

**Project Navigator
User Manual**

981-0313-002

September 1994

090-0511-002

Data I/O has made every attempt to ensure that the information in this document is accurate and complete. Data I/O assumes no liability for errors, or for any incidental, consequential, indirect or special damages, including, without limitation, loss of use, loss or alteration of data, delays, or lost profits or savings, arising from the use of this document or the product which it accompanies.

No part of this document may be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose without written permission from Data I/O.

Data I/O Corporation
10525 Willows Road N.E., P.O. Box 97046
Redmond, Washington 98073-9746 USA
(206) 881-6444

Acknowledgments:

Data I/O is a registered trademark and Synario, Project Navigator, Device-Independent Library, and ABEL-HDL are trademarks of Data I/O Corporation.

Data I/O Corporation acknowledges the trademarks of other organizations for their respective products or services mentioned in this document.

© 1993-94 Data I/O Corporation
All rights reserved

Table of Contents

Preface

Customer Support	vii
Technical Assistance	vii
Sending a Fax	vii
Using the BBS	viii
Warranty Information	viii
End User Registration and Address Change	viii
Where to Look for Information	ix
Conventions	x

1. Welcome to Synario

What Is Synario?	1-1
Freedom in Design Entry	1-1
Freedom in Design Processing	1-1
Designing with Synario	1-2
Design Entry	1-3
Functional Simulation	1-3
Optimization and Device Fitting	1-3
PLD Design Flow	1-4
Timing Simulation	1-4
What's in Synario	1-5
Projects and Sources	1-5
Processes	1-5
Customizing Processes	1-6
Simulating Your Design	1-7

Synario and Windows	1-8
Flexible Design Entry	1-9
Selecting a Device	1-9
Synario File Structure	1-11
Learning More About Synario	1-12
Exploring Synario's Documentation	1-12
Exploring Example Designs	1-12

2. Building a Design

Introduction to Projects and Sources	2-1
Opening and Saving Projects	2-3
Opening a Project	2-3
Creating a New Project	2-3
Saving a Project	2-4
Naming Sources and Projects	2-4
Creating Schematics and Behavioral Modules	2-4
Adding and Removing Source	2-5
Building a Hierarchical Project	2-5
Understanding Hierarchy	2-5
Building a Top-level ABEL-HDL Module	2-7
Building a Top-level Schematic	2-7
Building a Top-level VHDL Module	2-7
Test Fixture Sources	2-8
Building Other Project Sources	2-8

3. Editing Your Design

Editing Behavioral Modules and Test Stimulus	3-1
Editing Schematics	3-2
Editing Other Sources	3-2
Converting JEDEC Files to ABEL-HDL	3-2

4. Processing Your Design

Running Processes	4-1
-----------------------------	-----

Viewing Reports and Output Files	4-2
Processing Steps	4-3
Displaying Processes	4-3
Working with Processes	4-3
The Process Window	4-3
Changing How a Process Runs with Properties	4-5
Processing Problems	4-5
Error Messages	4-5
Debugging Problems	4-5
5. Designing for Device Independence	
Schematic Design Techniques	5-1
VCC and GND Symbols	5-1
Net Names and Symbol Instance Names	5-1
Create Bus and Net Names that are Unique	5-2
I/O Pin Guidelines	5-2
Other Design Considerations	5-2
Naming Restrictions	5-2
Don't Rely On Case Sensitivity	5-3
Keep Projects Separate	5-3
Use Text Documents	5-3
Use a Schematic for the Top-level Source	5-3
Test Fixture Include file	5-3
Designing for Device Independence	5-4
Device-independent Designs	5-4
Device-independent ABEL-HDL Modules	5-4
Device-independent Schematics	5-6
6. Working with Devices	
When You Need to Select a Device	6-1
Selecting a Device	6-1
Automatic Process Updating	6-2
What Happens When You Select a Device?	6-2
How Processes Change with a Device Change	6-3

How Properties Change with a Device Change	6-3
How Schematic Symbols Change with a Device Change	6-3
Device Kits	6-5
What Is a Device Kit?	6-5
How Do You Use a Device Kit?	6-5
Accessing Device Kit Help Files	6-5

A. Strategies and Properties

Properties	A-2
Advanced Properties	A-3
Default Properties	A-3
Setting Properties	A-3
Strategies	A-4
Creating New Strategies	A-6
Changing the Default Strategy	A-6
Deleting and Renaming Strategies	A-7
Saving Strategies	A-7
Importing Strategies	A-7

B. Synario Configuration

Synario Project Navigator	B-1
Text Editor and Report Viewer	B-1
Schematic Editor, Symbol Editor, and Hierarchy Navigator	B-2
Synario Project Navigator Configurable Menu	B-2

Preface

The Preface includes details about obtaining technical assistance and warranty service. The Preface also explains the Bulletin Board Service, where to find information, and typographic conventions used in the manuals.

Customer Support

For technical assistance and warranty service, contact your local distributor. An updated list of Synario distributors is included in your Synario Design Entry package.

Technical Assistance

You can contact your Synario distributor for technical assistance by calling, sending a fax, or electronic mail (e-mail). You can also access the Data I/O Bulletin Board Service (BBS) for product information.

To help us give you quick and accurate assistance, please provide the following information:

- ◆ Product version number
- ◆ Product serial number (if available)
- ◆ Detailed description of the problem you are experiencing
- ◆ Error messages (if any)
- ◆ Device manufacturer and part number (if device-related)

When you call, please be at your programmer or computer, have the product manual nearby, and be ready to provide the information listed above.

Sending a Fax

Fax the information listed above with your name, phone number, and address to your Synario distributor listed on the Synario Distributors sheet.

Using the BBS

To reach Data I/O via the BBS, include your name, phone number, e-mail address, and the information listed above in a message, and send it to the BBS.

From the Data I/O Bulletin Board Service (BBS) you can obtain a wide range of information on Data I/O products, including current product descriptions, new revision information, technical support information, application notes, and other miscellaneous information.

Using the BBS, you can access device support information, request support for a particular device, and leave messages for the BBS system operator or other customers. The BBS also includes many downloadable DOS utilities.

Multiple lines are available, all supporting 1200/2400/9600/19200 baud, with U.S. Robotics Dual/HST V.32bis/V.42bis modems. The modems are set to 8 data bits, 1 stop bit, and no parity. Online help files provide more information about the BBS and its capabilities.

The BBS numbers in the United States are +1-206-882-1976 and +1-206-861-6959

Warranty Information

Data I/O Corporation warrants this product against defects in materials and workmanship at the time of delivery and thereafter for a period of ninety (90) days.

The foregoing warranty and the manufacturers' warranties, if any, are in lieu of all other warranties, expressed, implied or arising under law, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose.

End User Registration and Address Change

If the end user for this product or your address has changed since you registered the product, please notify your Synario distributor. This ensures that you receive information about product enhancements. Be sure to include the product serial number.

Where to Look for Information

If you want to learn about	Look here
Installation	<i>Getting Started</i>
Menu operation	Online Help
Quick Start	Synario Online Tutorial Online Help (F1)
Design tutorials	<i>Getting Started</i>
Device-specific information	Device Kit manual Device Kit Help
ABEL-HDL syntax	Online Help <i>Synario ABEL-HDL Reference</i>
Schematic commands	Online Help <i>Synario Schematic Editor Reference</i>
Simulation	<i>Synario Simulator User Manual</i>
Complex PLD designs	PLD Device Kit <i>Getting Started</i>
FPGA designs	FPGA Device Kit <i>Getting Started</i>

Conventions

The conventions used in this manual are described below:



This symbol indicates that the information is available in Synario's online help system.

Keys and Key Combinations

An instruction for pressing two keys at once, such as ^Z (Control and Z), is represented by two keys separated by a plus, such as Ctrl + Z.

A key combination like Esc, Ctrl + T means press and release Esc, then press Ctrl and T at the same time.

Variable Input

Variable input is italicized and should be replaced with the requested information. For example, "enter copy *filename.hex*" means type "copy" just as you see it and replace *filename.hex* with the name of your file.

Also, some terms are italicized the first time they appear.

Optional Input

Optional items of a command are shown in brackets.

[option1] [option2]...[optionn]

Items separated by a vertical bar (for example, OR | OR | ...) are mutually exclusive; that is, only one of the options listed can be specified.

Displayed Text

Text displayed on the screen appears in a typewriter-like typeface.

You will see this text displayed on the screen.

Menu Items, Processes and Properties

Menu items are entered with the menu name, a colon, and the menu selection. For example, File: Exit means to select Exit from the File menu.

Processes and properties are entered as they appear on the screen.



Chapter 1

Welcome to Synario

The Synario Universal FPGA Design System brings together several FPGA design tools into a single Windows application. Synario streamlines the device design process, freeing you to concentrate on the details of your design. This chapter explains what Synario is and how you approach device design with it.

If you'd like to start using Synario right away, *Getting Started* gives you information on installing and starting Synario, along with tutorials to get you started designing in Synario.

What Is Synario?

Freedom in Design Entry

Synario is a design entry system for FPGAs. Synario helps you create designs that are

- ◆ Composed of either schematics or HDL modules, so you can use the best features of both design entry methods.
- ◆ Hierarchical: you can create designs from the bottom up or top down.
- ◆ Device independent, so you can easily retarget your designs using Synario's generic behavioral language and generic schematic symbols.

Freedom in Design Processing

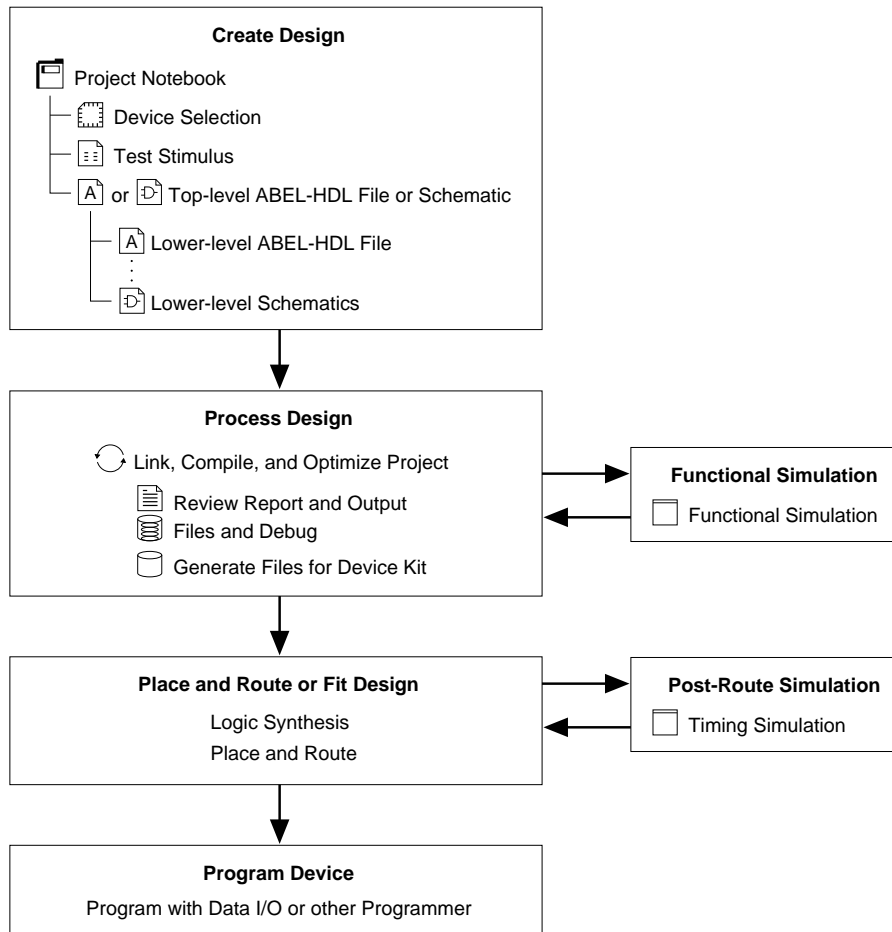
Synario also gives you freedom in processing your design. You can

- ◆ Design with or without selecting a device, and change the device at any time. Synario updates the processes for the new device.
- ◆ Develop and reuse different strategies for processing designs.
- ◆ Simulate both the function and timing of your design.
- ◆ Simulate your design before choosing a device, and after implementing in a specific device.
- ◆ Run any process, with Synario automatically running prerequisite processing.

Designing with Synario

Synario consists of many parts, all of which are accessed through the Synario Project Navigator. The Project Navigator helps you keep track of all of the parts of your design, and keeps track of the processing steps necessary to move the design from the conceptual stage through to implementation in an actual device.

Figure 1-1
Synario Design Flow



1993-1

Design Entry

Synario supports two primary design entry methods: schematic and HDL. You can enter your design using either of these entry methods, or mix schematic and behavior entry in a single design.

In addition to the schematics and HDL portions of your design, Synario allows you to enter and keep track of other files not directly associated with the logic of your design. These files could include things like word processor documents (for your project specification), simulation test fixtures, or graphic state diagrams. These accessory files are kept for you in the Synario Project Notebook. Virtually any type of file can be imported into Synario and kept in the Project Notebook.

Functional Simulation

Functional simulation is a way to verify that your design functions as intended before attempting to fit it into a device. The Synario Simulator is a full-featured simulator that accepts Verilog test stimulus and can display the simulation results in a variety of formats, including waveforms. For designs that include a top-level schematic, Synario also allows you to view simulation results right on the schematic, with actual circuit values displayed for each design input and output. This feature is called cross-probing.

Optimization and Device Fitting

Synario's optional device kits allow you to target your design to a wide range of programmable devices. The devices supported by Synario Device Kits range from simple PAL and PROM devices to complex PLDs and FPGAs.

The optimization and fitting steps that Synario performs for you are determined by the type of device that you choose for implementation. There are two broad categories of devices, however, that can be described in terms of design flows. These categories are FPGAs and PLDs.

FPGA Design Flow

FPGA and PLD device architectures require different design processing. An FPGA-type device differs from a PLD primarily in its need for a place-and-route step. Place-and-route software is typically provided by the FPGA vendors, so the processing steps performed in the Synario environment include the use of tools that may not be integrated directly into the Synario Project Navigator.

The FPGA design flow also differs from the PLD design flow in the types of entry formats supported. Because FPGA device fitters (and place-and-route software) operate on netlist format data, you can use both schematics and behavioral (HDL) design entry formats. When you enter a design that consists of a combination of schematics and HDL components, Synario converts all of the input forms to a netlist representation before the design is moved to the device-fitting phase. For most FPGA type devices, the individual project sources (the schematics and HDL files) are combined (merged) during device fitting.

PLD Design Flow

The design flow for PLDs varies depending on the type of PLD chosen. For most types of PLDs, the design flow includes compilation of schematic or HDL sources into the OPEN-ABEL format, and a linking step for designs that are composed of multiple source files. Unlike FPGA design, multiple source files are combined (by linking) prior to the device-fitting process.

The PLD design flow also includes the capability for another form of simulation that is specific to PLDs: test vectors. You can use test vectors to test the function of the design before device-specific synthesis has been performed, and after the JEDEC file has been created. Use the Equation simulator to simulate the design after the equations are compiled and linked. Use the JEDEC simulator to simulate the program-and-test process. JEDEC simulation is supported in Synario for most devices that are programmed using a JEDEC format programmer download file. See your device kit documentation for support information. See the *Equation and JEDEC Simulators User Manual* for more information on the Equation and JEDEC Simulators.

Timing Simulation

After your design is successfully implemented in a programmable device, you can verify the timing performance of the resulting circuit. Timing simulation allows you to verify that your design will operate correctly at the speeds that you will be using in your target system.

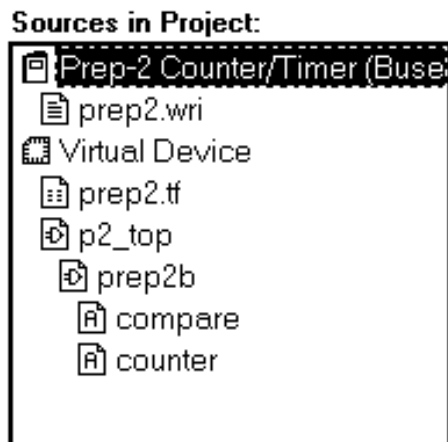
To accurately simulate your design with timing, Synario builds a timing model of the programmed device using timing data. Once you have this simulation model, you can use the Synario Simulator to check the operation of the design and timing-related circuit problems.

What's in Synario

Projects and Sources

Synario organizes a design by collecting all of the files into the Project Notebook. The Project Notebook lists the schematics and behavioral sources that create the logic of your design, testing files, and the device specification. The Project Notebook can also include any other design documents you want to keep with the design, such as design specifications, meeting notes or other supplementary files. Synario calls the notebook, documentation, and design pieces the *sources* of the project.

Figure 1-2
Sources in Project Window



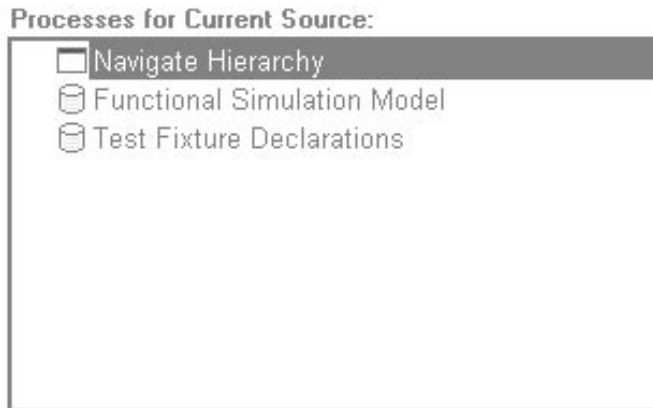
Each project is stored in its own directory to simplify archiving.

Processes

A Synario project may comprise many pieces, each of which needs to be handled in a different way. You might have schematics that must be drawn with a schematic capture program, tested, and then translated into netlists or other formats. You might also have behavioral modules that need to be compiled, optimized, tested, and translated into another format, and test fixtures that are used without processing. The manner in which each piece of a Synario project is processed is also dependent on the device selected.

All of the steps required to take your design from specification into a device are called *processes* in Synario, and Synario remembers them all for you. When you select a source or change the target device, Synario automatically displays the steps to get the job done, freeing you to concentrate on your design. All you have to do is double-click on a process to run it. Synario's Auto-make capability automatically runs any prerequisite processes before running the process you select.

Figure 1-3
Processes for Current Source Window



Customizing Processes

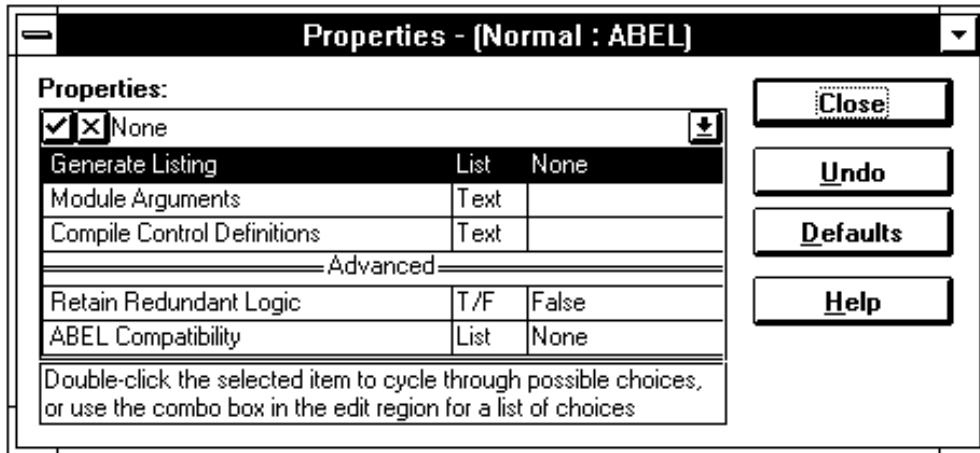
Although Synario processes are usually already optimized for the device and source selected, there may be times when you want to change the way a source is processed. You can do this by changing the options, called *properties*, of a process in the Properties dialog box. You can save multiple sets of processing properties, each into their own *strategy*. For more information on Strategies and Properties, see Appendix A.

See Also



Specific processes and properties for each source and device are available in Synario's online help system.

Figure 1-4
Properties Dialog Box



Simulating Your Design

The Verilog Simulator is an option that provides functional simulation for all designs, and timing simulation for devices that support simulation. Simulation is a process for a test stimulus source.

Designs targeted to most types of PLDs use ABEL-HDL test vectors to simulate the JEDEC programmer load file.

See Also

If you have the Verilog Simulator, see the *Verilog Simulator User Manual* for information on how to simulate your design.

For simulation support information, see your device kit documentation.

For Equation or JEDEC simulation, see the *Equation and JEDEC Simulators User Manual*.

Synario and Windows

Synario is a Windows program and uses standard Windows operations. If you are familiar with how a Windows application works, you already know a lot about how Synario works. Some of the ways you can use your Windows experience in Synario are described below:

Drag and Drop from the File Manager

You can drag and drop any file from the Windows' File Manager into a Synario project as a project source. Any source that Synario does not recognize as part of a device design is stored under the Project Notebook as a document. This function is equivalent to choosing Import from the Source menu in the Synario Project Navigator.

Double-click Sources to Edit

The files in a Synario project are shown in the Sources in Project window. The Sources window displays the Notebook name, device selected, and the other sources of the project. Double-click on the device selection to select a new device. Double-click on any source to run the appropriate program with the specified file loaded — the same way you can in the File Manager.

Note: Synario sets up a File Manager association for `.syn` during installation. You may need to set up file extension associations for other files. For instructions, see your Windows documentation.

Note: You may also need to enable Use File Associations from the Options: Environment dialog box.

Double-click Processes to Run Them

The steps to process the source highlighted in the Sources window are shown in the Processes for Current Source window. Double-click on a process in the Processes window to run that process. Synario automatically runs any prerequisite processes to complete the process you select.

Double-click on Reports to View Them

The reports generated by Synario processes are also listed in the Process window. You can double-click to view them, or highlight them and click the View button.

Shift+Double-click to Force One Process

If you hold down the shift key while double-clicking on a process, it will force that process to run, even if it is up-to-date.

Ctrl+Double-click to Force All Processes

If you hold down the shift key while double-clicking on a process, it will force that process and all prerequisite processes to run, even if they are up-to-date.

Double-click in the File Manager to Start Synario

Double-click on a Synario project file (*.syn) in the File Manager to start Synario with that project loaded.

Press F1 for Help

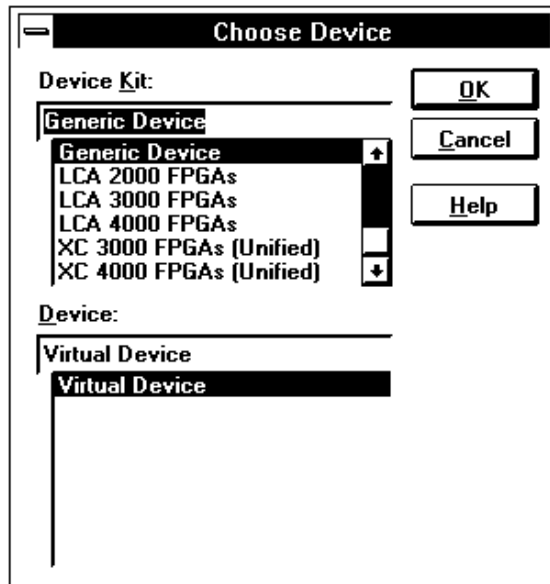
Synario comes with an extensive online help system. Press F1 to access context-sensitive help.

Flexible Design Entry

To take advantage of Synario's ability to change devices, enter your design so that it does not require a particular device; this type of design is called device independent. Synario's Schematic Editor and HDL language both have device-independent features:

Design Type	Device-independent Features
Schematic	Use the Synario Device-independent Library to enter your schematics. These symbols are supported by all FPGAs and are mapped to the device-specific symbols before place-and-route. See the <i>SCS Command Reference</i> for more information.
HDL	Use ABEL-HDL's device-independent language features. See the <i>ABEL-HDL Reference</i> for more information. See the <i>VHDL User Manual</i> for more information on VHDL.

Figure 1-5
Selecting a Device



Selecting a Device

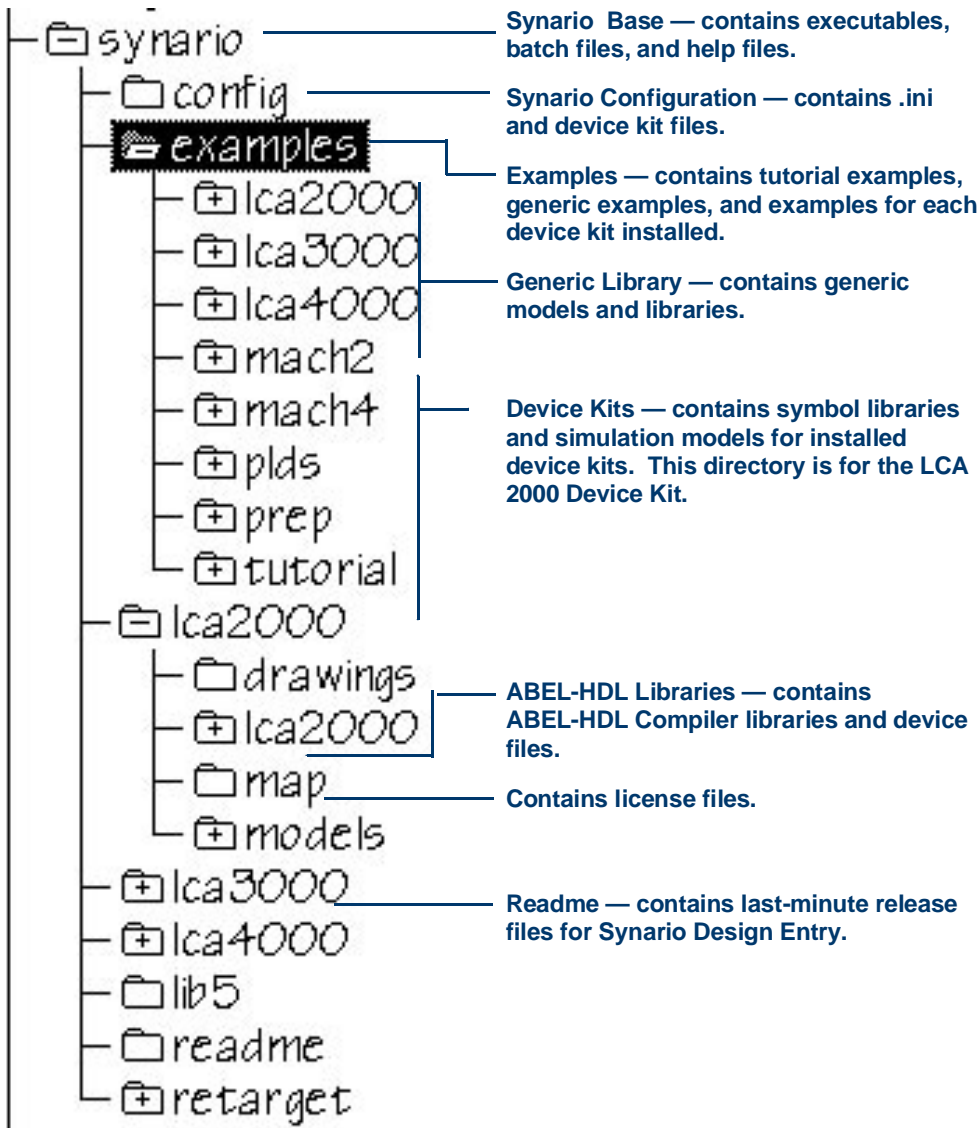
When you do want to select a device, Synario allows you to become increasingly more specific about the device you want to use. You will need to select (target) a device to do device-specific design processing, such as logic synthesis and place-and-route.

Device-specific functions are provided through device kits. You should have received one or more device kits with your Synario package. Synario device kits are available for FPGAs, such as LCAs and MAX devices. There are also device kits for PLDs and complex PLDs.

Synario File Structure

The files installed by Synario Design Entry are shown below.

Figure 1-6
Synario File Structure



Learning More About Synario

Exploring Synario's Documentation

The following table shows where to find things in the Synario documentation. Most information is also available in Synario's online help system.

To Learn About	See
Design Entry Topics:	
Device-independent designs	Chapter 5, "Designing for Device Independence" <i>ABEL-HDL Reference</i> <i>SCS Reference</i>
Device-independent Library	<i>SCS Reference</i>
Design Building Topics:	
Creating, importing, and editing sources	Chapter 3, "Building Your Design"
Design Processing Topics:	
Selecting a device	Chapter 6, "Working with Devices"
Processing your design	Chapter 4, "Processing Your Design"
Strategies and properties	Appendix A, "Strategies and Properties"

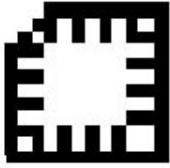
Exploring Example Designs

Synario comes with many example designs (called projects) that get you started creating your own designs. When you install a device kit, the installation program adds examples for the supported device.

To load an example design:

Select Open Example from the Synario Project Navigator File menu.

OR Double-click the *filename.syn* file in the Windows File Manager.



Chapter 2

Building a Design







This chapter introduces projects and sources, and gives instructions on how to build a design with them. For examples of building designs with Synario projects, see your *Getting Started* guide.








Introduction to Projects and Sources

In Synario, a design that will be targeted to a device is referred to as a project. A project contains sources that

- ◆ Identify the project and list included sources
- ◆ Define the design's logic
- ◆ Define the target device
- ◆ Provide test stimulus to test the design

Projects are built from the following types of sources. The type of source is indicated by the icon to the left of the source name in the Source Window.

Source	Icon	Example
Project notebook		hiermult.syn
Device family		lca.fdk
ABEL-HDL logic description		hierabel.abl
Structural logic description		hierschm.sch
VHDL logic description		hiervhdl.vhd
Verilog test fixtures		hierstim.tf

Source	Icon	Example
VHDL test bench		hiertb.vhd
ABEL-HDL test vectors		hiertv.abv (or .abl)
Waveform stimulus		waves.wdl
Undefined or incorrect source reference		
Schematic source that failed Update Hierarchy process		
ABEL-HDL source that failed Update Hierarchy process		
VHDL source that needs to be compiled		

See Also




Projects

- “Opening and Saving Projects”
- “Adding Source”
- “Adding Non-Synario Project Files”

Opening and Saving Projects

Opening a Project

To open an existing Synario project, select Open from the Synario File menu. Each Synario project is stored in its own directory. Example projects are installed in subdirectories under the Synario EXAMPLES directory.


- OR Double-click on a Synario project file, *filename.syn* in the Windows File Manager.
- OR Drag a Synario project file, *filename.syn* into Synario.
- OR Click on the Open Project button  in the toolbar.


Creating a New Project

To create a new Synario project:

1. Select New from the File menu.

OR

Click the New Project button  in the toolbar.

2. Select a directory for the project, or select the Create Directory button and enter a name for a new directory. All of the files in your project will be stored in this directory.
3. Enter a filename for the project file.
4. Double-click on the project notebook icon  in the Source window, and enter a title for the project. The project title can be as long as you like, but only the first 20 characters will show. The title can contain spaces and any other keyboard character except tabs and returns. The title is stored in the project file and is included in project output and report files created during processing.


Hint: To rename the project, double-click the project icon  in the Source window.

Note: *For some device kits, the project directory and project name should be the same.*

Saving a Project

To save a project:

Select Save or Save As from the File menu. If you select Save As, Synario asks for a filename to save the project to.

OR Click on the Save button  in the toolbar.

What is Saved

Saving a project saves a project file (.syn extension) with the following information:

- ◆ The title of the project
- ◆ The sources in the project
- ◆ The strategy associated with each source (.sty extension)

Synario also tells the schematic and text editors to save when you save a project.

When you select Save As to save a project to another directory, Synario copies all of the project files to that directory.

Naming Sources and Projects

Use the following guidelines when naming source files and projects:

- ◆ Avoid using ABEL-HDL, VHDL or Verilog keywords for module and signal names in any of your source files.
- ◆ Do not use any device-kit-specific macro functions to name a source.

Creating Schematics and Behavioral Modules

The design description (logic) for a project is contained within the following types of source:

- ◆ Schematics
- ◆ ABEL-HDL
- ◆ VHDL

One source file in a project is the top-level source for the design. The top-level source defines the inputs and outputs that will be mapped into the device, and references the logic descriptions contained in lower-level sources. The referencing of another source is called “instantiation.” Lower-level sources can also instantiate sources to build as many levels of logic as necessary to describe your design.

Note: *If you build a project with a single source, that source is automatically the top-level source.*

See Also

“Building a Hierarchical Project”
" Selecting a Device"

Adding and Removing Source

To add source to a project:

Select New from the Source menu, or click on the New button underneath the Sources in Project window.

OR Select Import from the Source menu.

OR Drag and drop a file from the Windows File Manager.

The new source is entered into the Source window in alphabetical order for each level of hierarchy following the project notebook and device entries.

To remove source from a project:

1. Select (highlight) the source in the Source window.
2. Select Remove from the Source window.

Note: *Removing a source from a project does not delete the underlying file.*

Building a Hierarchical Project

Understanding Hierarchy

Hierarchical FPGA and PLD design consists of a top-level source that contains functional blocks linked to create the overall design. The functional blocks referenced in the top-level source are placeholders for lower-level logic descriptions, which may in turn contain functional blocks that reference even lower-level logic descriptions. The referencing of a logic description is called “instantiation.”

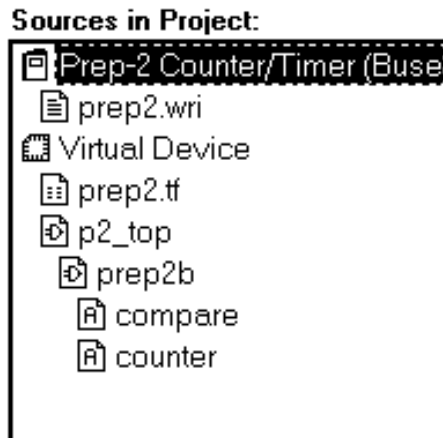
Both top-level and lower-level source can be either schematics or behavioral modules (in ABEL-HDL).

Note: *A source can be referenced (“instantiated”) more than once. Also, a source can be both a lower-level and top-level source. For example, "compare" in Figure 2-1 could instantiate another file.*

Figure 2-1 shows what a hierarchical design looks like in the Synario Sources in Project list.

Note: You can change to a "flat" hierarchy view (that shows only the top file) by removing the check on View: Hierarchy.

Figure 2-1
The Sources in Project Window



Prep-2 Counter Timer	The Project Notebook, which keeps track of all of the files that make up your project.
Virtual Device	The Device selected for the project (in this case, no device has been selected).
prep2.tf	A test fixture used to simulate the design implemented into the device.
p2_top	The top-level source for the project, which instantiates the lower-level (indented) sources. In this case, it is a schematic.
prep2b, compare, counter	Lower-level ABEL-HDL modules and schematic.

Note: You cannot instantiate a top-level source from a source instantiated below that source. For example, you can't instantiate "top" from the "compare" source.

Building a Top-level ABEL-HDL Module

In a top-level behavioral module in ABEL-HDL, you use the `Interface` and `Functional_block` keywords to instantiate lower-level files. You can also use the `Interface` keyword in lower-level files to link to upper-level ABEL-HDL modules (not upper-level schematics).

The ABEL-HDL file `pwmdac.abl` demonstrates the use of the `Functional_block` and `Interface` keywords in a top-level file. The file `counter.abl` demonstrates the use of the `Interface` keyword in a lower-level file.

See Also

“Interface (upper-level),” “Interface (lower-level),” and “Functional_block” in the *ABEL-HDL Reference*

Building a Top-level Schematic

In a top-level schematic (unless your design contains only one logic source), you use symbols to instantiate lower-level files. Select `Add: New Symbol` from the Schematic Editor menus to create a functional block symbol for a lower-level file.

If you are in a lower-level schematic, you can choose “This Block” in the “New Symbol” dialog box to automatically create a functional block symbol for the current schematic.

The Block Name is the name of the lower-level file, which can be another schematic or a behavioral module.

Note: *Some device-specific tools require that you not use busses in top-level schematics.*

See Also

SCS Reference

Building a Top-level VHDL Module

In a top-level VHDL file, you use component declarations to instantiate lower-level files.

The `arith.vhd` project (a tutorial in the *VHDL User Manual*) demonstrates the use of the component declaration.

Note: *Some device-specific tools require that you not use busses in top-level VHDL source.*

See Also

VHDL User Manual

Test Fixture Sources

If you create or import a test fixture file, Synario prompts you whether to associate the test fixture with the device source or HDL/Schematic source.

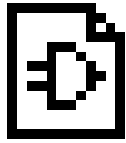
The project shown in Figure 2-1 includes two test fixtures (.tf extension). One associated with the device (time.tf) and one associated with the top-level source (func.tf).

You can double-click on a test fixture to edit it in the Text Editor.

Building Other Project Sources

You might have sources other than schematic and ABEL-HDL modules. These sources might include simulation test fixtures, documentation files, or other files related to Windows applications.

In most cases, you can drag and drop these files into your Synario project as needed. Any sources that are not part of the design logic description or simulation test fixtures are displayed under the Project Notebook.



Chapter 3

Editing Your Design

This chapter briefly explains the editors that come with Synario to help you start editing the sources that make up your logic design. More detailed information is provided in online help and the following manuals:

To Edit	Look Here
Behavioral Modules	<i>ABEL-HDL Reference</i>
Schematics	<i>SCS Reference</i>
Verilog Test Stimulus	<i>Verilog Simulator User Manual</i> (if you have this option)
VHDL	<i>VHDL User Manual</i> (if you have this option)

You can edit any of the sources that make up your project by double-clicking on them (if you have file associations set up in the Windows File Manager and have enabled "Use File Associations" in "Options: Environment").



In the editor for a source, you can press F1 or use the Help menus to access editor-specific help.

Editing Behavioral Modules and Test Stimulus

The Synario Text Editor provides several macros and templates to help you enter and edit test stimulus and behavioral modules written in ABEL-HDL and VHDL. You can edit an ABEL-HDL, VHDL, or test stimulus source by double-clicking on it, or by selecting it in the source list and choosing Source: Open.

You can also use any ASCII editor to edit behavioral modules and test stimulus. You then import them into your Synario project using drag-and-drop from the Windows File Manager or using Source: Import.

See Also



"Menus" for information on Synario Text Editor menus and using templates.
ABEL-HDL Language Reference for information on ABEL-HDL
Verilog Simulator User Manual
VHDL User Manual

Editing Schematics

Use the Synario Schematic Editor to edit schematic source. You can open the Schematic Editor on a schematic by double-clicking on the schematic name in the source list, or by selecting the source and choosing Source: Open.

See Also

The *SCS Reference* for information on creating schematics and on the Generic Library.

Your device kit documentation for device-specific symbol libraries and generic symbol to device symbol mapping charts.

Editing Other Sources

Other sources can be edited by double-clicking if you associate their file extensions with appropriate editing applications in the Windows File Manager. See your application documentation for editing information.

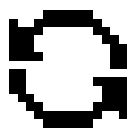
Note You can associate text files with the Synario Text Editor by choosing *File: Associate* from the Windows File Manager. See your Windows documentation for more information on file association in the File Manager.

Converting JEDEC Files to ABEL-HDL

If you have JEDEC files that you want to convert to ABEL-HDL files to use in a Synario project, you can use `jed2ahdl` in the Synario DOS window. You can bring up the Synario DOS window by pressing Ctrl+Alt+D.

The options for `jed2ahdl` are listed below:

```
jed2ahdl infile -o outfile -report mapfile
```

Chapter 4

Processing Your Design

This chapter explains how to process a project, including

- ◆ Running Processes
- ◆ Viewing Reports and Output Files
- ◆ Understanding Processing Steps
- ◆ Summary of Processes for Sources and Projects

See Also



Online help for Processes and Properties
Online tutorial

Chapter 2, “Building a Design”

Chapter 3, “Working with Sources”

Device kit documentation

Verilog Simulator User Manual

Equation and JEDEC Simulators User Manual

Running Processes

You can run a process with one of the following procedures:

Double-click on a Process:

1. Select the source you want to process. (To process the entire project, highlight the device.)
2. Double-click the end process you want. Synario automatically updates any intermediate steps necessary to complete the required process.

Note: *Shift+double-click forces one level to run; Ctrl+double-click forces all processes to run.*

OR Use the Start or View Buttons:

1. Select the source you want to process. (To process the entire project, highlight the device.)
2. Select the process you want.
3. Click on the Start button to run the process; click on the View button to run the process and view the resulting file (for report and viewable output files).

OR Use the Process Menu

1. Select the source you want to process. (To process the entire project, highlight the device.)
2. Select the process you want.
3. From the Process Menu (keyboard shortcut is Ctrl+P), select Start to run the process, or View to run the process and view the resulting file (for report and viewable output files). You can also select Force to run a process again that has already completed successfully.

See Also

“What Happens When You Select a Device?” in Chapter 6

Viewing Reports and Output Files

To view a report or output file:

1. Highlight the desired file.
2. Click on the View button or choose View from the Process menu.
3. When the process completes successfully, the File Viewer displays the file.

To view the log for the last process:

Click on the Log button or select Log from the Process menu.

Processing Steps

Displaying Processes

Processes are associated with sources, so the processes displayed are only those processes for the highlighted source. Table 4-1 shows which source some common processes are associated with.

Table 4-1

Source Associations for Processes

Type of Process	Source to Highlight
Compiling	Behavioral
Simulation	Test Fixtures
Generate Netlist	Schematic

Working with Processes

This section contains the following topics:






- ◆ The Process Window
- ◆ Changing How Processes Run with Properties

The Process Window

Each step required to process a project and its source is referred to as a Process. The processes available for the project and each type of source are different and vary based on the device selected. When you select (highlight) the project name or one of the sources in the Source Window, the processes available are displayed in the Process Window.




Types of Processes

The types of Processes available are shown below.

Icon	Process Type
	Run batch program
	Run Windows program
	Create Report File and View it
	Create Viewable Output File
	Create Non-viewable Output File

To see the status of a Process:

Look to the left of the process type icon for the process status icons:

Icon	Means process completed...
(No icon)	(Not started)
	Successfully (without errors or warnings)
	With warnings (subsequent processes can continue)
	With fatal errors (processing halts completely)

Changing How a Process Runs with Properties

You can change the way a process runs by changing the options (properties) of the process.

To change the properties for a process:

1. Select (highlight) the process in the Process window.
2. Click on the Properties button.

OR Select Properties from the Process menu.

The Property Editor displays the available properties for the selected process.

1. Double-click on a property to edit it. Changes take effect immediately and are saved to the current strategy.

To change properties for a different process or source:

Without closing the Property Editor, select the desired source and process. The Property Editor automatically updates the properties.

See Also

Appendix A, "Strategies and Properties"

Processing Problems

Error Messages

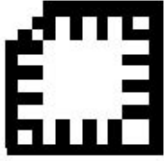


Explanations for error messages are available from online help by selecting Messages from the Problem Solving category on the help map.

Explanations for Device Kit error messages are available in the help file shipped with the device kit.

Debugging Problems

If you are having problems with Synario programs running under syndos, you can debug them by running a DOS window with the syndos.pif file. To access this window, press Ctrl+Alt+D. This gives you a DOS window with the same environment. Type **exit** to quit.



Chapter 5

Designing for Device Independence

This chapter covers project design techniques and designing for device independence. This information on conventions for designing with Synario will help you produce designs that are

- ◆ Easier to debug
- ◆ Easy to move to other device families
- ◆ Compatible with Synario designs created by other Synario users

Schematic Design Techniques

VCC and GND Symbols

The Synario Schematic Editor does not have VCC or GND symbols. Instead, you create connections to ground and VCC by naming the net "VCC" or "GND." Nets with those names are tied to the corresponding global signal. By creating a vertical net with the name "VCC" attached at the top, the VCC bar symbol appears. Likewise, by creating a vertical net with the name "GND" attached at the bottom, the ground symbol appears.

Net Names and Symbol Instance Names

During simulation, you may want to probe the design for certain nets or symbols. To improve post-synthesis debugging, you should name all of your intermediate nets with unique names, rather than having Synario provide automatic net names (which can be cryptic and difficult to sort out).

Create Bus and Net Names that are Unique

Bus names and net names, if identical, will "short out" in Verilog simulation because of the way Verilog breaks out bus signals. For example:

```
DATA[ 7:0 ]
```

and

```
DATA
```

will cross-connect during simulation and cause unpredictable errors. Avoid using buses and nets names that are identical.

I/O Pin Guidelines

- ◆ **Place I/O Pins on Top-level Only** — Keep I/O symbols (and pin assignments) in the top level source only. This makes designs much easier to simulate and understand later, as all of the device I/O can be found in one place, the top level source.
- ◆ **Avoid Using Ordered Buses** — on I/O pins for a top-level schematic, for example, data[7:0]. Because most fitters flatten buses, using ordered buses causes the Verilog test fixture created for functional simulation to be unusable for post-route simulation.
- ◆ **Use Alphanumeric Characters for Signal Names** — The first character should always be alphabetic.
- ◆ Use Global Net Names Vcc and GND only to tie unused inputs high or low.
- ◆ Do not end signal names with #_ (for example 3_).

Other Design Considerations

Naming Restrictions

You cannot assign an ABEL-HDL module name or VHDL entity name that is longer than 8 characters to a sub-module you want to instantiate in a schematic. The matching symbol name is limited by DOS filename restrictions.

Don't Rely On Case Sensitivity

Names should not rely on case to provide uniqueness. For example, don't use "FRED" and "fred" in the same design. Some tools in the Synario environment are inherently case sensitive, though many are not case sensitive. Duplicate names with different cases may internally "short" together during some back-end process, resulting in an invalid programming file and/or incorrect simulation results. Some processes also modify the case of signals and names.

Keep Projects Separate

Keep projects separate in their own directories to make them easier to archive, easier to move, and easier for other engineers to work on.

Use Text Documents

Place design notes in separate text files associated with your project rather than documenting directly in the schematics and ABEL-HDL modules to make the logic easier to follow.

Use a Schematic for the Top-level Source

Using a schematic for the top-level source allows you to take advantage of the design navigation and cross-probing features of the Hierarchy Navigator.

Test Fixture Include file

Use a Verilog ``include` statement to include the automatically-generated test fixture declarations (.tfx) in your test fixture (.tf) files to minimize re-working the test fixture files.

See Also

Verilog Simulator User Manual

Designing for Device Independence

This section describes how to create device-independent designs that can be easily moved to different devices. Designing for device independence has many advantages. Device-independent designs mean you can:

- ◆ Take advantage of new devices as they become available with minimal maintenance.
- ◆ Learn one behavioral language and one symbol library for all devices, instead of learning a new one for each device you use.
- ◆ Try a design in several architectures to determine the ones in which your design fits best or runs fastest.
- ◆ Not be committed to the architecture you start your design with.
- ◆ Re-use your Synario designs in whole or in part.
- ◆ Draw your schematic without committing to a device.
- ◆ Perform functional simulation before device-specific features are specified.

You can map a device-independent design into a similar device, or link together several smaller designs targeted for small devices into a larger device.

Device-independent Designs

Device-independent ABEL-HDL Modules

Most of the ABEL-HDL language is device independent (for example, equations, truth tables, state machine descriptions, and test vectors). However, the ABEL-HDL compiler needs some information about how signals function in logic descriptions.

Information needed	Provided in ABEL-HDL with
Signal specifications	Istype attributes in the signal declarations and dot extensions on signals in the logic description
Special functions	Property Statements

Most common signal functions can still be described independent of a device with the pin-to-pin signal declarations. Some special features available in only certain classes of devices require device-specific (“detailed”) signal declarations.

Conditional Symbols

Some devices also have special functions that can be accessed through conditionally-compiled syntax. Each device kit has one or more special compile-time symbols that can be used to create conditionally-compiled designs. The table below shows a sample of the conditional symbols for various device kits. Refer to your device kit for more information on the conditional symbols supported.

Device Kit	Conditional Symbols
LCA2000	<code>_LCA2000_</code> , <code>_LCA_</code>
LCA3000	<code>_LCA3000_</code> , <code>_LCA_</code>
LCA4000	<code>_LCA4000_</code> , <code>_LCA_</code>
PLDs	<code>_PLD_</code>
MAX5000	<code>_MAXPLUS2_</code> , <code>_MAX5000_</code>
MAX7000	<code>_MAXPLUS2_</code> , <code>_MAX7000_</code>
MACH2	<code>_MACH2_</code> , <code>_MACH_</code>
MACH4	<code>_MACH4_</code> , <code>_MACH_</code>

An example of how to use conditional symbols in your ABEL-HDL source file is shown below:

```
@IFDEF _LCA_
{
  <LCA macros>
}

@IFDEF _PLD_
{
  <PLD macros>
}
```

Pin-to-pin Signal Specifications

Using ABEL-HDL's pin-to-pin syntax allows you to create logic descriptions that can be migrated to different devices. Use the pin-to-pin syntax whenever you have a choice to allow the module to be migrated with little or no modification.

Detailed Signal Specifications

Detailed signal specifications provide you with access to unique features found only in certain device families.

Note: You can combine pin-to-pin and detailed signal declarations. In most cases, the ABEL-HDL compiler can reconcile the requirements of both types. In some cases, however, the circuit function may be ambiguous and the compiler displays an error message.

Syntax descriptions indicate whether the syntax is pin-to-pin or detailed.

See Also

“Pin-to-pin Vs. Detailed Module Descriptions” in the *Synario ABEL-HDL Language Reference*

Your device kit for device-specific ABEL-HDL property statements and dot extension restrictions

Device-independent Schematics

Synario Device-independent Library

Synario’s Device-independent Library gives you device-independent flexibility for your schematics. If you use device-specific symbols in a schematic, you may have to redraw the schematic if you later migrate your design to a different device.

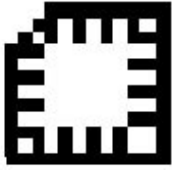
Device Kit Symbol Libraries

The symbol libraries that come with each device kit are device specific. If you plan to stay in one device family, these libraries are all you need. However, since device kit symbols are tied tightly to the device they describe, migrating a project with a device-specific schematic may require redrawing the schematic using the new target device’s symbol library. For this reason, we recommend the Synario Device-independent Library if you plan to use more than one type of device.

See Also

“Device-independent ABEL-HDL Modules”

Your device kit for device symbol libraries and Synario Device-independent Library mapping



Chapter 6

Working with Devices

There are many ways you can make it easier to map an existing design into a different device. These methods are discussed in Chapter 5, “Designing for Device Independence.”

When You Need to Select a Device

With Synario, you do not have to select a device to begin processing a design. You do need to select a device before you place and route your design or perform timing simulation (if you have the Synario Simulator).

You can, of course, specify a device at earlier stages in the design process.

Selecting a Device

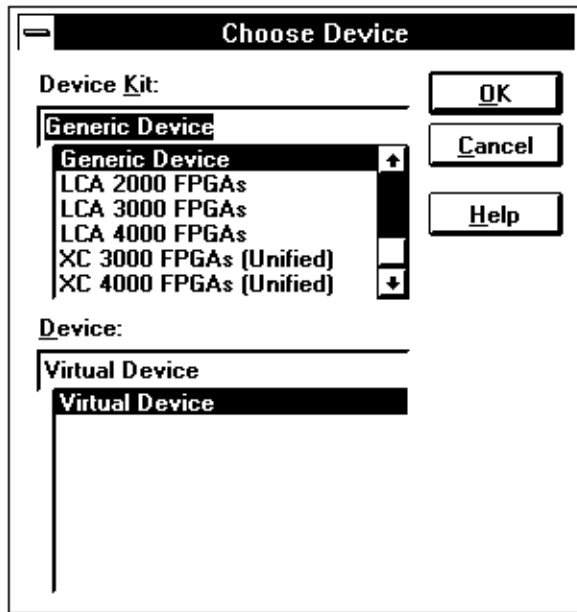
To choose any device, you must have a Synario Device Kit and the device support for the device installed.

You select the device for your project in the Choose Device dialog box. To access it, double-click on the Device icon in the Sources window, or select the Device icon and choose Open from the Source menu. The Choose Device dialog box is shown in Figure 6-1.

Device Kit	Lists the titles of the device kits currently licensed for this Synario installation. When you pick a device kit from this list, the Device box below is updated with the devices supported by that kit.
Device	Lists the supported device names licensed by the device kit. You can pick a device from this list, or enter a device name. If you enter a name, it must match a device in the listing (case doesn't have to match).

Note: *If the current project has sources that are not supported by the selected device, you will get a message box. Either select a different device or remove the unsupported sources from the project. (For example, standard PLDs do not support schematics, so selecting a P22V10 for a project that includes a schematic will cause this to happen.)*

Figure 6-1
Choose Device Dialog Box



If your device change involves changing the underlying design environment (which is done automatically), Synario asks you to confirm the change.

Automatic Process Updating

You do not have to remember how to process your designs for different devices: when you specify or change the device for your design, Synario re-configures the processing steps automatically to reflect the processes required for the selected device.

What Happens When You Select a Device?

Synario customizes itself for the currently-selected device. Changing the device to a device in a different device kit directly affects which processes, properties, and symbol libraries are available.

More detailed information about what is changed is given in the following sections.

How Processes Change with a Device Change

When you change to a device in a different device kit, the processing for the project changes. These changes are reflected in the following changes:

- ◆ The available processes change, especially device-specific process steps and netlists.
- ◆ The status of some processes that remain may return to “not processed” (that is, the green checks are removed and the process must be re-run for the new device).
- ◆ The properties available for processes.

How Properties Change with a Device Change

When you change to a device in a different device kit, the properties available for each Process change in the following ways:

- ◆ Properties are preserved for processes that did not change from the previous device kit (like the ABEL-HDL process “Compile Logic”).
- ◆ Properties for new processes, and new properties for existing processes have default values assigned.
- ◆ For processes that are not available with the new device, the properties are no longer used or visible, but the settings are saved (if you switch back to the previous device kit, these properties will be the same).

How Schematic Symbols Change with a Device Change

When you change to a device in a different device kit, the schematic symbols that are available change. Each device kit supports the Synario Generic Library and the symbol library for the devices in the kit. This means that after changing kits, Add: Symbol in the schematic editor lists the available symbols for the device kit.

CAUTION:

- ◆ You need to close and re-open the Schematic Editor and Hierarchy Navigator for the symbol availability changes to take effect.
- ◆ Schematics drawn with symbols from only the Synario Generic Library work without modification.
- ◆ Schematics drawn with symbols from a device-specific symbol library that is no longer available will be missing those symbols (you will get “Missing Symbol File” messages).

Note: *If you use the Synario Generic Library to draw all of your schematics, you'll rarely have to redo a schematic when you select a different device.*

Replacing Missing Symbols in a Migrated Schematic

To modify a schematic that is missing symbols after you've changed the device kit being used, do one of the following:

- ◆ Edit the schematic and select a different symbol from the new device kit library or the Synario Generic Library.
- OR*
- ◆ Create a new symbol and possibly an underlying schematic to support the new device kit.

For example:

Assume your original schematic was targeted to the MAX/FLEX device kit, and you used a symbol from the MAX/FLEX library. You then switch to the LCA3000 device kit, and this symbol is not supported. You get an error message "Missing symbol file *name.sym*" when you try to edit the schematic.

You can create a local symbol called *name.sym*, then draw a schematic called *name.sch* using symbols from the LCA3000 symbol library or the Synario Generic Library to provide the functionality of the MAX/FLEX symbol.

Device Kits

What Is a Device Kit?

A device kit contains all of the tools you need to create designs for that device or family of devices. So if you purchase the LCA device kit, you receive tools and documentation to help you create designs for LCAs. Device kits can contain all or some of the following items, depending on the processing requirements for the devices supported:

- ◆ Device Synthesis applications
- ◆ Place-and-route applications
- ◆ Functional and Timing Simulation models (for simulation with the optional Synario Simulator)
- ◆ Device-specific Synario project examples
- ◆ Device-specific design help files
- ◆ Device-specific symbol libraries
- ◆ Device optimization and pin assignment software

How Do You Use a Device Kit?

To use your device kit:

1. Install the device kit software.
2. In your Synario project, double-click on the device icon and choose a device supported by the device kit.

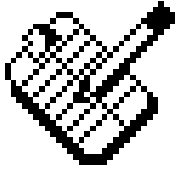
The installation process installs device kit files that Synario uses to update the processes available to you.

Accessing Device Kit Help Files

To access the separate help file installed with your device kit:

Choose the device kit help file from the Synario Project Navigator Help menu. This brings up the help file for the currently-selected device kit.

OR In Synario's main help file, click on the Device Kit button and choose the device kit help you want.



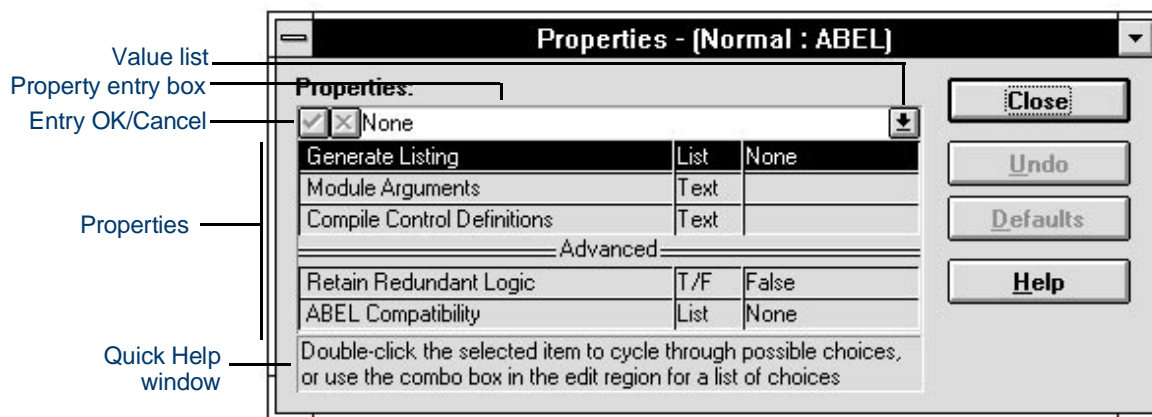
Appendix A

Strategies and Properties

Many processes have options, called properties, that tell Synario how you want the process carried out. You can modify these properties in the Properties dialog box.

To use the Properties dialog box, select a process in the Process list that you want to modify. If the process has properties, the Properties... button below the Process list will be active. Click on this button to open the Properties dialog box and edit the properties for the currently selected process.

Figure A-1
Properties Dialog Box



The title of the Properties dialog box window displays the name of the currently selected Strategy (Normal) and the type of source selected (Design). If you modify the properties shown above, you are modifying the properties in the Normal style, for Standard PLDs, for the currently-selected Design-level process.

Properties

There are four basic property types: True/False, Text, Num, and List .

True/False True/False properties have "T/F" in the second column of the Properties dialog box. To change the value of a True/False property:

Double-click on the property.

OR Select the property and then press the value-list drop-down button to select True of False.

OR Press T or F when a True/False property is selected.

You can undo a change by pressing the Undo button.

Text Text properties take text strings defined by that property. To modify text properties

1. Select the text property in the property list. The current value of the text property (if any) is displayed in the property entry box (see Figure 1-4).
2. In the edit region, type in the text value for the property.
3. Click on the check (OK) button next to the edit region to accept the change. Click on the X (Cancel) button to cancel changes.

Num Numeric properties take integer numbers. To modify numeric properties:

1. Select the numeric property in the property list. The current value of the numeric property (if any) is displayed in the property entry box (see Figure 1-4).
2. In the edit region, type in an integer value for the property.
3. Click on the check (OK) button next to the edit region to accept the change. Click on the X (Cancel) button to cancel changes.

List List properties allow you to select one value from a list of possible choices. To change the value:

1. Select the List property.
2. Click on the value-list drop-down button and select one of the values.

OR Double-click on the List property. Each double-click causes the List property to change to the next possible value in the list.

Advanced Properties

Some properties in the Properties dialog box are advanced. Be sure to review the online help for these properties before changing them.

Default Properties

You can reset properties to the defaults for your device kit by clicking the Defaults button.

Setting Properties

To set properties:

1. Select a source (for example, select the device to change project properties, or select a logic source (schematic or ABEL-HDL module) to change source properties).
2. Select a process from the Process window.
3. Click on the Properties button, or choose Properties from the Process menu. The Properties dialog box displays the standard and device-specific properties for that process.
4. Click on a property to select it.
5. Double-click on an option to toggle between choices. Click once to select an option and type in a new value, then select the green check to accept the change or the red X to undo.

If you change your mind, click on the Undo button.

6. To change the properties for another process, click on the desired source and/or process in the Synario main window. (You may need to move the Properties dialog box.)

The properties for the selected process are immediately displayed in the Properties dialog box window.

Strategies

Strategies are advanced features of Synario that allow you to set up different processing options for different parts of a project. For example, you can use strategies to use one set of synthesis options for part of your project, and another set for the rest.

A strategy contains all of the property settings required to process the sources (such as an ABEL-HDL module) in the Sources in Project list, and a strategy can be assigned to one or more of the following types of source in a project:



Design



Schematics



ABEL-HDL modules



VHDL sources

You can create multiple strategies and associate them with different sources in your design, allowing you to use different property settings for each source in your project.

When you associate a project source with a strategy, the source extracts its property settings from the strategy. For example, if you associate an ABEL-HDL source with a strategy named “Speed,” the ABEL-HDL module uses the ABEL-HDL process property settings in the strategy “Speed,” and ignores the property settings for the strategy “Normal.”

A Strategy is a collection of all the properties for all the processes. For example, you could set up a "Normal" strategy that has all of the ABEL-HDL Compile Logic and Reduce Logic properties set to optimize a module, and create a "No Optimize" strategy that has the Compile Logic and Reduce Logic properties set to not optimize the logic in a module (perhaps for timing purposes). You would then associate the "No Optimize" strategy with the sources that you do not wish to be optimized. Figure A-2 shows a graphical representation of a strategy.

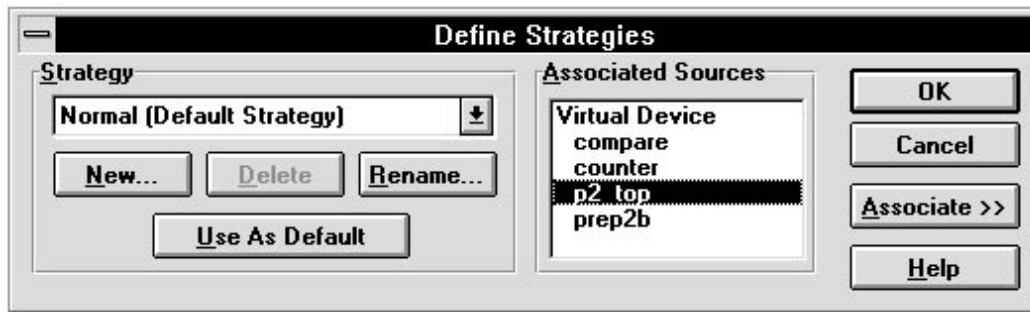
Figure A-2
Synario Strategies

Processes	Properties
<u>ABEL-HDL</u>	Generate Listing:None
Compile Logic	Retain Redundancy: False
Reduce Logic	Reduction Algorithm:By-Pin
XNF Netlist	Optimization Goal:Area
...	...
<u>Schematic</u>	
EDIF Netlist	Compress EDIF:False
...	...
<u>Device</u>	
Fit Design	Pin Preassignments:Keep
Create Fuse Map	Program Turbo Bits:False
Simulate JEDEC	Trace Type: Pins
Export to XACT	Back Annotation:XNFBA
...	...

Creating New Strategies

A project initially contains only the “Normal” strategy. You can create additional strategies and assign them to the different sources in your design. To create additional strategies, select Strategy from the Source menu to bring up the Define Strategies dialog box.

Figure A-3
Define Strategies Dialog Box



To create a new Strategy,

1. Click on the New button.
2. Type in a name for the new strategy.
3. Select an existing strategy to copy the new strategy’s properties from (or select System Defaults to get the default properties).
4. Click on OK (or press ENTER).

Changing the Default Strategy

The “Normal” Strategy is automatically created when you start a new project in Synario. Normal is the default strategy and is assigned to all sources when they are added or imported into a project.

You can change the default strategy for a project to one of the strategies you have created.

To set a new default strategy:

1. Select the Strategy from the Source menu to bring up the Define Strategies dialog box.
2. Select the strategy that you want to be the new default from the Strategy list box
3. Click on the Use As Default button.

The strategy you selected will now be assigned to all new and imported sources in your project.

Deleting and Renaming Strategies

To delete or rename a strategy:

1. Select Strategy from the Source menu to bring up the Define Strategies dialog box.
2. Select a strategy.
3. Click on the Delete or Rename... button.
4. If you are renaming the strategy, enter a new name and click on the OK button.

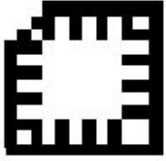
Note: *If you delete the Default strategy, the strategy at the top of the strategy list becomes the new Default Strategy. See “Changing the Default Strategy” if you want to change it.*

Saving Strategies

Strategies are automatically saved when you save your project.

Importing Strategies

Strategies are saved in a .sty file. You can import a set of strategies from another design by dragging and dropping the .sty file in the Windows File Manager, or by selecting Source: Import.



Appendix B

Synario Configuration

You can change the Synario environment by changing settings in different configuration files. You access the configuration files through the Options menu in the program or through the Project Navigator Options menu.

Tool	How to modify	Configuration File(s)
Synario Project Navigator (projnav.exe)	Options: Environment/Fonts	WINDIR\projnav.ini
Synario Text Editor (synedit.exe)	Options menu	WINDIR\synedit.ini
Synario Text Viewer (synview.exe)	Options: Environment/Fonts	WINDIR\synview.ini
Schematic Tools (projnav.exe)	Options: Schematic from the Project Navigator	\synario\config*.ini

Synario Project Navigator

You can set many environment variables in the Synario Project Navigator by choosing Options: Environment. The online help for the Environment dialog box discusses these options and the choices you have for changing them.

Text Editor and Report Viewer

You can select your preferences for the file modes, screen font, and customized key mappings from the Options menu in these programs. The online help discusses these options and the choices you have for changing them.

Schematic Editor, Symbol Editor, and Hierarchy Navigator

The Synario Schematic tools use special configuration files. There is a master INI file used for all schematics that is overlaid by device-specific INI files that add or override the behavior in the master INI file. A device-specific INI file is included with each Device Kit you install. You can change the settings in these files with the Synario INI Editor, including setting system colors and adding custom menus. All configuration files have an .ini extension and are installed in the Synario configuration directory (c:\synario\config is the default installation directory).

To edit the master INI file for all schematics:

Select Options: All Schematic...

To run the INI Editor with the currently-selected device kit INI file loaded:

Select Options: *device* Schematic in the Synario Project Navigator.

See Also



Online help in the INI Editor
"The SCS INI Editor" in the *SCS User Manual*

Synario Project Navigator Configurable Menu

You can add a menu to the Synario Project Navigator to access other Windows programs. This user-configurable menu is controlled by an INI-format file named synmenu.cfg installed in the synario\config directory. If synmenu.cfg is not found, only the standard Project Navigator menus are displayed.

The menus are specified first by listing the menu names in the [ProjNav Menus] section of SYNMENU.CFG, and then by listing menu items for each menu in separate sections of SYNMENU.CFG.

The basic format of the file is

```
[ProjNav Menus]
menu_name1=section_name1
menu_name2= . . .

[section_name1]
item_text1=program,valid_extension,command_line,message_bar_text
item_text2=. . .

[section_name2]
. . .
```

Note: *If you add an ampersand (&) in the text of the Menu Name or Item Text, the character following the ampersand is underlined, and the menu (or menu item) is available through the keyboard shortcuts (Alt+letter).*

The Message Bar Text should follow the format used by the fixed Synario menus.

Menu Name

menu_name

The Menu Name is the text that is displayed on the menu bar of the Synario Project Navigator between the Options and Windows menus. For example, if you enter

```
[&Tool]
```

a menu named "Tool" is inserted, accessed by Alt+T (underlined letter).

Section Name

section_name

The section name associates the menu_name with a section in SYNMENU.CFG where each menu item is found.

Menu Item

item_text

The Menu Item is the text for each menu item under the menu. Spaces are allowed.

Use & before a letter to allow keyboard access to the menu items (remember to give them unique letters).

Use a minus sign (-) at the beginning of the item_text line to place a separator line before that menu item.

For example, if you enter

```
-C&AD=program, valid_extension, command_line, message_bar_text
```

a menu item named CAD is inserted below a separator line, accessed with Alt+A (the underlined letter).

Program *program*

The Program is the name of the program that is run when the menu item is selected. It may contain a complete path specification if the program is not in your PATH environment variable search string (in your autoexec.bat file).

To use the Synario path:

Use the escape sequence \$\$ to substitute the path of the Synario executable.

For example, if projnav.exe is running in c:\synario, the program name \$\$\syndos.exe is translated to c:\synario\syndos.exe.

Extensions *valid_extension*

This field specifies which file extensions are supported by the program, and enables the menu when a source with a supported extension is selected

For reference, the Synario source extensions are shown below:

Source	Extension
Project Notebook	SYN
Device	SYN
ABEL-HDL Source	ABL
Schematic	SCH
Test Fixture	TF
VHDL Source	VHD
Test Vector Description	ABV
VHDL Test Bench	VHD
All Files	*

You can selectively enable the menu item for documents (which appear under the Project Notebook) based on their extension. For example, if your project contains both Word documents (.DOC) and Xilinx memory files (.MEM), a menu item with a *valid_extension* of DOC is enabled only for Word documents and not Xilinx memory files.

CAUTION: *The valid_extension field differs from the configurable menus in the Schematic Editor, which expect *. in front of the extension and a blank field instead of a single * for the All Files options.*

Command Line Options

command_line

The Command Line field specifies filenames and switches that are passed to and run with the program name. The command line can use escape sequences to customize the switches based on the filename extension of the selected source. The substitution escape sequences are as follows, and the examples are based on a selected source with the full path of C:\SYNARIO\TEST\TAZ.ABL:

Sequence	Meaning	Example Result
\$\$	Escape for a single \$	\$
\$B	Base name	TAZ
\$D	Drive and directory	C:\SYNARIO\TEST
\$E	Extension only, including period (.)	.ABL
\$F	Base name plus extension	TAZ.ABL
\$N	Complete name	C:\SYNARIO\TEST\TAZ.ABL
\$R	Drive + directory + base name	C:\SYNARIO\TEST\TAZ
\$Z	Ignored (required for blank argument field)	
%F	Full path name for the FDK file	C:\SYNARIO\CONFIG\LCA3000.FDK
%FN	Name of the FDK file	LCA3000

Note: *The \$Z parameter is required if the program has no arguments.*

Note that the following escape sequences are equivalent:

\$F is equivalent to \$B\$E
 \$R is equivalent to \$D\\$B
 \$N is equivalent to \$D\\$B\$E

Message Bar Text

message_bar_text

The Message Bar Text field is an optional field for text to be displayed in the message bar when the user selects the menu item. If no text is specified, a default message of "User-defined menu item" appears on the message bar. The message text for the menu (for example, Tools) is always "User-defined menu" and cannot be changed.

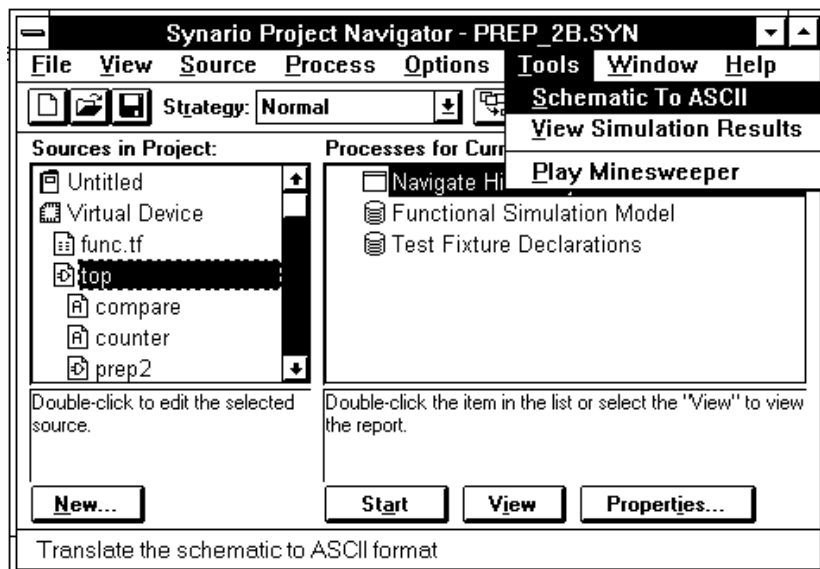
Note: *The Synario Schematic Editor configurable menus do not support the Message Bar Text field.*

Examples The following is an example of a menu configuration file.

```
[ProjNav Menus]
%Tool=ProjNav Tool
[ProjNav Tool]
&Schematic To ASCII=ascout.exe,sch,$F,Translate the schematic to ASCII format
&View Simulation Results=e:\foo\dir\waves.exe,tf,-nav $F,View the simulation waveforms
-&Play Minesweeper=winmine.exe,*,$Z,Hope the boss doesn't catch you
```

These lines would produce a Tool menu similar to that shown in Figure B-1.

Figure B-1
User-defined Tool Menu



Index

A

- ABEL-HDL modules 2-2
 - conditionally-compiled source 5-5
 - device-independent 5-4
 - editing 3-1
 - See also* Synario ABEL-HDL Reference
- Address changes viii
- Advanced properties A-3

B

- Behavioral source 2-2
- Bulletin Board Service viii

C

- Case-sensitivity 5-3
- Changing your address viii
- Configuration B-1

D

- Data I/O
 - Bulletin Board Service viii
- Debug window 4-5
- Default properties A-3
- Design entry 1-3
- Design flow 1-2, 4-3
 - for FPGAs 1-3
 - for PLDs 1-4
- Designs
 - See* projects
- Device kits 6-5
 - help files for 6-5
- Device-independent design 1-9, 5-1, 5-4
 - ABEL-HDL modules 5-4
 - schematics 5-6
- Devices
 - device family 2-2
 - how processes change 6-3
 - how properties change 6-3
 - how schematic symbols change 6-3
 - selecting 6-1
 - what happens when you change 6-2
 - when to select 6-1
- Directory structure 1-11

E

- Editing
 - ABEL-HDL modules 3-1
 - other sources 3-2
 - schematics 3-2
- End user registration viii
- Environment settings B-1
- Error log
 - viewing 4-2
- Errors 4-5
- Examples, loading 1-12

F

- File structure 1-11
- Fitting 1-3
- Flexible design entry 1-9
- Flow 1-2
- Functional simulation 1-3

G

- Generic Library 5-6
- Getting Started 1-1
 - See also Synario Getting Started manual
- Green check 4-4

H

- Help
 - on device kits 6-5
- Hierarchy
 - building a hierarchical project 2-5
 - top-level behavioral module 2-7
 - top-level schematic 2-7
- Hierarchy Navigator
 - setting preferences for B-2

I

- Instantiation 2-5
- Introduction 1-1

J

- JEDEC files, converting 3-2

L

- List properties A-2
- Logic description
 - contained in sources 2-4

M

- Menus
 - user-configurable B-2
- Messages 4-5
- Missing symbols 6-3 - 6-4

N

- Notebook
 - definition 1-5

O

- Optimization 1-3
- Options
 - See properties
- Other sources
 - editing 3-2
- Output files
 - viewing 4-2

P

- Post-route Simulation 1-4
- Problems 4-5
- Processes
 - automatic updating 6-2
 - changing how they run 4-5
 - customizing 1-6
 - definition 1-5
 - problems 4-5
 - processes window 4-3
 - running 4-1
 - status of 4-4
 - steps 4-3
 - types of 4-4
 - which source to highlight 4-3
- Project Notebook 2-2
- Projects
 - creating new 2-3
 - definition 1-5
 - introduction to 2-1
 - opening 2-3
 - saving 2-4
 - what is saved 2-4
- Properties 4-5
 - advanced A-3
 - default A-3
 - dialog box A-1
 - introduction A-1
 - setting A-3
 - types of A-2

R

- Red X 4-4
- Registration viii
- Report files
 - viewing 4-2
- Report Viewer
 - setting preferences for B-1

S

- Schematic Editor
 - setting preferences for B-2
- Schematics 2-2
 - design techniques 5-1
 - editing 3-2
 - See also* Synario Schematic Editor Reference
- Setting properties A-3
- Simulation 1-3 - 1-4, 1-7, 2-8
 - See also* Synario Simulator User Manual
- Sources
 - adding and removing 2-5
 - definition 1-5
 - editing 3-1
 - introduction to 2-1
 - other 2-8
 - top-level behavioral module 2-7
 - top-level schematic 2-7
- Start button 4-2
- Stimulus files
 - See* test fixtures
- Strategies
 - changing the default A-6
 - creating new A-6
 - deleting and renaming A-7
 - description A-4
 - introduction A-1
 - saving A-7
- Symbol Editor
 - setting preferences for B-2
- Symbol libraries
 - device kit libraries 5-6
 - Retargeting Library 5-6
- syndos problems 4-5

T

- Technical assistance vii
- Test fixtures 2-2, 2-8
 - See also* Synario Simulator User Manual
- Text Editor
 - setting preferences for B-1
- Text properties A-2
- Timing Simulation 1-4
- Tools menu B-2
- True/False properties A-2
- Typographic conventions x

V

Verilog
 See test fixtures
View button 4-2

W

Warranty
 information and service viii
Windows 1-8

Y

Yellow exclamation point 4-4