FPGA *Express*

Geting Started

Version 3.4, March 2000

Comments?
E-mail your comments about Synopsys
documentation to doc@synopsys.com

SYNOPSYS®

Copyright Notice and Proprietary Information

Right to Copy Documentation

Destination Control Statement

All technical data contained in this publication is subject to the export control laws of the United States of America. Disclosure to nationals of other countries contrary to United States law is prohibited. It is the reader's responsibility to determine the applicable regulations and to comply with them.

Disclaimer

SYNOPSYS, INC., AND ITS LICENSORS MAKE NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS MATERIAL, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

Registered Trademarks

Trademarks

Service Marks

Printed in the U.S.A.

FPGA *Express* Getting Started, Version 3.4

Contents

Figures

Tables

About This Manual

This Getting Started guide provides information for users of FPGA *Express*. The guide explains how to apply the program's basic features to your chosen design flow and programmable logic device (PLD) architecture. For information about advanced features not covered in this manual, see the online help or man pages for individual commands.

This guide supports FPGA *Express* version 3.4.

This preface includes the following sections:

- Audience
- Related Publications
- FPGA *Express* Online Help
- FPGA *Express* Man Pages
- SOLV-IT! Online Help
- Customer Support
- Conventions

Audience

This guide is for logic designers or engineers who use the FPGA *Express* synthesis tool to implement PLD designs.

Related Publications

Related publications include:

- *FPGA* Express *Installation Guide*
- *FPGA Compiler II /* Express *VHDL Reference Manual (online)*
- *FPGA Compiler II /* Express *Verilog HDL Reference Manual (online)*

All are delivered with the program in PDF (portable document format) files.

The program's Help menu also provides access to a selection of documents provided by Altera.

FPGA *Express* Online Help

FPGA *Express* includes these forms of online help:

- Topic-based
- Context-sensitive

To access the topic-based online help system, choose Help Topics from the Help menu. You can then access specific help topics through the table of contents, the alphabetic index, or the text-search feature. The help system also includes a glossary of FPGA *Express* terminology.

When working in FPGA *Express*, you can display a context-sensitive description of a specific feature. Click the question mark button on the product's tool bar or press Shift-F1, and then click an object on the screen.

FPGA *Express* Man Pages

The FPGA *Express* man pages contain the most up-to-date command usage and syntax information. To access the man pages, enter `man`, followed by the command of interest, in the fe_shell. For example,

```
fe_shell > man script_chip
```

SOLV-IT! Online Help

SOLV-IT! is the Synopsys electronic knowledge base, which contains information about Synopsys and its tools and is updated daily. For more information about SOLV-IT!, do one of the following:

- Go to the Synopsys Web page at
  http://www.synopsys.com
  and click SolvNET.
- Send an e-mail message to
  solvitfb@synopsys.com

Customer Support

For technical support, Altera provides several points of contact regarding FPGA *Express*.

- Send e-mail to Altera Applications at sos@altera.com.
- Call Altera Applications at 1-800-800-EPLD (United States and Canada).

As an FPGA *Express* customer, you have access to the Altera World Wide Web site at http://www.altera.com. The web site offers technical and general information, including

- Automated assistance
- Altera product descriptions and announcements
- Descriptions of Altera training and support services
- The Altera Technical Support (AtlasSM) database

Conventions

The following conventions are used in Synopsys documentation.

<table>
<tr><th>Convention</th><th>Description</th></tr>
<tr><td><code>Courier</code></td><td>Indicates command syntax.
In command syntax and examples, shows system prompts, text from files, error messages, and reports printed by the system.</td></tr>
<tr><td><em>italic</em></td><td>Indicates a user specification, such as <code><em>object_name</em></code></td></tr>
<tr><td><strong>bold</strong></td><td>In interactive dialogs, indicates user input (text you type).</td></tr>
<tr><td>[ ]</td><td>Denotes optional parameters, such as <code><em>pin1</em> [<em>pin2 ... pinN</em>]</code></td></tr>
<tr><td>|</td><td>Indicates a choice among alternatives, such as <code>low | medium | high</code>
(This example indicates that you can enter one of three possible values for an option: low, medium, or high.)</td></tr>
<tr><td>_</td><td>Connects terms that are read as a single term by the system, such as <code>set_annotated_delay</code></td></tr>
<tr><td>Control-c</td><td>Indicates a keyboard combination, such as holding down the Control key and pressing c.</td></tr>
<tr><td>\</td><td>Indicates a continuation of a command line.</td></tr>
<tr><td>/</td><td>Indicates levels of directory structure.</td></tr>
<tr><td>Edit &gt; Copy</td><td>Indicates a path to a menu command, such as opening the Edit menu and choosing Copy.</td></tr>
</table>

1

FPGA *Express*

FPGA *Express*, your Synopsys programmable logic device (PLD) logic-synthesis solution, brings a powerful combination of Synopsys logic-synthesis and optimization technology, high-level design methodology, and an easy-to-use user interface to the PLD design desktop.

This chapter includes the following sections:

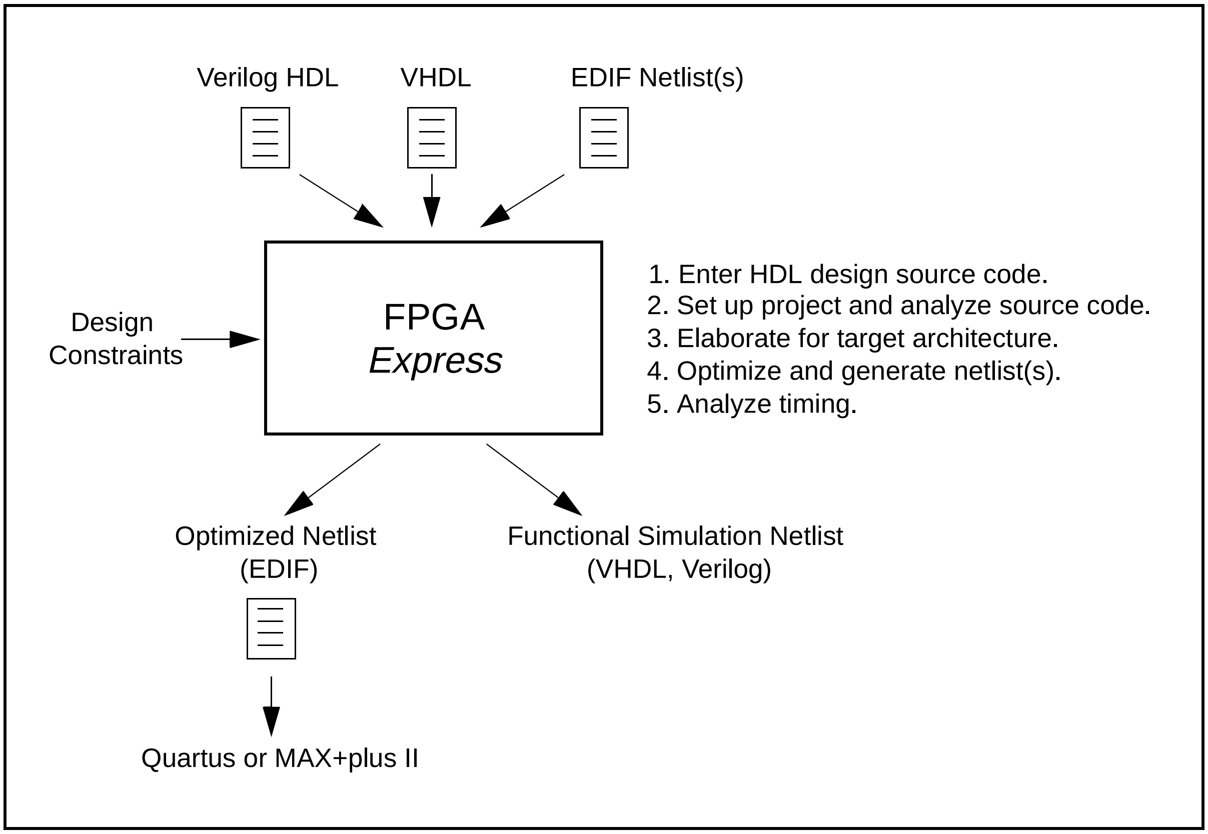
- What is FPGA *Express*?
- Features
- Benefits
- Using FPGA *Express* in Your Design Flow

What Is FPGA *Express*?

FPGA *Express* is a powerful synthesis tool for leading PLD architectures. The OEM version for Altera is specifically tailored for Altera devices. With the tool, you can create optimized netlists from VHDL code, Verilog HDL code, and existing, unoptimized EDIF netlists.

Figure 1-1 summarizes the way that FPGA *Express* fits in your design flow.

Figure 1-1 FPGA Express Design Flow Overview


Verilog HDL
VHDL
EDIF Netlist(s)
Design Constraints
FPGA *Express*
1. Enter HDL design source code.
2. Set up project and analyze source code.
3. Elaborate for target architecture.
4. Optimize and generate netlist(s).
5. Analyze timing.
Optimized Netlist (EDIF)
Functional Simulation Netlist (VHDL, Verilog)
Quartus or MAX+plus II

In this flow, you use FPGA *Express* to perform the following steps:

1. Use your favorite text editor to enter VHDL or Verilog HDL source code for your design.

   FPGA *Express* accepts any combination of VHDL, Verilog HDL, and EDIF netlist files as sources for a design. For example, you can use functions or subdesigns created through schematic capture and Verilog within a VHDL top-level design.

2. Create your project folder, add the source files to it, and analyze the HDL (VHDL and Verilog HDL) design source files for correct syntax.

   Begin by creating an FPGA *Express* project — a working directory folder to hold your files.

   When you add the design source files to the project, FPGA *Express* automatically analyzes them, using the Synopsys industry-standard HDL language support. If the source files contain errors, the Output window helps you find and correct problems. You can use the integrated text editor on the design source files to facilitate easy debugging.

3. Elaborate logic from VHDL, Verilog HDL,and EDIF netlist source files, targeting a specific Altera architecture and device.

   FPGA *Express* elaborates the logic of your design, using architecture-specific algorithms to target Altera devices. In this part of the design flow, the program elaborates each subdesign module and creates and links the design hierarchy to form a unique design implementation.

   After the elaboration, FPGA *Express* generates a schematic that represents the hierarchy of your design.

4. Optimize logic for speed or area as directed by your design constraints, generating an EDIF netlist file that is ready for place and route by MAX+plus II or Quartus.

   With the FPGA *Express* graphical user interface (GUI), you can enter constraints for your design in editable tables. The constraints contain performance requirements and optimization options for architecture-specific optimization engines.

   After optimization, FPGA *Express* generates a netlist ready for place and route by MAX+plus II or Quartus. Note that Quartus can be launched from within the GUI of FPGA Express.The program can also generate reports and an optimized schematic of your design.

5. Analyze timing. Extract and display accurate post-synthesis delay information for timing analysis and debugging.

   FPGA *Express* displays timing information beside your design constraints and highlights timing violations. This information is linked directly to the schematic for easy debugging.

Features

The FPGA *Express* core technology was developed specifically for FPGA and programmable logic device (PLD) architectures. The technology includes the following features:

- Architecture-specific mapping and optimization for Altera devices
- Industry-leading quality of results (QOR)
- Easy-to-use design flows and graphical user interfaces
- Integrated static timing analysis with TimeTracker

- Vista (visual tools for analysis) including schematic viewing with tight links to TimeTracker
- Tcl-based language for scripting

Benefits

FPGA *Express* adds control to the design process and helps you migrate to an HDL design methodology while using common, familiar systems.

Adding Control to the Design Process

Using FPGA *Express*, you can produce the results you want without multiple design iterations. You eliminate most or all iterations because you enter the target performance and select speed or area as the primary design goal before you synthesize. Other constraints you can specify include system clock speed, path delays, port delays, and hierarchical requirements. These additional constraints are passed to the place-and-route tool.

Migrating to HDL Design Methodology

If you are migrating from a schematic-based to an HDL-based design methodology, FPGA *Express* adds HDL logic synthesis and optimization to your current PLD design environment. You can completely define a design with HDL source code, or use a mixture of schematics and HDL source code to enter a design into FPGA *Express*.

Using an HDL-based design methodology increases productivity because HDL source code is device-independent, retargetable, and reusable. The built-in optimization algorithms, which are specifically tuned for Altera devices, provide the smallest, fastest designs in the shortest time.

Using Common, Familiar Systems

The FPGA *Express* Windows-compliant GUI uses standard commands and procedures to accept all your input values; no command scripts are required.

A complete set of Tcl-based commands is available for batch or interactive command-line operation. For more information about these commands, see the online help or the man pages available from the scripting shell.

Synthesizing With Flexible Design Flows

FPGA *Express* accommodates a variety of design styles and performance goals with multiple synthesis design flows. Your PLD design can be either complex or simple. It can push the limits of minimum area or maximum speed, or it can be driven by a minimum design cycle.

Using FPGA *Express* in Your Design Flow

Your design methodology determines how you use FPGA *Express* in your design flow.

Entering Designs Into FPGA *Express*

FPGA *Express* supports the following methods of describing PLD designs:

- HDL-based design methodology uses only HDL source code for design entry.

  You can enter one or more VHDL or Verilog HDL files describing any hierarchical design structure. You can then split the design into hierarchical, functional blocks, or create a single flattened design description. This feature makes it easy to reuse modules from a common design library or design source.

- Schematic-based design methodology uses a schematic editor for design entry. The design is transferred from the schematic editor to the synthesis tool through an EDIF netlist.

  This design methodology adds only one step to a traditional PLD design process, and can improve area and performance significantly.

- Mixed design methodology uses a combination of HDL source code and schematic netlists for design entry.

  In this design methodology, the HDL source code is elaborated, combined with netlists from schematic entry, and optimized into a netlist ready for MAX+plus II or Quartus.

FPGA *Express* allows you to divide a design between HDL and schematic input in any proportion and create virtually any design hierarchy. You can also use functional blocks made from both schematics and HDL to reuse modules from common design libraries or design sources.

Choosing Your Synthesis Design Flow

FPGA *Express* supports a variety of design flows. Choose the flow that meets your design's requirements.

- Push-Button

  In only two steps you can go from an HDL description or schematic netlist to an optimized netlist ready for place and route. The steps are simple: First, add your design's source files to the FPGA *Express* project. Then select the target device (family, package, and speed grade).

- Constraint-Driven

  With FPGA *Express* you can specify design requirements and constraints details in the TimeTracker static timing analysis tool to tailor the results to your needs.

- Multiple-Device

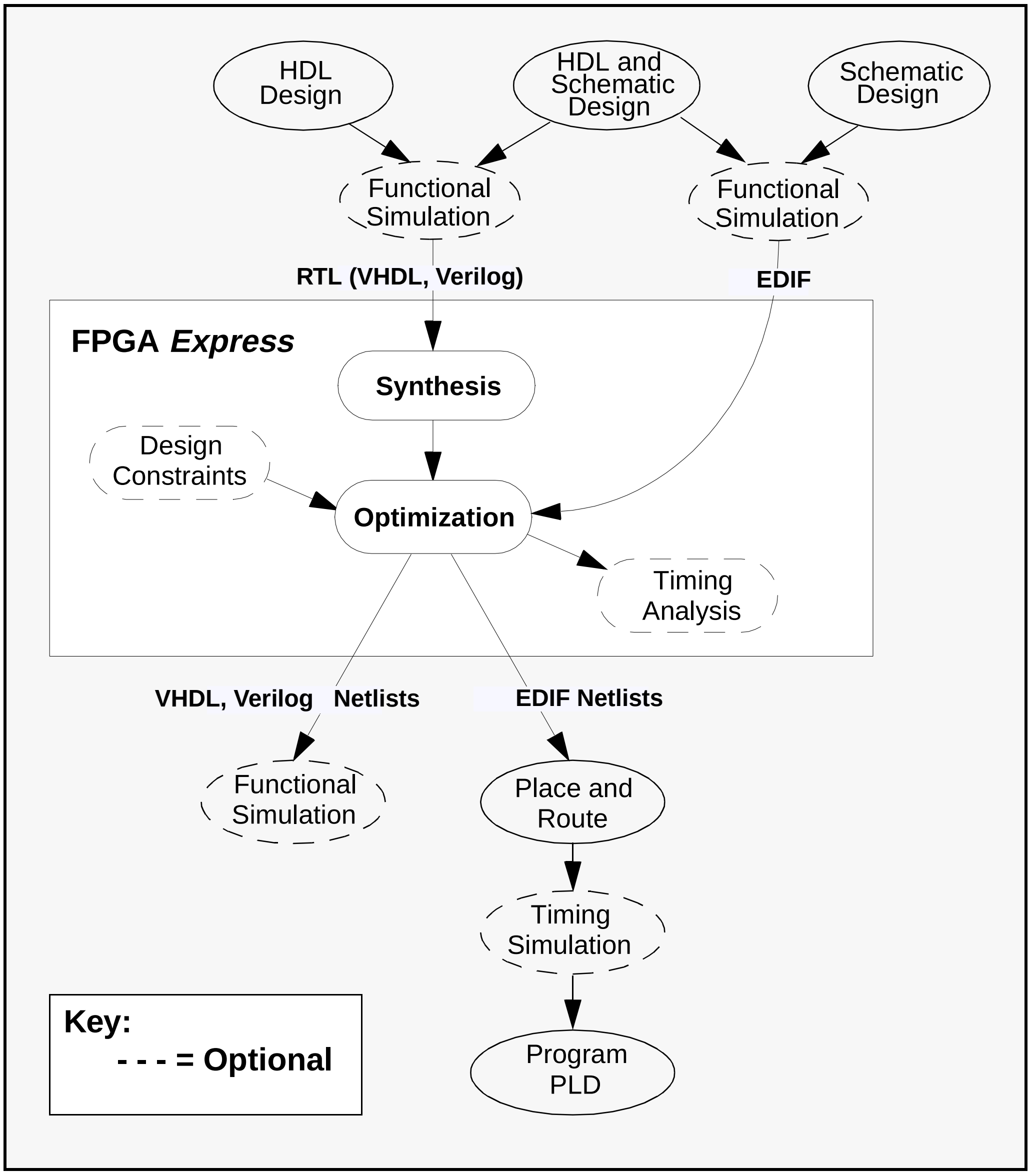
  The FPGA *Express* project manager allows you to design several devices in the same project. You can add the source files for all the chips on a board. After you identify the top-level design file of each chip, the program automatically links the rest of the design files in the chip hierarchy.

- Script-Based

  You can access all of the functionality of FPGA *Express* via the FPGA Scripting Tool (FST), a Tcl-based command-line language. You can use FST to synthesize many designs, either in batch mode or interactively, from the shell command line.

Figure 1-2 shows the flow of FPGA *Express* with other tools in these design environments. The steps in this procedure are similar for each environment; only the method of entering the design into FPGA Express differs. Note that simulation is optional.

Figure 1-2 FPGA Express in Your Design Environment



To use FPGA *Express* in your design environment,

1. Create the design:

   - For an HDL-based design methodology, write the HDL source code for the design.

   - For a schematic-based methodology, capture the design schematics and export the design in VHDL, Verilog or EDIF format.

   - For a mixed (HDL and schematic) design methodology, write the HDL source code for the parts of the design you decide to describe in HDL. Capture the schematics for the rest of the design.

2. (Optional) Verify the design functionality. For HDL code, use an HDL simulator. For schematics, use a gate-level simulator.

3. In FPGA *Express*, set up the design project and analyze the source files.

4. In FPGA *Express*, select the target device and optionally enter design constraints to elaborate and optimize the HDL source code and/or schematic capture netlists.

5. In FPGA *Express*, export an optimized EDIF netlist for place and route.

6. (Optional) In FPGA *Express*, analyze timing information to verify circuit performance.

7. (Optional) Generate VHDL or Verilog netlists for functional simulation.

   Using any standard VHDL or Verilog simulator, you can verify that the optimized netlist is functionally identical to the RTL input of FPGA *Express*.

8. Place and route the design with MAX+plus II or Quartus. Note that Quartus can be launched from within the GUI of FPGA *Express.*
9. (Optional) Simulate the design with back-annotated timing delays.
10. Program the Altera device.

2

Basic Operations

This chapter introduces you to many of the basics of running FPGA *Express*. It includes the following sections:

- Launching the Program
- Using the Quick Tour Video and the Tutorial
- Using the Tool Bar
- Learning More
- Additional Tips

For more information on advanced features not listed here, see the online help and man pages.

Launching the Program

To launch FPGA *Express*, choose Start > Programs > Synopsys > FPGA Express from the Windows Start menu.

To launch the scripting tool (FST), enter `fe_shell` in a shell or command window.

Using the Quick Tour Video and the Tutorial

If you are running FPGA *Express* on a PC, before using the program, run the online video, Quick Tour. Quick Tour is available from the Help menu.

The Quick Tour answers many of your basic questions about FPGA *Express*. You can replay sections of the video by using the menu or the Play button.

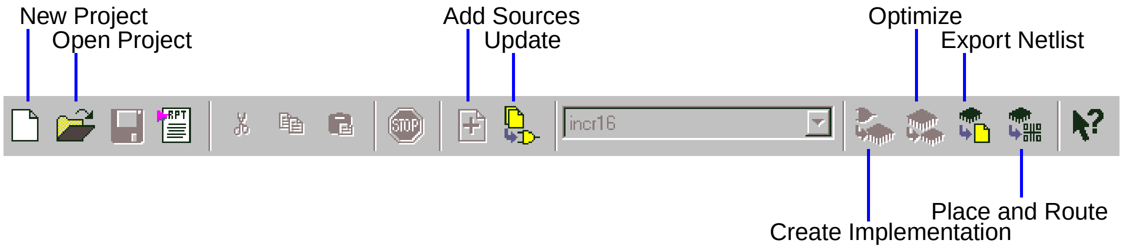
Also see “Using FPGA Express in Your Design Flow” on page 1-7 of this manual to become familiar with the design flow.

For a more detailed introduction to FPGA *Express*, step through the tutorial that is included in this manual.

Using the Tool Bar

You can use the program’s tool bar to guide you through the design flow. Clicking each button on the tool bar, from left to right, automatically takes you through the entire design flow—from project creation to netlist generation.

Figure 2-1 Tool Bar



Learning More

To learn more about FPGA *Express*, including the dialog boxes and other aspects of the GUI, use this manual and the tools described in the next sections:

- Tip Bar
- Output Window
- Topic-Based Help System
- Context-Sensitive Help System
- *FPGA Express Man Pages*
- *Altera-Provided Documents*
- Compiler Reference Manuals

Tip Bar

The tip bar provides information at each stage of the design flow. It automatically detects the state of the design, identifies the next logical step in the design flow, and provides a brief explanation of the function and purpose of that step.

Output Window

A separate window, the Output window displays errors, warnings, and other messages about your actions as you use FPGA *Express*.

Topic-Based Help System

FPGA *Express* includes an extensive topic-based online help system. To access the system, choose Help Topics from the Help menu. You can then access specific help topics through the table of contents, the alphabetic index, and the text-search feature. You can also access a glossary of FPGA *Express* terminology.

Context-Sensitive Help System

When working in FPGA *Express*, you can display context-sensitive descriptions of specific features. Click the question mark button on the product's tool bar or press Shift-F1, and then click a GUI object.

FPGA *Express* Man Pages

The FPGA *Express* man pages contain the most up-to-date command usage and syntax information. To access the man pages, enter `man`, followed by the command of interest, in the fe_shell. For example,

```
fe_shell > man script_chip
```

Altera-Provided Documents

A selection of documents provided by Altera is available from the Help menu. You might find these documents helpful in answering some of your Altera-specific questions.

Compiler Reference Manuals

Online compiler reference manuals for VHDL and Verilog HDL contain complete descriptions of the languages and their application to FPGA *Express* synthesis and optimization.

You can find all compiler reference manuals on the FPGA *Express* CD-ROM or in the program installation.

Additional Tips

The following tips will help you use FPGA *Express* more efficiently.

Table 2-1 Mouse Behavior in FPGA Express

Action	Result
Left button click	Selects and deselects.
Left button double-click	In the Design Sources and Chips windows, expands or contracts the hierarchy of the project, library, or file. On an error in the Output window, starts the text editor and opens the design source file to the selected error.
Right button click	Displays a context menu of actions for The source file in the Design Sources window and the editor Unoptimized or optimized implementations in the Chips window The messages in the Output window Subpaths in the constraints table

Table 2-2 Common Tasks in FPGA Express

If you want to . . .	Do this . . .
Expand or contract a project, library, or file tree	Double-click the icon.
Add to a library	Click the Add Sources button on the tool bar or choose Synthesis > Add Source Files.
View extended help about an error or warning	In the Output window, double-click the error number. Use the Help Index to locate the error number.

Table 2-2 Common Tasks in FPGA Express (continued)

If you want to . . .	**Do this . . .**
Modify a design source file using the editor	In the Design Sources window, right-click the file icon and choose Edit File. In the Output window, double-click the error text.
Enter constraints for an implementation	Right-click the implementation and choose Edit Constraints.
Specify timing constraints on arbitrary timing paths	In the constraints table, right-click a path and choose New Subpath.
Copy data within a row or column	Copy the selection with Edit > Copy, and paste it into the destination cell with Edit > Paste. Select the cell containing the data you want to copy and the cells to which the data should be copied. Press Control-d or choose Edit > Fill down.
Run timing analysis on an implementation	Right-click the optimized implementation and choose View Results.
Launch Quartus	Right-click the optimized implementation and choose Place and Route Chip.
Run the program from the command line	Start the FST shell.
Run batch files of FST commands	Start the FST shell and execute your batch files.

3

Tutorial

One of the fastest ways to learn to use FPGA *Express* is to complete this tutorial, which follows the program design flow. The tutorial includes the following sections:

- Creating Design Source Files
- Setting Up the Project
- Elaborating the Design
- Entering Design Constraints and Controls
- Optimizing the Design Implementation
- Analyzing Timing
- Viewing Schematics

- Generating Netlists and Reports
- Launching Quartus
- Running the FPGA Scripting Tool

Creating Design Source Files

The first step in the design flow is to create HDL source files. You can use any text editor to enter VHDL and Verilog HDL source code for your design. FPGA *Express* source files can be any combination of VHDL, Verilog HDL, or EDIF netlist files.

This step is usually performed outside of FPGA *Express*. However, you can use the integrated HDL Editor (described in the section “Debugging the Source Files” on page 3-7) to edit a design source file once it has been read into a project.

Setting Up the Project

An FPGA *Express* project is a working directory that holds a project file and subdirectories for internal and intermediate files. The key components of a project are the source files that describe the PLD design or designs. The types of source files FPGA *Express* supports are VHDL, Verilog HDL, and EDIF netlists. Netlist files usually originate from schematic capture systems or preoptimized macro libraries.

To set up a project in FPGA *Express*,

1. Create the project.
2. Add the design source files to the project and analyze them.
3. Debug the source files.

The following sections describe these steps.

Creating the Project

After you prepare the design source files, start the design flow by creating a project. To create a new project, FPGA *Express* creates a new folder (directory) in which to store the project information.

To create a new project in FPGA *Express*,

1. Launch FPGA *Express* as described on page 2-2. This step opens the main window, which includes the tool bar, tip bar, Design Sources window, Chips window, and Output window. For more information about these window features, see Chapter 2, “Basic Operations.”
2. When you start FPGA *Express*, you can either open an existing project or create a new one. For now, create a new project by clicking the New Project button on the tool bar or by choosing New Project from the File menu.

   In the main window, FPGA *Express* displays the Create New Project dialog box.

   The dialog box includes a default location and name for the project. You can change the location, name, or both. Use the drop-down list in the Save In field to navigate your directory tree, or click the Create New Folder icon to create a new directory.

3. Type the name `tutor` for the project folder (directory) in the Name field. The `tutor` directory is where project files will be stored.
4. Click the Create button to create the new project.

   After the program creates the `tutor` project, the Add Sources dialog box opens.

Adding the Design Source Files to the Project

After you create the project, you add the design source files to it. When you add these files to a project, FPGA *Express* automatically analyzes them for errors.

Whenever you create a project, a default library named WORK automatically appears in the project folder. You add the source files to the library. FPGA *Express* source files can be any combination of VHDL, Verilog HDL, or EDIF netlist files.

VHDL environments sometimes require multiple libraries. You can add a library at any time by right-clicking in the Design Sources window and choosing New Library. Then add design source files to that library in the same way as you do for WORK.

When you add source files to a project library, FPGA *Express* automatically analyzes them for syntax errors, conditions that produce errors and warnings, and other problems. To understand the problems, read the topic-based help system's explanations of the messages.

Note:

> FPGA *Express* does not copy design source files. When you add these files, the program analyzes them in their current location. If you changed the files (a ? icon beside a file name indicates a

change), you must click the Update button on the tool bar or right-click in the Design Sources window and choose Update File so that FPGA *Express* reanalyzes the files.

To add source files,

1. Be sure the Add Sources dialog box is open.

   If it is not open, you can access it by clicking ⊞ on the tool bar, by choosing Synthesis > Add Source Files, or by right-clicking WORK in the Design Sources window and choosing Add Sources in WORK.

2. Select source files in the Add Sources dialog box.

   For this tutorial, use the `tutor` design files written in VHDL (`tutor.v/tutor.vhd`, `counter.vhd`) or Verilog (`tutor.v`, `counter.v`). These files are located in the *installation_directory*/ samples/tutorial/vhdl or verilog directories.

3. After you select your design source files, click Open.

   Note:

   On a PC, you can also drag and drop source files into the Design Sources window to add them to the project.

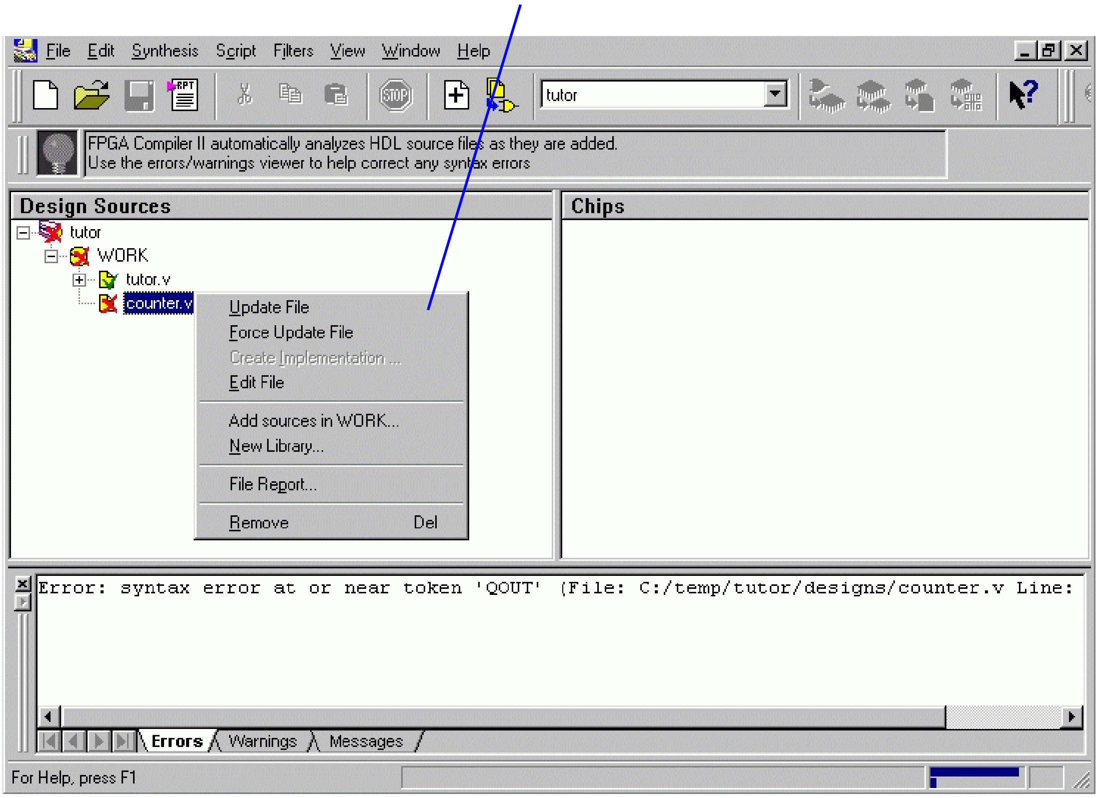
   After you select your design source files, FPGA *Express* displays the project window and extends the menu bar. The project window has two subwindows—the Design Sources window and the Chips window (see Figure 3-1):

   - The Design Sources window displays the name, location, and status of each of the design’s source files.
   - The Chips window displays information for individual design implementations, such as name and device type.

Note that the project window's title bar displays the name of the project.

FPGA *Express* automatically analyzes each source file as it is added in the Design Sources window. The icon to the left of each file name indicates the results of the analysis, as illustrated in Figure 3-1.

Figure 3-1 The Project Window After Adding Design Source Files



To display this menu, right-click a source file name.

FPGA Compiler II automatically analyzes HDL source files as they are added.
Use the errors/warnings viewer to help correct any syntax errors

```
Error: syntax error at or near token 'QOUT' (File: C:/temp/tutor/designs/counter.v Line:
```

For Help, press F1

In this tutorial, the counter file has at least one error (indicated by the red cross) that must be corrected. The green check marks next to the other files indicate that the files have no errors or warnings.

The error in the problem source file is reflected up the hierarchy. Therefore the library and the project icons are also marked with a red cross. (For a list and explanation of each analysis status icon, see the topic-based online help.)

Note that the Output window at the bottom of the screen displays error messages about the selected source file.

Debugging the Source Files

Use the Errors, Warnings, and Messages tabs of the Output window to help find and correct errors and warnings in source files. The Output window shows the file name, line number, and type of each error or warning. After viewing the messages, you can use the built-in text editor to investigate and fix the source files.

To view and correct errors and warnings,

1. Select the design `counter` in the Design Sources window, view the errors in the Output window. Read the error and warning messages.
2. Correct any errors in the design source file. There are two ways to open the text editor window directly to an error in a design source file:
   - At the location of the *first* error, open the source file for editing by right-clicking in the Design Sources window and choosing Edit File.
   - At the location of a *specific* error, open the source file for editing by double-clicking the text for that error in the Output window.

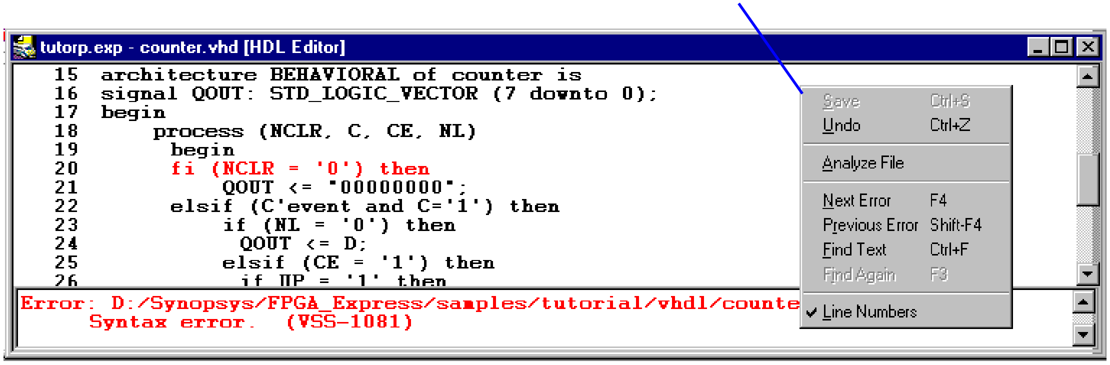
If you selected the internal source editor as the default source file editor (from the Synthesis > Options > General screen), FPGA *Express* uses the built-in text editor to edit the selected file. You can use the Edit menu to navigate in the text file. You can also use the text editor menu that is displayed when you right-click in the text editor window.

If you did not select the internal source editor, choosing Edit File starts the editor associated with the file's type and opens the file.

Figure 3-2 shows the text editor window and its menu of commands.

Figure 3-2 Text Editor Window

To display this menu, right-click in the HDL Editor window.



```
architecture BEHAVIORAL of counter is
signal QOUT: STD_LOGIC_VECTOR (7 downto 0);
begin
      process (NCLR, C, CE, NL)
        begin
        fi (NCLR = '0') then
              QOUT <= "00000000";
        elsif (C'event and C='1') then
              if (NL = '0') then
                QOUT <= D;
              elsif (CE = '1') then
                if UP = '1' then
```

```
Error: D:/Synopsys/FPGA_Express/samples/tutorial/vhdl/counte
        Syntax error.  (VSS-1081)
```

3. The text editor indicates that the errors in the design `counter` are caused by a typing error in the `if` statement. Change the text from `fi` to `if` in the `counter` file.

4. Save the file. The `tutor.vhd` icon contains a question mark, indicating that the file is out of date.

5. Update the file: Either go to the Design Sources window and click on the tool bar, or right-click in the text editor window and choose Analyze File before closing the text editor window.

Depending on the extent of your modifications, you update the file, the library, or the project. FPGA *Express* reanalyzes only the files you modify.

If your changes are not evident to FPGA *Express* (for example, if the changes are in a dependent file), right-click in the Design Sources window and choose Force Update File (Library or Project).

After adding, analyzing, and debugging source files, you are ready to elaborate the design.

Elaborating the Design

FPGA *Express* uses design configurations to specify a target device and design constraints, leading to optimized PLD netlists. Each design configuration, together with the set of design files, determines a unique design implementation.

In this step, FPGA *Express* elaborates a unique, unoptimized design implementation from your source files. The program begins by using generic logic to create a technology-independent description of the design, and then maps this description to a specific device technology. After you have completed this step, you can set design constraints and controls before optimizing.

To elaborate the design,

1. Identify the top-level design.
2. Specify the target architecture.
3. Set the target clock frequency.
4. Choose synthesis options and create the design implementation.

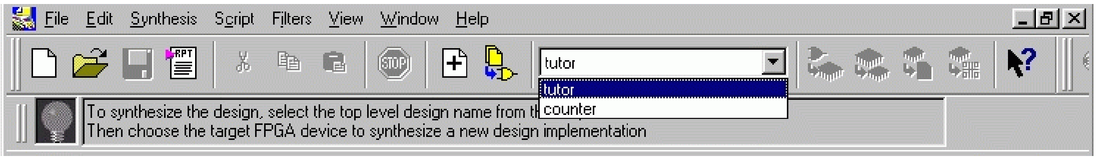
Identifying the Top-Level Design

To begin building a design implementation from analyzed source files, you identify the top-level entity (VHDL), module (Verilog), or schematic netlist. FPGA *Express* uses this information to start building design hierarchy and interconnections.

For this example, choose tutor as the top-level entity. You can identify the top-level design in any of these ways:

- Click the drop-down list of top-level designs on the tool bar to see a list of all entities, modules, and netlist designs in the source files. Scroll through the list to find and choose the top-level design. See Figure 3-3.)

Figure 3-3 Identifying the Top-Level Design



To synthesize the design, select the top level design name from th
Then choose the target FPGA device to synthesize a new design implementation

- Alternatively, in the Design Sources window (shown earlier) you can double-click the source file you want to use as the top level . This step expands the file and displays an icon for the top-level design. Right-click the icon and choose Create Implementation.

- As a third alternative, you can click the Create Implementation button on the tool bar.

Specifying the Target Architecture

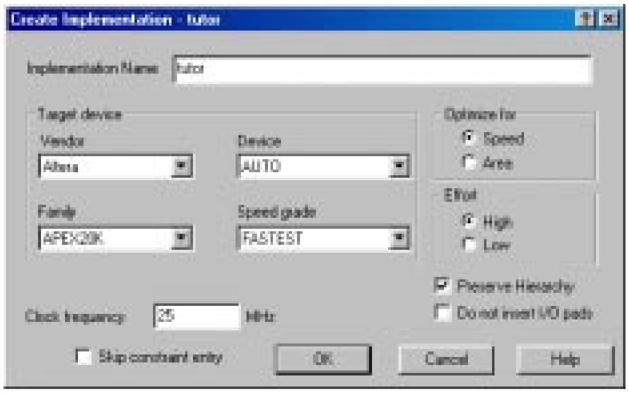
After selecting a top-level design, specify a target architecture in the Create Implementation dialog box (see Figure 3-4).

To specify a target architecture,

1. Enter an implementation name. If you do not enter a name, FPGA *Express* automatically creates a unique implementation name based on the name of the top-level design.
2. Specify the target device and speed grade for the design.

For this tutorial, the design is small enough to fit into any device supported by FPGA *Express*.

Figure 3-4 Create Implementation Dialog Box



Create Implementation - tutor

Implementation Name: tutor

Target device

Vendor: Altera

Device: AUTO

Family: APEX20K

Speed grade: FASTEST

Optimize for: ● Speed ○ Area

Effort: ● High ○ Low

☑ Preserve Hierarchy

□ Do not insert I/O pads

Clock frequency: 25 MHz

□ Skip constraint entry

Setting the Target Clock Frequency

Set the target clock frequency for the design in the Create Implementation dialog box. This target frequency is used as the default value for all clocks in the design. After you elaborate the design, you can change target clock frequencies in the design constraint tables.

For this tutorial, enter a target clock frequency of 25 MHz.

Note:

> Overconstraining a design can adversely affect the place and route results. Therefore, specify only what is really required.

Choosing Synthesis Options and Creating the Design Implementation

After you set the target architecture and clock frequency, FPGA *Express* is ready to create a design implementation.

To create the implementation, the program elaborates logic for each source file. It determines the complete hierarchical structure and topology of the design, including multiple-level links and references between subdesigns. With this information, FPGA *Express* produces an intermediate, unoptimized design implementation.

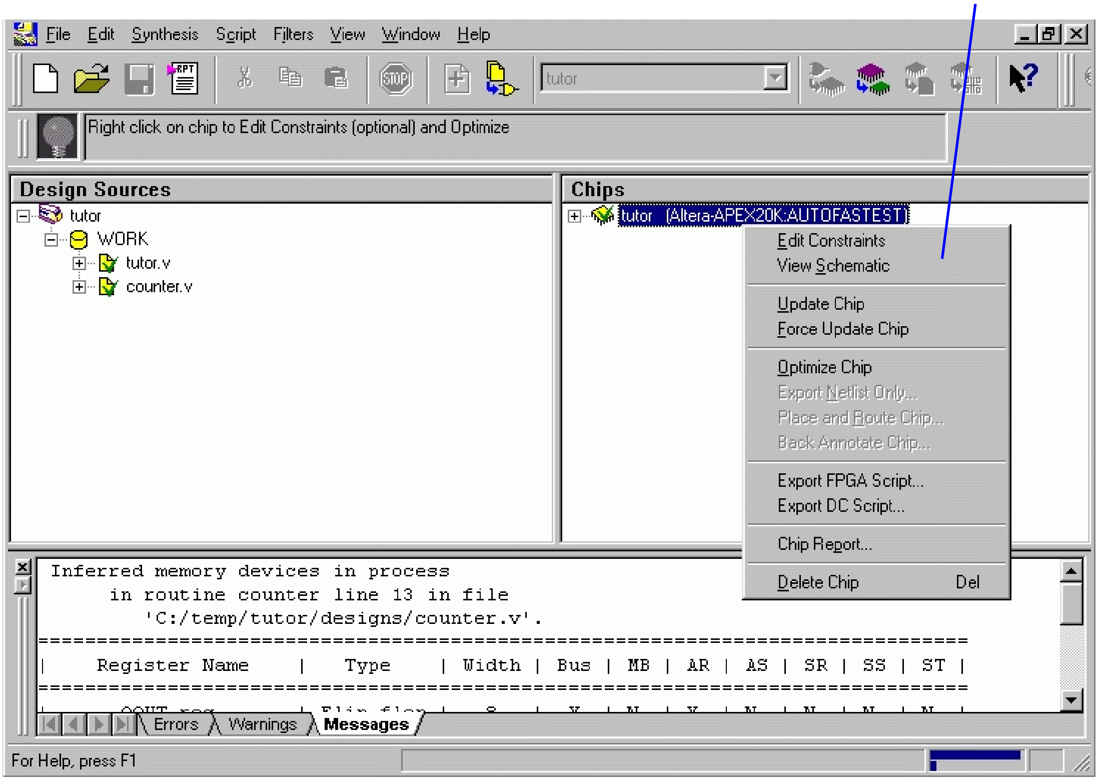
To create a design implementation,

1. In the Create Implementation dialog box, verify the target architecture, clock frequency, and implementation name.
2. (Optional) You can also specify other options in this dialog box, including:
    - Choosing to optimize for speed or area with high or low CPU effort.
    - Choosing to preserve hierarchy in the design.
    - Choosing not to insert I/O pads, if appropriate.
    - Choosing to skip constraint entry at this time (see the Push-Button Design Flow topic of the topic-based help). For this tutorial, do not select Skip constraint entry.
3. Click OK.

    When FPGA *Express* finishes creating the design implementation, an implementation icon, its name, and target device appear in the Chips window. The icon indicates the implementation’s status. See Figure 3-5.
4. Check the Output window to investigate errors and warnings.

Figure 3-5 Creating the Design Implementation

To display this menu, right-click an implementation icon.



```
Inferred memory devices in process
     in routine counter line 13 in file
        'C:/temp/tutor/designs/counter.v'.
===============================================================================
|    Register Name    |    Type    | Width | Bus | MB | AR | AS | SR | SS | ST |
===============================================================================
```

Entering Design Constraints and Controls

Before you start to optimize a design for a target device, you can set performance constraints, attributes, and optimization controls. Design constraints guide FPGA *Express* with specific optimization requirements.

Although this step is optional, it is highly recommended. Entering your requirements in the constraint tables can significantly improve place and route results.

For example, entering constraints for an output port with restrictive speed requirements makes it easier for the place and route tool to fulfill those requirements. If a design is very large and has many hierarchical levels, you can improve place and route results by entering hierarchy constraints. Note, however, that overconstraining a design can adversely affect place and route results. Therefore, specify only what is really required.

FPGA *Express* separates constraint entries into logically related groups (for example, clocks, ports, and paths). The program automatically extracts design-specific information, such as clock names, port names, and design hierarchy, from the design and displays it in tables. You enter performance constraints, attributes, and optimization options directly into the tables.

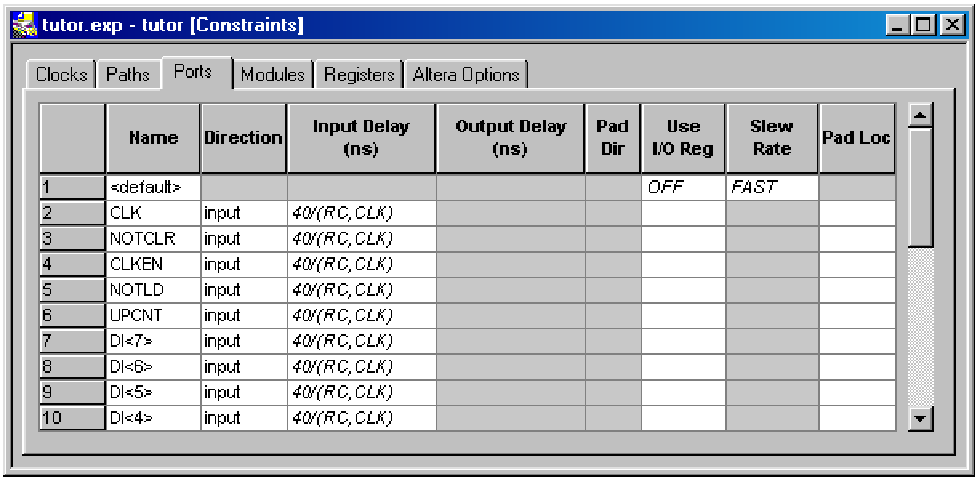
Each set of constraint tables and dialog boxes is specific to a particular target PLD architecture. Controls for some target technologies are available through a vendor-specific dialog box that is displayed as another tab for the constraint tables.

To enter design constraints, attributes, and options,

1. Right-click the elaborated implementation and choose Edit Constraints to open the design constraint and optimization-control tables.
2. Figure 3-6 shows how the constraints and controls are logically grouped by function into separate Clocks, Paths, Ports, and Modules tabs. An additional Registers tab is also available for APEX20K and APEX20KE devices. Click the tabs to display their tables.

   Use the topic-based online help if you need instructions and information about specific constraints, attributes, and optimization options.

Figure 3-6 Constraint Tables



	Name	Direction	Input Delay (ns)	Output Delay (ns)	Pad Dir	Use I/O Reg	Slew Rate	Pad Loc
1	<default>					*OFF*	*FAST*	
2	CLK	input	*40/(RC,CLK)*					
3	NOTCLR	input	*40/(RC,CLK)*					
4	CLKEN	input	*40/(RC,CLK)*					
5	NOTLD	input	*40/(RC,CLK)*					
6	UPCNT	input	*40/(RC,CLK)*					
7	DI<7>	input	*40/(RC,CLK)*					
8	DI<6>	input	*40/(RC,CLK)*					
9	DI<5>	input	*40/(RC,CLK)*					
10	DI<4>	input	*40/(RC,CLK)*					

3. Open and explore the tables for the tutor implementation.

   The contents of the tables reflect the architecture you chose earlier. Note that the Clocks and Paths constraint tables are preloaded with the clock frequency (and corresponding period) that you entered for the target clock frequency.

4. After entering constraint, attribute, and option information, close the constraint window. This action automatically saves any changes.

Optimizing the Design Implementation

After entering constraint, attribute, and option information, you are ready to optimize the design implementation.

In this step, you optimize the design implementation elaborated earlier, guided by the constraints and controls you entered in the constraint tables.

To optimize an elaborated implementation,

1. Click the elaborated implementation in the Chips window to select it. Its name is displayed on the top-level design field of the tool bar.
2. Right-click the implementation and choose Optimize Chip, or click on the tool bar.

A new optimized implementation icon appears beneath the original implementation.

When you optimize a design implementation, FPGA *Express* analyzes the actual timing of your design against your requirements. After optimization, the design implementation tables display the constraints you specified with your design results so that you can compare them.

Analyzing Timing

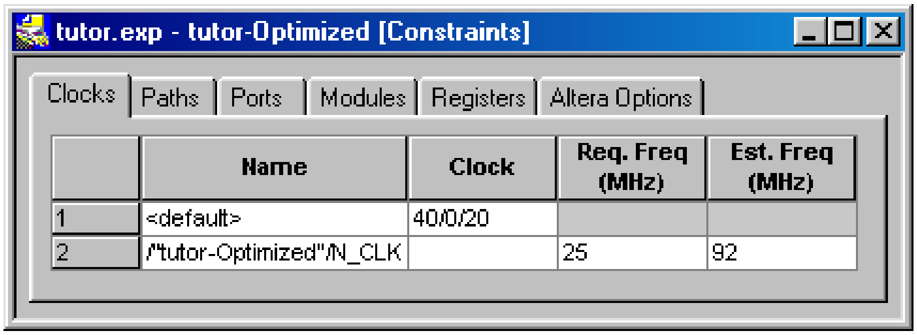
After you create an elaborated implementation, you can determine circuit performance by checking the results of optimization and analyzing timing information. The post-synthesis timing data is displayed in the same formats as the tables you used to enter constraints.

To view the results of optimization,

1. Open an optimized implementation by right-clicking it and choosing View Results.

2. Check the Clocks constraint table to see the maximum clock frequencies that FPGA *Express* estimated for each of the clocks in the design. Clock frequency violations appear in red. Figure 3-7 shows the Clocks constraint table after optimization.

Figure 3-7 Optimization Results in the Clocks Constraint Table



	Name	Clock	Req. Freq (MHz)	Est. Freq (MHz)
1	<default>	40/0/20		
2	/"tutor-Optimized"/N_CLK		25	92

3. Check the Paths constraint table for more information about timing violations:

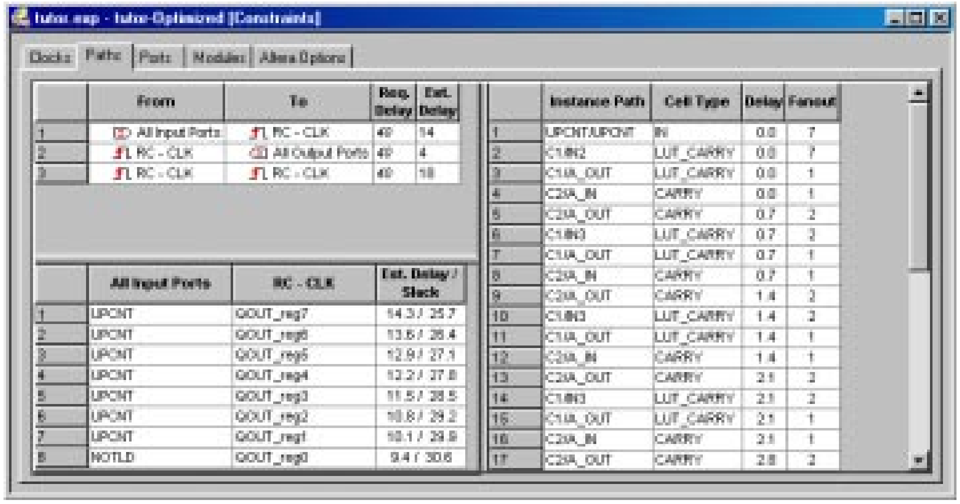
   - Select a path group to see a list of paths in that group.
   - Select a path from the list to see the details of path composition, cumulative delays, and fanout.
   - See the topic-based online help for a description of timing analysis in FPGA *Express*.

   Figure 3-8 shows the Paths constraint table after optimization.

   Note:

   All pins on the timing path are displayed. Therefore two rows on the right pane correspond to a single cell (source and load).

Figure 3-8 Optimization Results in the Paths Constraint Table



tutor.exp - tutor-Optimized [Constraints]

	From	To	Req. Delay	Est. Delay
1	All Input Ports	RC - CLK	40	14
2	RC - CLK	All Output Ports	40	4
3	RC - CLK	RC - CLK	40	18

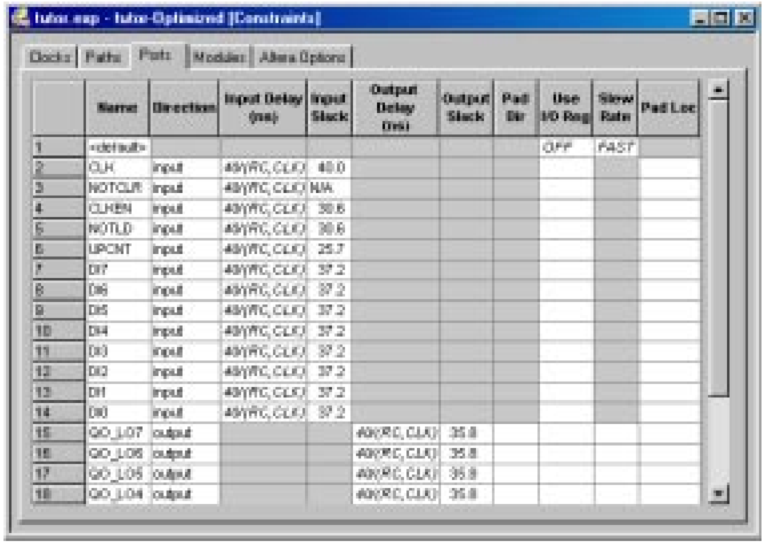
	All Input Ports	RC - CLK	Est. Delay / Slack
1	UPCNT	QOUT_reg7	14.3 / 25.7
2	UPCNT	QOUT_reg6	13.6 / 26.4
3	UPCNT	QOUT_reg5	12.9 / 27.1
4	UPCNT	QOUT_reg4	12.2 / 27.8
5	UPCNT	QOUT_reg3	11.5 / 28.5
6	UPCNT	QOUT_reg2	10.8 / 29.2
7	UPCNT	QOUT_reg1	10.1 / 29.9
8	NOTLD	QOUT_reg0	9.4 / 30.6

	Instance Path	Cell Type	Delay	Fanout
1	UPCNT/UPCNT	IN	0.0	7
2	C1/IN2	LUT_CARRY	0.0	7
3	C1/A_OUT	LUT_CARRY	0.0	1
4	C2/A_IN	CARRY	0.0	1
5	C2/A_OUT	CARRY	0.7	2
6	C1/IN3	LUT_CARRY	0.7	2
7	C1/A_OUT	LUT_CARRY	0.7	1
8	C2/A_IN	CARRY	0.7	1
9	C2/A_OUT	CARRY	1.4	2
10	C1/IN3	LUT_CARRY	1.4	2
11	C1/A_OUT	LUT_CARRY	1.4	1
12	C2/A_IN	CARRY	1.4	1
13	C2/A_OUT	CARRY	2.1	2
14	C1/IN3	LUT_CARRY	2.1	2
15	C1/A_OUT	LUT_CARRY	2.1	1
16	C2/A_IN	CARRY	2.1	1
17	C2/A_OUT	CARRY	2.8	2

4. Check the Ports constraint table for information about input and output delays.

   Figure 3-9 shows the Ports constraint table. The results include the slack for input arrival time and output delay for each port.

Figure 3-9 Optimization Results in the Ports Constraint table



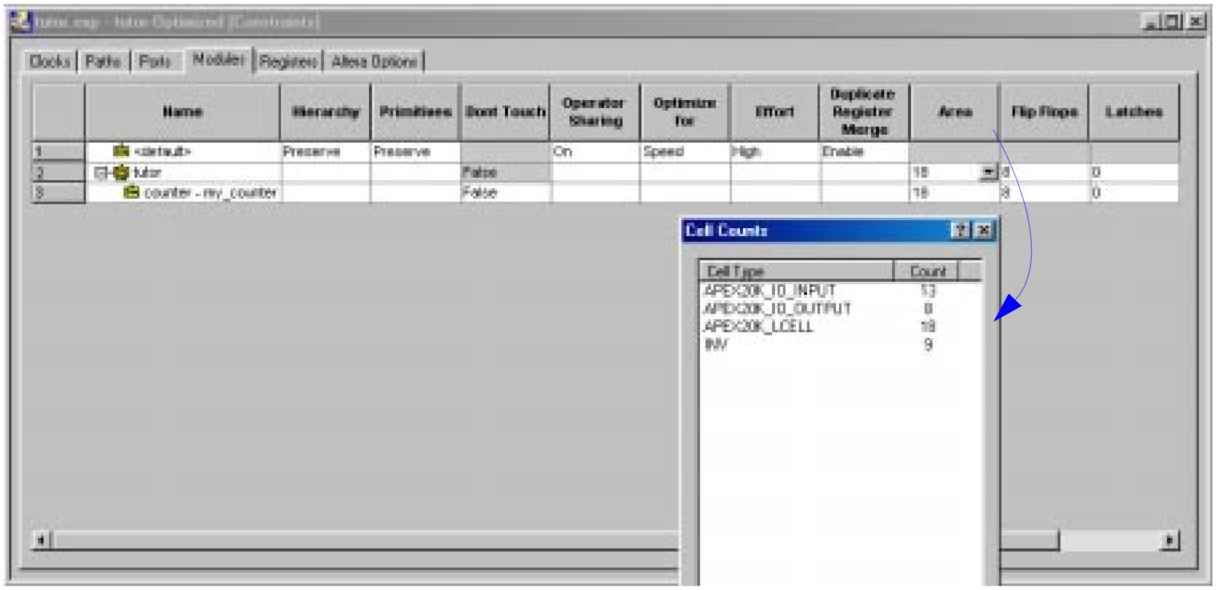
tutor.exp - tutor-Optimized [Constraints]

	Name	Direction	Input Delay (ns)	Input Slack	Output Delay (ns)	Output Slack	Pad Dir	Use I/O Reg	Slew Rate	Pad Loc
1	<default>							OFF	FAST	
2	CLK	input	40/(RC,CLK)	40.0						
3	NOTCLR	input	40/(RC,CLK)	N/A						
4	CLKEN	input	40/(RC,CLK)	30.6						
5	NOTLD	input	40/(RC,CLK)	30.6						
6	UPCNT	input	40/(RC,CLK)	25.7						
7	DI7	input	40/(RC,CLK)	37.2						
8	DI6	input	40/(RC,CLK)	37.2						
9	DI5	input	40/(RC,CLK)	37.2						
10	DI4	input	40/(RC,CLK)	37.2						
11	DI3	input	40/(RC,CLK)	37.2						
12	DI2	input	40/(RC,CLK)	37.2						
13	DI1	input	40/(RC,CLK)	37.2						
14	DI0	input	40/(RC,CLK)	37.2						
15	QO_LO7	output			40(RC,CLK)	35.8				
16	QO_LO6	output			40(RC,CLK)	35.8				
17	QO_LO5	output			40(RC,CLK)	35.9				
18	QO_LO4	output			40(RC,CLK)	35.8				

5. Check the Modules constraint table for information about the device resources used. Double-click the items in the Area column for details about cell count.

   Figure 3-10 shows the Modules constraint table after optimization.

Figure 3-10 Viewing Optimization Results in the Modules Constraint Table



	Name	Hierarchy	Primitives	Dont Touch	Operator Sharing	Optimize for	Effort	Duplicate Register Merge	Area	Flip Flops	Latches
1	<default>	Preserve	Preserve		On	Speed	High	Enable			
2	tutor			False					18	8	0
3	counter - my_counter			False					18	8	0

Cell Type	Count
APEX20K_IO_INPUT	13
APEX20K_IO_OUTPUT	8
APEX20K_LCELL	18
INV	9

Viewing Schematics

You can view and analyze your design graphically using the integrated schematic viewer and the TimeTracker feature. You can view an RTL view of the design, as shown in Figure 3-11, or an optimized (mapped) view of the design, as shown in Figure 3-12:

Figure 3-11   Schematic Viewer—RTL View

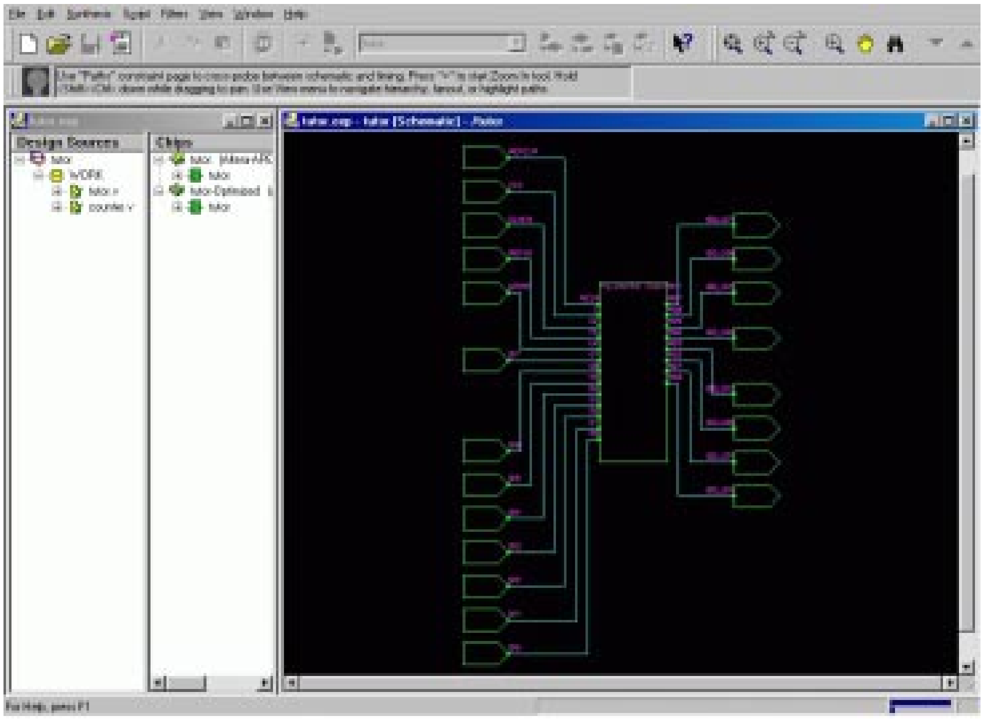
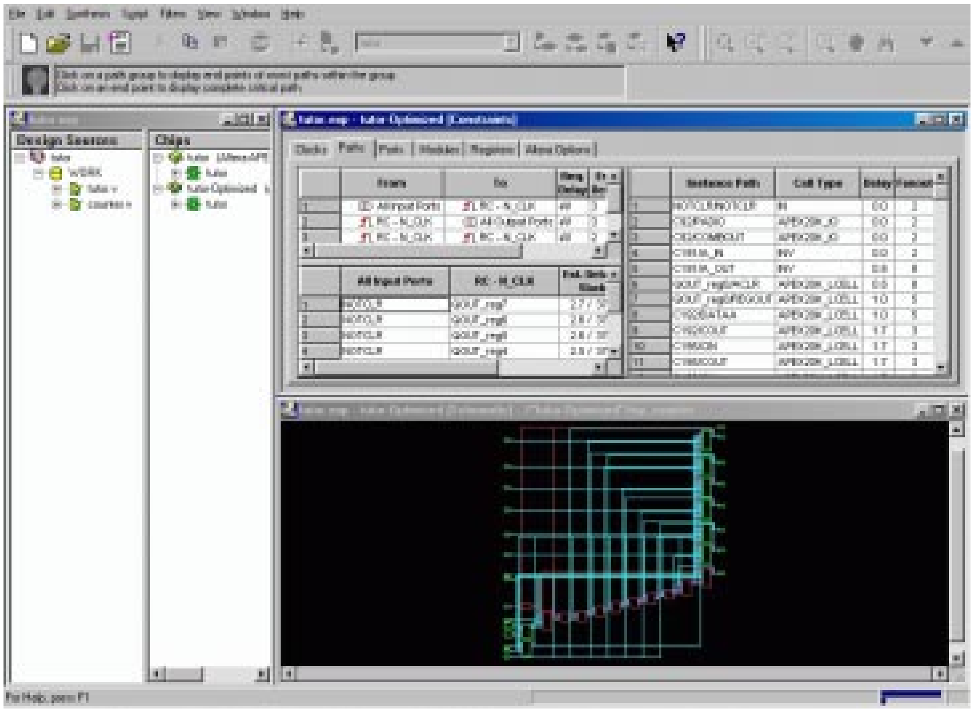

File Edit Synthesis Script Filters View Window Help
Use "Paths" constraint page to cross-probe between schematic and timing. Press "V" to start Zoom In tool. Hold <Shift>/<Ctrl> down while dragging to pan. Use View menu to navigate hierarchy, fanout, or highlight paths.
tutor.cnp - tutor [Schematic] - /tutor
Design Sources
tutor
WORK
tutor.v
counter.v
Chips
tutor
tutor-Optimized
tutor
For Help, press F1

Figure 3-12 Schematic Viewer—Optimized View



Click on a path group to display end points of worst paths within the group.
Click on an end point to display complete critical path

tutor.exp - tutor Optimized [Constraints]

	From	To	Req. Delay	
1	All Input Ports	RC - N_CLK	All	3
2	RC - N_CLK	All Output Ports	All	3
3	RC - N_CLK	RC - N_CLK	All	2

	All Input Ports	RC - N_CLK	Est. Orig. / Slack
1	NOTCLR	QOUT_reg7	2.7 / 37
2	NOTCLR	QOUT_reg6	2.8 / 37
3	NOTCLR	QOUT_reg5	2.8 / 37
4	NOTCLR	QOUT_reg4	2.8 / 37

	Instance Path	Cell Type	Delay	Fanout
1	NOTCLR/NOTCLR	IN	0.0	2
2	CIO/PADIO	APEX20K_IO	0.0	2
3	CIO/COMBOUT	APEX20K_IO	0.0	2
4	C1918_IN	INV	0.0	2
5	C1918_OUT	INV	0.8	8
6	QOUT_reg0/ACLR	APEX20K_LCELL	0.0	8
7	QOUT_reg0/REGOUT	APEX20K_LCELL	1.0	5
8	C1928/DATAA	APEX20K_LCELL	1.0	5
9	C1928/COUT	APEX20K_LCELL	1.7	3
10	C1960/CIN	APEX20K_LCELL	1.7	3
11	C1960/COUT	APEX20K_LCELL	1.7	3

Viewing an RTL or Generic Design

To view an RTL view of a design,

1. In the Chips window, right-click the elaborated implementation called `tutor`.
2. From the pop-up menu, choose View Schematic. The windows auto-arrange to maximize the viewable area.
3. Maximize the main program window in the screen area.
4. Navigate around the schematic by clicking the Zoom In, Zoom Out, Zoom In Tool, and Zoom Full-Fit buttons on the tool bar.

5. To view an area, click the Zoom In Tool button on the tool bar, left-drag the mouse pointer over the area of interest, and then release the mouse button.

6. View the contents of the `my_counter(counter)` block by double-clicking the block.

7. Return to the next level up by right-double-clicking anywhere in the schematic.

8. Navigate around the hierarchy using the Chips window:

   - Double-click a level of hierarchy to expand it.

   - Select one of the blocks to view that level.

9. Note that all cells are generic—not mapped to a particular technology at this stage—and that all hierarchy is preserved, because operators have not been implemented at this stage, and they appear as primitive cells.

10. When you are comfortable with the navigation features, close the RTL view.

Viewing an Optimized (or Mapped) Design

To view an optimized design,

1. In the Chips window, right-click the optimized implementation called `tutor-Optimized`.

2. From the pop-up menu, choose View Schematic.

   The windows auto-arrange, displaying the project window on the left, and the TimeTracker and schematic windows on the right.

3. Navigate around the schematic by clicking the Zoom in, Zoom Out, Zoom In Tool, and Zoom Full-Fit buttons on the tool bar.
4. To view an area, click the Zoom In Tool button on the tool bar, left-drag the mouse pointer over the area of interest, and then release the mouse button.

Viewing Timing Results Graphically Using TimeTracker

To view timing results in the schematic viewer,

1. Select the Paths tab to view the Paths constraint table.
2. Select the RC–CLK RC–CLK row in the path groups pane (the upper-left pane).

   The paths constrained by this path group are now displayed in the paths area (bottom-left pane).

   For any paths that fail timing, the timing values are displayed in red in the TimeTracker window, and in the schematic the registers are highlighted.
3. Select the first path from this group of paths.

   The right side of the TimeTracker window displays the pins contained in this path. The instance path name, cell type, delay, and fanout are shown for each pin.

   In the schematic window, the critical path is highlighted in red, allowing you to see the path graphically.
4. Select a path and pause the mouse pointer over each cell of the path in turn.

A pop-up window displays the cell name, cell type, pin numbers, delay, fanout, and slack values.

5. Select a pin from the pin list.
6. Note that the cell to which the pin is attached is highlighted in yellow in the schematic window.
7. Move along the path by clicking the Previous Pin and Next Pin buttons on the tool bar, or by clicking each pin in the path in TimeTracker.
8. With a cell selected and highlighted in yellow, click the Fan In and Fan Out buttons on the tool bar to display the fanin and fanout logic cones.

Generating Netlists and Reports

When the design implementation is optimized, you can generate two kinds of netlist files:

- EDIF files formatted for place and route by MAX+plus II or Quartus
- VHDL or Verilog files for functional simulation

You can also generate a report file to review and document the project.

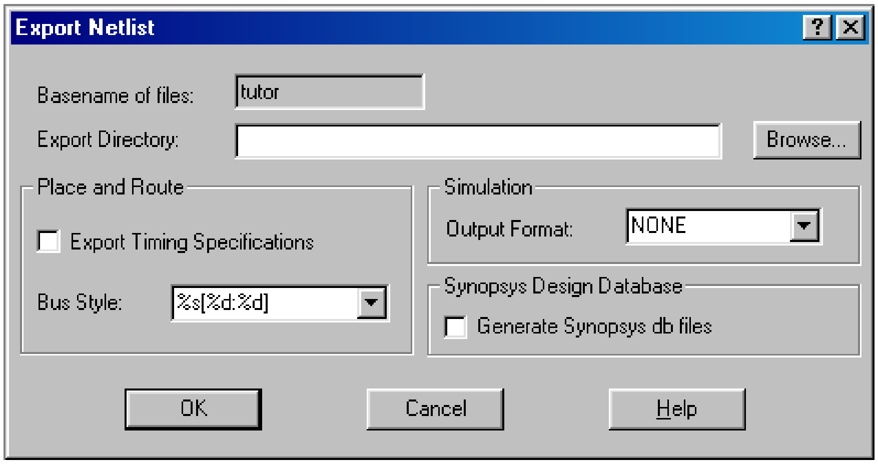
Generating Netlist Files

FPGA *Express* generates EDIF netlists that MAX+plus II or Quartus can process directly. FPGA *Express* can also generate VHDL and Verilog netlists for simulation.

To generate netlist files,

1. Either select the optimized implementation and click on the tool bar, or right-click the implementation and choose Export Netlist. The Export Netlist dialog box appears (see Figure 3-13).

Figure 3-13 Export Netlist Dialog Box



Export Netlist

Basename of files: tutor

Export Directory:

Place and Route

☐ Export Timing Specifications

Bus Style: %s[%d:%d]

Simulation

Output Format: NONE

Synopsys Design Database

☐ Generate Synopsys db files

2. Choose an export directory for the netlist files.

3. To change directories, either type the new directory name or click Browse.

4. Specify whether or not to export timing constraints with the netlist, using the Export Timing Specifications check box.

5. Select an output format for your netlist:

   - Select NONE in the Output Format field to export only the netlist for place and route.

   - Select Verilog or VHDL to also export a netlist for simulation.

6. Click OK.

Caution!

To avoid overwriting your source files, always export netlists into a separate directory to be sure that they are manageable. Many files might be exported from a single design.

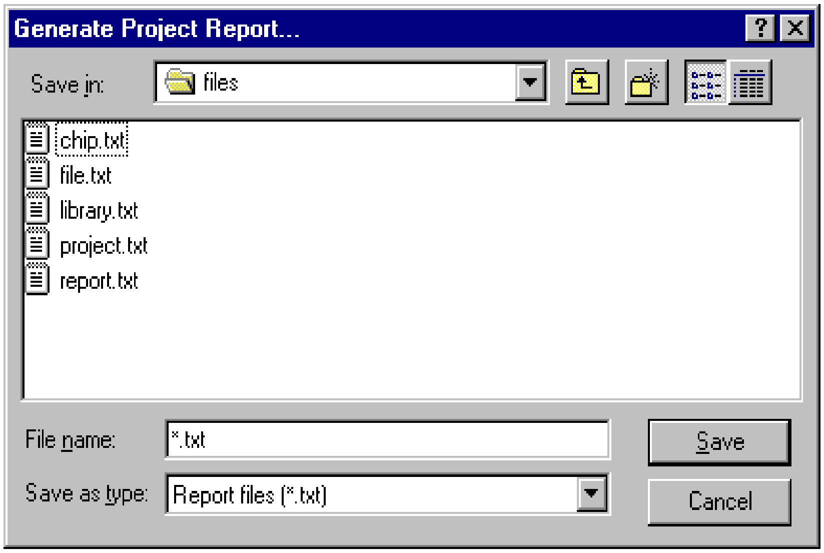
Generating a Report

You can generate a report on a project, library, file, or chip. Project reports document the design through the synthesis and optimization design flow and include information such as design source data, constraints, and optimization options.

To generate a report,

1. Either select the project, library, file, or chip in the project window and click RPT on the tool bar or right-click the project, library, file, or chip and choose Report.
2. In the dialog box that appears (see Figure 3-14), specify the name and location for the report.

Figure 3-14   Generate Project Report Dialog Box



Save in: files

chip.txt
file.txt
library.txt
project.txt
report.txt

File name: *.txt

Save as type: Report files (*.txt)

3. Click Save. FPGA *Express* creates a text file containing summary information for the whole project, the library, the design file, or the chip.

4. Open the report file in a text editor or word processor.

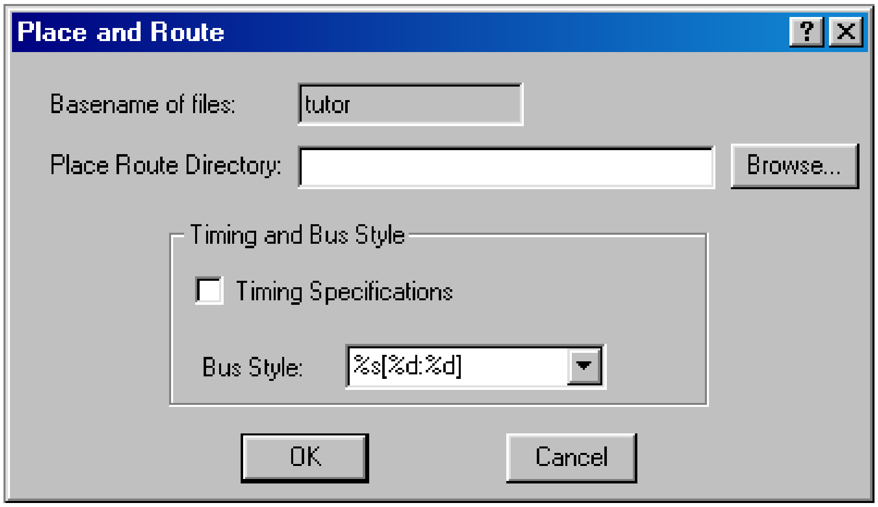
Launching Quartus

FPGA *Express* allows you to launch Quartus from within the GUI. All the files that Quartus requires are automatically generated in the directory specified in the Place and Routedialog box shown in Figure 0-1.

To launch Quartus,

1. Either select the optimized implementation and click on the tool bar or right-click the implementation and choose Place and Route Chip. The Place and Route dialog box appears.

Figure 3-15 Place and Route Dialog Box



Place and Route

Basename of files: tutor

Place Route Directory:

Timing and Bus Style

□ Timing Specifications

Bus Style: %s[%d:%d]

2. Choose a place and route directory.

3. To change directories, either type the new directory name or click Browse.

4. Specify whether or not to export timing specifications with the netlist, using the Timing Specifications check box.

Running the FPGA Scripting Tool

The FPGA Scripting Tool (FST) implements a Tcl-based command-line interface to all of the synthesis and optimization features of FPGA *Express*. FST is designed for users who

- Prefer command-line or keyboard-based interfaces to graphical interfaces.

- Understand the capabilities of the tool fully and want to specify operations as quickly as possible. For example, a specification that requires multiple mouse clicks and keyboard entries could be entered as a single command.

You can run FST from the command line of the FPGA *Express* shell, or execute batch files of FST commands.

To run FST in a shell window:

1. Start the shell either by choosing Start > Programs > Synopsys > FPGA *Express* Shell in Windows or by typing `fe_shell` in a UNIX shell or command window.
2. In the FST shell, execute FST commands. All of the commands are documented in the man pages.
   - To access FST man pages, type the following at the FST prompt:

     ```
     man commandname
     ```

   - To display help about the man pages, type

     ```
     man help
     ```

Summary

You have now completed the FPGA *Express* design flow tutorial. You have learned how to perform these tasks:

1. Creating design source files
2. Setting up an FPGA *Express* project, and adding, analyzing, and debugging source files
3. Creating a design implementation, specifying the top-level design and target device, and elaborating the design
4. Entering performance constraints and controls to guide the optimization process
5. Optimizing the design
6. Analyzing timing
7. Viewing schematics
8. Generating netlists and report files
9. Launching Quartus

Now you are ready to use FPGA *Express* for your own designs.

Index