# Cadence®

Interface Guide

UNIX® Environments

**Actel Corporation, Sunnyvale, CA 94086**

Printed in the United States of America

Part Number: 5579000-2

Release: April 1999

## Trademarks

# Table of Contents

# List of Figures

# Introduction

The *Cadence Interface Guide* contains detailed information about using Cadence software to create desings for Actel devices. Refer to the *Designing with Actel* manual for additional information about using the Designer software and the Cadence documentation for information about using Cadence tools.

## Document Organization

The *Cadence Interface Guide* is divided into the following sections:

**Chapter 1 - Setup** contains information and procedures about setting up Cadence tools to creat Actel Designs.

**Chapter 2 - Actel-Cadence Design Flows** illustrates and describes the design flow for:

- Creating Actel Designs using Cadence Concept schematic capture tool and Verilog-XL simulator.
- Creating Actel Designs using Cadence Concept schematic capture tool and Leapfrog VHDL simulator.
- Creating Actel Designs using Cadence Composer schematic capture tool and Verilog-XL simulator

**Chapter 3 - Actel-Concept Design Considerations** contains information and procedures to assist you in creating Actel designs with Cadence Concept.

**Chapter 4 - Actel-Composer Design Considerations** contains information and procedures to assist you in creating Actel designs with Cadence Composer.

**Chapter 5 - Simulation Using Verilog-XL** contains information and procedures about simulating Actel designs with Verilog-XL.

**Chapter 6 - Simulation Using Leapfrog** contains information and procedures about simulating Actel designs with Leapfrog.

**Chapter 7 - Integrating Synthesis Tools with Cadence** contains information about importing Actel optimized logic blocks into Cadence-generated designs.

**Appendix A - Product Support** provides information about contacting Actel for customer and technical support.

## Document Assumptions

The information in this manual is based on the following assumptions:

1. You have installed the Designer Series software.
2. You have installed the Cadence software.
3. You are familiar with UNIX operating system environments.
4. You are familiar with Actel FPGA architecture and FPGA design software.

## Document Conventions

The following conventions are used throughout this manual.

Information that is meant to be input by the user is formatted as follows:

```
keyboard input
```

The contents of a file is formatted as follows:

```
file contents
```

Messages that are displayed on the screen appear as follows:

```
Screen Message
```

The following variables are used throughout this manual.

- Actel FPGA family libraries are shown as <act_fam>. Substitute the desired Actel FPGA family (act1, act2 (for ACT 2 and 1200XL devices), act3, 3200dx, 40mx, 42mx, and 54sx) as needed. For example:

  ```
  va2adl fam:<act_fam> <design>
  ```

  <ACT_FAM> refers to the same, but with the name in upper case.
- Compiled VHDL libraries are shown as <vhd_fam>. Substitute <vhd_fam> for the desired VHDL family (act1, act2 (for ACT 2 and 1200XL devices), act3, a3200dx, a40mx, a42mx, and a54sx) as needed. VHDL design flows require that the library names begin with an alpha character.
- The Actel installation directory is shown as <alsdir>.

## Actel Manuals

The Designer Series software includes printed and on-line manuals. The on-line manuals are in PDF format on the CD-ROM in the "/ manuals" directory. These manuals are also installed onto your system when you install the Designer software. To view the on-line manuals, you must install Adobe® Acrobat Reader® from the CD-Rom.

The Designer Series includes the following manuals, which provide additional information on designing Actel FPGAs:

*Designing with Actel.* This manual describes the design flow and user interface for the Actel Designer Series software, including information about using the ACTgen Macro Builder and ACTmap VHDL Synthesis software.

*Actel HDL Coding Style Guide.* This guide provides preferred coding styles for the Actel architecture and information about optimizing your HDL code for Actel devices.

*ACTmap VHDL Synthesis Methodology Guide.* This guide contains information, optimization techniques, and procedures to assist designers in the design of Actel devices using ACTmap VHDL.

*Silicon Expert User's Guide*. This guide contains information and procedures to assist designers in the use of Actel's Silicon Expert tool.

*DeskTOP Interface Guide*. This guide contains information about using the integrated VeriBest® and Synplicity® CAE software tools with the Actel Designer Series FPGA development tools to create designs for Actel Devices.

*Cadence® Interface Guide*. This guide contains information and procedures to assist designers in the design of Actel devices using Cadence CAE software and the Designer Series software.

*Mentor Graphics® Interface Guide*. This guide contains information and procedures to assist designers in the design of Actel devices using Mentor Graphics CAE software and the Designer Series software.

*MOTIVE™ Static Timing Analysis Interface Guide*. This guide contains information and procedures to assist designers in the use of the MOTIVE software to perform static timing analysis on Actel designs.

*Synopsys® Synthesis Methodology Guide*. This guide contains preferred HDL coding styles and information and procedures to assist designers in the design of Actel devices using Synopsys CAE software and the Designer Series software.

*Viewlogic Powerview® Interface Guide*. This guide contains information and procedures to assist designers in the design of Actel devices using Powerview CAE software and the Designer Series software.

*Viewlogic Workview Office Interface Guide*. This guide contains information and procedures to assist designers in the design of Actel devices using Workview Office CAE software and the Designer Series software.

*VHDL Vital Simulation Guide*. This guide contains information and procedures to assist designers in simulating Actel designs using a Vital compliant VHDL simulator.

*Verilog Simulation Guide*. This guide contains information and procedures to assist designers in simulating Actel designs using a Verilog simulator.

*Activator and APS Programming System Installation and User's Guide.* This guide contains information about how to program and debug Actel devices, including information about using the Silicon Explorer diagnostic tool for system verification.

*Silicon Sculptor User's Guide.* This guide contains information about how to program Actel devices using the Silicon Sculptor software and device programmer.

*Silicon Explorer Quick Start.* This guide contains information about connecting the Silicon Explorer diagnostic tool and using it to perform system verification.

*Designer Series Development System Conversion Guide UNIX® Environments.* This guide describes how to convert designs created in Designer Series versions 3.0 and 3.1 for UNIX to be compatible with later versions of Designer Series.

*Designer Series Development System Conversion Guide Windows Environments.* This guide describes how to convert designs created in Designer Series versions 3.0 and 3.1 for Windows to be compatible with later versions of Designer Series.

*Actel FPGA Data Book.* This guide contains detailed specifications on Actel device families. Information such as propagation delays, device package pinout, derating factors, and power calculations are found in this guide.

*Macro Library Guide.* This guide provides descriptions of Actel library elements for Actel device families. Symbols, truth tables, and module count are included for all macros.

*A Guide to ACTgen Macros.* This Guide provides descriptions of macros that can be generated using the Actel ACTgen Macro Builder software.

## On-Line Help

The Designer Series software comes with on-line help. On-line help specific to each software tool is available in Designer, ACTgen, ACTmap, Silicon Expert, Silicon Explorer, Silicon Sculptor, and APSW.

# 1 Setup

This chapter contains information about setting up Cadence software to create Actel designs. Refer to the Cadence documentation for additional information about setting up Cadence software.

Included in this chapter are software requirements; details about the user setup, details about migration libraries, and details about project setup for Cadence Composer and Cadence Concept.

## Software Requirements

The information in this guide applies to the Actel Designer Series software release R1-1999 or later and Cadence software. For specific information about which versions are supported with this release, go to the Guru automated technical support system on the Actel Web site (http://www.actel.com/guru) and type the following in the Keyword box:

```
third party
```

## User Setup

Before you create designs with the Actel library, you must set up your account to properly access the Actel and Cadence software. The following sections describe how to configure your user account for Designer Series in the Cadence environment. Refer to the *Designing With Actel* manual for Designer Series environment setup.

### Compiling Libraries for Cadence's Leapfrog Simulator

You must compile the Actel FPGA library models before using the Actel VHDL VITAL library models with Leapfrog. The following procedures describe how to compile libraries for the Cadence Leapfrog simulator.

### To compile the Actel FPGA library models:

This procedure compiles an Actel VITAL library in the "$ALSDIR/lib/vtl/95/lfrog" directory. The FPGA library models must be compiled for the Actel VITAL 95 libraries to work properly.

1. **Create a directory called "lfrog" in the "$ALSDIR/lib/vtl/95" directory.**
2. **Change to the "$ALSDIR/lib/vtl/95/lfrog" directory.**
3. **Create a directory named <vhd_fam>.**
4. **Map the library.** Compile the models and create the "cds.lib" file as follows:

   ```
   INCLUDE $CDS/tools/leapfrog/files/cds.lib
   DEFINE <vhd_fam> $ALSDIR/lib/vtl/95/lfrog/<vhd_fam>
   ```

5. **Compile the library.** Type the following command at the prompt:

   ```
   cv -work <vhd_fam> -messages -file $ALSDIR/lib/vtl/95/
   <act_fam>.vhd
   ```

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

   ```
   cv -work a40mx -messages -file $ALSDIR/lib/vtl/95/40mx.vhd
   ```

   Note: Perform the following step only if you are compiling the migration library.

6. **(Optional) Compile the migration library.** Type the following command at the prompt:

   ```
   cv -work <vhd_fam> -messages -file $ALSDIR/lib/vtl/95/
   <act_fam>_mig.vhd
   ```

## Migration Libraries

In addition to the standard Actel libraries, Actel provides a set of migratrion libraries. These libraries contain macros that were supported in earlier versions of the Designer Series software and macros that may be needed to retarget designs to a different Actel family. Actel does not recommend using the migration libraries on new designs.

## Project Setup

The user must setup the project to access Actel libraries and generate schematics with Cadence Concept or Composer.

### Concept Project Setup

To access Actel libraries with Cadence Concept, you must edit the *master.local*, *global.cmd*, and *hdldir.cmd* files as shown in the following examples:

#### *Example "global.cmd" file:*

The following is an example of a "global.cmd" file:

```
master_library "master.local";
library "<act_fam>";
use "project.wrk";
```

#### *Example "master.local" file:*

```
FILE_TYPE = MASTER_LIBRARY
'<act_fam>' '<alsdir>/lib/va/<act_fam>/<act_fam>.lib';
end.
```

#### *Example "hdldir.cmd" file:*

The "hdldir.cmd" file defines the option values used by the HDL Direct utility. The following example shows the information required for the "hdldir.cmd" file:

```
vlog_uppercase=TRUE
lang=verilog&vhdl&scald
```

## Concept Project Setup with Migration Libraries

To access Actel migration libraries with Cadence Concept, you must edit the *master.local*, *global.cmd*, and *hdldir.cmd* files as shown in the following examples:

### Example “global.cmd” file:

The following is an example of a “global.cmd” file:

```
master_library “master.local”;
library “<act_fam>”, “<act_fam>_mig”;
use “project.wrk”;
```

### Example “master.local” file:

```
FILE_TYPE = MASTER_LIBRARY
'<act_fam>' '<alsdir>/lib/va/<act_fam>/<act_fam>.lib';
'<act_fam>_mig' '<alsdir>/lib/va/<act_fam>/<act_fam>_mig.lib';
end.
```

### Example “hdldir.cmd” file:

The “hdldir.cmd” file defines the option values used by the HDL Direct utility. The following example shows the information required for the “hdldir.cmd” file:

```
vlog_uppercase=TRUE
lang=verilog&vhdl&scald
```

## Composer Project Setup

To access Actel libraries with Cadence Composer, you must set the path in the “cds.lib” file as shown in the following example:

Note: Variables should be upper case and lower case, as shown in the example.

### *Example “cds.lib” file:*

```
DEFINE <ACT_FAM> <alsdir>/lib/cds/<act_fam>/<ACT_FAM>
```

## Composer Project Setup with Migration Libraries

To access Actel migration libraries with Cadence Composer, you must set the path in the “cds.lib” file as shown in the following example:

Note: Variables should be upper case and lower case, as shown in the example.

### *Example “cds.lib” file:*

```
DEFINE <ACT_FAM> <alsdir>/lib/cds/<act_fam>/<ACT_FAM>
DEFINE <act_fam>_mig <alsdir>/lib/cds/<act_fam>/<act_fam>_mig
```

2

# Actel-Cadence Design Flows

This chapter illustrates and describes the design flow for creating Actel designs using Cadence design and simulation tools.

## Concept/Verilog-XL Schematic-Based Flow Illustrated

Figure 2-1 shows the schematic-based design flow for an Actel FPGA using Designer software and Concept schematic capture software.[1]

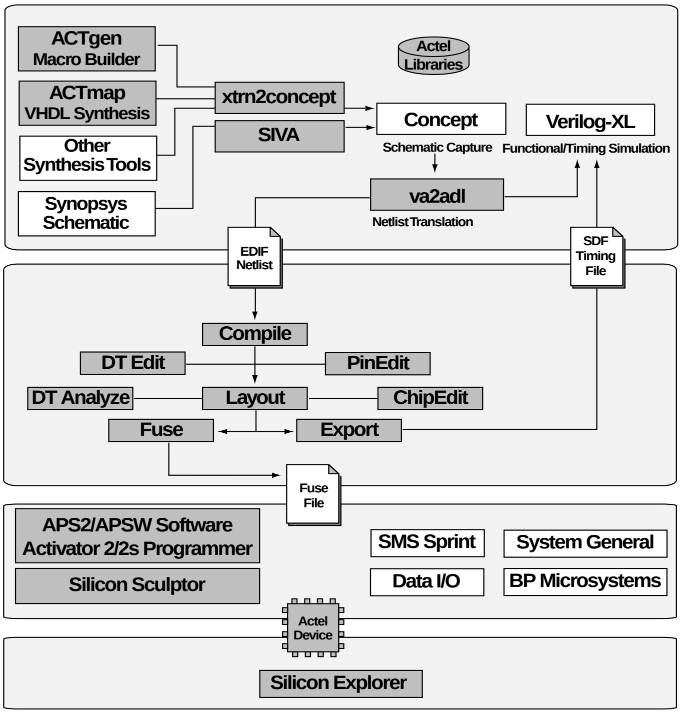

*Figure 2-1. Concept/Verilog-XL Design Flow*

1. *Actel-specific utilities/tools are denoted by the grey boxes in Figure 2-1.*

## Concept/Verilog-XL Schematic-Based Flow Overview

The Actel-Concept/Verilog schematic-based design flow has four main steps; Design Creation/Verification, Design Implementation, Programming, and System Verification. These steps are described in detail in the following sections.

### Design Creation/ Verification

During design creation/verification, a schematic representation of a design is captured using Cadence Concept software. After design capture, a pre-layout, functional simulation can be performed with Verilog-XL software. Finally, an EDIF netlist is generated for use in Designer.

#### Schematic Capture

Enter your schematic in Concept. Refer to the Cadence Concept documention for information about using Concept.

#### Functional Simulation

You perform a functional simulation of your design using Verilog-XL before generating an EDIF netlist for place and route. Functional simulation verifies that the logic of the design is correct. Unit delays are used for all gates during functional simulation. Refer to “Simulation Using Verilog-XL” on page 45 for information about performing functional simulation with Verilog-XL.

#### EDIF Netlist Generation

After you have captured and verified your design, you must generate an Actel EDIF netlist for place and route in Designer. Refer to “Capturing the Design using Concept” on page 31 for information about generating an EDIF netlist with Concept.

### Design Implementation

During design implementation, a design is placed and routed using Designer. Additionally, static timing analysis is performed on a design in Designer with the DT Analyze tool. After place and route, post-layout (timing) simulation is performed in Verilog-XL software.

### *Place and Route*

Use Designer to place and route your design. Refer to the *Designing with Actel* manual for information about using Designer.

### *Static Timing Analysis*

Use the DT Analyze tool in Designer to perform static timing analysis on your design. Refer to the *Designing with Actel* manual for information about using DT Analyze.

### *Timing Simulation*

You perform a timing simulation on your design after placing and routing it. Timing simulation requires information extracted and back annotated from Designer. Refer to "Simulation Using Verilog-XL" on page 45 for information about performing timing simulation with Verilog-XL.

## *Programming*

You can program a device with programming software and hardware from Actel or a supported 3rd party programming system. Refer to the *Designing with Actel* manual and the *APS Programming System User's Guide* or *Silicon Sculptor User's Guide* for information about programming an Actel device.

## *System Verification*

You can perform system verification on a programmed device using the Silicon Explorer. Refer to the *APS Programming System User's Guide* or *Silicon Explorer Quick Start* for information on using Silicon Explorer.

## Concept/Leapfrog Schematic-Based Flow Illustrated

Figure 2-2 shows the schematic-based design flow for an Actel FPGA using Designer software and Concept schematic capture software.[1]

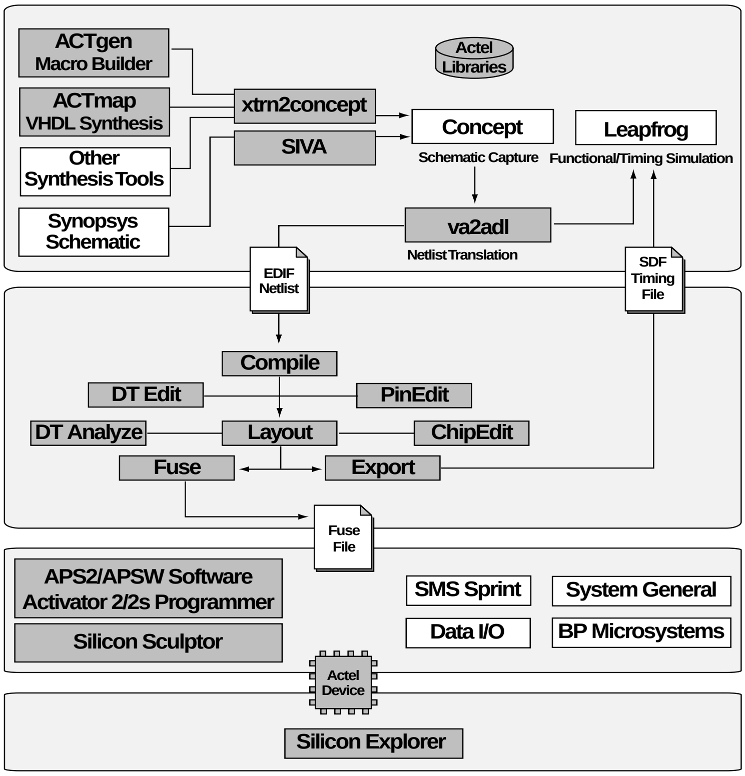

*Figure 2-2. Concept/Leapfrog Design Flow*

*1. Actel-specific utilities/tools are denoted by the grey boxes in Figure 2-2.*

# Concept/Leapfrog Schematic-Based Flow Overview

The Actel-Concept/Leapfrog schematic-based design flow has four main steps; Design Creation/Verification, Design Implementation, Programming, and System Verification. These steps are described in detail in the following sections.

## Design Creation/ Verification

During design creation/verification, a schematic representation of a design is captured using Cadence Concept software. After design capture, a pre-layout, functional simulation can be performed with Leapfrog software. Finally, an EDIF netlist is generated for use in Designer.

### Schematic Capture

Enter your schematic in Concept. Refer to the Cadence Concept documention for information about using Concept.

### Functional Simulation

You can perform a functional simulation of your design using Leapfrog before generating an EDIF netlist for place and route. Functional simulation verifies that the logic of the design is correct. Unit delays are used for all gates during functional simulation. Refer to “Simulation Using Leapfrog” on page 49 for information about performing functional simulation.

### EDIF Netlist Generation

After you have captured and verified your design, you must generate an Actel EDIF netlist for place and route in Designer. Refer to “Capturing the Design using Concept” on page 31 for information about generating an EDIF netlist with Concept.

## Design Implementation

During design implementation, a design is placed and routed using Designer. Additionally, timing analysis is performed on a design in Designer with the DT Analyze tool. After place and route, post-layout (timing) simulation is performed in Leapfrog software.

### Place and Route

Use Designer to place and route your design. Refer to the *Designing with Actel* manual for information about using Designer.

### Static Timing Analysis

Use the DT Analyze tool in Designer to perform static timing analysis on your design. Refer to the *Designing with Actel* manual for information about using DT Analyze.

### Timing Simulation

You perform a timing simulation on your design after placing and routing it. Timing simulation requires information extracted and back annotated from Designer. Refer to “Simulation Using Leapfrog” on page 49 for information about performing timing simulation.

## Programming

You program a device with programming software and hardware from Actel or a supported 3rd party programming system. Refer to the *Designing with Actel* manual and the *APS Programming System User’s Guide* or *Silicon Sculptor User’s Guide* for information about programming an Actel device.

## System Verification

You can perform system verification on a programmed device using the Silicon Explorer. Refer to the *APS Programming System User’s Guide* or *Silicon Explorer Quick Start* for information on using Silicon Explorer.

## Composer/Verilog-XL Schematic-Based Flow Illustrated

Figure 2-3 shows the schematic-based design flow for an Actel FPGA using Designer software and Composer schematic capture software.[1]

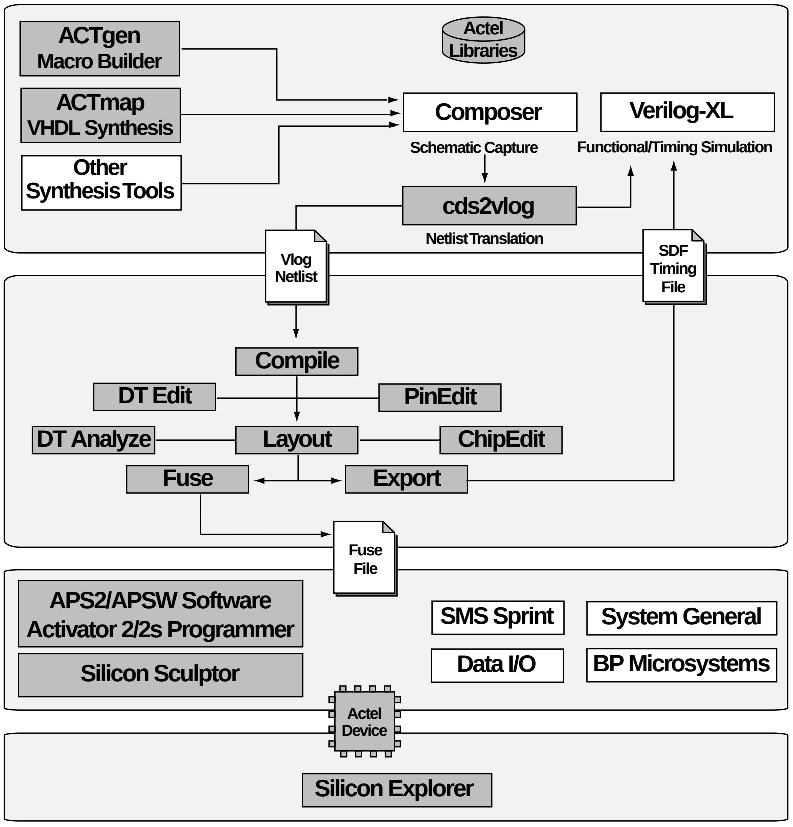

*Figure 2-3. Composer/Verilog-XL Design Flow*

*1. Actel-specific utilities/tools are denoted by the grey boxes in Figure 2-3.*

## Composer/Verilog-XL Schematic-Based Flow Overview

The Actel-Composer/Verilog-XL schematic-based design flow has four main steps; Design Creation/Verification, Design Implementation, Programming, and System Verification. These steps are described in detail in the following sections.

### Design Creation/ Verification

During design creation/verification, a schematic representation of a design is captured using Cadence Composer software. After design capture, a pre-layout, functional simulation can be performed with Leapfrog software. Finally, a Verilog netlist is generated for use in Designer.

#### Schematic Capture

Enter your schematic in Composer. Refer to the Cadence Composer documention for information about using Composer.

#### Functional Simulation

You perform a functional simulation of your design using Verilog-XL before place and route. Functional simulation verifies that the logic of the design is correct. Unit delays  are used for all gates during functional simulation. Refer to “Simulation Using Verilog-XL” on page 45 for information about performing functional simulation.

### Design Implementation

During design implementation, a design is placed and routed using Designer. Additionally, timing analysis is performed on a design in Designer with the DT Analyze tool. After place and route, post-layout (timing) simulation is performed in the Verilog-XL software.

#### Place and Route

Use Designer to place and route your design. Refer to the *Designing with Actel* manual for information about using Designer.

### Static Timing Analysis

Use the DT Analyze tool in Designer to perform static timing analysis on your design. Refer to the *Designing with Actel* manual for information about using DT Analyze.

### Timing Simulation

You perform a timing simulation on your design after placing and routing it. Timing simulation requires information extracted and back annotated from Designer. Refer to “Simulation Using Verilog-XL” on page 45 for information about performing timing simulation using Verilog-XL.

## Programming

You program a device with programming software and hardware from Actel or a supported 3rd party programming system. Refer to the *Designing with Actel* manual and the *APS Programming System User’s Guide* or *Silicon Sculptor User’s Guide* for information about programming an Actel device.

## System Verification

You can perform system verification on a programmed device using the Silicon Explorer. Refer to the *APS Programming System User’s Guide* or *Silicon Explorer Quick Start* for information on using Silicon Explorer.

3

# Actel-Concept Design Considerations

When creating a schematic with Cadence Concept for Actel devices, there are some conventions that should be observed, as described in the following sections.

## Updating Concept Designs

Designs created prior to the Cadence Design kit released with the Designer Series 3.0 require conversion. The “EXOR” macro has been replaced as “XOR2.” As the Verilog-XL libraries do not contain the “EXOR” macro, if you use the Verilog-XL simulator, you must replace all “EXOR” cells.

To replace “EXOR” macros with “XOR2” macros, type the following at the command line in the project directory where design files exist:

```
exor2xor2 <scald map file>
```

Designs created prior to the Cadence Design kit released with the latest version of Designer Series also require conversion. The “XNOR” macro has been replaced as “XNOR2.” As the regular Verilog-XL libraries do not contain the “XNOR” macro, if you use the Verilog-XL simulator, you must replace all “XNOR” cells.

To replace “XNOR” macros with “XNOR2” macros, type the following at the command line in the project directory where design files exist:

```
xnor2xnor2 <scald map file>
```

## Converting SCALD Schematics into HDL Direct Schematics

Actel and Cadence recommend using the HDL Direct utility to generate Verilog netlists for designs. Users with SCALD schematics will need to modify their schematics before using them with HDL Direct. After editing the schematics, run HDL Direct and Verilog-XL to check them.

The following changes must be made to SCALD schematics:

- If your SCALD drawing has multiple pages, change the ONE_ARCHITECTURE property on the DECLARATIONS symbol to a value of MANY_PAGES.
- Replace SCALD BIT TAP symbols with the HDL Direct SLICE symbols. SCALD BIT TAP symbols use the BIT property to specify the bit-to-tap from a vectored signal. If the range of the vectored signal does not start with zero, the tap number needs to be changed to reference the actual bit that is to be tapped.
- Replace SCALD LSB TAP and MSB TAP symbols with HDL Direct TAP symbols.
- Replace SCALD SIGN EXTEND and SLASH symbols with equivalent wiring.
- Add HDL Direct port symbols to the ports of the schematic.
- Remove \I from all signal names.
- Remove SCALD FLAG symbols from your schematics.
- Replace signals that end with \G to start with a forward slash (/).
- Do not connect wires or signals to pass thru pins on parts. Only connect to the main pin.
- Replace SCALD NOT bodies with wires. Use the *bubble_check off* option with the SCALD compiler.
- Rename all signals and symbol names starting with numbers to valid Verilog and VHDL identifiers.
- Replace the SCALD signal concatenation operator colon (:) with the HDL Direct signal concatenation operator ampersand (&).
- Replace the SCALD signal replication operator “\R number” with a concatenation of the required number of signals. The SCALD REPLICATE symbol should be replaced in the same way.
- Replace SCALD SYNONYM symbols with HDL Direct ALIAS symbols.
- Ensure all property values adhere to Verilog and VHDL rules. For example, you must change the SIZE=4B property to SIZE=4.

HDL Direct converts SCALD signal names to Verilog and VHDL signal names.

SCALD TAP symbols use the BN property to specify the bit to tap from a vectored signal. HDL Direct handles SCALD TAP symbols in the same way it handles HDL Direct SLICE symbols.

HDL Direct supports SCLAD MERGE symbols and treats them like HDL Direct CONCAT symbols. In HDL Direct, the output of a MERGE or CONCAT symbol must be an unnamed signal.

HDL Direct supports SCALD DRAWING symbols. These symbols are netlisted only in the SCALD connectivity file. They are ignored when creating Verilog modules or VHDL text.[1]

## Capturing the Design using Concept

You must schematically capture the design using the Concept and Actel libraries, import external blocks created, or use both methods. Use one of the Actel symbol family libraries.

Note: Properties assigned using Concept schematic capture tool are not preserved when the netlist is created.

The following steps you through capturing a design using the Concept schematic capture tool:

Note: For each design, do not use symbols from more than one Actel family library.

1. **Setup "global.cmd," "master.local, " "hdldir.cmd" files.** Refer to "Setup" on page 13 for detailed information.
2. **Invoke Concept.** From the project directory, type:

   ```
   concept
   ```

3. **Import the external blocks.** For additional information, refer to "Adding ACTgen Macros" on page 34, "Adding ACTmap Blocks" on page 35, and "Integrating Synthesis Tools with Cadence" on page 55.

*1. This section was copied from the "Using HDL Direct with SCALD Applications" section of the HDL Direct User Guide supplied by Cadence.*

4. **Instantiate either the Actel symbols or the symbols generated for external blocks.** Use the "add" command to instantiate the symbols and create the design's logic.

5. **Add a flag body for each I/O macro in the top level schematic.** Use a wire segment to connect an I/O macro pad pin to the flag body. Use the "SIGNAME" command to name this wire segment. This net name is the port name for the I/O macro.

   Use inport bodies for the macro inputs, outport bodies for the outputs, and ioport bodies for the bi-directional signals. Without the bodies, the corresponding Verilog code will not have the correct direction declaration and I/O macro list. Instantiate all I/O macros in the top level schematic.

   Figure 3-1 illustrates I/O buffer examples and buffer connections to the pins:

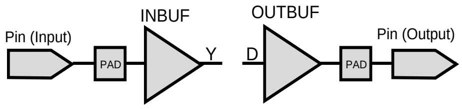

*Figure 3-1. I/O Buffers in Concept*

   Note: Use the bodies found in the hdl_direct_lib library.

6. **Name all incoming and outgoing cell signals.** Use flag bodies for each user-created macro or cell schematic.

7. **Set the HDL Direct interface ON.** Select the "Block" icon. Select the "HDL Direct On" menu option. Refer to your Cadence tutorials and documentation for additional information regarding the HDL Direct interface.

8. **Save the design and write the Verilog and VHDL netlist files.** Create the Verilog and VHDL netlist by writing the design from Concept as follows:

```
write <design>
```

Replace the “<design>” option with the name of your design.

Note: To avoid netlist translation errors, you must start net names with a letter. Design names must use only alphanumeric characters (A to Z or 0 to 9). Do not use slashes, underscores, dots, etc.

9. **Create a hierarchical Verilog description for the design using va2adl.** Va2adl creates both a hierarchical Verilog, VHDL, EDIF, and ADL descriptions of the design. From the command line type:

```
va2adl FAM:<act_fam> <design>
```

Note: Replace the “<design>” option with the name of your design.

The following error message will appear if the hdldir.cmd file is not correct:

```
Error: The hdldir.cmd file does not exist. Please create
an hdldir.cmd file and rewrite your design with HDLDirect
ON.

Example hdldir.cmd file:
vlog_uppercase=TRUE
lang=verilog & vhdl & scald
```

Note: Designs written before the hdldir.cmd file was created must be rewritten.

## Adding ACTgen Macros

With the ACTgen Macro Builder, you can create macros using a procedural description. You can generate netlists from the high-level language macros you create with ACTgen.

The ACTgen software generates a variety of critical structural macros such as, adders, subtracters, counters, comparators, and multiplexors in a shorter time than the typical HDL compilers. In addition, you can integrate ACTgen-based blocks with schematics and simulation tools such as Cadence simulation tools. To integrate an ACTgen-generated macro with schematic captures, you must create and instantiate a symbol in your design.

### Generating an ACTgen Macro

Refer to the *Designing with Actel* manual for information regarding using the ACTgen software.

### Integrating ACTgen with Concept

You can import an ACTgen macro directly into Concept by selecting Concept as the Output Format in ACTgen before generating the macro. ACTgen calls the “xtrn2concept” command and creates a Concept symbol, a connectivity file, verilog, and VHDL descriptions of the block.

A schematic is not generated for the block. To use “xtrn2concept,” the Cadence EDIF netlist reader, “redifnet,” must be in your Unix path and must have an available license.

1. **Setup “global.cmd” and “master.local” files.** Refer to “Setup” on page 13 for more information.
2. **Generate a Concept symbol and netlist for the ACTgen macro.** Select “Output Format” from the “File” submenu of the ACTgen Macro Builder window. Specify Concept for the Netlist/CAE Options. Select OK.
3. **Invoke Concept.** Invoke the program in your project directory.

   ```
   concept
   ```

4. **Instantiate the ACTgen block symbol in the design and connect the I/O to other design logic.** Once the symbol has been added, the macro is fully instantiated in the design. Refer to "Simulation Using Verilog-XL" on page 45 and to "Simulation Using Leapfrog" on page 49 for information regarding performing functional simulation and performing back annotated simulation. Refer to "Capturing the Design using Concept" on page 31 for information on generating an EDIF netlist.

## Adding ACTmap Blocks

The ACTmap VHDL Synthesis program can translate logic blocks described in ACTmap VHDL to either EDIF or ADL netlists. The output netlists are optimized to fit a particular ACT family architecture, and can be imported into Concept designs. To integrate a logic block optimized using ACTmap with schematic capture, you must create and instantiate a symbol in the design. Refer to the *Designing with Actel* manual for information on creating an ACTmap block.

### Generating an ACTmap Logic Block

For information regarding the ACTmap VHDL language description or the ACTmap software use, refer to the *Designing With Actel* manual.

### Integrating ACTmap with Concept

You can import an ADL, EDIF or verilog description of the ACTmap logic block into Concept using the "xtrn2concept" command. The "xtrn2concept" command creates a Concept symbol, a connectivity file, verilog, VHDL descriptions and netlist for the macro.

1. **Setup "global.cmd," "master.local, " "hdldir.cmd" files.** Refer to "Setup" on page 13 for detailed information.

2. **Generate a Concept symbol and netlist for the ACTmap logic block.** Use the "xtrn2concept" command by typing the following:

```
xtrn2concept FAM:<act_fam>
[INFORMAT:{adl,edif,verilog}]
[INFILE:<Input File>]
[GENSYM:{on,off}]
<block_name>
```

   Note: In this command, specify "adl," "edif," or "verilog" as the "INFORMAT" option. Use the appropriate Actel library as the "FAM" as the option. Use the file name containing the ACTmap block's description as the "INFILE" option. Specify "on" for the "GENSYM" option to create a Concept symbol for your block. Specify "off" for the "GENSYM" option to use your own symbol. Specify the ACTmap block's name as the "<block_name>" value.

3. **Invoke Concept.** Invoke the program where the ACTmap block is instantiated.

```
concept
```

4. **Instantiate the ACTmap block symbol in the design and connect the symbol pins to other design logic.** Once the symbol has been added, the logic block is fully instantiated in the design. Refer to "Simulation Using Verilog-XL" on page 45 for information regarding performing functional simulation and performing back annotated simulation. Refer to "Capturing the Design using Concept" on page 31 for information on generating an Actel netlist.

4

# Actel-Composer Design Considerations

When creating a schematic with Cadence Composer for Actel devices, there are some conventions that should be observed, as described in the following sections.

## Updating Composer Designs

If you have been using version 4.3.x of the Design Framework II software, you need to convert libraries, technology files, Analog Artist simulation CDF, timing views, custom SKILL code, and databases with parameterized cells (pcells), to Cadence's 4.4 formats.

From the Cadence Composer Command Interpreter Window (CIW), choose Tools--Conversion Tool Box to bring up the Conversion Tool Box, then click on one of the following buttons to start conversion, as required.

- **Read Me First** displays flowcharts of the conversion process.
- **Convert DFII-DM Libraries** brings up a form where you can specify the libraries to convert, designate the directories where they will go, and run the conversion.
- **Merge Display Resource Files** brings up a form that lets you merge the multiple display resource files (display.drf's) created during library conversion into a single file.
- **Convert Artist CDF Data** brings up a form that lets you convert Analog Artist simulation CDF.
- **Convert Timing Views** brings up an interface to two utilities you need to run, the TV2TLF translator and the tlfc compiler.
- **Check SKILL Code** brings up the SKILL Checker, a utility that checks SKILL code for conversion-related problems. Run this utility on your 4.3.x code and use the messages as a guide to rewriting the code. Some code (like SKILL-based pcells) requires other conversion steps.

Refer to your Cadence Composer documentation for more information.

## Capturing the Design Using Composer

You must schematically capture the design using the Composer and Actel libraries, import external blocks created, or use both methods. Use one of the Actel symbol family libraries.

Note: Properties assigned using Composer schematic capture tool are not preserved when the netlist is created.

### *To capture a design with Composer schematic capture tool:*

Note: For each design, do not use symbols from more than one Actel family library.

Set up the "cds.lib" file. Refer to "Setup" on page 13 for detailed information.

Once the library search path is set, the Composer Library Browser includes an Actel library.

When capturing the design, you must only use the Actel library macros. The design must contain I/O buffers that are instantiated at the design's top level. These buffers represent the pin-out of the Actel device.

Once you capture the design, create the design pins within Composer. Figure 4-1 illustrates I/O buffer examples and buffer connections to the pins.

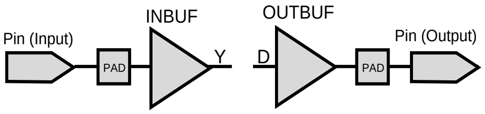

*Figure 4-1. I/O Buffers in Composer*

# Adding ACTgen Macros

With the ACTgen Macro Builder, you can create macros using a procedural description. You can generate netlists from the high-level language macros you create with ACTgen.

The ACTgen software generates a variety of critical structural macros such as, adders, subtracters, counters, comparators, and multiplexors in a shorter time than the typical HDL compilers. In addition, you can integrate ACTgen-based blocks with schematics and simulation tools such as Cadence simulation tools. To integrate an ACTgen-generated macro with schematic captures, you must create and instantiate a symbol in your design.

## Generating an ACTgen Macro

Refer to the *Designing with Actel* manual for information regarding using the ACTgen software.

## Integrating ACTgen with Composer

You can import a Verilog description of the ACTgen macro into Composer using the Cadence “Verilog In” program. The “Verilog In” utility creates a Composer symbol and schematic for the block.

1. **Generate a Verilog description of the ACTgen block.** Select “Output Format” from the “File” submenu of the ACTgen Macro Builder window. Specify Verilog for the Netlist/CAE Options. Select OK.
2. **Display the Verilog In screen.** Select the Import option from the file submenu of the “icds” program. Specify Verilog from the Import submenu.
3. **Specify the library search path.** Enter the directory paths to the Cadence basic and sample libraries, the Actel macro symbol library and the project library in the Library Search Path field.

   ```
   <alsdir>/lib/cds/<act_fam> /project /cds/dfII/etc/cdslib
   /cds/dfII/samples/cdslib
   ```

4. **Specify the library name.** Type the following in the Library Name field:

```
PROJECT
```

5. **Specify the reference libraries.** Enter the basic, sample and Actel macro symbol library names in the Reference Libraries field. The Actel library must be specified first.

```
<act_fam> basic sample
```

6. **Create a simulation command file.** The following is an example of the syntax for a simulation command file, “run.cmd:”

```
<macro_name>.v
-y <alsdir>/lib/vlog/<act_fam>
+libext+.v
```

7. **Specify the Verilog Options.** Type the following in the Verilog Options field:

```
-f run.cmd
```

8. **Specify the Ignore Modules File.** Type the following in the Ignore Modules File field:

```
<alsdir>/lib/vlog/<act_fam>/cells.lst
```

9. **Specify the Log File.** Type the following in the Log File field:

```
./verilogIn.log
```

10. **Select OK.** A schematic and a symbol will be generated for the block.

    Note: Verilog In text fields should be left blank that have not been previously specified.

11. **View the schematic or instantiate the symbol in a design.** Use the Composer Library Browser to view the schematic. Once the symbol has been added, the logic block is fully instantiated in the design.

## Adding ACTmap Blocks

The ACTmap VHDL Synthesis program can translate logic blocks described in ACTmap VHDL to either EDIF or ADL netlists. The output netlists are optimized to fit a particular ACT family architecture, and can be imported into Cadence Composer designs. To integrate a logic block optimized using ACTmap with schematic capture, you must create and instantiate a symbol in the design.

### Generating an ACTmap Logic Block

For information regarding the ACTmap VHDL language description or the ACTmap software use, refer to the *Designing With Actel* manual.

### Integrating ACTmap with Composer

You can import a Verilog description of the ACTmap logic block into Composer using the Cadence "Verilog In" program. The "Verilog In" utility creates a Composer symbol and schematic for the block.

1. **Generate a Verilog description of the block.** Use the ACTmap Translate screen to select a Verilog Netlist as the Output Format.
2. **Write out the Verilog netlist.** Select the Run button. ACTmap writes the verilog netlist in the same directory as the optimized EDIF netlist (.edo file) as "<block_name>.vlo."
3. **Display the Verilog In screen.** Select the Import option from the file submenu of the "icds" program. Specify Verilog from the Import submenu.
4. **Specify the library search path.** Enter the directory paths to the Cadence basic and sample libraries, the Actel macro symbol library and the project library in the Library Search Path field.

   ```
   <alsdir>/lib/cds/<actel_fam> /project/cds/dfII/etc/cdslib
   /cds/dfII/samples/cdslib
   ```

5. **Specify the library name.** Type the following in the Library Name field:

   ```
   PROJECT
   ```

6. **Specify the reference libraries.** Enter the basic, sample and Actel macro symbol library names in the Reference Libraries field. Use the proper Actel library family respectively for the Actel macro library specification. The Actel library must be specified first.

   ```
   <act_fam> basic sample
   ```

7. **Create a simulation command file.** The following is an example of the syntax for the simulation command file, “run.cmd:”

   ```
   <block_name>.vlo
   -y <alsdir>/lib/vlog/<act_fam>
   +libext+.v
   ```

8. **Specify the Verilog Options.** Type the following in the Verilog Options field:

   ```
   -f run.cmd
   ```

9. **Specify the Ignore Modules File.** Type the following in the Ignore Modules File field:

   ```
   <alsdir>/lib/vlog/<act_fam>/cells.lst
   ```

10. **Specify the Log File.** Type the following in the Log File field:

    ```
    ./verilogIn.log
    ```

11. **Select OK.** A schematic and a symbol will be generated for the block.

    Note: Verilog In text fields should be left blank that have not been previously specified.

12. **View the schematic or instantiate the symbol in a design.** Use the Composer Library Browser to view the schematic. Once the symbol has been added, the logic block is fully instantiated in the design.

## Creating the Verilog Source File from Composer

Generate a Verilog netlist to perform a functional simulation of the design as follows.

1. **Select simulation from the Tools menu and choose Verilog-XL.**
2. **Click OK on the Setup Environment window to accept the default.**
3. **On the Verilog-XL Integration Control window, select "Netlist" from the Setup menu.** Make sure that the netlisting options, "Netlist Explicitly" and "Generate Pin Map" are selected.
4. **Select "verilog" from the Stimulus menu.** This generates a gate-level description of the design.
5. **Perform a functional simulation of your design with Verilog-XL simulator.** Refer to your simulator documentation for additional information.

# 5 Simulation Using Verilog-XL

This section describes steps to perform Functional simulation (Behavioral and Structural), Timing simulation, and board-level simulation for Actel devices using Verilog-XL Verilog simulator:

## Functional Simulation

Use the following procedure to perform a functional simulation of your design with unit delays.

1. **Create or modify a stimulus file in the project directory.** Use this file to apply test vectors or patterns. It is important to add a timescale definition to the stimulus file as follows:

   ```
   ‘timescale 1 ns/100 ps
   ```

2. **Generate a simulation command file as follows:**

   ```
   <design_name>.sim
   <design_name>.v
   -y <alsdir>/lib/vlog/<act_fam>
   +libext+.v
   -a
   ```

3. **Run functional simulation.** Type:

   ```
   verilog -f <simulation_command_file>
   ```

## Timing Simulation

After you place and route the design using the Designer Series software, and after the design's postlayout delays have been extracted, you can perform timing simulation to evaluate the design's performance.

Follow the next steps to run backannotation simulation:

1. **Place and route your design in Designer.** Refer to the *Designing with Actel* manual for information about using Designer.
2. **Extract timing information for your design.** Chose the Extract command from the Tools menu or click the Extract button. The Extract dialog box is displayed. Create a (design_name>.sdf file by specifying SDF as the CAE type. Click OK.
3. **Add an SDF routine to the stimulus file.** You must add this routine to perform post-layout simulation. The "sdf" file generated by the "sdfgen" command contains derated post-layout delays. The following is the SDF routing syntax:

   ```
   $sdf_annotate("<design_name>.sdf", <instance_name>);
   ```

   The following is an example of the "$sdf_annotate" construct added to a test stimulus file name "run.cmd":

   ```
   'timescale 1ns/100ps
   module test;

   //Declare inputs and outputs
   wire...
   reg...

   //Instantiate the module TOP in the test module
   TOP instance1 (...Pin list...);

   //Stimulus patterns
   initial
   begin
   ...
   end

   //Invoke SDF routine to backannotate
   initial
   $sdf_annotate("top.sdf", instance1);
   endmodule
   ```

4. **Invoke a simulation session using the post-layout delays.** Once you add the “$sdf_annotate” construct to the stimulus file, invoke the Verilog-XL simulator using one of the following commands:

```
verilog -f <simulation_command_file> +mindelays
verilog -f <simulation_command_file> +typdelays
verilog -f <simulation_command_file> +maxdelays
```

   For this command, use your simulation command file’s name for the “<simulation_command_file>” value.

6

# Simulation Using Leapfrog

This chapter describes steps to perform Functional (Behavioral and Structural) and Timing simulation for Actel devices using the Cadence Leapfrog simulator.

You must compile Cadence Leapfrog VITAL libraries before performing simulation with Leapfrog. Refer to “Compiling Libraries for Cadence’s Leapfrog Simulator” on page 13 for information on compiling Leapfrog VITAL libraries.

## Simulation

This section describes steps to perform Functional and Timing simulation for Actel devices using the Synopsys, ACTmap, or other synthesis tools with Cadence Leapfrog simulator.

### Behavioral Simulation

After the VHDL descriptions of the logic blocks have been coded, test and debug the design using the Leapfrog simulator.

1. **Create a work directory and a cds.lib file in the project directory.** At the UNIX prompt, enter:

   ```
   mkdir work
   ```

   Using a text editor, create a cds.lib file and enter the following lines:

   ```
   INCLUDE $CDS/tools/leapfrog/files/cds.lib
   ```

2. **Map to the Actel VITAL and FPGA libraries.** If any Actel macros are instantiated in your VHDL source, add the following lines to your cds.lib file (2.3) to map them to the Actel VITAL and FPGA libraries in $ALSDIR.

   ```
   DEFINE <vhd_fam> $ALSDIR/lib/vtl/95/lfrog/<vhd_fam>
   DEFINE WORK ./work
   ```

   Add the following lines to your VHDL design files to reference the Actel Family library in your VHDL design files:

   ```
   library <vhd_fam>;
   use <vhd_fam>.components.all;
   ```

3. **Compile the VHDL design and testbench files.** Type:

   ```
   cv -work work -messages -file <vhdl_design_file>
   cv -work work -messages -file <vhdl_testbench_file>
   ```

   Note: If you are using ACTmap Asyl Packages for synthesis you must compile the Asyl packages first. For more information on Asyl packages refer to the *ACTmap VHDL Synthesis Methodology Guide*.

4. **Elaborate the design.** Type:

   ```
   ev -work work -messages <configuration_name>
   ```

   Where <configuration_name> is the name of the configuration in the test bench.

   For example:

   ```
   ev -work work -messages test_add_behave
   ```

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

   If you have instantiated Actel Library Cells in your behavioral VHDL, you must use the -COMPATABILTY switch when elaborating your design as shown below:

   ```
   ev -work work -messages -COMPATIBILITY <configuration_name>
   ```

5. **Perform a behavioral simulation of your design.** Type:

```
sv -work work <configuration_name>
```

Where <configuration_name> is the name of the configuration in the test bench.

For example:

```
sv - work work -batch -run test_add_behave
```

## Structural Simulation

You can generate a gate level VHDL from your EDIF netlist by either exporting it from Designer or by using the edn2vhdl program.

1. **Generate a structural VHDL netlist.**

   If you are using Synopsys or ACTmap, generate the structural VHDL netlist from these tools.

### *To generate a netlist using Edn2vhdl:*

Type:

```
edn2vhdl fam:<act_fam> <design_name>
```

### *To generate a netlist using Designer Series software:*

Select the Import Netlist command from the File menu of the Designer window. Specify EDIF as the Netlist Type, GENERIC as the Edif Flavor, and VHDL as the Naming Style. Use the Browse utility to search for the EDIF netlist of your design.

Select the Export command from the Designer File menu and select Netlist VHDL.

Note: The VHDL generated by both Designer and edn2vhdl will use std_logic for all ports. The bus ports will be in the same bit order as they appear in the EDIF netlist.

2. **Map the VITAL and the Actel FPGA libraries.** Create a cds.lib file in the project directory as follows:

```
INCLUDE $CDS/tools/leapfrog/files/cds.lib
DEFINE <vhd_fam> <alsdir>/lib/vtl/95/lfrog/<vhd_fam>
DEFINE WORK ./work
```

3. **Compile the VHDL design and testbench files.** Type:

```
cv -work work -messages -file <vhdl design file>
cv -work work -messages -file <vhdl testbench file>
```

4. **Elaborate the design in compatibility mode.** Type:

```
ev -work work -messages -compatibility <configuration_name>
```

   Where <configuration_name> is the name of your configuration that binds the test bench entity and architecture.

   For example:

```
ev -work work -messages -compatibility test_adder_structure
```

   In the above example, “test_adder_structure” is the name of configuration for the test bench.

5. **Perform a structural simulation of your design.** Type:

```
sv -work work -batch -run <configuration_name>
```

   Where <configuration_name> is name of the test bench configuration.

   For example:

```
sv -work work -batch -run test_adder_structure
```

   Here the name of the test bench configuration is test_adder_structure.

## Timing Simulation

This section describes timing simulation using Cadence Leapfrog simulator.

1. **Place and route your design in Designer.** Refer to the *Designing with Actel* manual for information about using Designer.
2. **Extract timing information for your design.** Chose the Extract command from the Tools menu or click the Extract button. The Extract dialog box is displayed. Create a (design_name>.sdf file by specifying SDF as the CAE type. Click OK.
3. **Map the VITAL and the Actel FPGA libraries.** If not already done for structural simulation, create a cds.lib file in the project directory as shown below:

```
INCLUDE $CDS/tools/leapfrog/files/cds.lib
DEFINE <vhd_fam> $ALSDIR/lib/vtl/95/lfrog/<vhd_fam>
DEFINE WORK ./work
```

4. **Compile the VHDL design and testbench files.** If not already done for structural simulation, type:

```
cv -work work -messages -file <vhdl design file>
cv -work work -messages -file <vhdl testbench file>
```

5. **Elaborate the design in compatibility mode.** Type:

```
ev -work work -messages -compatibility -bsdf ./<design
name>.sdf -bmtm [max|typ|min] -bscope <UUT>
<configuration_name>
```

Where <design_name> is name of top level entity, <UUT> is the instance of the top level entity in the test bench, and <configuration_name> is the name of your configuration that binds the test bench entity and architecture.

For example:

```
ev -work work -messages -compatibility -bsdf adder.sdf -bmtm
max -bscope dut test_adder_structure
```

In the above example, “adder” is the name of the top level entity, “test_adder_structure” is the name of configuration for the test bench and “dut” is instance of the top-level entity “adder” in the test bench.

6. **Perform a timing simulation of your design.** Type:

```
sv - work work -batch -run <configuration_name>
```

Where <configuration_name> is name of the test bench configuration.

For example:

```
sv -work work -batch -run test_adder_structure
```

Here the name of the test bench configuration is test_adder_structure.

7

# Integrating Synthesis Tools with Cadence

EDIF, ADL or Verilog netlists that are optimized to fit a particular ACT family architecture can be imported into Cadence Composer or Cadence Concept designs. To integrate a logic block optimized using a synthesis tool with schematic capture, you must create and instantiate a symbol in the design.

## Integrating Synthesized Blocks with Concept

You can import an ADL, EDIF or verilog description of the logic block into Concept using the “xtrn2concept” command. The “xtrn2concept” command creates a Concept symbol and netlist for the macro.

Note: Any assigned properties in a netlist that is imported into Concept will not be preserved.

1. **Generate a Concept symbol and netlist for the logic block.** Use the “xtrn2concept” command by typing the following:

   ```
   xtrn2concept FAM:<act_fam>
   [INFORMAT:{adl,edif,verilog}]
   [INFILE:<Input File>]
   [GENSYM:{on,off}]
   <block_name>
   ```

   Note: In this command, specify “adl,” “edif,” or “verilog” as the “INFORMAT” option. Use the file name containing the ACTmap block’s description as the “INFILE” option. Specify “on” for the “GENSYM” option to create a Concept symbol for your block. Specify “off” for the “GENSYM” option to use your own symbol. Specify the ACTmap block’s name as the “<block_name>” value.

2. **Invoke Concept.** Invoke the program where the ACTmap block is instantiated by typing:

   ```
   concept
   ```

3. **Instantiate the block symbol in the design and connect the symbol pins to other design logic.** Once the symbol has been added, the logic block is fully instantiated in the design.

## Integrating Synthesized Blocks with Composer

You can import a Verilog description of the logic block into Composer using the Cadence "Verilog In" program. The "Verilog In" utility creates a Composer symbol and schematic for the block.

Note: Any assigned properties in a netlist that is imported into Composer will not be preserved.

1. **Generate a Verilog description of the block.** Use the adl2vlog command to generate a Verilog netlist from an ADL netlist by typing the following:

   ```
   adl2vlog FAM:<act_fam>
   [ ADL:<Adl_file> ]
   [ VLGOUT:<Verilog_file> ]
   [ VNM:<VerilogNamesMap_file> ]
   [ SIMTEMP:<Template_file> ]
   [ USEADLNAME:T ]
   [ BUSFORMAT:^%s^%d ]
   <Design>
   ```

   Use the edn2vlog command to generate a Verilog netlist from an EDIF netlist by typing the following:

   ```
   edn2vlog  FAM:<act_fam>
   [ EDNIN:<Edif_File1>[+<Edif_File2...>] ]
   [ VLGOUT:<Verilog File> ]
   [ SIMTEMP:<stimulus_template_file>]
   <Design>
   ```

2. **Display the Verilog In screen.** Select the Import option from the file submenu of the "icds" program. Specify Verilog from the Import submenu.

3. **Specify the library search path.** Enter the directory paths to the Cadence basic and sample libraries, the Actel macro symbol library and the project library in the Library Search Path field.

   ```
   <alsdir>/lib/cds/<act_fam> /project /cds/dfII/etc/cdslib
   /cds/dfII/samples/cdslib
   ```

4. **Specify the library name.** Type the following in the Library Name field:

   ```
   PROJECT
   ```

5. **Specify the reference libraries.** Enter the basic, sample and Actel macro symbol library names in the Reference Libraries field. The Actel library must be specified first.

   ```
   <act_fam> basic sample
   ```

6. **Create a simulation command file .** The following is an example of the syntax for the simulation command file, "run.cmd:"

   ```
   <block_name>.vlo
   -y <alsdir>/lib/vlog/<act_fam>
   +libext+.v
   ```

7. **Specify the Verilog Options.** Type the following in the Verilog Options field:

   ```
   -f run.cmd
   ```

8. **Specify the Ignore Modules File.** Type the following in the Ignore Modules File field:

   ```
   <alsdir>/lib/vlog/<act_fam>/cells.lst
   ```

9. **Specify the Log File.** Type the following in the Log File field:

   ```
   ./verilogIn.log
   ```

10. **Select OK.** A schematic and a symbol will be generated for the block.

    Note: Verilog In text fields should be left blank that have not been previously specified.

11. **View the schematic or instantiate the symbol in a design.** Use the Composer Library Browser to view the schematic. Once the symbol has been added, the logic block is fully instantiated in the design.

## Integrating Synopsys Schematics into Concept

Users with Synopsys schematic files can use Actel's SIVA program to import the design into Concept.

1. **Run SIVA command by typing the following:**

   ```
   SIVA <design>
   ```

2. **This will produce a file called "<design>.edc."**
3. **Invoke the Cadence program "redifsch" by typing the following:**

   ```
   redifsch <design>.edc
   ```

   This will produce the Concept schematic files for the Synopsys Schematic design.

# 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: 410.381.3289
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