and resets). Taking advantage of these dedicated routing resources reduces overall routing congestion and improves design performance. The following command specifies the use of a global resource:

```
set_pad_type -exact GL33U Input_signal_name
```

Note: The global selection of nets and ports can also be controlled in an ASICmaster constraint file. ASICmaster can also automatically allocate global resources to high fanout nets to improve routing. For details, refer to the *ASICmaster User's Guide*. For a full listing of available global buffers, see the *ProASIC Macro Library Guide.*

Design Environment

The wire load model, and the operating conditions, must be specified in the design in synthesis in order to be coherent to the post-layout results.

Wire Load Models

Actel A500K libraries for Synopsys contain a general wire load model. You should set the default wire load model to "A500K":

```
set_wire_load A500K
```

Note: If you do not set a wireload, the interconnect delay will be 0, unless you have previously back annotated timing into the design.

In the general wire load model, pad cell delays are in reference to an external load of 35pF. If these loads have to be changed, individual pad loads can be specified in the synthesis environment if the "A500K_padext" model is used. To do so, group all pads at the top level with no other leaf cells in this hierarchy, except a hierarchal core block. Then set the wire load model to "A500K_padext" for the top level and apply appropriate output capacitances to the I/O pads. Make sure that, for the core hierarchy, the "A500K" wire load model is still in use. Where pads with pull-ups are present, pull-up resistance has been added to ensure a correct delay calculation.

Operating Conditions

No default operating conditions have been set in the current library. The choices are "Best", "Typical", or "Worst". "Best" corresponds to best process, a temperature of 0^{O}C and a voltage of 2.75V. "Typical" corresponds to worst process, a temperature of 25^{O}C and a voltage of 2.5V. "Worst" corresponds to worst process, a temperature of 70^{O}C and a core voltage of 2.25V. Using these parameters allows you to explore the robustness of the design in different operating conditions. In a Design Compiler script, set the operating conditions as:

```
set_operating_conditions "WORST"
```

Timing Constraints

To inform the synthesis tool of the timing requirements for surrounding designs and chips, timing constraints must be specified to the tool, specifying clocks, input delays and output delays as well as false paths and multi cycle paths.

To set constraints, Actel recommends:

1. **Create the clocks of the design.**
2. **Before adding any other constraints, compile once to get an initial idea of the design's performance.**
3. **To set the timing or area constraints, type the following commands:**

```
set_max_delay, or set_max_area, create_clock,
set_input_delay, set_output_delay
```

Actel recommends to set rational constraints.

4. **Re-run the compilation.**
5. **If the results are worse than the initial compilation, relax the constraints and recompile the design.**
6. **To forward-annotate timing data to ASICmaster, use the "write_constraints -format" sdf command to instruct Design Compiler to write out timing constraints.** Generally, you should not pass all of the Synopsys synthesis optimization constraints along to ASICmaster.

Design Compilation

The map_effort option of the compile command can be set to low, medium, or high. In most cases, you should start with medium as the default effort level.

Note: Make sure all memories generated by MEMORYmaster have the "dont_touch" attribute set to true. If this is not done, the resulting netlist will not match with the constraints file for the memories generated by MEMORYmaster.

Some designs benefit you perform an additional compile step at the high effort level, flatten for cross-boundary optimization, or perform an incremental compile. Generally, you should not use map_effort low.

Reports Generation

To become familiar with the gate level design, the report options of Design Compiler should be use extensively. For additional information, refer to the *Synopsys Design Compiler Family Reference Manual*.

Forward Constraints

The Synopsys Design Compiler writes timing constraints for use in ASICmaster. After compiling the design, you can use the write_constraints command to create an SDF file that contains timing constraints. Make sure that all the name rules specified in the setup section

are applied by using the change_names command for all hierarchies before writing the SDF file. For example:

```
change_names -h
write_constraints -o timing.sdf -format sdf-v2.1 -cover_design
```

The checker program in ASICmaster checks for syntactical and semantic errors in the SDF file. It also performs an initial timing estimation, using intrinsic cell delays and estimated wire delays to determine whether the constraint is likely to be met.

Example Script

This following provides an example of Synopsys Design Compiler synthesis script:

```
/* ~~~~~~~~~~~~~~~~~~~~~~~~~~~~
   Technology dependent Setting
   ~~~~~~~~~~~~~~~~~~~~~~~~~~~~ */
AMHOME = "/net/jedi/xwing/Solaris/gatefield/am5b2"
AMLIB = " " + AMHOME + "/etc/deskits/synopsys/lib/"

search_path = {. synopsys_root + "/libraries/syn " AMLIB };

/* ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
   Create a Directory and a library WORK
   ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ */
sh mkdir WORK

define_design_lib WORK -path "./WORK"

/* ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
    bus naming style and name_rules
   ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ */
bus_naming_style = %s_%d;
bus_dimension_separator_style = "_"
define_name_rules NO_SLASH -restricted "/" \
                    -replacement_char "_"
default_name_rules = NO_SLASH

/* ~~~~~~~~~~~~~~~~~~~~~~~~
   VHDL Files Read/Analyze
   ~~~~~~~~~~~~~~~~~~~~~~~~ */
read -format vhdl {"../source/file1.vhd"}
read -format vhdl {"../source/file2.vhd"}

read -format vhdl {"../source/topdesign.vhd"}
```

```
current_design = AM29116_CRC
link

/* ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
   Clock Creation and Protection
   ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ */
create_clock “CP” -p 20 -waveform { 0 10.0 }
set_clock_skew -uncertainty 0.2 CP
dont_touch_port_list = {CP}
dont_touch_network dont_touch_port_list

/* ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
   Operating Conditions and Wire Load Setting
   ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ */
set_operating_conditions WORST -library A500K
set_wire_load A500K -library A500K

/* ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
    Uniquify / Flatten: Design dependent
   ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ */
ungroup -all -flatten

/* ~~~~~~~~~~~~~~~~~~~~~~~~~~
   Pad Insertion and setting
   ~~~~~~~~~~~~~~~~~~~~~~~~~~ */
set_port_is_pad
set_dont_use { find (cell, “A500K/IO*25*”) \
               find (cell, “A500K/O*25*”) }
set_pad_type -slewrate high find (port, “*”)
insert_pads   -respect_hierarchy

/* ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
    Timing or Boolean optimization or none
    ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ */
/* ~~~~~~~~~~~~~~~
   Mapping Effort
   ~~~~~~~~~~~~~~~ */
 compile -map_effort medium

/* ~~~~~~~~~~~~~~~~~~
   Reports Generation
   ~~~~~~~~~~~~~~~~~~ */
sh mkdir ./reports

report_area ./reports/area.txt
report_timing ./reports/time.txt

/* ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
   DB Saving and Netlist Generation
```

```
   ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ */
sh mkdir ./netlists

vhdlout_bit_type = std_logic;
vhdlout_bit_vector_type = std_logic_vector;

change_names -hierarchy
write -f vhdl -h -o ./netlists/top.vhd
write -f db -h -o ./netlists/top.db

/* ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
   Post Synthesis SDF file generation
   ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ */
write_constraints -format sdf-v2.1 -cover_design \
                        -output ./netlists/top.sdf
quit
```

Back Annotation Scripting

The following script presents general guidelines for Synopsys Design Compiler backannotation scripts for Actel's ProASIC devices. The basic steps are similar to a synthesis script, except that the design will not be compiled, but the timing file will be read before reporting the timing. The steps to be performed are as follows:

1. **Specify technology dependent settings.**
2. **Read netlist.**
3. **Set timing constraints.**
4. **Read SDF timing.**
5. **Write timing report.**

Example Script

The following provides an example of Synopsys back annotation script:

```
/* ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
   Technology dependent Setting
   ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ */
AMHOME = "/net/jedi/xwing/Solaris/gatefield/am5b2"
AMLIB = " " + AMHOME + "/etc/deskits/synopsys/lib/"
```

```
search_path = {. synopsys_root + "/libraries/syn " AMLIB };

/* ~~~~~~~~~~~~~~~~~~~~~~
   Read VHDL Netlist Files
   ~~~~~~~~~~~~~~~~~~~~~~ */
read -format vhdl {"../netlists/top.vhd"}

current_design = AM29116_CRC
link

/* ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
   Clock Creation and Protection
   ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ */
create_clock "CP" -p 20 -waveform { 0 10.0 }
set_clock_skew -propagated CP

/* ~~~~~~~~~~~~~~~~~~
   Read SDF timing File
   ~~~~~~~~~~~~~~~~~~ */
read_timing ../sdf/top.sdf

/* ~~~~~~~~~~~~~~~~~~
   Reports Generation
   ~~~~~~~~~~~~~~~~~~ */
report_timing ./reports/backanno_timing.rpt

quit
```

A

Product Support

Actel backs its products with various support services including Customer Service, a Customer Applications Center, a Web and FTP site, electronic mail, and worldwide sales offices. This appendix contains information about using these services and contacting Actel for service and support.

Actel U.S. Toll-Free Line

Use the Actel toll-free line to contact Actel for sales information, technical support, requests for literature about Actel and Actel products, Customer Service, investor information, and using the Action Facts service.

The Actel Toll-Free Line is (888) 99-ACTEL.

Customer Service

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

From Northeast and North Central U.S.A., call (408) 522-4480.
From Southeast and Southwest U.S.A., call (408) 522-4480.
From South Central U.S.A., call (408) 522-4434.
From Northwest U.S.A., call (408) 522-4434.
From Canada, call (408) 522-4480.
From Europe, call (408) 522-4252 or +44 (0) 1256 305600.
From Japan, call (408) 522-4743.
From the rest of the world, call (408) 522-4743.
Fax, from anywhere in the world (408) 522-8044.

Customer Applications Center

The Customer Applications Center is staffed by applications engineers who can answer your hardware, software, and design questions.

All calls are answered by our Technical Message Center. The center retrieves information, such as your name, company name, phone number and your question, and then issues a case number. The Center then forwards the information to a queue where the first available application engineer receives the data and returns your call. The phone hours are from 7:30 a.m. to 5 p.m., Pacific Standard Time, Monday through Friday.

The Customer Applications Center number is (800) 262-1060.

European customers can call +44 (0) 1256 305600.

Guru Automated Technical Support

Guru is a Web based automated technical support system accessible through the Actel home page (**http://www.actel.com/guru/**). Guru provides answers to technical questions about Actel products. Many answers include diagrams, illustrations and links to other resources on the Actel Web site. Guru is available 24 hours a day, seven days a week.

Web Site

Actel has a World Wide Web home page where you can browse a variety of technical and non-technical information. Use a Net browser (Netscape recommended) to access Actel's home page.

The URL is **http://www.actel.com**. You are welcome to share the resources we have provided on the net.

Be sure to visit the "Actel User Area" on our Web site, which contains information regarding: products, technical services, current manuals, and release notes.

FTP Site

Actel has an anonymous FTP site located at **ftp://ftp.actel.com**. You can directly obtain library updates, software patches, design files, and data sheets.

Electronic Mail

You can communicate your technical questions to our e-mail address and receive answers back by e-mail, fax, or phone. Also, if you have design problems, you can e-mail your design files to receive assistance. The e-mail account is monitored several times per day.

The technical support e-mail address is **tech@actel.com**.

Worldwide Sales Offices

Headquarters

Actel Corporation
955 East Arques Avenue
Sunnyvale, California 94086
Toll Free: 888.99.ACTEL

Tel: 408.739.1010
Fax: 408.739.1540

US Sales Offices

California

Bay Area
Tel: 408.328.2200
Fax: 408.328.2358

Irvine
Tel: 949.727.0470
Fax: 949.727.0476

San Diego
Tel: 619.938.9860
Fax: 619.938.9887

Thousand Oaks
Tel: 805.375.5769
Fax: 805.375.5749

Colorado

Tel: 303.420.4335
Fax: 303.420.4336

Florida

Tel: 407.677.6661
Fax: 407.677.1030

Georgia

Tel: 770.831.9090
Fax: 770.831.0055

Illinois

Tel: 847.259.1501
Fax: 847.259.1572

Maryland

Tel: 443.255.1346
Fax: 410.290.3291

Massachusetts

Tel: 978.244.3800
Fax: 978.244.3820

Minnesota

Tel: 612.854.8162
Fax: 612.854.8120

North Carolina

Tel: 919.376.5419
Fax: 919.376.5421

Pennsylvania

Tel: 215.830.1458
Fax: 215.706.0680

Texas

Tel: 972.235.8944
Fax: 972.235.965

International Sales Offices

Canada
Suite 203
135 Michael Cowpland Dr,
Kanata, Ontario K2M 2E9

Tel: 613.591.2074
Fax: 613.591.0348

France
361 Avenue General de Gaulle
92147 Clamart Cedex

Tel: +33 (0)1.40.83.11.00
Fax: +33 (0)1.40.94.11.04

Germany
Bahnhofstrasse 15
85375 Neufahrn

Tel: +49 (0)8165.9584.0
Fax: +49 (0)8165.9584.1

Hong Kong
Suite 2206,
Parkside Pacific Place,
88 Queensway

Tel: +011.852.2877.6226
Fax: +011.852.2918.9693

Italy
Via Giovanni da Udine No. 34
20156 Milano

Tel: +39 (0)2.3809.3259
Fax: +39 (0)2.3809.3260

Japan
EXOS Ebisu Building 4F
1-24-14 Ebisu Shibuya-ku
Tokyo 150

Tel: +81 (0)3.3445.7671
Fax: +81 (0)3.3445.7668

Korea
135-090, 18th Floor,
Kyoung AmBldg
157-27 Samsung-dong
Kangnam-ku, Seoul

Tel: +82 (0)2.555.7425
Fax: +82 (0)2.555.5779

Taiwan
4F-3, No. 75, Sec. 1,
Hsin-Tai-Wu Road,
Hsi-chih, Taipei, 221

Tel: +886 (0)2.698.2525
Fax: +886 (0)2.698.2548

United Kingdom
Daneshill House,
Lutyens Close
Basingstoke,
Hampshire RG24 8AG

Tel: +44 (0)1256.305600
Fax: +44 (0)1256.355420

Index

G

H

I

L

M

O

P

R

S

T

V

W